ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

The Full-Stack Software Developer as a System Cohesion Architect: A Definitive Competency-Based Redefinition based on the K72T004 National Standard (Sri Lanka)

Dr. W. A. Lakdimuthu Pasinda Weerasuriya[1];
Dr. Senarath Mudiyanselage Nirodha Rupasinghe[2];
Dr. J. H. U. S. Bhatiya Tissera[3]; K. A. Achintha Chamara Lakshan[4];
Dr. M. A. Tharaka Pathum Sankalpa[5]

[1]Ph.D. in Software Engineering, IIC University of Technology, Cambodia; [1]MSc in Information Technology, Cardiff Metropolitan University, UK; [1]BEng in Software Engineering, IIC University of Technology; [1]Sun Certified Java Programmer, Sun Microsystems, USA; [1]Oracle Certified Professional – Mobile Application Developer, Oracle University, USA; [1]Oracle Certified Professional – Web Component Developer, Oracle University, USA.

[2]Ph.D. in Software Engineering, IIC University of Technology, Cambodia; [2]MSc in Information Technology, Cardiff Metropolitan University, UK; [2]BEng in Software Engineering, IIC University of Technology; [2]Oracle Certified Professional – Java Programmer, Oracle University, USA; [2]Oracle Certified Professional – Mobile Application Developer, Oracle University, USA

[3]Ph.D. in Software Engineering, IIC University of Technology, Cambodia; [3]MSc in Information Technology, Cardiff Metropolitan University, UK; [3]BEng in Software Engineering, IIC University of Technology; Sun Certified Java Programmer, Sun Microsystems, USA; [3]Sun Certified Mobile Application Developer, Sun Microsystems, USA; [3]Sun Certified Web Component Developer, Sun Microsystems, USA; [3]Sun Certified Business Component Developer, Sun Microsystems, USA

[4]SCQF Level 9 - Graduate Diploma in Software Engineering, Scottish Qualifications Authority, Scotland; [4]SCQF Level 8 - Higher Diploma in Software Engineering, Scottish Qualifications Authority, [4]Scotland; SCQF Level 7 - Diploma in Software Engineering, Scottish Qualifications Authority, Scotland.

[5]Ph.D. in Software Engineering, IIC University of Technology Cambodia; [5]BSc in Software Engineering; [5]SCQF Level 9 - Graduate Diploma in Software Engineering, Scottish Qualifications Authority, Scotland; [5]SCQF Level 8 - Higher Diploma in Software Engineering, Scottish Qualifications Authority, [5]Scotland; SCQF Level 7 - Diploma in Software Engineering, Scottish Qualifications Authority, Scotland.

Publication Date: 2025/11/24

Abstract: The role of the Full-Stack Software Developer (FSD) is often ambiguously defined, primarily focusing on broad implementation skills across front-end and back-end development. This report addresses this definitional crisis by rigorously analyzing the Sri Lankan National Competency Standard for Full Stack Software Development (K72T004). The K72T004 standard mandates a two-tiered progression: NVQ Level 5 (Implementation Baseline) and NVQ Level 6 (Governance). Our analysis reveals that Level 6 competencies, encompassing comprehensive system integration management, holistic security ownership, and robust process governance, elevate the FSD beyond a mere generalist coder. We assert that the FSD, as defined by K72T004, functions as a System Cohesion Architect, responsible for designing and governing the integrity of data flow, security models, and quality pipelines across the entire application stack. This redefinition, grounded in a nationally

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

recognized vocational standard, provides a clear framework for professionalization, mitigating architectural friction and aligning global industry expectations with the strategic value of an FSD possessing both deep implementation empathy and crucial architectural oversight.

How to Cite: Dr. W. A. Lakdimuthu Pasinda Weerasuriya; Dr. Senarath Mudiyanselage Nirodha Rupasinghe; Dr. J. H. U. S. Bhatiya Tissera; K. A. Achintha Chamara Lakshan; Dr. M. A. Tharaka Pathum Sankalpa (2025). The Full-Stack Software Developer as a System Cohesion Architect: A Definitive Competency-Based Redefinition based on the K72T004 National Standard (Sri Lanka). *International Journal of Innovative Science and Research Technology*, 10(11), 1275-1284. https://doi.org/10.38124/ijisrt/25nov808

I. INTRODUCTION

➢ *The Crisis of Ambiguity and the Redefinition Mandate*

- *The Definitional Crisis of the Modern Full-Stack Developer*

The role of the Full-Stack Software Developer (FSD) occupies a pivotal position in contemporary software development, yet it remains one of the most loosely and ambiguously defined roles in the global technology sector.[1] Traditionally, the FSD has been relegated to the status of a generalist coder—a professional competent across both the Front-End (client-side) and Back-End (server-side) implementation layers. This vague, implementation-focused definition has precipitated several critical organizational challenges, including inconsistent hiring standards, ill-defined career progression paths, and an immense cognitive load placed upon professionals tasked with managing systemic complexity without the commensurate formal architectural mandates.[1]

The fundamental flaw in defining the FSD based solely on implementation breadth is the failure to recognize the unique systemic oversight required to manage the interaction points between layers. When developers are responsible for the entire vertical slice of an application, their core value pivots from generating high volumes of code to ensuring the stability and integrity of the whole structure.

This report addresses this persistent definitional gap by conducting a rigorous analysis of the National Competency Standard for Full Stack Software Development (Competency Standard Code: K72T004), developed in Sri Lanka and formally endorsed by the Tertiary & Vocational Education Commission (TVEC).[1, 1] The structured nature of K72T004, which formalizes the FSD profession across two additive tiers—NVQ Level 5 (Implementation) and NVQ Level 6 (Governance)—provides an evidence-based solution to the global ambiguity.

The central assertion of this analysis is that the FSD, as rigorously defined by the K72T004 standard, must be formalized and globally recognized as a System Cohesion Architect. This definition is strategically necessary, predicated upon the mandatory inclusion of Level 6 governance competencies that prioritize architectural integrity, comprehensive system integration management, and holistic security oversight over mere hands-on coding volume.[1]

- *The K72T004 National Competency Standard as a Regulatory Solution*

The K72T004 standard is a foundational policy document within Sri Lanka's National Vocational Qualification (NVQ) framework. Developed by industry practitioners and validated by the National Apprentice & Industrial Training Authority (NAITA), it carries the regulatory weight necessary to establish standardized curriculum and professional expectations.[1] The structure explicitly mandates progression: Level 5 forms the Diploma foundation, establishing core technical proficiency, and Level 6 forms the Higher National Diploma, introducing advanced managerial and architectural responsibilities.[1]

This structure elevates the definition of the FSD from a loose industry consensus based on technological trends (e.g., using specific JavaScript or Python frameworks [3]) to a nationally recognized vocational and academic policy position. The regulatory endorsement from bodies like the TVEC ensures that training curricula, assessment procedures, and certification pathways are standardized and measurable.[5] This contrasts sharply with informal, market-driven job descriptions that fluctuate based on regional technological demand.

The implications of this structured approach extend beyond national boundaries. By establishing a clear, two-tiered model encompassing foundational skill (Level 5) and architectural mandate (Level 6), the Sri Lankan framework provides a blueprint for other nations seeking to professionalize their IT workforce and align academic qualifications with robust industry needs.[6] It offers a globally transferable model for standardizing the FSD role, thereby ensuring better employability, economic growth, and global competitiveness.[5]

A visual representation of this structure highlights the compulsory progression inherent in the NVQ framework.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

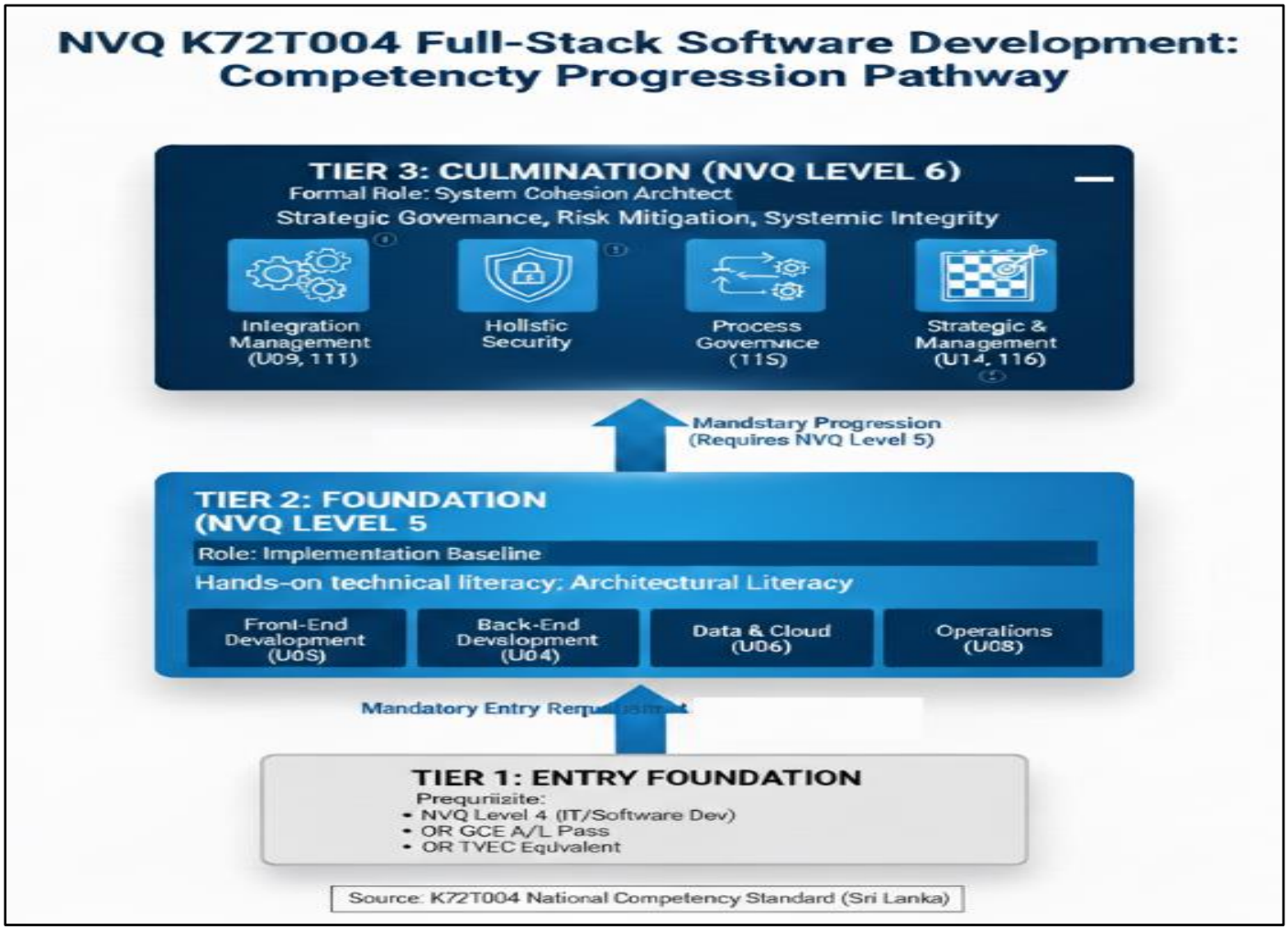

Fig 1 NVQ K72T004 Full-Stack Software Development: Competency Progression Pathway

II. THE FOUNDATIONAL PILLARS

➢ *Implementation and Architectural Literacy (NVQ Level 5)*

- *Level 5: The Non-Negotiable Prerequisite*

The NVQ Level 5 competencies establish the essential technical literacy that serves as the foundation for the FSD role. These competencies are non-negotiable prerequisites because they equip the professional with the hands-on understanding necessary to analyze the technical context, appreciate component limitations, identify potential failure points, and accurately estimate technical debt across the entire application.[1] Without this direct, experiential knowledge, subsequent architectural decisions risk becoming impractical or divorced from contemporary implementation realities.

The core units in Level 5 ensure comprehensive breadth across the software development lifecycle:

✓ *K72T004U01 (Conduct Feasibility Study):*

This foundational unit requires the gathering of requirements (categorized as functional and non-functional) and verifying the technical, operational, and financial feasibility of a project.[1] This serves as the initial step in architectural planning, ensuring the design aligns with business constraints.

✓ *K72T004U02 (Prepare Software Design Documents):*

This unit mandates the creation of the Software Requirement Specifications (SRS), design diagrams (including Use Case, Activity, and Sequence diagrams), UI/UX wireframes/prototypes, and Entity Relationship (ER)/Extended Entity Relationship (EER) diagrams.[1] Proficiency here ensures the FSD can translate high-level requirements into structured, understandable documentation, a critical requirement for any architect.

✓ *K72T004U03 (Develop Front-End of the System):*

This competency requires implementing UI/UX components using selected programming languages/frameworks, ensuring compatibility and responsiveness on various devices, and applying backend integration.[1] It provides direct knowledge of client-side constraints and security surface area.

✓ *K72T004U04 (Develop Back-End of the System):*

The focus here is on setting up the development environment, configuring version control, selecting appropriate back-end languages/frameworks, and developing business logic components that adhere to the system architecture document.[1] This knowledge is vital for

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

understanding data processing limitations and core system behavior under load.

✓ *K72T004U06 (Manage Relational and NoSQL Databases and Cloud Services):*

This unit is critical, covering data normalization, relational schema creation, selection and configuration of database management systems (DBMS), executing CRUD operations, and managing cloud storage services.[1] This ensures foundational expertise in data persistence integrity and optimizing the data layer.

✓ *K72T004U08 (Deploy and Maintain Software System):*

This unit acts as the functional bridge to Level 6. It requires the integration of the Front-End and Back-End (P.C. 1.1), preparation of system release notes, and deployment of the tested system.[1] This hands-on experience in the release cycle is a necessary precursor for the managerial oversight required in Level 6 DevOps (U15).

- *The Non-Friction Architect: Pragmatic Design*

The explicit requirement for mandatory hands-on implementation skills at Level 5 before advancing to Level 6 management functions fundamentally shapes the type of architect the FSD becomes. Traditional Software Architects are frequently criticized for creating impractical, overly complex, or outdated designs because they lack continuous, daily exposure to implementation realities, leading to a "detachment from the code" problem.[1]

By contrast, the K72T004 standard mandates that the System Cohesion Architect retains this vital implementation empathy. Their Level 6 architectural decisions are inherently informed by current, tangible knowledge of component limitations, deployment complexity (as gained in U08), and technical debt management (L5 units U03, U04, U06).[1]

This mandatory foundation ensures that the FSD acts as a Non-Friction Architect, one capable of translating high-level strategy into pragmatic, deployable, and maintainable code bases. This competency structure effectively bridges the conventional chasm between theoretical pure architecture and daily development execution, providing a unique advantage in complex environments.

The following table summarizes the strategic purpose of the Level 5 units in establishing this architectural literacy.

Table 1 Level 5 Competencies: Establishing Architectural Literacy

Unit Code	Title	Key Function	Architectural Implication
K72T004U03	Develop Front-End	UI/UX implementation, responsiveness, frameworks.	Understanding client-side constraints and security surface area.
K72T004U04	Develop Back-End	Business logic implementation, version control.	Understanding data processing, scalability constraints, and core system behavior.
K72T004U06	Manage DB/Cloud	Data modeling, normalization, cloud storage configuration.	Ensuring data integrity and optimizing the persistence layer (NFRs).
K72T004U08	Deploy and maintain	System integration, deployment, release notes preparation.	Functional knowledge of system release lifecycle (prerequisite for U15 DevOps governance).

III. THE CORE MANDATE

➢ *Governance and Systemic Integrity (NVQ Level 6)*

The definitive aspect of the FSD role lies in the NVQ Level 6 competencies, which introduce a necessary pivot from operational execution to supervisory governance. Level 6 provides the structural evidence confirming that the modern FSD is inherently a System Cohesion Architect, moving their focus from *how* to build individual components to *how* to ensure the entire system interacts reliably and sustainably.[1]

The 8 core units of Level 6 (K72T004U09 to K72T004U16) explicitly introduce responsibilities centered around governance, risk mitigation, and systemic integrity.[1] A critical observation of the Level 6 units is the consistent use of managerial language in their titles and performance criteria, such as "Manage," "Execute," and "Finalize".[1] This confirms that the expected performance involves planning, oversight, strategic decision-making, and coordination, signifying a true elevation to an architectural role.

➢ *Domain 1: Integration Management - The Nexus of Cohesion (U09 & U11)*

System cohesion is defined by the quality and reliability of the interfaces between components. The FSD's mandatory ownership of these boundaries is the primary structural validation for the "Cohesion Architect" title.

- *Internal Cohesion: Defining the Stack Contract (U09)*

The integrity of the internal architecture is managed through K72T004U09 (Manage Front-End and Back-End integration). This unit necessitates expertise far beyond simply writing endpoints. It requires the FSD to manage the stability of the communication *contract* between the client and server layers, including defining communication protocols (such as REST or GraphQL), data serialization formats, and ensuring consistent interface stability.[1]

The performance criteria of U09 demand not only the coding or development of components but also the formal finalization and testing of the integration as per the system requirement (P.C. 4.1, 4.2).[1] This management function moves the FSD from merely contributing code to owning the structural design that dictates how data flows and components interact reliably. The FSD must leverage the implementation knowledge gained in Level 5 (U03 and U04) to design pragmatic, non-friction internal contracts that prevent integration bottlenecks.[8]

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

- *External Cohesion: Architecting the System Boundary (U11)*

When an application interacts with external systems or provides services to partners, the FSD's role expands to that of the external cohesion architect. This responsibility is codified in K72T004U11 (Manage APIs and web services development).

This unit mandates several high-level functions that are traditionally reserved for specialized integration architects.[9] Specifically, the FSD must design the architecture of the API/Web services (P.C. 1.3), utilize necessary authentication modules (P.C. 2.2), and rigorously test not only functionality (P.C. 3.1) but also vulnerability (P.C. 3.2) and performance (P.C. 3.3).[1] Furthermore, the FSD is responsible for the API document preparation (P.C. 2.4), ensuring that the external interface is robust, secure, and comprehensively documented for consumers.[1]

This control over the system boundary ensures that the FSD owns the definition of external contracts, manages API versioning, and validates performance guarantees—all core architectural responsibilities.

- *Enforcing Consistency via Interface Governance*

The structural weakness in software systems often manifests when interfaces—internal or external—degrade due to inconsistent contract definition or technical drift.[1] By making the System Cohesion Architect responsible for designing and managing the stability of these interfaces (U09, U11) and enforcing their security (U11 P.C. 3.2), the K72T004 standard mandates proactive interface governance.

This function ensures that components adhere to structural standards from the outset, preempting cohesion failure. The mandatory Level 5 implementation base provides the essential technical context, allowing the Level 6 FSD to design internal contracts and external interfaces that minimize friction, reduce integration bottlenecks, and maintain a unified project vision throughout the development lifecycle.[8] This systematic approach to cohesion is crucial for maintaining the long-term integrity of complex application landscapes.

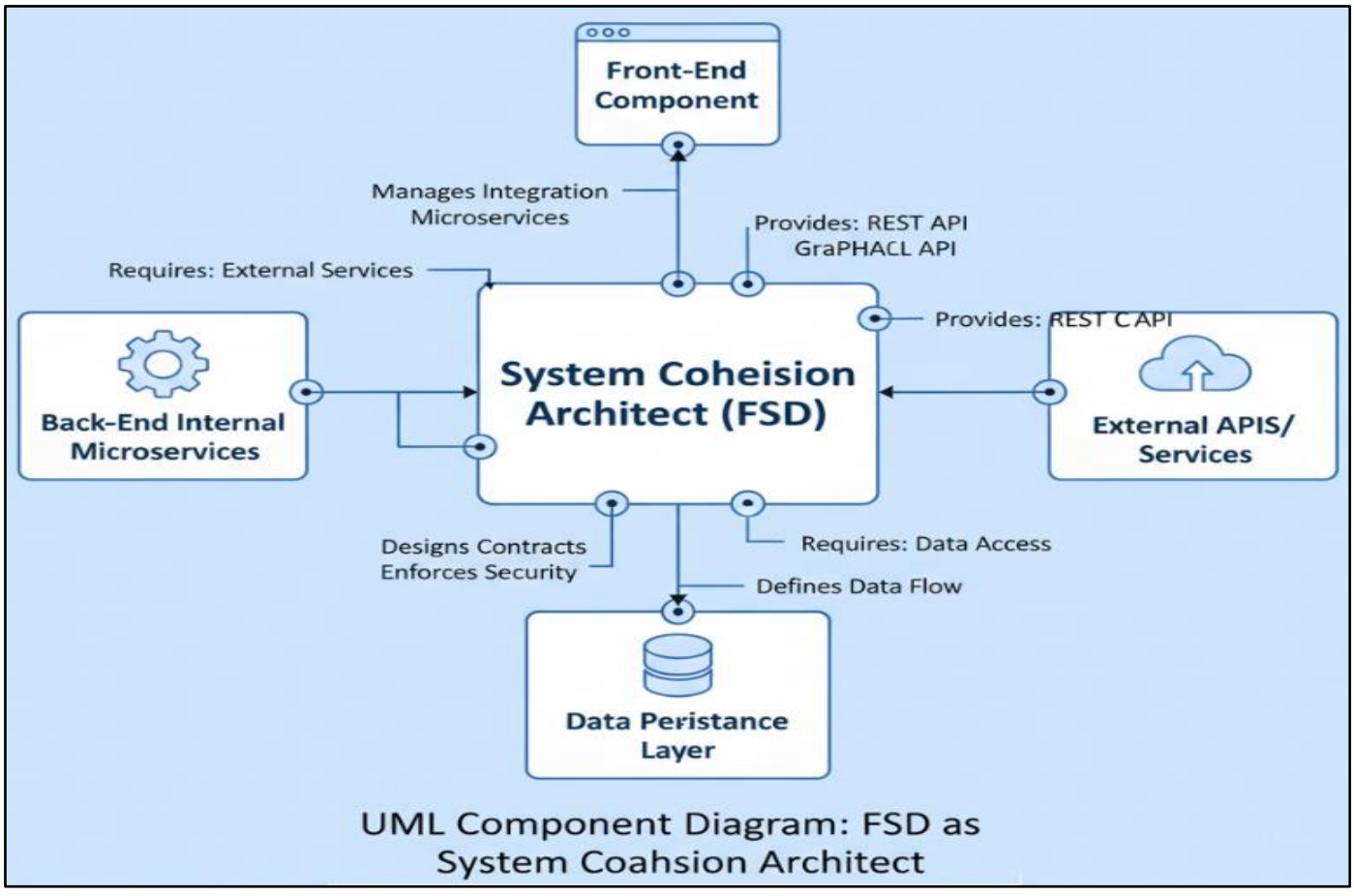

Fig 2 UML Component Diagram: FSD as System Cohesion Architect

➢ *Domain 2: Holistic Security Ownership (U12)*

Unit K72T004U12 (Manage security of the full stack development) offers the most direct evidence that the FSD carries an architectural, cross-layer accountability for risk management, a duty fundamentally different from that of a specialist developer.

- *System-Wide Threat Modeling Mandate*

The definition of the FSD as a System Cohesion Architect is validated by the unit's mandate to identify current and predict possible security threats (P.C. 1.1, 1.2) and categorize them, essentially requiring the FSD to conduct continuous, system-wide threat modeling.[13]

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

Crucially, the unit demands the application of security best practices across five distinct, interlinked architectural domains: Front-End, Back-End, Database, Cloud, and API/Web services (P.C. 2.1–2.5).[1] This holistic view stands in stark contrast to the siloed security concerns of single-layer specialists—where a database administrator handles SQL access control, and a front-end developer focuses on XSS mitigation.[1]

The FSD must design the overarching security model, managing the delicate and often flawed authorization flow across the entire stack. This includes overseeing client-side token storage, server-side authentication and authorization logic, and ensuring proper database hardening techniques are applied (a competency further detailed in U10).[1]

- *The Central Custodian of Systemic Trust*

Security vulnerabilities frequently exploit the "seams" of the application, originating at the integration boundaries between layers—the exact domain owned by the FSD.[1] For instance, failures in sanitizing user input can lead to XSS or SQL injection attacks that traverse multiple layers.[10] The FSD's mandatory oversight ensures these cross-layer threat vectors are consistently addressed.

Furthermore, U12 mandates actively evaluating and updating applied security measures (P.C. 3.1), conducting systematic security audits and vulnerability assessments (P.C. 3.2), and monitoring and responding to security incidents (P.C. 3.4).[1] These are clear governance functions that confirm the FSD's role as the central custodian of the system's trust boundaries and integrity.

- *Risk Centralization and Accountability*

The traditional fragmentation of security responsibility across specialized teams often creates blind spots, leading to systemic failures. By centralizing security ownership under U12, K72T004 ensures that the FSD, as the System Cohesion Architect, is the single point of accountability for systemic risk. This demands the strategic design of an interwoven security fabric rather than a collection of isolated, siloed defenses.

The mandate to manage security holistically ensures that Non-Functional Requirements (NFRs) like security and data integrity are integrated into the core architectural design, not treated as afterthoughts.[14] The FSD must leverage their comprehensive technical understanding from Level 5 to design secure authentication mechanisms, enforce secure storage, and mandate validation and sanitization procedures across all system inputs.[1] This strategic management of NFRs is a fundamental justification for the designation of architectural authority.

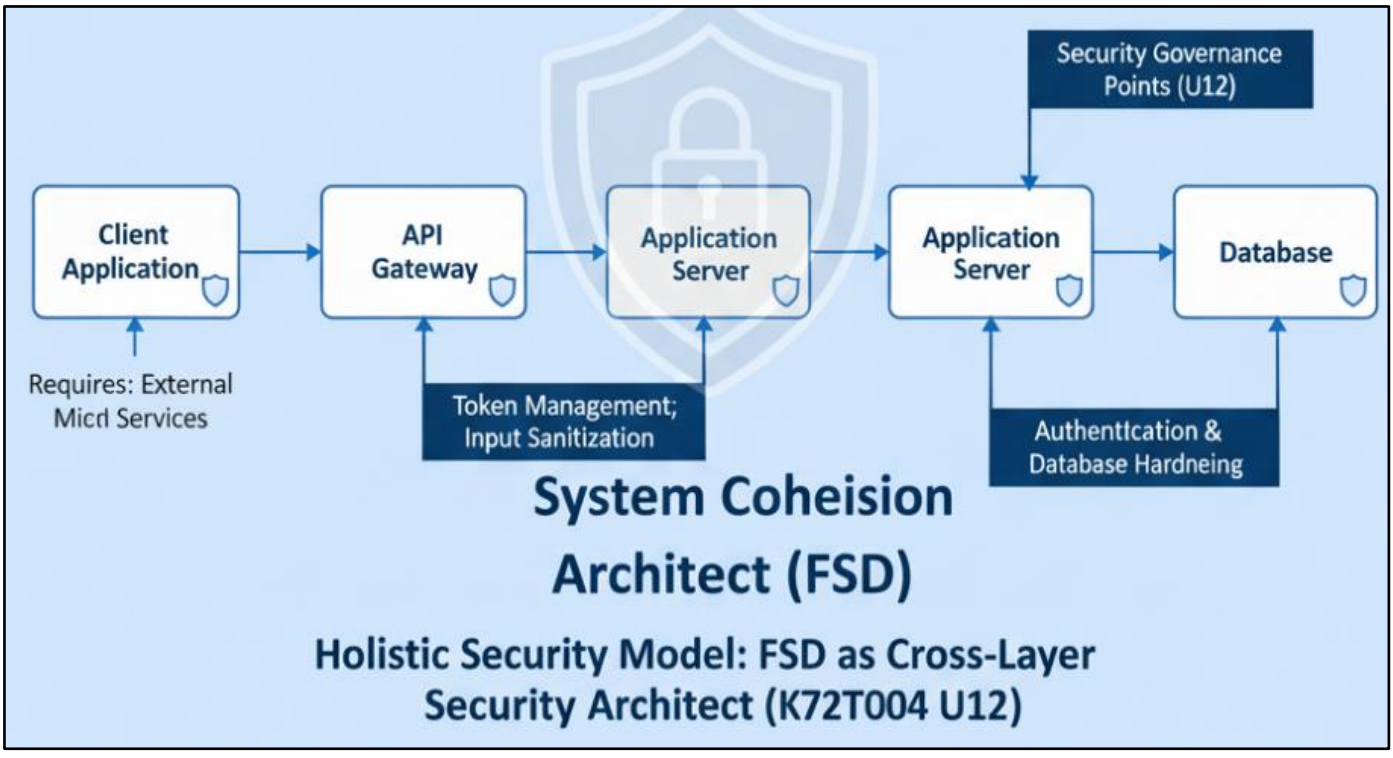

Fig 3 Holistic Security Model: FSD as Cross-Layer Security Architect (K72T004 U12)

➢ *Domain 3: Quality and Process Governance (U13 & U15)*

The System Cohesion Architect's responsibility extends into defining the system's long-term sustainability and the reliability of its delivery process. This governance is codified in the Quality Assurance and DevOps units.

- *Quality Design and Assurance (U13)*

K72T004U13 (Manage quality assurance) defines the FSD's responsibility for quality system design. This moves the role beyond personal code debugging and testing to establishing system-wide quality standards and procedures.[1]

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

This unit requires the preparation of a Quality Management Plan (P.C. 1.3), mapping requirements to the testing life cycle (P.C. 1.2), and verifying overall test processes.[1] The FSD must define and enforce the quality gates that ensure the system is viable beyond initial deployment. Given that insufficient testing is directly linked to massive software failure rates [15], this mandate to manage test case verification (P.C. 2.1) and utilize test automation tools (P.C. 3.1) signifies a crucial managerial function. The FSD must ensure that performance, security, and functional quality are systematically checked across the integrated system, providing quality oversight for the entire stack.

- *Process Cohesion Architect: Governing DevOps (U15)*

The implementation of Continuous Integration/Continuous Delivery (CI/CD) pipelines is a technical task often delegated to DevOps Engineers.[16] However, K72T004U15 (Manage DevOps) defines the FSD as the process cohesion architect—the professional who governs the automation process to ensure it adheres strictly to the architectural vision and maintains systemic integrity.[1]

U15 mandates key architectural decisions related to deployment: configuring the production environment (P.C. 1.2), setting up necessary tools (CI/CD, monitoring, logging, version control) (P.C. 1.4), and implementing critical Non-Functional Requirements like High Availability and Failover mechanisms (P.C. 1.5).[1] While a DevOps Engineer focuses on tool functionality, the FSD (U15) focuses on the *architectural outcome* of the automated process.

- *Architecting Reliability via Automation (IaC Enforcement)*

The reliability and scalability of modern architectures depend heavily on consistent, repeatable deployments, typically achieved through Infrastructure as Code (IaC).[18] The K72T004 structure forces the System Cohesion Architect to understand and govern this domain.

The Level 6 FSD must utilize their foundational Level 5 deployment knowledge (U08) and leverage their governance mandate (U15) to strategically enforce IaC standards within the CI/CD pipeline.[19] This includes ensuring that security policies established in U12 are translated into "Policy as Code" [18], preventing environment drift, and reducing the incidence of manual misconfigurations.[18]

By managing the DevOps process (U15), the System Cohesion Architect ensures that the architectural blueprint—covering infrastructure configuration, high availability, and security protocols—is consistently and securely realized in production environments. This ensures process cohesion by systematically mitigating the high cognitive burden associated with complex, manual operations.[1]

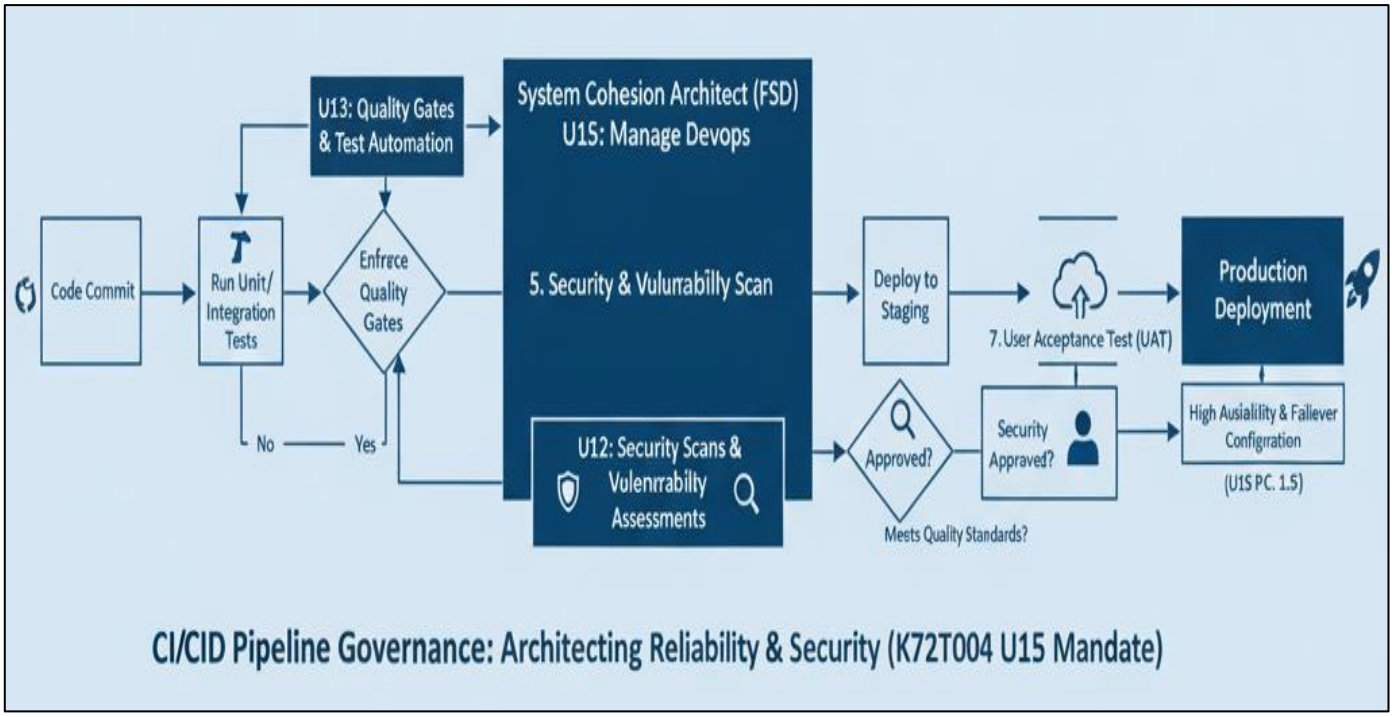

Fig 4 CI/CD Pipeline Governance: Architecting Reliability and Security (K72T004 U15 Mandate)

➢ *Domain 4: Strategic and Operational Management (U10, U14, U16)*

The remaining Level 6 units formalize the FSD's managerial and strategic scope, complementing the cohesion mandates by addressing the long-term viability, project feasibility, and business continuity of the system.

- *Advanced Data and Enterprise Management (U10)*

K72T004U10 (Manage enterprise databases and cloud databases) moves the FSD beyond the functional data management of Level 5 (U06) to enterprise-level architectural concerns. The focus shifts to optimization, performance, and advanced security.[1]

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

This unit requires restructuring tables to maximize storage efficiency (P.C. 2.1), optimizing queries for data retrieval performance (P.C. 2.2), and implementing advanced database features like triggers and stored procedures (P.C. 1.7).[1] Furthermore, U10 emphasizes database hardening configuration (P.C. 1.2), including fault tolerance and high availability planning (P.C. 4.6). These activities are crucial for governing the data layer's long-term performance and security, integral NFRs for scalable enterprise applications. The FSD manages the technical debt inherent in the most critical component of data-driven systems.

- *Project and Resource Execution (U14)*

The architectural authority must be paired with effective resource management. K72T004U14 (Execute project management functions) formalizes the FSD's role in project governance.

This unit mandates identifying project scope and objectives (P.C. 1.1), selecting the appropriate project management methodology (e.g., Agile or Waterfall), and developing a comprehensive project plan, including timelines, milestones, and resource allocation (P.C. 1.3).[1] The System Cohesion Architect must integrate the highly technical requirements (U09, U12, U15) into a coherent delivery schedule, coordinating teams and stakeholders to ensure the architectural vision is delivered on time and within budget.[2] This unit confirms the FSD's supervisory role in operational execution.

- *Viability and Business Continuity (U16)*

The final architectural concern is the system's life cycle and continued operational existence. K72T004U16 (Manage maintenance and support services) assigns the FSD accountability for post-deployment health.

This unit requires the negotiation and finalization of maintenance and support agreements (P.C. 1.2), the management of change requests and system issues (P.C. 2.1–2.3), and continuous performance optimization and security enhancement (P.C. 4.2, 4.3).[1] Crucially, the FSD is mandated to implement regular backups and disaster recovery plans for business continuity (P.C. 5.1). This competency confirms that the FSD's mandate is strategic, covering the system's entire lifespan and ensuring its long-term viability and adherence to regulatory compliance.

- *The Integration of Non-Functional Requirements into Strategic Planning*

The mandate of the System Cohesion Architect is defined by their comprehensive ownership of Non-Functional Requirements (NFRs). These architectural requirements, such as performance, security, scalability, and maintainability, determine the viability of an application.[14]

A direct examination of the Level 6 units reveals systematic coverage of the primary NFR spectrum: U10 (Performance/Hardening), U12 (Security/Risk), U13 (Quality), U15 (High Availability/Failover), and U16 (Disaster Recovery/Maintenance).[1] By packaging these NFR responsibilities within the FSD role and mandating their integration with project planning (U14), the K72T004 standard requires that the System Cohesion Architect treats NFRs as essential parts of the strategic plan, rather than allowing them to be secondary concerns left to external specialists. This structured assignment of responsibility guarantees that architectural decisions prioritize long-term systemic health and cohesion over short-term implementation expediency.

IV. GLOBAL COMPARISON AND POLICY IMPLICATIONS

The standardized definition provided by the K72T004 framework offers a structured advantage over informal industry definitions, allowing for clearer differentiation from related professional roles.

➢ *Differentiating the Cohesion Architect from the Specialist*

The K72T004 model uniquely positions the FSD at the intersection of architectural authority and hands-on implementation:

- *FSD vs. Traditional Architect:*

Traditional architects, focused on high-level conceptual design, often suffer from a literacy gap, resulting in impractical or non-optimal designs.[1] The K72T004 FSD explicitly avoids this pitfall by mandating Level 5 implementation literacy, ensuring their architectural directives are pragmatic and informed by contemporary technical constraints.

- *FSD vs. Specialist Developer:*

A specialist may possess advanced skills in one Level 5 area (e.g., expert SQL tuning in U06 or highly optimized server coding in U04) but fundamentally lacks the mandated Level 6 systemic oversight necessary for integration (U09) and holistic security (U12).[1] Their focus remains on component depth, whereas the FSD's focus is on component interaction and integrity.

- *FSD vs. DevOps Engineer:*

While the DevOps Engineer specializes in infrastructure automation, tools, and monitoring (CI/CD pipeline execution) [16], the FSD (U15) governs the *process configuration* to ensure high availability, security (U12), and quality (U13) are structurally designed into the pipeline itself.[1] The FSD dictates *what* the pipeline must enforce architecturally; the DevOps engineer implements *how* it is automated.

➢ *The FSD as Cognitive Load Mitigator*

A significant challenge in modern, highly decomposed software systems is the exponential increase in cognitive load placed upon development teams tasked with integrating numerous specialized services and managing complex, distributed architectures.[1]

The System Cohesion Architect's primary function—mandatory ownership of system integrity across all boundaries (U09, U11, U12)—directly addresses this complexity. By possessing end-to-end oversight and

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

managing the integration points, the FSD proactively identifies technical bottlenecks and conflicts that specialized, siloed teams would inevitably overlook.[8] This cross-layer visibility streamlines communication between design, data, and operations teams, reduces integration friction, and accelerates product iteration.[10] The FSD defined by K72T004 thus serves as a strategic resource for mitigating cognitive load by centralizing systemic knowledge and crucial architectural decision-making.

The following table rigorously validates the distinction between the K72T004 FSD and common industry roles:

Table 2 Role Comparison: K72T004 System Cohesion Architect Model

Role	K72T004 Competency Profile	Primary Risk Mitigated	Value Proposition
Specialist Developer	Advanced L5 skills (e.g., U04 coding), lacks mandatory L6 governance.	Technical Sub-optimization within a single silo.	Deep technical execution and optimization in one layer.
Traditional Architect	Possesses L6 mandate, potentially lacks up-to-date L5 implementation literacy.	Impractical/Outdated Design and technical debt accumulation.	High-level conceptual design and foundational documentation.
System Cohesion Architect (K72T004 FSD)	Mandatory L5 implementation + mandatory L6 governance (U09, U12, U15).	Systemic Friction, Cohesion Failure, and Cross-Layer Risk.	Mandatory ownership of system-wide integrity, pragmatic architecture, and holistic risk management.

➢ *Implications for Global Curricula and Hiring*

The K72T004 standard offers a definitive structure for professionalization that is superior to purely technology-driven frameworks.[3] It provides a blueprint for global technology education bodies and corporate talent acquisition teams to stabilize career pathways.

For academic and vocational institutions, the Level 6 governance competencies provide clear objectives for advanced curriculum design. Global training providers (such as those offering professional certificates [20]) should align their advanced programs with these governance mandates, ensuring that FSD graduates are competent not only in implementation but also in the strategic, architectural oversight necessary for systemic integrity, making them universally valuable in high-demand roles worldwide.

V. CONCLUSION

➢ *The Formal Definition and Strategic Value*

- *Synthesis of the Argument*

The ambiguity surrounding the Full-Stack Software Developer role has long hindered professional standardization. The rigorous, competency-based framework established by the Sri Lankan National Competency Standard K72T004 provides the definitive, evidence-based solution to this crisis. The analysis unequivocally demonstrates that the NVQ Level 6 mandates—particularly in Integration Management (U09, U11), Holistic Security Ownership (U12), and Process Governance (U15)—transcend mere technical execution and constitute core architectural responsibilities. The FSD, under this standard, is responsible for governing the structural relationships, risk profile, and delivery pipeline of the entire application.

- *Formal Definition and Policy Recommendations*

The strategic value derived from the K72T004 standard justifies the formal elevation of the professional title.

➢ *Formal Proposed Definition:*

A Full-Stack Software Developer is a System Cohesion Architect responsible for designing, implementing, and governing the integrity of the data flow, security model, and quality pipeline across all layers of an application stack. This role requires the implementation breadth of NVQ Level 5 to inform the critical governance oversight mandates of NVQ Level 6.[1]

It is recommended that international governmental, academic, and industrial bodies formally recognize and adopt this comprehensive, competency-based definition. This measure will stabilize global career pathways, standardize qualification recognition, reduce architectural friction, and accurately align professional expectations with the high demands and unique systemic value generated by the System Cohesion Architect in the modern digital economy. This redefinition transforms the FSD from a generalist coder into a critical strategic leader, responsible for the long-term health and stability of the entire software ecosystem.

REFERENCES

[1]. Tertiary & Vocational Education Commission (TVEC), National competency standard for full stack software development (competency standard code: K72T004). Sri Lanka, 2024.
[2]. Tertiary and Vocational Education Commission (TVEC), Full stack software development supervisor NVQ level 5. Career One – Connecting competencies with opportunities. 2025, (https://www.careerone.gov.lk/career-guidance/career-information/contents/details/NTE=)
[3]. Talent500, Top 10 full stack developer frameworks in 2025. https://talent500.com/blog/top-full-stack-developer-frameworks/
[4]. IBM / Coursera, IBM full stack cloud developer professional certificate. 2025, https://www.coursera.org/professional-certificates/ibm-full-stack-cloud-developer
[5]. Tertiary and Vocational Education Commission

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25nov808

(TVEC), Sri Lanka's TVET system: pathways to competency and career progression. 2025, https://tvec.procons.lk/trainee/
[6]. Asian Development Bank (ADB), Framework for training reform in Sri Lanka. 2025, https://www.adb.org/sites/default/files/publication/28291/framework-training-reform-sri.pdf
[7]. Prezi, Understanding the national vocational qualifications (NVQ) framework: pathways to competency and career progression. 2025, https://prezi.com/p/kr3vpyd_aqvb/understanding-the-national-vocational-qualifications-nvq-framework-pathways-to-competency-and-career-progression/
[8]. Amazon Web Services (AWS), What is full-stack development? 2025, https://aws.amazon.com/what-is/full-stack-development/
[9]. Spotterful, Integration architect responsibilities and required skills. 2025, https://spotterful.com/blog/job-description-template/integration-architect-responsibilities-and-required-skills
[10]. Wow Remote Teams, Full-stack developer job description: roles, responsibilities, and skills. 2025, https://wowremoteteams.com/blog/full-stack-developer-job-description/
[11]. Microsoft Support, Create a UML component diagram. 2025, https://support.microsoft.com/en-us/office/create-a-uml-component-diagram-aa924ecb-e4d2-4172-976e-a78fa157b074
[12]. IBM Docs, Component diagrams in UML. 2025, https://www.ibm.com/docs/en/dma?topic=diagrams-component
[13]. Black Duck, what is threat modeling? 2025, https://www.blackduck.com/glossary/what-is-threat-modeling.html
[14]. Vfunction, Software architect vs. Software engineer. 2025, https://vfunction.com/blog/software-architect-vs-software-engineer/
[15]. Index.dev, The ultimate checklist for releasing your full-stack web application. 2025, https://www.index.dev/blog/full-stack-web-application-release-checklist
[16]. Abhinowww (Dev.to), Difference between DevOps and Full-Stack Engineer roles. 2025, https://dev.to/abhinowww/difference-between-devops-and-full-stack-engineer-roles-responsibilities-and-key-skills-1dcl
[17]. Geeksfor Geeks, Requirements to become a full stack developer. 2025, [https://www.geeksforgeeks.org/hr/requirements-to-become-a-full-stack-developer/] (https://www.geeksforgeeks.org/hr/requirements-to-become-a-full-stack-developer/)
[18]. National Security Agency (NSA), Enforce secure automated deployment practices through Infrastructure as Code. 2025, (https://media.defense.gov/2024/Mar/07/2003407857/-1/-1/0/CSI-CloudTop10-Infrastructure-as-Code.PDF)
[19]. OWASP Cheatsheet Series, Infrastructure as Code security cheatsheet. 2025, (https://cheatsheetseries.owasp.org/cheatsheets/Infrastructure_as_Code_Security_Cheat_Sheet.html)