International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Decomposing Vocational Higher Education Monoliths via Strategic Domain-Driven Design: Context Mapping and Microservices Architectural Framing at State Polytechnic of Madiun

Susilo Veri Yulianto[1], Gus Nanang Syaifuddiin[2], Rahmania Kumalasari[3], Indikasari Tita Rahmadani[4]

[1]Software Engineering Technology, Polytechnic State of Madiun, Madiun, Indonesia, 60133
[2]Software Engineering Technology, Polytechnic State of Madiun, Madiun, Indonesia, 60133
[3]Information Technology, Polytechnic State of Madiun, Madiun, Indonesia, 60133
[4]Information Technology, Polytechnic State of Madiun, Madiun, Indonesia, 60133

Abstract**—***Monolithic academic information systems in vocational higher education institutions frequently struggle with high architectural coupling, rigid scalability, and complex maintenance cycles. As vocational curricula emphasize practical, competency-based, and package-based course delivery, domain boundary identification requires specific tailoring compared to traditional university systems. This study presents a systematic application of Strategic Domain-Driven Design (DDD) to decompose the legacy monolithic architecture of State Polytechnic of Madiun into a resilient microservices ecosystem. Through iterative domain storytelling, event storming, and context mapping, we extract and boundary 18 fine-grained domain services specifically structured for vocational operations, spanning core academic tracking, package-based study planning, internship management, final projects, and single tuition fee validation. We formulate an upstream-downstream relationship matrix, context maps, and integration patterns (Shared Kernel, Customer-Supplier, Anti-Corruption Layer) to establish a sustainable migration roadmap. The resulting architectural model provides a scalable blueprint for modernizing legacy academic software in vocational institutions while preserving domain cohesion and operational autonomy.*

Keywords**—** *Domain-Driven Design, Microservices, Monolith Decomposition, Context Mapping.*

I. INTRODUCTION

Higher education institutions heavily rely on Academic Information Systems (AIS) to manage complex administrative, academic, financial, and pedagogical processes. Historically, many polytechnics and universities deployed enterprise software using monolithic architectural patterns, where all business functionalities—ranging from student enrollment to grading and graduation—share a single codebase, unified data schema, and monolithic deployment pipeline. At State Polytechnic of Madiun, the legacy monolithic system served administrative needs reliably during early institutional growth. However, as student enrollments expanded and vocational educational workflows diversified, the monolithic architecture encountered severe scalability bottlenecks, high risk during deployments, inter-module code ripple effects, and database connection exhaustion during peak academic events such as course registration and grade submission.

Modern software engineering advocates microservices architecture as an architectural style to achieve independent deployability, decentralized data management, and operational resilience [1]. Nevertheless, transitioning from a monolithic legacy system to microservices poses substantial risks if boundaries are defined arbitrarily along technical layers or organizational silos rather than business domain boundaries [2]. Arbitrary service decomposition leads to distributed monoliths characterized by high latency, cascading failures, and complex distributed transactions.

Domain-Driven Design (DDD), originally articulated by Eric Evans, provides a disciplined methodology to bridge complex business domains and software architecture. Specifically, Strategic DDD focuses on understanding the problem space, identifying Ubiquitous Language, establishing Bounded Contexts, and mapping organizational and technical relationships through Context Mapping [3]. While DDD has been widely adopted in fintech, e-commerce, and logistics domains, its specialized application within vocational higher education remains under-explored.

Vocational higher education institutions differ fundamentally from academic research universities. Vocational curricula exhibit rigid structural constraints, such as package-based study plans, mandatory industry internships, competency-based assessments, and tight synchronization between tuition payment validation and academic eligibility. Thus, applying generic higher education software templates yields sub-optimal architectural alignment.

This paper presents the strategic application of Strategic Domain-Driven Design at State Polytechnic of Madiun to decompose its legacy monolithic AIS into an event-driven microservices architecture. The core contributions of this paper are: (1) A comprehensive strategic domain analysis tailoring DDD techniques to vocational higher education workflows; (2) The formal derivation and boundary definition of 18 vocational-specific bounded contexts; and (3) A context

Susilo Veri Yulianto, Gus Nanang Syaifuddiin, Rahmania Kumalasari, and Indikasari Tita Rahmadani, "Decomposing Vocational Higher Education Monoliths via Strategic Domain-Driven Design: Context Mapping and Microservices Architectural Framing at State Polytechnic of Madiun," *International Research Journal of Advanced Engineering and Science*, Volume 11, Issue 4, pp. 87-90, 2026.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

map detailing integration patterns (Customer-Supplier, Shared Kernel, Anti-Corruption Layer) and upstream-downstream relationships required to guide monolithic migration.

II. Literature Review

Monolith decomposition into microservices has emerged as a central research theme in cloud-native software evolution. Newman [1] emphasizes that the success of a microservices transformation relies heavily on identifying bounded contexts that minimize cross-service communication while maximizing domain autonomy. Fritzsch et al. [2] conducted an empirical survey on microservices refactoring approaches, highlighting that domain-centric strategies consistently outperform static/dynamic code analysis approaches in preserving business semantics.

Strategic Domain-Driven Design has gained significant traction as the primary methodology for domain-centric service decomposition. Bucchiarone et al. [4] demonstrated that leveraging DDD bounded contexts prevents microservice granularity mismatch, ensuring services align with organizational boundaries and domain experts' mental models. Similarly, Rademacher et al. [5] proposed a model-driven framework integrating DDD context maps directly with microservices API modeling, illustrating that explicit relationship definitions reduce integration friction during legacy migration.

In the higher education sector, digital transformation initiatives have increasingly targeted legacy software modernization. Recent studies by Al-Malaise et al. [6] and Lorenzo et al. [7] examine enterprise architecture frameworks for university administration; however, these works primarily address high-level business capability mapping rather than technical bounded context isolation and database decoupling. Furthermore, vocational education domain specifics—such as block-package course scheduling, mandatory industrial placement verification, and integrated competency graduation—require explicit context mapping strategies that standard university systems do not account for [8]. Our work fills this gap by delivering an empirical case study of strategic DDD applied to a public vocational polytechnic in Indonesia.

III. Methodology

Our research methodology follows an iterative, domain-centric qualitative and analytical workflow structured into three main phases:

A. *Phase* 1: Knowledge Discovery & Event Storming

We conducted structured Event Storming workshops involving domain experts across administrative departments at State Polytechnic of Madiun, including academic staff, department heads finance sub-departments, academic affairs, and IT system administrators. Participants explored business events across the student lifecycle using color-coded sticky notes to identify Domain Events, Triggers, Aggregates, and Read Models.

B. *Phase 2: Bounded* Context Isolation

Domain events were grouped around business boundaries where dynamic linguistic shifts (Ubiquitous Language variations) occurred. For instance, the term "Student" carries distinct attributes and business rules within the tuition payment context versus the final project evaluation context. This linguistic isolation yielded discrete candidate Bounded Contexts.

C. *Phase 3: Strategic* Context Mapping

To prevent integration chaos, relationships between identified bounded contexts were categorized according to standard DDD strategic patterns: Customer-Supplier (CS), Shared Kernel (SK), Separate Ways (SW), and Anti-Corruption Layer (ACL). Upstream (U) context changes were mapped against Downstream (D) impacts.

IV. Result

Through the strategic DDD analysis of business processes at State Polytechnic of Madiun, we identified 18 specific candidate bounded contexts essential for vocational operations. Each context encapsulates specific business rules, data schemas, and domain models, described in TABLE I:

TABLE 1. Bounded Contexts Identification

No	*Bounded Contexts*	*Descriptions*
1	Study Program Context	Manages structural data regarding academic departments and specific study programs, establishing accreditation and degree offerings.
2	Organizational Unit Context	Handles hierarchical administrative structures, including academic sub-departments, general administration, finance units, and study program management.
3	Academic Term Context	Manages academic calendars, semester operational timelines (odd/even terms), and active system registration windows.
4	Personnel Context	Manages comprehensive human resource records for both teaching faculty (dosen) and administrative staff.
5	Student Context	Encapsulates student demographic lifecycle data, status transitions (active, leave, drop-out), and biographical records.
6	Course Catalog Context	Regulates master course definitions, learning outcomes, credit units, and departmental course ownership.
7	Curriculum Structure Context	Manages versioned curriculum frameworks, prerequisite mappings, and vocational competency structures across study programs.
8	Class Section Context	Controls class section allocation, capacity constraints, and cohort distribution per academic term.
9	Course Scheduling Context	Handles room assignments, instructor timetables, time-slot conflict checking, and facility allocation.
10	Lectures Context	Oversees operational lectures, teaching log delivery, subject delivery tracking, and course material distribution.
11	Attendance Context	Tracks student and faculty attendance records for scheduled lectures, enforcing minimum attendance eligibility rules.
12	Internship/Industrial Placement Context	Manages vocational industrial placements, mentor assignments, internship evaluation reports, and company partnership records.
13	Grading Context	Accumulates course assessment components, quiz/exam scores, letter grade conversions, and raw grade submissions.

Susilo Veri Yulianto, Gus Nanang Syaifuddiin, Rahmania Kumalasari, and Indikasari Tita Rahmadani, "Decomposing Vocational Higher Education Monoliths via Strategic Domain-Driven Design: Context Mapping and Microservices Architectural Framing at State Polytechnic of Madiun," *International Research Journal of Advanced Engineering and Science*, Volume 11, Issue 4, pp. 87-90, 2026.

No	Bounded Contexts	Descriptions
14	Re-registration & Tuition Context	Validates Single Tuition Fee payments, financial clearance, and student re-registration eligibility per semester. Validates Single Tuition Fee payments, financial clearance, and student re-registration eligibility per semester.
15	Study Planning Context	Executes semester course enrollment using vocational package-based systems based on prescribed cohort curricula.
16	Academic Performance Context	Computes semester grade point averages, generates transcripts, and tracks academic standing.
17	Graduation Determination Context	Conducts formal graduation eligibility audits, credit fulfillment checks, and official academic degree conferring.
18	Final Project Context	Oversees final project proposal submission, advisor assignment, defense examination scheduling, and publishing.

To formalize the domain classification, each context was categorized into Core, Supporting, or Generic domains based on strategic value to State Polytechnic of Madiun, as summarized in Table II.

TABLE II. Domain Categorization and Categorization Criteria

Domain Category	Bounded Contexts	Strategic Value & Rationale
Core Domain	Study Planning, Internship/Industrial Placement, Final Project, Graduation Determination, Curriculum Structure	Provides competitive advantage; contains unique vocational business rules (packaged study plans, industrial internship tracking, graduation audits).
Supporting Domain	Lectures, Attendance, Grading, Academic Performance, Class Section, Course Scheduling	Essential for daily operational execution of vocational teaching; custom tailored but highly operational.
Generic Domain	Study Program, Organizational Unit, Academic Term, Personnel, Student, Course Catalog, Re-registration & Tuition	Standard administrative and master data capabilities; off-the-shelf patterns or external systems (e.g., payment gateway) can be integrated.

Mapping the relationships between bounded contexts is critical to prevent tight runtime coupling in microservices. In strategic DDD, an Upstream (U) context provides services or data, while a Downstream (D) context consumes them.

A. Downstream/Upstream Analysis

The strategic dependencies follow strict business invariants. Re-registration & Tuition acts as an upstream gatekeeper to Study Planning; a student cannot execute course enrollment until financial clearance is published. Study Planning subsequently provides valid enrollment rosters to Lectures and Attendance. The Graduation Determination context acts as the terminal downstream aggregator, consuming completion events from Hasil Studi, Internship/Industrial Placement, and Final Project.

B. Integration Pattern Specifications

We established four primary integration patterns across the 18 microservices:

1) Customer-Supplier (CS): Applied between Re-registration & Tuition (Supplier) and Study Planning (Customer). The Study Planning service relies on financial status notifications. As a Customer, its operational requirements directly influence the API contracts exposed by Re-registration & Tuition.

2) Anti-Corruption Layer (ACL): Implemented in the Graduation Determination core domain when interacting with legacy finance or external government databases (e.g., PDDikti). The ACL translates legacy data schemas into clean domain models, preventing external model pollution.

3) Shared Kernel (SK): Utilized between Curriculum Structure and Course Catalog context boundaries, sharing a tightly controlled domain library for credit structure definitions and prerequisite evaluation algorithms.

4) Published Language / Open Host Service (OHS/PL): The Student and Personnel master data services expose standard REST/gRPC endpoints with standardized JSON schemas consumed across all operational bounded contexts.

TABLE III. Strategic Context Relationship & Pattern Matrix

Upstream Context	Downstream Context	Pattern	Data Flow & Integration Mechanism
Re-registration & Tuition	Study Planning	Customer-Supplier	Asynchronous event (Payment Verified) via Kafka event broker.
Curriculum Structure	Study Planning / Class Section	Shared Kernel	Shared domain SDK for prerequisite and package validation logic.
Study Planning	Lectures / Attendance	OHS / PL	gRPC synchronous query for roster validation; async events for registration.
Lectures	Grading / Attendance	Customer-Supplier	REST API for lecture session verification before grade posting.
Academic Performance, Internship/ Industrial Placement, Final Project	Graduation Determination	Anti-Corruption Layer	ACL adapter mapping multi-service completion events to degree audit rules.

V. CONCLUSION

This paper presented the strategic decomposition of the legacy monolithic Academic Information System at State Polytechnic of Madiun into a microservices architecture using Domain-Driven Design. By applying domain storytelling, event storming, and context mapping, we successfully bounded 18 domain services specialized for vocational higher education. The derived context map and upstream-downstream relationship matrix establish clear integration guidelines through Customer-Supplier, Shared Kernel, and Anti-Corruption Layer patterns. This structured domain model eliminates monolithic coupling, enhances service autonomy, and guarantees high availability during peak academic periods.

Susilo Veri Yulianto, Gus Nanang Syaifuddiin, Rahmania Kumalasari, and Indikasari Tita Rahmadani, "Decomposing Vocational Higher Education Monoliths via Strategic Domain-Driven Design: Context Mapping and Microservices Architectural Framing at State Polytechnic of Madiun," *International Research Journal of Advanced Engineering and Science*, Volume 11, Issue 4, pp. 87-90, 2026.

REFERENCES

[1] S. Newman, Building Microservices: Designing Fine-Grained Systems, 2nd ed. Sebastopol, CA: O'Reilly Media, 2021.

[2] J. Fritzsch, J. Bogner, M. Zimmermann, and S. Wagner, "From monolith to microservices: A classification scheme for refactoring approaches," IEEE Transactions on Software Engineering, vol. 49, no. 4, pp. 2560–2580, Apr. 2023.

[3] E. Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software. Boston, MA: Addison-Wesley, 2004.

[4] A. Bucchiarone, N. Dragoni, S. Dustdar, P. Lago, and M. Mazzara, "Microservices: Yesterday, Today, and Tomorrow," IEEE Software, vol. 38, no. 1, pp. 40–50, Jan.-Feb. 2021.

[5] T. Rademacher, S. Sachweh, and A. Zündorf, "A modeling environment for domain-driven microservice architecture design," Journal of Systems and Software, vol. 183, p. 111104, Jan. 2022.

[6] T. Al-Malaise, A. Al-Ghamdi, and M. Bader, "Enterprise architecture framework for modernizing university academic information systems," IEEE Access, vol. 10, pp. 45120–45135, May 2022.

[7] R. Lorenzo, M. Sanchez, and A. Santos, "Decomposing legacy higher education systems into microservices," Computers & Electrical Engineering, vol. 108, p. 108680, May 2023.

[8] H. Kurniawan and A. P. Widodo, "Academic information system refactoring using domain-driven design in vocational education," Journal of Applied Computer Science and Technology, vol. 4, no. 2, pp. 112–121, Dec. 2023.