

Enterprise Architecture for Multi-Sided E-Commerce Marketplaces: A Completeness-Driven TOGAF-Zachman Approach

Stephen Edlin^{1)*}, Handri Santoso²⁾, Erick Dazki³⁾, Januponsa Dio Firizqi⁴⁾, Richardus Eko Indrajit⁵⁾

¹⁾²⁾³⁾⁴⁾⁵⁾Department of Information Technology, Pradita University, Tangerang, Banten, Indonesia

¹⁾stephen.edlin@student.pradita.ac.id*, ²⁾handri.santoso@pradita.ac.id,

³⁾erick.dazki@pradita.ac.id, ⁴⁾januponsa.dio@pradita.ac.id, ⁵⁾eko.indrajit@gmail.com

Submitted : Aug 8, 2026 | **Accepted** : Sep 6, 2026 | **Published** : Oct 1, 2026

Abstract: Indonesian micro, small, and medium enterprises (MSMEs) are moving into digital commerce, but the move remains scattered. National statistics show that 94.76% of e-commerce businesses sell through instant-messaging applications and only 17.23% through a marketplace platform, while 48.03% still arrange delivery themselves and cash remains the dominant settlement method. Multi-sided marketplace platforms can consolidate this activity, but designing them is architecturally demanding. Existing enterprise architecture (EA) studies also seldom verify whether every fundamental descriptive dimension of the enterprise has been addressed. This study proposes a completeness-driven EA for a multi-sided e-commerce marketplace. It integrates the TOGAF Architecture Development Method (ADM) as the design process, ArchiMate 3 as the modeling language, and the Zachman Framework's 5W1H ontology (What, How, Where, Who, When, Why) as an explicit completeness lens. Within a Design Science Research approach, the artifact is developed across TOGAF ADM Phases A to D and evaluated through architectural consistency checking, gap analysis, and a structured completeness assessment that maps every model element onto a perspective-by-aspect matrix under a stated population rule. Of the 24 cells assessed, 20 are fully populated, 3 are only partially populated, and 1 is weak. The What, How, Where, Who, and Why aspects are answered at all four perspectives, while the temporal When aspect accounts for every cell that falls short. The study contributes a reusable blueprint for marketplaces that advance SME digitalization, an escrow mechanism modeled end-to-end including its dispute path and regulatory driver, and a replicable procedure for assessing architectural completeness.

Keywords: ArchiMate; Architectural completeness; Enterprise architecture; SME digitalization; TOGAF ADM

INTRODUCTION

Indonesia is one of Southeast Asia's largest and fastest-expanding digital economies, with electronic commerce as its primary growth driver. Micro, small, and medium enterprises (MSMEs) are its most numerous participants and largest source of employment, so their digital transition carries national economic consequence (BPS-Statistics Indonesia, 2025; Lubis et al., 2025). Adoption has accelerated under government initiatives and platform incentives, yet many enterprises remain digitally present but operationally immature (Indiani et al., 2025).

In everyday practice, many Indonesian MSMEs still transact through fragmented and informal channels. National survey data quantify the pattern. Of Indonesian e-commerce

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

businesses, 94.76% sell through instant-messaging applications and 29.66% through social media, while only 17.23% sell through a marketplace platform (BPS-Statistics Indonesia, 2025; Ministry of Trade of the Republic of Indonesia, 2025). A further 48.03% deliver orders themselves rather than through a logistics partner. The most consequential weakness in this pattern is the absence of a neutral trust mechanism. Without escrow-style protection, a buyer who pays in advance bears the full hazard of non-delivery, while a seller who ships first bears the full hazard of non-payment. That dilemma is long documented in electronic-markets research (Hu et al., 2004).

A multi-sided marketplace platform is the structural response to these frictions, because it replaces informal bilateral arrangements with standardized commerce processes and institutional trust. Designing one is an enterprise-scale problem: at least three constituencies with differing incentives, several parallel revenue streams, and trust itself must be coordinated within a single design.

Enterprise architecture (EA) keeps business strategy, information systems, and technology mutually consistent, and mature EA practice is associated with digital-transformation success (Alghamdi, 2024; Dumitriu & Popescu, 2020; Lukáš et al., 2025). This study adopts TOGAF and its Architecture Development Method (ADM) as the design process and ArchiMate as the modeling notation, as Section 2 explains (The Open Group, 2022a, 2022b).

Two gaps in the literature motivate this study. The first is a domain gap. Existing TOGAF-based designs target single enterprises or the internal information technology of one SME (Izzati et al., 2020; Purnama et al., 2026; Sentosa et al., 2024; Widjaja et al., 2025). Studies of e-commerce platforms concentrate on technical qualities such as scalability without adopting any enterprise framework (Guntakandla, 2025). The literature reviewed here was identified by combining the terms enterprise architecture, TOGAF, ArchiMate, marketplace, and multi-sided platform in Google Scholar and the Indonesian SINTA index, restricted to 2019–2026 and to studies reporting an applied architecture artifact. Within that set, no study models a multi-sided e-commerce marketplace end-to-end through TOGAF ADM and ArchiMate. The second is a methodological gap. EA studies routinely validate their designs through gap analysis, which contrasts a current and a target state. They seldom verify architectural completeness, that is, whether the design answers every fundamental descriptive question about the enterprise. Completeness is exactly what the Zachman Framework addresses through its six interrogative aspects (Sowa & Zachman, 1992; Zachman, 1987).

This study bridges both gaps with a completeness-driven enterprise architecture. TOGAF ADM supplies the design process, ArchiMate 3 the modeling language, and the Zachman 5W1H ontology an explicit completeness lens over the resulting model. The design object is a multi-sided e-commerce marketplace intended to advance SME digitalization, with escrow-based trust as a first-class architectural concern.

Accordingly, the study pursues four research questions. RQ1: What motivation architecture aligns the marketplace's stakeholders, strategic goals, architecture requirements, and platform resources at the Architecture Vision (Phase A)? RQ2: How should the marketplace's revenue-generating core processes—commission, payment, and fulfillment—be architected across the business, application, and technology layers (Phases B to D)? RQ3: What structural gaps separate the fragmented As-Is condition of SME selling from the To-Be architecture, and how does the design close them? RQ4: To what extent is the architecture complete when assessed against Zachman's 5W1H aspects, and which aspects remain under-specified?

The paper makes three contributions. The first is a reusable end-to-end EA blueprint for an SME-empowering multi-sided marketplace, with explicit monetization and an escrow mechanism modeled down to its dispute path and regulatory driver. The second is a replicable method for integrating the Zachman 5W1H ontology as a completeness lens over a TOGAF-Archimate design. The third is a completeness assessment that verifies coverage and localizes the least-specified aspect as actionable future work.

LITERATURE REVIEW

Over three decades, EA has shifted from documenting information systems toward steering digital transformation, and recent evidence ties disciplined EA practice to transformation success

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

(Alghamdi, 2024; Lukáš et al., 2025). Two foundational schools continue to shape practice, and the contrast between them is what this study exploits. The Zachman Framework (Sowa & Zachman, 1992; Zachman, 1987) is a classification scheme rather than a procedure: six interrogatives—What, How, Where, Who, When, and Why—are crossed with six viewpoints. Its enduring contribution is the principle of completeness: a populated cell signals that a descriptive dimension has been addressed, while an empty one marks a blind spot (Iyamu, 2023). TOGAF contributes the complementary strength of process: its ADM sets out a cyclical path from the Architecture Vision through the Business, Information Systems and Technology Architectures to migration and governance (The Open Group, 2022b). The ADM can be tailored to different organizational contexts (Kornysheva & Deneckère, 2022) and applied to greenfield design (Oberle et al., 2023), directly analogous to specifying a new marketplace from first principles.

The connecting element between the two schools is a modeling language. ArchiMate is now the usual choice for rendering ADM deliverables across the business, application, and technology layers, and there is empirical evidence that it improves how architectural models are visualized and communicated (The Open Group, 2022a; Zhi & Zhou, 2022). Digital-platform business models are argued to require this kind of dedicated architectural framing rather than ordinary application design (Höfflin et al., 2025). What remains scarce is verification of coverage. Systematic reviews of EA evaluation catalogue the models and methods in use (Khairina et al., 2024). Across the studies compared in Table 1, validation is reported through gap analysis, descriptive reporting, performance testing, or conceptual comparison, and none reports a deliberate test of whether the design answers every descriptive dimension. That absence is the methodological opening this study exploits.

The applied EA literature is rich in sectoral case studies, particularly in Indonesia. TOGAF ADM has been applied to real-estate revenue processes (Widjaja et al., 2025), a challenger digital bank (Sentosa et al., 2024), advertising services (Pratama, 2024), e-government (Afarini & Hindarto, 2023) and healthcare imaging integration (Purnama et al., 2026). Hamdani et al. (2026) pair TOGAF-based EA with Design Science Research, a precedent for treating a blueprint as a research artifact. For the MSME segment, Izzati et al. (2020) design an IT blueprint for one enterprise and Hindarto and Putra (2023) architect a property-marketing platform. What unites these studies is a single organization or a one-sided service; none addresses a multi-sided marketplace that aggregates and monetizes the commerce of many independent SMEs.

On the e-commerce side, the literature divides into adoption and technology streams. Indiani et al. (2025) find perceived risk and infrastructure readiness decisive for SME e-commerce adoption, and Lubis et al. (2025) document how Indonesian MSMEs use digital marketing to widen the reach of creative products. On the technology side, Guntakandla (2025) shows that microservices and modular architecture improve scalability and fault isolation, but optimizes implementation without connecting platform technology to business strategy through any enterprise framework. Bridging strategy and technology is exactly EA's role, which is where this study sits. Hu et al. (2004) establish escrow as a trusted third party substituting institutional assurance for absent interpersonal trust. Escrow nevertheless rarely appears as an explicit architectural element in published EA blueprints, a further gap this study addresses. Table 1 positions this study against the closest prior work along two axes: the domain object and the validation approach.

Table 1. Comparative analysis of related research.

Authors	Focus / object	Framework / validation	Gap addressed by this study
Widjaja et al. (2025)	Five revenue processes in one firm	TOGAF ADM + ArchiMate; gap analysis	Single enterprise; not a multi-sided platform; completeness not assessed

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Authors	Focus / object	Framework / validation	Gap addressed by this study
Sentosa et al. (2024)	Digital-bank basic features	TOGAF ADM + ArchiMate; gap analysis	Financial product; no commerce aggregation; completeness not assessed
Izzati et al. (2020)	Internal IT of one SME	TOGAF ADM; descriptive	Architects one SME, not the aggregating platform
Guntakandla (2025)	E-commerce scalability	Microservices (no EA); performance testing	No business–IT alignment; no EA framework
Iyamu (2023); Reiter (2025)	Framework comparison	Conceptual; literature analysis	Discuss frameworks generally; no applied marketplace artifact
This study	Multi-sided marketplace empowering SMEs (commission, escrow, fulfillment)	TOGAF ADM + ArchiMate + Zachman 5W1H; consistency + gap + completeness	—

METHOD

This study follows the Design Science Research (DSR) paradigm, whose central output is a deliberately constructed artifact—here an enterprise-architecture blueprint—built for a specified class of problems and then judged against that intended purpose (Hevner & Chatterjee, 2010). DSR suits this study because the contribution is a designed solution rather than an empirical measurement. Its environment is Indonesian SME commerce with its sellers, buyers, logistics and payment partners, regulator and platform organization; its knowledge base is the TOGAF standard, the ArchiMate language, the Zachman ontology and the EA research reviewed above.

The artifact targets a class of problems—fragmented SME selling—rather than a single named firm. Its requirements are therefore derived from the documented characteristics of that class, as established in the reviewed literature and consolidated in a marketplace business model canvas, rather than from organization-specific interviews. This choice is consistent with prior TOGAF-based DSR work (Hamdani et al., 2026) and means the resulting design should be read as the ideal To-Be condition for the problem class.

The research proceeds in five stages, illustrated in Figure 1. The first stage is the literature review reported in Section 2. The second stage is requirements definition, conducted through a marketplace business model canvas. The canvas specifies the platform’s vision and mission, its twelve strategic resources, its partner and customer segments, and its ten revenue streams. Six of the twelve are carried into the Architecture Vision under an explicit selection rule. A resource is retained when it is the direct enabler of exactly one strategic goal, so that the mapping discriminates between goals. The remaining six are shared or supporting assets that enable several goals indistinguishably. Where one goal carries two requirements, as the trust goal does, the same resource enables both. The third stage is architecture design: TOGAF ADM Phases A to D, with every deliverable expressed in ArchiMate 3 using the Archi tool. The Preliminary Phase covers only framework, language and tool selection, plus the architecture principles reported in Section 4.1. Consistent with comparable studies, the scope is deliberately bounded at Phase D, because Phases A to D constitute the fundamental architecture-definition cycle, whereas the later phases

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

concern implementation governance that presupposes an operational organization (Purnama et al., 2026; Widjaja et al., 2025).

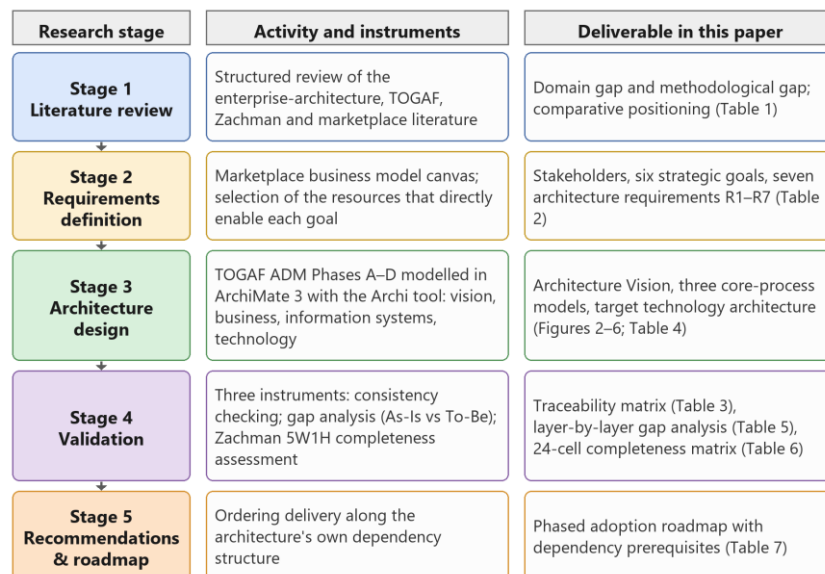


Fig. 1. Research methodology: the five-stage Design Science Research process, the instruments used at each stage, and the deliverable each stage contributes to this paper.

The fourth stage—central to this study’s methodological contribution—evaluates the artifact through three complementary instruments. The first is architectural consistency checking. It traces three chains: that every business process is served by an identified application service, that every application service accesses a defined data object, and that every data object is realized by an artifact deployed on the database server. No element is therefore left as a dangling dependency across layers. The second is gap analysis, which contrasts the As-Is baseline with the To-Be architecture layer by layer. The third and distinctive instrument is a structured completeness self-assessment based on the Zachman ontology. Every element of the model is mapped onto a matrix of the four architectural perspectives against the six Zachman aspects. The population rule is stated operationally in the next paragraph (Iyamu, 2023; Sowa & Zachman, 1992).

The three grades of population are defined so that a reader can reapply them. A cell is fully populated when at least one named element, table row, figure, or explicitly enumerated statement in the paper answers the corresponding interrogative at that perspective. The System/How cell, for example, is answered by the eleven application services named in the traceability matrix of Section 4.2. A cell is partially populated when the answer can only be derived from a relation between elements rather than read off an element of its own. The Business/When cell is the example, because ordering follows from ArchiMate triggering relationships while no element carries a duration. A cell is weakly populated when neither an element nor a relation answers it, as at the Technology/When cell, where no runtime timing or batch schedule is represented. A cell answered by neither an element nor a relation nor an implicit ordering would be recorded as empty; no cell in either assessment falls into that category. To give the resulting score a reference point, the same rule was applied a second time to a published TOGAF-ArchiMate design in this journal (Widjaja et al., 2025), using only the artifacts that paper presents.

The choice of four perspectives rather than Zachman’s full six is deliberate and follows from the scope of the design. The four used here—Contextual, Business, System, and Technology—are the ones that TOGAF ADM Phases A to D actually deliver. Zachman’s remaining two viewpoints, Detailed Representation and Functioning Enterprise, describe implementation-level and running-system artifacts, which fall outside the architecture-definition boundary set above; including them would guarantee empty cells that record the study’s scope rather than any defect in the design. Restricting the lens to the four architecturally relevant perspectives therefore keeps

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

every assessed cell answerable from the delivered model, and yields the 24 cells reported here rather than the full 36.

The fifth and final stage consolidates the findings into strategic recommendations and a phased adoption roadmap (Section 4.6).

RESULTS

4Architecture Vision (Phase A)

The design process begins with the Architecture Vision, which functions as the strategic contract between the platform's stakeholders and its architecture. Figure 2 presents the model and Table 2 sets out the resulting alignment chain. Six stakeholder groups influence six strategic goals, and Table 2 traces each goal to its requirement, its enabling resource and the services that implement it. Read as a chain, the model answers RQ1 by stating which stakeholder concern motivates each goal, which requirement makes that goal testable, and which platform resource and service make it attainable. Six of the seven requirements are realized by application services within Phases A to D. The seventh, R6, concerns analytics on the consolidated transaction data. It is enabled at the strategy layer by the machine-learning resource, but is deliberately left to Phase 4 of the roadmap in Section 4.6, consistent with the treatment of the remaining revenue streams in Section 4.2.

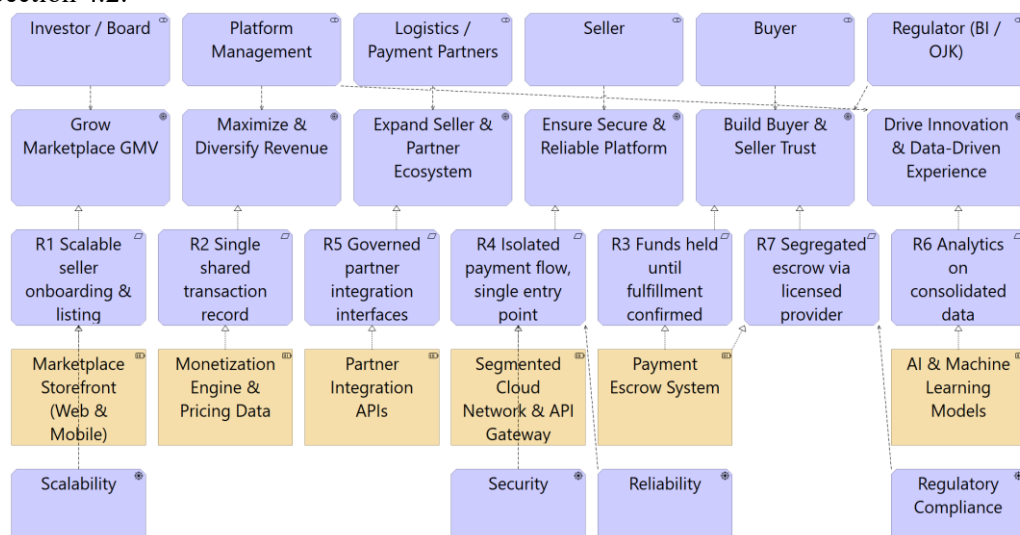


Fig. 2. Architecture Vision: six stakeholder groups, six strategic goals, the seven architecture requirements derived from them, their enabling resources, and the non-functional drivers.

Table 2. Strategic alignment of stakeholders, goals, architecture requirements, and enabling resources.

Stakeholder group	Strategic goal	Architecture requirement	Enabling resource	Realized by
Investor / board	Grow marketplace gross merchandise value (GMV)	R1 Seller self-service onboarding and product publication at scale	Marketplace storefront (web & mobile)	SellerOnBoard; ProductManager
Platform management	Maximize & diversify revenue	R2 Commission computed from a single shared transaction record	Monetization engine & pricing data	CommissionEngine; CheckoutEngine
Buyer	Build buyer & seller trust	R3 Buyer funds held until fulfillment is confirmed	Payment escrow system	PayoutService; OrderService

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Seller	Ensure a secure & reliable platform	R4 Payment flow isolated behind a single controlled entry point	Segmented cloud network & API gateway	PaymentGateway
Logistics / payment partners	Expand seller & partner ecosystem	R5 Partner couriers and payment providers integrated through governed interfaces	Partner integration APIs	LogisticsConnect; PaymentGateway
Platform management	Drive innovation & data-driven experience	R6 Analytics and recommendation on a single consolidated data source	AI & machine-learning models	Deferred to roadmap Phase 4 (Table 7)
Regulator (BI / OJK)	Build buyer & seller trust	R7 Escrow funds segregated and settled through a licensed payment provider	Payment escrow system	PayoutService; DisputeService

Three design decisions deserve emphasis. First, trust is a first-class strategic goal rather than a downstream safeguard. For a marketplace whose sellers are largely unrated SMEs, escrow is the structural substitute for absent interpersonal trust (Hu et al., 2004). Anchoring it at the motivation layer means every later decision inherits it as a requirement. Second, holding funds that belong to other parties is a licensed activity. The regulator is therefore modeled as a stakeholder and regulatory compliance as a driver, which together yield requirement R7: escrow balances are segregated and settled through a licensed payment provider rather than held on the platform's own account. Third, tying each requirement to a concrete resource gives platform management a governance instrument, because any investment proposal can be judged by whether it strengthens a resource that realizes a stated requirement.

Four architecture principles carried over from the Preliminary Phase govern the design, and every later decision is checked against them. They are P1, transactional data has a single source of truth; P2, trust mechanisms are architectural rather than operational; P3, external dependencies are reached only through governed interfaces; and P4, revenue flows are isolated from one another. P1 supports the revenue goal, P2 the trust goal, P3 the ecosystem goal and P4 the reliability goal; together with the six strategic goals, these enumerated statements are what answer the Contextual/Why cell of the completeness matrix. The four non-functional drivers, in turn, are attached in Figure 2 to the requirements they constrain rather than to the goals, so that each driver has a testable target.

Business and Information Systems Architecture (Phases B–C)

The strategic vision is operationalized through three revenue-generating core processes, each modeled end-to-end from the triggering business actor down to the data objects and hosting technology. Figures 3 to 5 present the models and Table 3 collects their vertical traceability, so the discussion concentrates on design decisions rather than restating each chain.

The first core process, in Figure 3, is the Commission Fee process, the marketplace's primary monetization flow. A seller registers, uploads a product and publishes it; a buyer purchases and the transaction completes; the platform deducts its commission; and the net payment is released to the seller. Four application services carry the flow—SellerOnBoard, ProductManager, CheckoutEngine and CommissionEngine—each accessing a distinct data object and hosted on the marketplace core server through its application-hosting service, as Table 3 records.

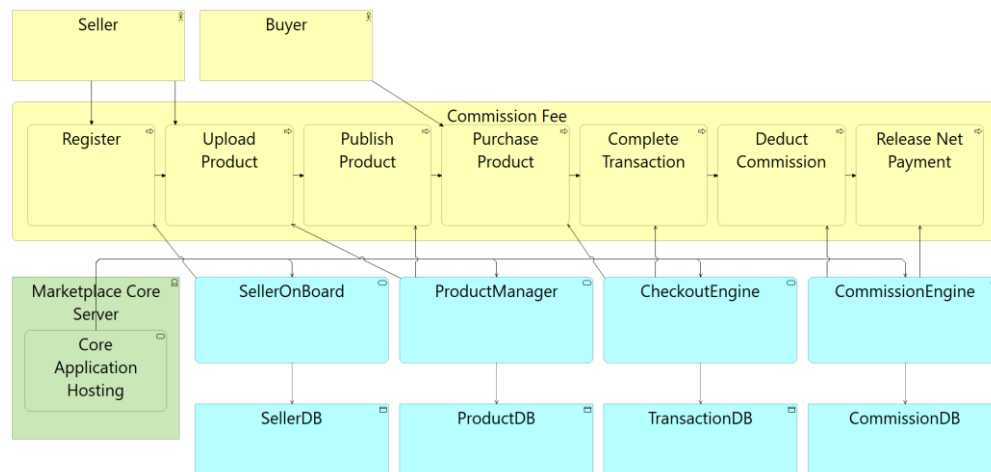


Fig. 3. Commission Fee process architecture: business process, application services, data objects, and hosting node.

The second core process, Payment Processing (Figure 4), embeds the escrow mechanism directly into the transaction flow and so realizes the trust goal set in the Architecture Vision. After checkout and payment-method selection, an external gateway such as Midtrans or Xendit processes and verifies the payment; the order is confirmed and invoiced; the funds are held in escrow; and they are released to the seller only upon completion. CheckoutEngine, PaymentGateway, OrderService, PayoutService and DisputeService support the flow on a dedicated payment server, and the external gateway appears in Figure 4 as a governed technology node that serves PaymentGateway rather than as an unmodeled dependency.

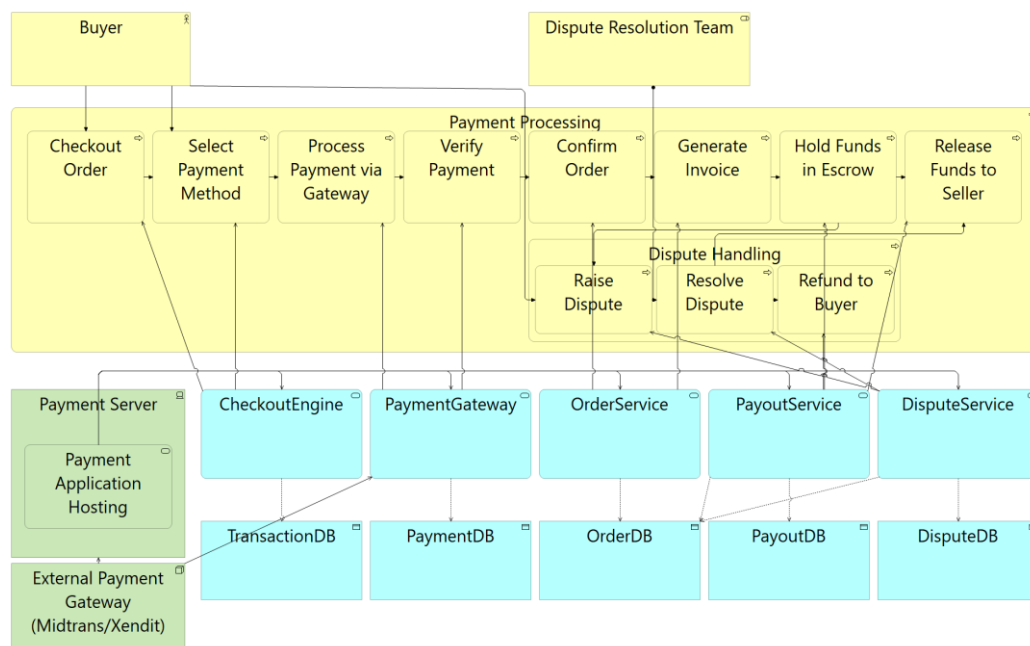


Fig. 4. Payment Processing and escrow architecture: buyer funds held in escrow and released to the seller upon completion, or refunded to the buyer after dispute resolution.

The escrow mechanism deserves to be stated as architecture rather than as a single step, because it is the structural substitute for absent interpersonal trust (Hu et al., 2004). It is modeled as an explicit progression of states carried by two data objects. On successful gateway verification the captured amount is recorded against the transaction in PaymentDB together with its escrow state, while OrderDB carries the fulfillment state that governs release. OrderService advances the

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

order state as fulfillment evidence arrives. PayoutService is the only service authorized to move an amount out of the held state: forward to the seller once delivery is confirmed, or back to the buyer when the order is canceled or a dispute is upheld.

The dispute path is modeled explicitly rather than left implicit. While funds are held, the buyer can raise a dispute. DisputeService records it in DisputeDB and reads the order state from OrderDB, and a dispute-resolution role assigned to the resolution process decides the outcome. Figure 4 groups the three steps as a Dispute Handling sub-process. Only that decision triggers either the refund to the buyer or the release to the seller, so both exits from the held state have a named owner and an audit record instead of appearing as an unattributed branch.

Two consequences follow for the design. First, the release condition is bound to the order state rather than to elapsed time, so the invariant that no payout precedes completion can be enforced as a data constraint rather than left to operational discipline. Second, CheckoutEngine and TransactionDB are shared with the Commission Fee process—as single model elements, not as duplicates—so the amount released to the seller and the amount retained as commission are computed from one transaction record and always reconcile.

The third core process, Fulfillment (Figure 5), monetizes logistics as a platform service. Sellers submit orders to the platform warehouse, the platform processes fulfillment, partner couriers collect the shipments, tracking runs until delivery, and a fulfillment fee is charged to the seller. WarehouseManagement, LogisticsConnect and BillingService support the flow on a logistics server. Taken together, the three processes answer RQ2. Each of commission, payment and fulfillment is realized as an explicit chain from business process through application service and data object to the artifact and node that persist it. The two shared elements—CheckoutEngine and TransactionDB—mark the platform's natural integration boundaries. The remaining revenue streams divide in two: payment-processing fees and flash-sale slots reuse the charge-on-transaction pattern, whereas advertising, seller subscriptions, featured stores, buyer-premium subscriptions, insurance and data services introduce different actors and settlement cycles and are left for future elaboration.

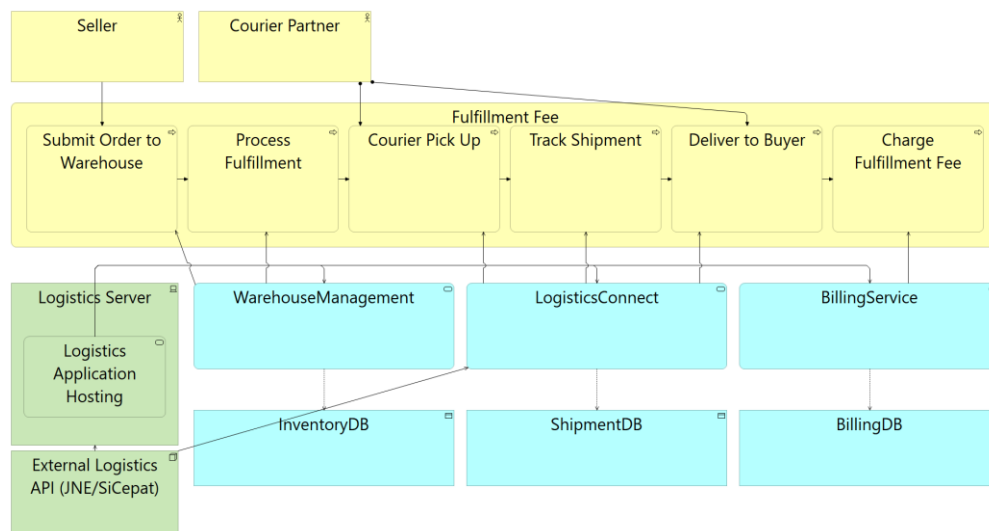


Fig. 5. Fulfillment process architecture: logistics monetized as a platform service through partner-courier integration.

Table 3. Traceability matrix of the three core revenue-generating processes; the node column names the compute node that hosts the application services, while all data objects are persisted on the consolidated database server (Figure 6).

Core process	Business process	Application service	Data object	Hosting node (services)
Commission	Register	SellerOnBoard	SellerDB	Marketplace Core Server

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Core process	Business process	Application service	Data object	Hosting node (services)
Commission	Upload Product; Publish Product	ProductManager	ProductDB	Marketplace Core Server
Commission	Purchase Product; Complete Transaction	CheckoutEngine	TransactionDB	Marketplace Core Server
Commission	Deduct Commission; Release Net Payment	CommissionEngine	CommissionDB	Marketplace Core Server
Payment	Checkout Order; Select Payment Method	CheckoutEngine	TransactionDB	Payment Server
Payment	Process Payment via Gateway; Verify Payment	PaymentGateway	PaymentDB	Payment Server
Payment	Confirm Order; Generate Invoice	OrderService	OrderDB	Payment Server
Payment	Hold Funds in Escrow; Release Funds to Seller	PayoutService	PayoutDB	Payment Server
Payment	Raise Dispute; Resolve Dispute	DisputeService	DisputeDB; OrderDB	Payment Server
Payment	Refund to Buyer	PayoutService	PayoutDB; OrderDB	Payment Server
Fulfillment	Submit Order to Warehouse; Process Fulfillment	WarehouseManagement	InventoryDB	Logistics Server
Fulfillment	Courier Pick Up; Track Shipment; Deliver to Buyer	LogisticsConnect	ShipmentDB	Logistics Server
Fulfillment	Charge Fulfillment Fee	BillingService	BillingDB	Logistics Server

Technology Architecture (Phase D)

Figure 6 consolidates the technology layer, and Table 4 maps each node to the technology service it contributes, the application services it carries, and its security or scalability mechanism. The stack falls into four functional groups, all composed within a single cloud platform. Security and gateway handles inbound traffic through a firewall and a web application firewall (WAF), a content delivery network (CDN) that terminates transport layer security (TLS), and an application programming interface (API) gateway that is the single controlled entry point. Application hosting is separated into three dedicated servers—marketplace core, payment and logistics—so that a fault or load spike in one revenue flow does not propagate to the others. Data storage is consolidated on one database server, which carries the Marketplace Database Schema artifact realizing all eleven data objects and makes the shared transaction record a single source of truth. Partner integration is confined to two governed boundaries, the external payment gateway and the external courier API, neither of which is reachable from any other node.

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

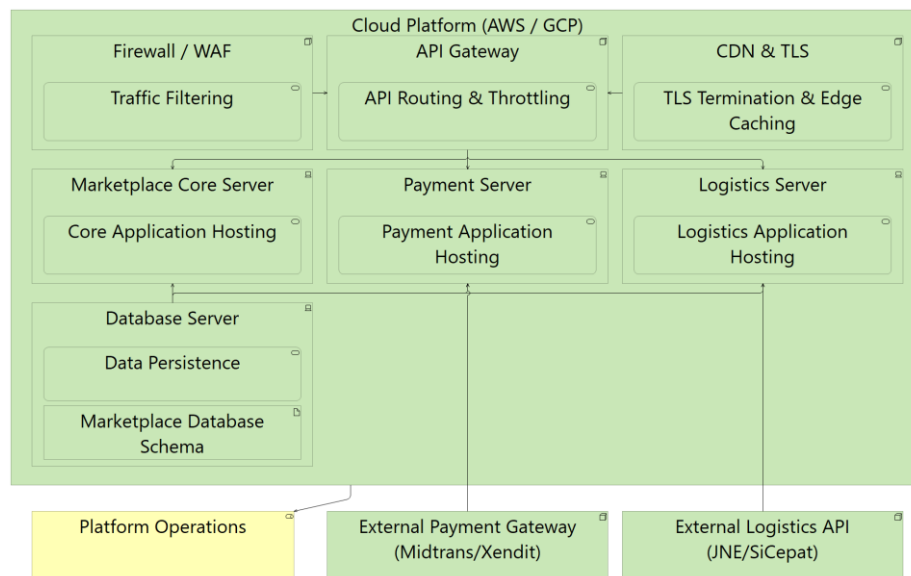


Fig. 6. Target technology architecture: every node is composed within the cloud platform and contributes one technology service, and the database server carries the schema artifact that realizes the eleven data objects.

Each of the seven nodes composed within the cloud platform contributes exactly one technology service, as Table 4 lists. The application services of Section 4.2 therefore consume infrastructure through a named service rather than through a node directly. The infrastructure in turn serves a platform-operations role, the human actor accountable for running it, which Figure 6 shows alongside the two governed external systems. Together these groups complete the answer to RQ2 at the technology layer.

Table 4. Technology nodes, the technology service each contributes, the application services they carry, and their security and scalability mechanisms.

Technology node	Technology service	Application services carried	Security / scalability mechanism
Firewall / WAF	Traffic Filtering	All inbound traffic	Request filtering at the network edge
CDN & TLS	TLS Termination & Edge Caching	All client traffic	Transport encryption; edge caching
API Gateway	API Routing & Throttling	Routing to the three servers	Single controlled entry point; routing and throttling
Marketplace Core Server	Core Application Hosting	SellerOnBoard, ProductManager, CheckoutEngine, CommissionEngine	Horizontal scaling on cloud compute
Payment Server	Payment Application Hosting	CheckoutEngine (shared), PaymentGateway, OrderService, PayoutService, DisputeService	Isolated payment zone; governed gateway integration
Logistics Server	Logistics Application Hosting	WarehouseManagement, LogisticsConnect, BillingService	Governed courier API integration
Database Server	Data Persistence	Marketplace Database Schema artifact realizing all eleven data objects	Consolidated single source of truth
Cloud Platform (AWS / GCP)	—	Composed of the seven nodes above (Figure 6)	Elastic capacity without capital expenditure

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Gap analysis (As-Is versus To-Be)

To answer RQ3, a gap analysis contrasts the As-Is baseline of SME selling with the To-Be architecture layer by layer, as summarized in Table 5. The baseline is a documented characterization of the problem class rather than a survey of one firm. It comprises five practices: catalogs published as social-media posts, orders negotiated in chat, direct transfers without a third-party safeguard, records kept on personal devices, and shipping arranged per order. Section 1 quantifies the channel and delivery practices from national e-commerce statistics, and one further figure is decisive here: cash is the dominant settlement method at 74.79% (BPS-Statistics Indonesia, 2025). Most of that is cash on delivery, which moves the whole of the counterparty risk onto the seller, who dispatches goods before being paid, and which leaves no consolidated transaction record on either side of the exchange.

At each layer, every major deficit of the baseline is closed by an identifiable group of elements in the target design, as Table 5 records; the mapping is reported at layer level rather than element level.

Table 5. Gap analysis of the marketplace enterprise architecture.

Layer	Baseline (As-Is)	Target (To-Be)	Gap / solution
Business	Fragmented channels (social media, chat); direct transfers without protection; ad-hoc shipping	Standardized processes: onboarding, listing, checkout, escrow payment, fulfillment, commission settlement	Formalized workflow with built-in trust (escrow)
Application	No integrated systems; separate chat, spreadsheets, personal banking apps	Integrated services (SellerOnBoard, ProductManager, CheckoutEngine, CommissionEngine, PaymentGateway, OrderService, PayoutService, DisputeService, WarehouseManagement, LogisticsConnect, BillingService)	Unified application landscape with shared services
Data	Records scattered on personal devices; no single source of truth	Centralized domain databases (seller, product, transaction, commission, payment, order, payout, dispute, inventory, shipment, billing)	Consolidated, consistent transactional data
Technology	Personal devices; no dedicated infrastructure or security governance	Cloud platform, API gateway, firewall/WAF, CDN/TLS, dedicated servers, governed external integrations	Scalable, secure infrastructure with managed trust boundaries

The consistency check traced each element across layers, and Table 3 reports the result: the three core processes yield 13 process-to-service mappings drawn from 11 distinct application services and 11 distinct data objects across three hosting servers. CheckoutEngine and TransactionDB are each shared between the commission and payment processes, the deliberate integration boundary described in Section 4.2, and PayoutService occupies two rows because the held state has two exits. No process, service or data object lacks its counterpart in the layer below, so no dangling dependency remains within the modeled scope. The check is not a formality: an earlier iteration of the model left the refund branch out of the held state with no owning application service, and that finding is what introduced DisputeService and DisputeDB into the design.

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Zachman completeness assessment

Gap analysis establishes that the To-Be design improves upon the baseline, but it does not establish that the design is descriptively complete; a design can close every identified gap and still omit a whole dimension that never entered the baseline comparison. To answer RQ4, every element of the architecture was mapped onto the Zachman completeness matrix in Table 6. The matrix crosses the four architectural perspectives with the six 5W1H aspects, and for each of the 24 cells it asks whether the architecture answers the corresponding interrogative at that level of abstraction.

Table 6. Zachman 5W1H completeness matrix of the marketplace architecture.

Perspective	What (Data)	How (Function)	Where (Network)	Who (People)	When (Time)	Why (Motivation)
Contextual (Scope)	Core entities: seller, product, order, payment, shipment (§1; Table 3)	Core flows: list & sell, pay, fulfill (§1; Figs. 3–5)	National marketplace; cloud + partner network (§1; §3; Table 4)	Six stakeholder groups incl. regulator (Fig. 2)	Partial: transaction & settlement cycles (implicit)	Six strategic goals and four architecture principles (§4.1)
Business (Conceptual)	Eleven domain data objects, SellerDB to BillingDB (Table 3)	Business processes, Register to Charge Fulfillment Fee (Table 3)	Channels: platform, warehouse, courier (Figs. 3–5)	Actors buyer, seller & courier partner; dispute-resolution role (Figs. 3–5)	Partial: process sequence via triggering	Six goals and seven requirements (Table 2)
System (Logical)	Eleven data objects with their service access paths (Table 3)	Eleven application services, SellerOnBoard to DisputeService (Table 3)	Service distribution across core / payment / logistics servers (Table 4)	Users of the services: buyer, seller, courier partner, dispute-resolution team; partner systems (Figs. 3–5)	Partial: service orchestration (event triggers)	Requirements R1–R7 traced from the six goals (Table 2)
Technology (Physical)	Database server & Marketplace Database Schema artifact (Fig. 6)	Seven technology services: filtering, TLS, routing, hosting, persistence (Table 4)	Cloud Platform (AWS / GCP), CDN, node composition (Fig. 6)	Platform Operations role; external systems as governed nodes (Fig. 6)	Weak: runtime timing / batch settlement (under-specified)	Drivers: scalability, security, reliability, regulatory compliance (Fig. 2)

The matrix yields a clear result: of the 24 cells assessed, 20 are fully populated, 3 are only partially populated, and 1 is weak, and every cell that falls short of full population lies in the same column. The findings group into three observations.

Fully populated aspects (What, How, Who, Why). The architecture states plainly what data exists, how processes and services behave, who the actors and external systems are, and why each element exists. The Who column reaches beyond the two transacting actors to the dispute-resolution role and the governed external systems. The Why column is answered at every

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

perspective, from the strategic goals and the four enumerated architecture principles down to the non-functional drivers modeled in Figure 2.

Contextual expansion (Where). The Where aspect is answered at every perspective. At the contextual level it is little more than a national market and a partner network; it becomes concrete once the Technology Architecture makes the cloud platform, the network topology and the external integrations explicit. With the four aspects above, these five columns account for the 20 fully populated cells.

The completeness gap (When). The single aspect under-specified throughout is When. Ordering is represented through ArchiMate triggering relationships, but explicit temporal properties are not: escrow hold periods, settlement cycles, service-level timing and batch schedules are all absent, and these absences account for every cell that falls short. At the contextual, business and system perspectives the column is partial, because ordering can be inferred while no element carries a duration; at the technology perspective it is weak, because runtime timing and batch settlement are not represented at all. The gap follows from the scope decision rather than from any absence of temporal constructs in the language. ArchiMate 3.2 provides a plateau and a gap in its Implementation and Migration Layer (The Open Group, 2022a), but those express architectural states and their transitions rather than operational timing. That layer also falls outside the Phase A to D boundary set in Section 3.

The assessment therefore answers RQ4 directly. The design is complete on five of the six descriptive dimensions at every perspective, and the sixth is a specific, named omission rather than a vague weakness.

Phased adoption roadmap (Stage 5)

Stage 5 of the method consolidates the design into an adoption sequence. The order is not a schedule but a dependency order read off the architecture itself: a phase may begin only once the elements it consumes already exist. Table 7 states each phase, its scope, and the prerequisite that fixes its position.

Table 7. Phased adoption roadmap and its dependency prerequisites.

Phase	Scope	Dependency prerequisite
1. Core commerce	Seller onboarding, product listing, checkout, commission settlement (SellerOnBoard, ProductManager, CheckoutEngine, CommissionEngine)	None; the transaction record established here is consumed by every later phase
2. Escrow payment and dispute	Gateway integration, order state, escrow hold and release, dispute resolution (PaymentGateway, OrderService, PayoutService, DisputeService)	Requires the transaction record from Phase 1 and a licensed payment partner (R7)
3. Fulfillment and partner integration	Warehouse handover, courier integration, tracking, fulfillment billing (WarehouseManagement, LogisticsConnect, BillingService)	Requires the confirmed order state from Phase 2 to trigger handover
4. Analytics and recommendation	Consolidated reporting and recommendation over the persisted transaction data (R6); the services that deliver it lie outside the Phase A–D scope of this design	Requires the accumulated data of Phases 1 to 3

The sequence also makes the platform revenue-bearing from Phase 1, because commission settlement is available before the more integration-heavy phases begin.

DISCUSSION

The results extend the enterprise-architecture literature on two fronts, one substantive and one methodological. Substantively, the study moves TOGAF-based design from single-enterprise optimization to multi-sided platform design, which elevates shared application services and

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

governed trust boundaries from incidental details to central design elements. Against the technology-first stream the contribution is complementary rather than competing: microservice research settles how a platform should be implemented, whereas this blueprint settles what it must do and why. The eleven application services identified here map onto candidate microservice boundaries. Table 8 sets out the comparative position in detail.

Table 8. Comparative position on the completeness and monetization dimensions.

Dimension	Widjaja et al. (2025)	Guntakandla (2025)	This study
Multi-sided platform modeled	No	Partial	Yes (seller–buyer–partner–regulator)
Monetization architecture	Partial	No	Full (commission, escrow, fulfillment)
Escrow modeled as architecture	No	No	Yes (motivation, process, service, technology)
Completeness assessment	No	No	Yes (24-cell 5W1H matrix)
Cells fully populated under the same rule	18 / 24	Not assessable (no EA model)	20 / 24
Aspects accounting for the shortfall	When and Why	—	When only
Validation instruments	Gap analysis	Performance testing	Consistency + gap + completeness

Methodologically, laying the Zachman 5W1H ontology over a TOGAF-ArchiMate design proved valuable beyond presentation. The assessment did not uncover an unexpected defect; what it did was make the boundary of the design explicit, localized, and measurable. Gap analysis and consistency checking both reason relative to a baseline or to internal references, so neither can state which descriptive dimension is unaddressed. The completeness lens answers that question in a form another architect can re-derive from the artifacts (Iyamu, 2023; Reiter, 2025). The design also places a trust mechanism the literature leaves unplaced. Hu et al. (2004) established escrow as an institutional substitute for interpersonal trust without specifying where it belongs in an enterprise architecture. Here it appears at four layers. It is a strategic resource realizing a motivation-layer requirement, a mandatory step with an explicit dispute path in the payment process, a dedicated payout service at the application layer, and a governed integration at the technology layer.

The score also has a reference point. Applying the same population rule to the artifacts published by Widjaja et al. (2025), a TOGAF-ArchiMate design for five revenue processes in one real-estate firm published in this journal by two of the present authors, yields 18 of the 24 cells fully populated. The six that fall short are the four When cells and the Why cells at the system and technology perspectives. There, motivation elements appear only at the contextual level, and the link from goals down to applications and nodes has to be traced through intermediate relations. Two things follow. First, the instrument does not simply reward any competent design: a peer-reviewed TOGAF-ArchiMate blueprint scores 18, not 24. Second, and more usefully, the two designs differ less in coverage than in where the coverage stops: there the shortfall spans two descriptive dimensions, while here it is confined to one. Part of the difference is scope: that study aimed at five business processes in one firm, not at motivation-layer completeness.

Several limitations bound these results. First, the validation is architectural and analytical: the blueprint is checked for internal consistency and descriptive completeness. No platform has been implemented, so runtime qualities such as latency, operating cost and user acceptance remain untested. The completeness mapping was performed by one of the authors and inter-rater agreement was not measured, and the comparison with a published design likewise rests on the artifacts that study reports rather than on its underlying model. Second, because the artifact targets a class of problems rather than one organization, its requirements come from national statistics,

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

the reviewed literature and a constructed business model canvas rather than from primary organizational data. Empirical validation with MSME owners, operators and partners, together with expert review, is the priority next step. Third, the design assumes the Indonesian partner and regulatory landscape, so transferring it would require localizing the external integrations and compliance provisions. Fourth, the temporal dimension is represented only as ordering: hold periods, settlement cycles and batch schedules are durations rather than states and are not yet modeled, and the security and compliance architecture, while identified, is not elaborated to Phase-D depth.

6. CONCLUSION

This study designed a completeness-driven enterprise architecture for a multi-sided e-commerce marketplace serving MSME digitalization. It integrates TOGAF ADM as the design process, ArchiMate 3 as the modeling language, and the Zachman 5W1H ontology as an explicit completeness lens, within a Design Science Research approach. The completeness assessment reports that 20 of the 24 assessed cells are fully populated, 3 are partial and 1 is weak, and that every cell falling short lies in the temporal When column. The lens therefore localizes the boundary of the design and turns it into a stated research agenda.

For adopters, three recommendations follow: implement in the dependency order set out in Table 7; treat the identified application services as candidate microservice boundaries; and preserve the traceability discipline, so that every proposed feature either traces to a stakeholder goal or is questioned.

Future work follows the same order of priority. The temporal dimension surfaced by the completeness assessment should be modeled explicitly, beginning with escrow hold periods and settlement cycles. The comparison begun here with one published design should be extended to a larger set, so that a completeness score can be read against a distribution rather than against a single reference point. The architecture should be extended into TOGAF ADM Phases E to G, and the security and compliance architecture elaborated to full Phase-D depth. Finally, the blueprint should be evaluated through independent expert review and a pilot implementation with SME participants, so that its architectural promise can be tested against operational reality.

Conflict of interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgement

The authors thank the Department of Information Technology, Pradita University, for the academic guidance and support provided during this research.

References

- Afarini, N., & Hindarto, D. (2023). The proposed implementation of enterprise architecture in e-government development and services. *International Journal Software Engineering and Computer Science (IJSECS)*, 3(3), 219–229. <https://doi.org/10.35870/ijsecs.v3i3.1756>
- Alghamdi, H. (2024). Assessing the impact of enterprise architecture on digital transformation success: A global perspective. *Sustainability*, 16(20), 8865. <https://doi.org/10.3390/su16208865>
- BPS-Statistics Indonesia. (2025). *Statistik e-commerce 2024*. Badan Pusat Statistik.
- Dumitriu, D., & Popescu, M. A. M. (2020). Enterprise architecture framework design in IT management. *Procedia Manufacturing*, 46, 932–940. <https://doi.org/10.1016/j.promfg.2020.05.011>
- Guntakandla, A. R. (2025). Microservices and modular architecture: Revolutionizing e-commerce scalability. *Journal of Computer Science and Technology Studies*, 7(4), 133–137. <https://doi.org/10.32996/jcsts.2025.7.4.15>
- Hamdani, I., Indrajit, R. E., & Firizqi, J. D. (2026). Orchestrating the circular smart palm ecosystem: A design science research using TOGAF-based enterprise architecture. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 10(2), 937–950. <https://doi.org/10.33395/sinkron.v10i2.15828>

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

- Hevner, A., & Chatterjee, S. (2010). Design science research in information systems. In *Design research in information systems* (pp. 9–22). Springer. https://doi.org/10.1007/978-1-4419-5653-8_2
- Hindarto, D., & Putra, T. D. (2023). Building digital platform for property marketing sales with an enterprise architecture approach. *Journal of Computer Networks, Architecture and High Performance Computing*, 6(1), 67–76. <https://doi.org/10.47709/cnahpc.v6i1.3361>
- Höfflin, J., Goschnick, P., Brecht, P., & Hahn, C. H. (2025). DPAF—Digital platform software architecture framework: Designing software architecture for digital platform business models linked to cyber-physical systems. *Procedia CIRP*, 136, 272–277. <https://doi.org/10.1016/j.procir.2025.08.048>
- Hu, X., Lin, Z., Whinston, A. B., & Zhang, H. (2004). Hope or hype: On the viability of escrow services as trusted third parties in online auction environments. *Information Systems Research*, 15(3), 236–249. <https://doi.org/10.1287/isre.1040.0027>
- Indiani, N. L. P., Keshminder, J. S., Wiratama, N. I., & Amertha, G. S. (2025). Unlocking e-commerce potential in SMEs: An integrative framework for adoption in emerging markets. *Humanities and Social Sciences Communications*, 12, Article 766. <https://doi.org/10.1057/s41599-025-04952-3>
- Iyamu, T. (2023). The Zachman framework for the implementation of enterprise architecture. In *The concept of enterprise architecture from theory to practice* (pp. 188–213). CRC Press. <https://doi.org/10.1201/9781003390879-10>
- Izzati, B. M., Fajrillah, A. A. N., Saputri, R. A. A., Oktavian, I. T., & Widyastri, L. A. (2020). IT blueprint design using TOGAF ADM to support digital transformation at UMKM. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 4(3), 404–417. <https://doi.org/10.29207/resti.v4i3.1763>
- Khairina, D. M., Purwanto, P., & Nugraheni, D. M. K. (2024). Systematic literature review on evaluation models and methods in enterprise architecture research. *Bulletin of Electrical Engineering and Informatics*, 13(4), 2851–2864. <https://doi.org/10.11591/eei.v13i4.6943>
- Kornysheva, E., & Deneckère, R. (2022). A proposal of a situational approach for enterprise architecture frameworks: Application to TOGAF. *Procedia Computer Science*, 207, 3499–3506. <https://doi.org/10.1016/j.procs.2022.09.408>
- Lubis, U. S., Wulandari, N., Wahyudi, I., Widiyanto, A., Wibowo, E. A., Hakim, L., Apriyansyah, B., & Hardiati, N. (2025). Digital marketing strategies in expanding the market for MSME creative products in Indonesia—State of art. *Frontiers in Communication*, 10, Article 1647391. <https://doi.org/10.3389/fcomm.2025.1647391>
- Lukáš, M., Ulman, M., Pavelka, M., Ambruz, P., & Vaněk, J. (2025). Support of project & program management of digital transformation initiatives by holistic enterprise architecture in sectors of national economy. *Procedia Computer Science*, 256, 1542–1549. <https://doi.org/10.1016/j.procs.2025.02.289>
- Ministry of Trade of the Republic of Indonesia. (2025). *Kinerja perdagangan melalui sistem elektronik (PMSE) Indonesia tahun 2025*. Badan Kebijakan Perdagangan, Kementerian Perdagangan.
- Oberle, M., Yesilyurt, O., Schlereth, A., Risling, M., & Schel, D. (2023). Enterprise IT architecture greenfield design combining IEC 62264 and TOGAF by example of battery manufacturing. *Procedia Computer Science*, 217, 136–146. <https://doi.org/10.1016/j.procs.2022.12.209>
- Pratama, Y. (2024). Enterprise architecture design in Sonic Advertising Company using TOGAF framework. *IJIS: International Journal of Informatics and Information Systems*, 7(1), 44–56. <https://doi.org/10.47738/ijis.v7i1.194>
- Purnama, W., Indrajit, R. E., & Firizqi, J. D. (2026). An enterprise architecture blueprint for HL7-based RIS-PACS integration using TOGAF ADM: A mobile diagnostic access framework. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 10(2), 905–916. <https://doi.org/10.33395/sinkron.v10i2.15825>

- Reiter, M. (2025). Comparative analysis of enterprise architecture frameworks: TOGAF, Zachman and FEAF. In *Eurasian studies in business and economics* (pp. 141–162). Springer. https://doi.org/10.1007/978-3-032-03321-5_8
- Sentosa, S., Indrajit, R. E., & Dazki, E. (2024). Enterprise architecture of the basic banking feature for a new challenger of digital banking in Indonesia. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 8(4), 2197–2211. <https://doi.org/10.33395/sinkron.v8i4.14116>
- Sowa, J. F., & Zachman, J. A. (1992). Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, 31(3), 590–616. <https://doi.org/10.1147/sj.313.0590>
- The Open Group. (2022a). *ArchiMate 3.2 specification* (Document No. C226).
- The Open Group. (2022b). *The TOGAF standard* (10th ed.). Van Haren Publishing.
- Widjaja, H., Indrajit, R. E., & Dazki, E. (2025). Transforming real estate: Leveraging TOGAF ADM for digital optimization in enterprise architecture. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 9(1), 220–231. <https://doi.org/10.33395/sinkron.v9i1.14219>
- Zachman, J. A. (1987). A framework for information systems architecture. *IBM Systems Journal*, 26(3), 276–292. <https://doi.org/10.1147/sj.263.0276>
- Zhi, Q., & Zhou, Z. (2022). Empirically modeling enterprise architecture using ArchiMate. *Computer Systems Science and Engineering*, 40(1), 357–374. <https://doi.org/10.32604/csse.2022.018759>