FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Multi-Organisation Ticket Management in B2B Service Ecosystems

Salif Henrique Faustino



Mestrado em Engenharia de Software

Supervisor: Prof. Jorge Melegati

July, 2026

© Salif Henrique Faustino, 2026

Multi-Organisation Ticket Management in B2B Service Ecosystems

Salif Henrique Faustino

Mestrado em Engenharia de Software

July, 2026

Resumo

Os ecossistemas de serviços Business-to-Business (B2B) modernos dependem de múltiplas organizações interdependentes para entregar valor aos clientes finais. Quando ocorrem incidentes, a sua resolução exige coordenação que ultrapassa as fronteiras organizacionais. Contudo, ferramentas comerciais de Gestão de Serviços de TI (do inglês, *IT Service Management* — ITSM) amplamente usadas, como ServiceNow, Jira Service Management e Zendesk, continuam a organizar a colaboração entre entidades sobretudo através de registos locais, ligações, partilhas ou sincronização entre sistemas. Na prática, quando é necessário envolver parceiros externos, os tickets tendem a ser replicados ou referenciados entre sistemas, a coordenação faz-se por canais informais ou integrações parciais, e aceita-se a consequente perda de contexto, duplicação de esforço e fragmentação de visibilidade.

Esta fragmentação também afecta os técnicos que executam o trabalho operacional: a necessidade de reintroduzir informação, acompanhar estados em sistemas distintos e coordenar respostas entre parceiros aumenta a carga repetitiva e reduz a visibilidade do contributo individual. Trata-se, assim, de um segundo problema — de natureza humana — distinto do problema arquitectural da fragmentação. É para este problema de envolvimento dos técnicos, e não para o arquitectural, que a gamificação se apresenta como uma possível solução, apoiando-os sem comprometer a qualidade do serviço.

A presente dissertação aborda duas questões de investigação: *(RQ1) Como conceber soluções centralizadas de gestão de tickets que suportem o encaminhamento formal de incidentes entre organizações em ecossistemas B2B?* e *(RQ2) De que forma podem os elementos de gamificação ser concebidos e integrados no ciclo de vida do ticket inter-organizacional?* A investigação segue uma abordagem de Design Science Research, organizada em três contribuições.

Em primeiro lugar, a dissertação contribui com uma Revisão Sistemática da Literatura (do inglês, *Systematic Literature Review* — SLR) que sintetiza 16 estudos primários em três domínios: arquitecturas de gestão de falhas inter-organizacionais, desafios dos processos ITSM em ambientes subcontratados e multi-fornecedor, e arquitecturas de plataformas multi-inquilino (*multi-tenant*) para gestão de serviços. A revisão fornece a base teórica para o modelo, identifica sete requisitos e sete lacunas de investigação, e confirma que a gestão de tickets inter-organizacional permanece uma área severamente subexplorada: o trabalho existente trata a integração de tickets como um problema de sincronização de dados entre sistemas independentes, sem propor uma entidade canónica de ticket com semântica formal de encaminhamento.

Em segundo lugar, a contribuição central da dissertação é um modelo arquitectural e de domínio para tratar o ticket inter-organizacional como entidade de primeira classe: um registo único e autoritativo, partilhado pelas organizações envolvidas, no qual a responsabilidade é transferida através de eventos formais em vez de cópia entre sistemas. O modelo define ainda visibilidade configurável por organização, métricas extremo-a-extremo e por organização a partir da mesma fonte de dados, e integração de mecanismos de gamificação sem acoplamento à lógica de negócio central. O conhecimento de design subjacente é articulado como padrão arquitectural transferível

— o *Canonical Inter-Organisational Aggregate* — aplicável a outros domínios de coordenação entre organizações.

Em terceiro lugar, esse modelo é concretizado no artefacto Flowzao, uma prova de conceito funcional. A plataforma é tratada como uma instanciação da arquitectura proposta, permitindo avaliar se o ticket canónico, os eventos formais de encaminhamento, a visibilidade configurável e a integração de gamificação são tecnicamente viáveis em fluxos reais coordenados. A validação do problema é conduzida separadamente através de um questionário público multilingue, que examina os sintomas de fragmentação junto de respondentes que nunca utilizaram a plataforma, fornecendo um sinal industrial independente. A avaliação do artefacto combina uma avaliação estruturada face aos seis requisitos no âmbito da implementação (R1–R4, R6, R7) e um piloto em ecossistema, no qual organizações colaboradoras utilizam a plataforma sobre os seus próprios fluxos de trabalho coordenados. Os resultados mostram que os seis requisitos no âmbito da implementação (R1–R4, R6, R7) são satisfeitos nesse âmbito e que o ticket canónico e o encaminhamento formal operam em fluxos reais — cerca de um terço dos tickets do piloto atravessou fronteiras organizacionais. O feedback ao nível dos requisitos foi positivo e o questionário corroborou de forma independente os sintomas de fragmentação, ainda que a usabilidade (SUS) tenha revelado um sinal misto; dada a escala reduzida do piloto, estes resultados constituem evidência formativa e direcional, não uma generalização estatística.

Palavras-chave: Gestão de Serviços de TI, ITSM, ecossistemas B2B, gestão de tickets inter-organizacional, encaminhamento de tickets, arquitectura multi-tenant, padrões arquitecturais, gamificação, Design Science Research

Abstract

Modern Business-to-Business (B2B) service ecosystems depend on multiple interdependent organisations to deliver value to end customers. When incidents occur, their resolution requires coordination that crosses organisational boundaries. Yet widely used commercial IT Service Management (ITSM) tools, such as ServiceNow, Jira Service Management, and Zendesk, still organise cross-entity collaboration mainly through local records, links, sharing agreements, or synchronisation between systems. In practice, when external partners must be involved, tickets tend to be replicated or referenced across systems, coordination relies on informal channels or partial integrations, and the resulting loss of context, duplicated effort, and fragmented visibility are treated as unavoidable.

This fragmentation also affects the technicians who perform the operational work: re-entering information, tracking status across separate systems, and coordinating responses with partners increase repetitive effort and reduce visibility of individual contribution. This is, therefore, a second and distinct problem — a human one — separate from the architectural problem of fragmentation. It is this engagement problem, not the architectural one, that gamification is brought in to address, supporting technicians without compromising service quality.

This dissertation addresses two research questions: *(RQ1) How can centralised ticket-management solutions be architected to support formal incident routing across organisations in B2B service ecosystems?* and *(RQ2) How can gamification elements be designed and integrated into the inter-organisational ticket lifecycle?* The research follows a Design Science Research approach, organised around three contributions.

First, the dissertation contributes a Systematic Literature Review (SLR) that synthesises 16 primary studies across three domains: inter-organisational fault and incident management architectures, ITSM process challenges in outsourced and multi-vendor environments, and multi-tenant platform architectures for service management. The review provides the theoretical basis for the model, identifies seven requirements and seven research gaps, and confirms that inter-organisational ticket management remains a severely underexplored area: existing work treats ticket integration as a data synchronisation problem between independent systems, without introducing a canonical ticket entity with formal routing semantics.

Second, the dissertation's central contribution is an architectural and domain model for treating the inter-organisational ticket as a first-class entity: a single authoritative record shared across participating organisations, in which responsibility is transferred through formal events rather than by copying tickets between systems. The model also defines organisation-scoped visibility, end-to-end and per-organisation metrics from the same data source, and gamification mechanisms integrated without coupling to the core ticket logic. The underlying design knowledge is articulated as a transferable architectural pattern — the *Canonical Inter-Organisational Aggregate* — applicable to other cross-organisational coordination domains.

Third, this model is materialised in Flowzao, a functional proof of concept. The platform is treated as an instantiation of the proposed architecture, allowing the canonical ticket, formal rout-

ing events, configurable visibility, and gamification integration to be evaluated in real coordinated workflows. The problem is validated separately through a public multilingual questionnaire, which probes the fragmentation symptoms with respondents who have not used the platform, providing an industry-side signal independent of the artefact. The evaluation of the artefact combines a structured assessment against the six requirements within scope (R1–R4, R6, R7) and an in-ecosystem pilot in which collaborating organisations exercise the platform under their own coordinated workflows. The results show that the six in-scope requirements (R1–R4, R6, R7) are satisfied within that scope and that the canonical ticket and formal routing operate in real workflows — about a third of the pilot's tickets crossed organisational boundaries. Requirement-level feedback was positive and the questionnaire independently corroborated the fragmentation symptoms, although usability (SUS) returned a mixed signal; given the limited scale of the pilot, these results constitute formative and directional rather than statistically generalisable evidence.

Keywords: IT Service Management, ITSM, B2B ecosystems, inter-organisational ticket management, ticket routing, multi-tenant architecture, architectural patterns, gamification, Design Science Research

UN Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs) comprise 17 goals that provide a shared framework to address the world's most pressing social, economic, and environmental challenges. This dissertation contributes to three of these goals through the design and evaluation of a centralised, multi-tenant architecture for inter-organisational ticket management in B2B service ecosystems.

The core contribution, modelling fragmented per-organisation ticket workflows as a shared canonical routing problem, has direct implications for operational productivity (SDG 8), digital infrastructure resilience (SDG 9), and multi-stakeholder collaboration (SDG 17). The integration of gamification mechanisms further supports equitable recognition of technician contributions within these cross-organisational workflows.

The specific goals and targets addressed are:

SDG 8: Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all.

SDG 9: Build resilient infrastructure, promote inclusive and sustainable industrialisation and foster innovation.

SDG 17: Strengthen the means of implementation and revitalise the Global Partnership for Sustainable Development.

SDG	Target	Contribution	Indicators
8	8.2	Reduces duplicated ticket handling across organisations and manual coordination overhead, enabling technicians to focus on diagnostic and resolution work rather than administrative re-entry.	Ratio of cross-organisational tickets resolved without manual duplication; mean resolution time with and without platform adoption.
	8.3	Gamification mechanics (achievements, points, leaderboards) provide transparent, data-driven recognition of individual contributions across organisational boundaries. The design follows opt-in principles to avoid coercive dynamics.	Achievement unlock rate per technician; point distribution equity measured via Gini coefficient; voluntary engagement rate with the reward and recognition system.
9	9.1	The event-driven, multi-tenant architecture replaces fragile point-to-point integrations between ITSM tools with a shared infrastructure that supports formal ticket routing and preserves end-to-end traceability.	Number of organisations and ecosystems supported concurrently; platform availability under concurrent load; mean time to onboard a new participant organisation into an ecosystem.
	9.c	The platform provides a single shared entry point for structured service delivery, enabling smaller organisations within an ecosystem to participate in formal ticket workflows without deploying their own ITSM infrastructure.	Distribution of participating organisation sizes measured by company size tier; percentage of ecosystem member organisations actively routing tickets through the shared platform.
17	17.16	Operationalises multi-stakeholder partnerships by providing formal ticket routing semantics and configurable visibility controls that span organisational boundaries.	Number of active inter-organisational ticket routing flows; percentage of tickets with a complete end-to-end audit trail from creation to closure.

Acknowledgements

To my mother, the reason behind every effort, every late night, and every decision to keep going. You are the foundation of everything I am and everything I aspire to become. This work is, above all, for you.

I owe my deepest gratitude to Professor Jorge Melegati, my supervisor, whose sharp questions and patient guidance shaped this work from a vague intuition into a structured research contribution. His ability to challenge assumptions while preserving creative freedom made all the difference.

To Bernardo, my partner in Flowzao, thank you for sharing the vision that this platform can reach every company and grow into something far larger than a dissertation. Your commitment to turning a research prototype into a real product gave this work a purpose that extends well beyond academia.

I am grateful to the Faculty of Engineering of the University of Porto and the Master in Software Engineering programme for providing a rigorous academic environment that consistently pushed me to think beyond implementation and towards the *why* behind every design decision.

To my colleagues and friends in the MESW programme: the shared work sessions, the honest feedback, and the small victories along the way made this process as rewarding as the result.

Salif Henrique Faustino

"The single biggest problem in communication is the illusion that it has taken place."

George Bernard Shaw

Contents

1 Introduction **1**
- 1.1 Context and Motivation . . . 1
- 1.2 Problem Statement . . . 3
- 1.3 Research Questions . . . 4
- 1.4 Research Methodology . . . 5
- 1.5 Contributions . . . 6
- 1.6 Dissertation Structure . . . 7

2 State of the Art **9**
- 2.1 Conceptual Foundations . . . 9
  - 2.1.1 B2B Service Ecosystems and Distributed Service Delivery . . . 9
  - 2.1.2 IT Service Management Frameworks . . . 10
- 2.2 Systematic Literature Review Methodology . . . 10
  - 2.2.1 Objective and Scope . . . 10
  - 2.2.2 Search Strategy . . . 11
  - 2.2.3 Inclusion and Exclusion Criteria . . . 11
  - 2.2.4 Selection Process and Data Extraction . . . 12
- 2.3 Systematic Review Results . . . 15
  - 2.3.1 Inter-Organisational Fault Management Architectures . . . 15
  - 2.3.2 Integration Models: Replication versus Routing . . . 16
  - 2.3.3 The Inter-Organisational Ticket and Single Source of Truth . . . 16
  - 2.3.4 ITSM Process Challenges in Multi-Vendor Environments . . . 17
  - 2.3.5 Multi-Tenant Architectures for Service Management Platforms . . . 18
  - 2.3.6 Composite SLA Management . . . 18
  - 2.3.7 Inter-Organisational Workflow Coordination . . . 18
  - 2.3.8 Commercial ITSM Tools and Multi-Organisation Support . . . 19
- 2.4 Gamification in Organisational and ITSM Contexts . . . 20
  - 2.4.1 Definitions and Theoretical Foundations . . . 20
  - 2.4.2 Empirical Evidence on Effectiveness . . . 22
  - 2.4.3 Gamification in the Workplace and IT Operations . . . 22
  - 2.4.4 Risks and Dark Patterns in Gamification . . . 22
  - 2.4.5 Gamification in Inter-Organisational Contexts . . . 23
- 2.5 Software Architecture Approaches . . . 23
  - 2.5.1 Domain Modelling, CQRS, and Event-Driven Communication . . . 23
  - 2.5.2 Architectural Alternatives and Trade-Off Analysis . . . 24
- 2.6 Synthesis: Requirements and Research Gaps . . . 25
  - 2.6.1 Workflow Fragmentation as the Core Gap . . . 25
  - 2.6.2 Requirements for a Multi-Organisation Ticket System . . . 26

2.6.3 Research Gaps . . . 27
2.6.4 Threats to Validity . . . 28
2.6.5 Positioning of This Dissertation . . . 29
2.7 Chapter Summary . . . 29

3 Industry Problem Validation **31**
3.1 Protocol and Instrument . . . 31
3.2 Respondent Pool . . . 32
3.3 Operational Backdrop . . . 33
3.4 Pain Validation (SLR Cross-Check) . . . 33
3.5 Capability Priorities . . . 33
3.6 Qualitative Findings . . . 35
3.7 Threats to Validity . . . 35
3.8 Chapter Summary . . . 38

4 Proposed Architecture and Domain Model **39**
4.1 Core Contribution: The Canonical Inter-Organisational Aggregate Pattern . . . 40
4.1.1 Intent . . . 40
4.1.2 Context and Forces . . . 40
4.1.3 Solution Structure . . . 41
4.1.4 Design Guidelines . . . 42
4.1.5 Consequences . . . 43
4.1.6 Applicability to Adjacent Domains . . . 43
4.1.7 Anti-Patterns and Variations . . . 43
4.1.8 Pattern Summary . . . 44
4.2 Architectural Principles and Patterns . . . 44
4.2.1 Hexagonal Architecture . . . 44
4.2.2 Domain-Driven Design . . . 45
4.2.3 CQRS and Event-Driven Communication . . . 45
4.3 Bounded Context Decomposition . . . 46
4.4 Domain Model . . . 48
4.4.1 Organisation Aggregate . . . 48
4.4.2 Ecosystem Aggregate . . . 49
4.4.3 Billing Aggregates . . . 49
4.4.4 Ticket Aggregate . . . 51
4.5 Inter-Organisational Ticket Routing . . . 52
4.5.1 Routing Model . . . 52
4.5.2 Ecosystem as Routing Scope . . . 53
4.5.3 End-to-End Traceability . . . 55
4.6 Gamification Subsystem Design . . . 55
4.6.1 Domain Model . . . 55
4.6.2 Integration with the Ticket Lifecycle . . . 55
4.6.3 Achievement Trigger Types . . . 56
4.6.4 Design for Fairness . . . 56
4.7 Orchestration Layer . . . 57
4.7.1 Guarded Workflows . . . 57
4.7.2 Cascade Workflows . . . 58
4.7.3 Event-Driven Reactions . . . 58
4.7.4 Cross-Cutting Guards . . . 58

4.8 Runtime Constraints . . . 59
4.9 Design Traceability . . . 59
4.10 Architectural Decision Summary . . . 59
4.11 Design Limitations and Trade-offs . . . 59
4.12 Chapter Summary . . . 62

5 Implementation **64**
5.1 Technology Stack . . . 64
5.2 Implementation Boundary . . . 64
5.3 Core Platform Implementation . . . 64
5.4 Ticket Lifecycle and Routing Implementation . . . 65
5.4.1 Transfer Implementation . . . 66
5.5 Gamification Engine . . . 66
5.6 Frontend Applications . . . 66
5.7 User Interface . . . 66
5.8 Artefact Metrics . . . 69
5.9 End-to-End Walkthrough . . . 70
5.10 Key Technical Challenges . . . 70
5.11 Chapter Summary . . . 70

6 Evaluation **71**
6.1 Requirements Evaluation . . . 71
6.1.1 R1: Canonical Inter-Organisational Ticket Model . . . 71
6.1.2 R2: Formal Routing Events . . . 72
6.1.3 R3: Multi-Level Visibility . . . 72
6.1.4 R4: End-to-End and Segment Metrics . . . 73
6.1.5 R5: Composite SLA Management . . . 73
6.1.6 R6: Multi-Tenant Architecture . . . 74
6.1.7 R7: Single Source of Truth (SSOT) . . . 74
6.2 Evaluation Summary . . . 75
6.3 Gamification Evaluation (RQ2) . . . 75
6.4 Empirical Evaluation with B2B Partner Organisations . . . 76
6.4.1 Protocol Design . . . 76
6.4.2 Setup and Participants . . . 77
6.4.3 Quantitative Findings from Pilot Telemetry . . . 78
6.4.4 System Usability Scale Results . . . 78
6.4.5 Per-Requirement and Per-Gamification Likert Results . . . 79
6.4.6 Qualitative Findings from Free-Text Responses . . . 79
6.4.7 Comparative Assessment . . . 81
6.5 Limitations and Threats to Validity . . . 81
6.6 Chapter Summary . . . 82

7 Conclusions and Future Work **83**
7.1 Summary of Contributions . . . 83
7.2 Answers to the Research Questions . . . 84
7.2.1 RQ1: Architecture for Inter-Organisational Ticket Routing . . . 84
7.2.2 RQ2: Gamification in Inter-Organisational Ticket Management . . . 85
7.3 Future Work . . . 85
7.4 Final Remarks . . . 87

References **88**

A Domain Event Catalogue **93**

B Problem Evaluation Survey Instrument **99**

C Evaluation Protocol **104**

D Data Model **113**

E Canonical Inter-Organisational Aggregate: Pattern Card **117**

F Detailed Review Selection and Supporting Data **122**

List of Figures

1.1 Multi-organisation incident fragmentation. 2
1.2 Design Science Research process model. 5

2.1 Study selection process. 13
2.2 Temporal distribution of the primary studies. 15

4.1 Proposed architecture at a glance. 40
4.2 Canonical Inter-Organisational Aggregate structure. 42
4.3 Hexagonal architecture within a bounded context. 45
4.4 Bounded context map of the proposed architecture. 46
4.5 Core domain model. 49
4.6 Ticket status state machine. 51
4.7 Consent-based transfer request protocol. 54
4.8 Gamification integration flow. 55
4.9 Orchestration layer mechanisms. 57

5.1 Ticket detail view. 67
5.2 Cross-organisational transfer dialogue. 67
5.3 Ecosystem list. 68
5.4 Ecosystem detail view. 68
5.5 Team settings view. 69
5.6 Analytics view. 69

D.1 Persistence-layer bounded-context map. 113

List of Tables

2.1 Inclusion and exclusion criteria for study selection. 12
2.2 Quality assessment rubric. 13
2.3 Primary studies included in the systematic review. 14
2.4 Secondary studies (systematic reviews) retained as background context, not counted among the primary studies. 14
2.5 Commercial ITSM platform comparison. 20
2.6 Gamification elements mapped to SDT, Octalysis, and MDA. 22
2.7 Comparison of architectural alternatives against key requirements. 25
2.8 Requirements for a multi-organisation ticket system and their literature sources. . 27

3.1 Respondent role distribution (attention-passed cohort). 32
3.2 Respondent sector distribution. 33
3.3 Years of experience with inter-organisational tickets. 33
3.4 External organisations coordinated with in a typical week. 34
3.5 Reported share of tickets that cross at least one organisational boundary. 34
3.6 Primary ITSM tool reported by respondents. 34
3.7 Coordination mechanisms in use today (multi-select; sums to more than 100%). . 34
3.8 Problem-validation pain Likert results. 34
3.9 Platform capability priorities. 35
3.10 Recent cross-organisational incident responses (translated from Portuguese). . . . 36
3.11 Shared-platform adoption barrier responses (translated from Portuguese). 37

4.1 Integration patterns between bounded contexts. 47
4.2 Ecosystem permission categories. 47
4.3 Domain-event responsibilities across bounded contexts. 48
4.4 Organisation and Ecosystem invariants. 50
4.5 Principal business invariants enforced by the Ticket aggregate. 53
4.6 Requirements-to-design traceability. 60
4.7 Architectural decision summary. 61

5.1 Technology stack used by the Flowzao proof of concept. 65

6.1 Summary of the requirements evaluation. 75
6.2 Quantitative findings from the platform's usage over the pilot window. 78
6.3 Per-respondent SUS scores. 79
6.4 Pilot requirements and gamification Likert results. 79
6.5 Most valuable feature responses (translated from Portuguese). 80
6.6 Missing or unsatisfactory feature responses (translated from Portuguese). 80
6.7 Comparative assessment against existing ITSM tools. 81

A.1 Complete domain event catalogue organised by bounded context. 93

F.1 Detailed study-selection funnel (per-stage counts and criteria). 122

List of Acronyms

API	Application Programming Interface
B2B	Business-to-Business
B2C	Business-to-Consumer
CI/CD	Continuous Integration / Continuous Delivery
CIOA	Canonical Inter-Organisational Aggregate (architectural pattern)
CMDB	Configuration Management Database
CQRS	Command Query Responsibility Segregation
DDD	Domain-Driven Design
DSR	Design Science Research
E2E	End-to-End
HTTP	Hypertext Transfer Protocol
ITSM	IT Service Management
ITIL	Information Technology Infrastructure Library
JSONB	PostgreSQL binary JSON column type
MDA	Mechanics–Dynamics–Aesthetics (game design framework)
ORM	Object-Relational Mapping
REST	Representational State Transfer
RQ	Research Question
SaaS	Software as a Service
SDG	Sustainable Development Goal
SDT	Self-Determination Theory
SIAM	Service Integration and Management
SLA	Service-Level Agreement
SLR	Systematic Literature Review
SPA	Single-Page Application
SQL	Structured Query Language
SSOT	Single Source of Truth
SUS	System Usability Scale
UI	User Interface

Chapter 1

Introduction

1.1 Context and Motivation

In many Business-to-Business (B2B) service ecosystems, responsibility for diagnosing and resolving an IT incident[1] is distributed across organisational boundaries rather than concentrated in a single firm. Prior work on outsourced and inter-organisational fault management describes this loss of end-to-end visibility as a recurring coordination problem [32–34]. In such ecosystems, value is co-created by multiple actors such as suppliers, customers, platform owners, and service providers through shared digital infrastructures [11, 50]; in IT service delivery, a service degradation may therefore traverse several organisations before the root cause is identified. Each organisation may open its own ticket, in its own system, with its own identifier. By the time the incident is resolved, no participant necessarily holds a complete picture of what happened.

Consider the following scenario: an enterprise customer reports a degradation to their systems integrator, who creates ticket `TK-4521`. Investigation reveals the root cause lies in a third-party cloud provider, so the integrator manually opens a new case `INC-8832` in the provider's system, copying a subset of the diagnostic information. The provider, in turn, escalates to a specialised database vendor, who creates yet another ticket `DB-0091`. Three independent tickets now exist across three organisations, with no shared identifier, no unified timeline, and no way to measure end-to-end resolution time. When the database vendor resolves its ticket, the information must be manually propagated back through the chain. Each handoff introduces delay, potential data loss, and ambiguity about who is responsible for the next action.

Figure 1.1 illustrates this pattern. It is not an exceptional case. ITSM and Information Technology Infrastructure Library (ITIL)-oriented research has largely studied process implementation and improvement within organisational settings [26, 35, 36]. Inter-organisational fault-management studies show that collaborative providers often maintain separate ticket systems that

[1] Throughout this dissertation, "incident" is used in the ITIL sense — an unplanned interruption to, or reduction in the quality of, an IT service [4] — rather than in the reliability-engineering sense of a specific fault type (e.g., a particular fault-tolerance failure mode). The contribution concerns how responsibility for a service-affecting incident is coordinated *across organisations*, and is deliberately agnostic to the underlying fault taxonomy (hardware fault, software defect, configuration error, and so on).

require correlation or integration [31, 33]. When an external partner must be involved, a common operational pattern is to open a new ticket in the partner's system, copy some information over, and coordinate through e-mail or phone calls. The result is multiple tickets representing the same underlying problem, without a global identifier or any formal notion of routing.

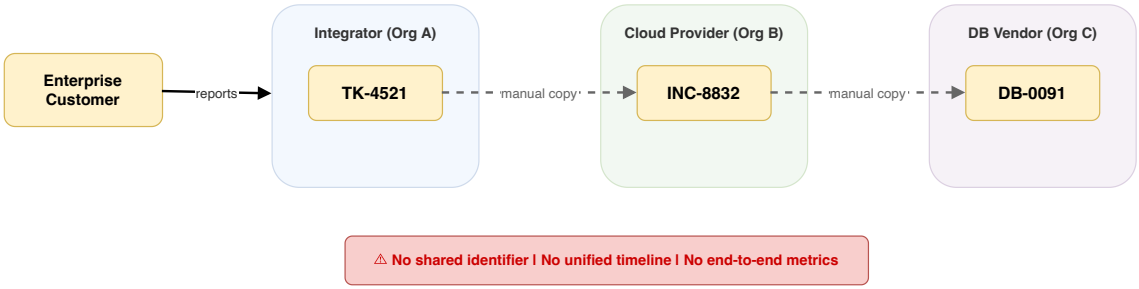


Figure 1.1: Multi-organisation incident fragmentation.

The documented consequences include partial incident views, loss of diagnostic context across system boundaries, redundant or correlated tickets, and weaker end-to-end visibility [32, 34, 45]. Duplicated effort, delayed handoffs, and "ticket ping-pong" are operational implications of this fragmentation, especially in high-stakes services where delays carry significant economic or reputational consequences.

The problem also has practical scale. The global IT service management market is forecast to grow from USD 13.5 billion in 2024 to USD 29.9 billion by 2030 [20], while enterprise IT increasingly relies on outsourced and multi-provider delivery [35]. The Service Integration and Management (SIAM) framework recognises this multi-supplier reality and provides guidance for operating models, governance, process interaction, and reporting across service providers [47]; it does not, however, define the ticket-level routing semantics required here. At the operational level, the inter-organisational literature shows that independent ticket systems create partial views, context loss, and duplicated coordination effort [32, 34, 45].

Precise quantitative estimates of the resulting overhead vary across sources, but the underlying mechanisms are documented: cross-boundary incident handling often involves partial incident views, status chasing, and navigation across partner processes, weakening end-to-end visibility and increasing coordination effort [32, 34, 45]. Across multi-organisation incident chains, even modest per-incident inefficiencies can accumulate into slower resolution and additional coordination effort.

Despite this growing practical urgency, academic attention to inter-organisational ticket management has been notably sparse. The seminal work of Marcu and colleagues [31–34] and the ticket correlation model of Salah et al. [45] established the conceptual foundations between 2009 and 2016. Since then, the systematic review reported in Chapter 2 finds no sustained successor research programme; the ITSM research community has focused predominantly on intra-organisational process improvement, leaving multi-organisation coordination as a peripheral concern.

The technical fragmentation has a human counterpart. Incident resolution work is inherently repetitive, and the cognitive load of navigating across organisational boundaries amplifies the risk of disengagement among the technicians who perform it. Gamification — the use of game-design elements in non-game contexts [16] — has shown context-dependent positive effects on engagement and use in some empirical settings, including work-related and crowdsourcing systems [21, 30], yet its application to inter-organisational ITSM workflows remains unexplored.

Taken together, these observations point to a twofold gap: limited design knowledge on how to model the inter-organisational ticket as a first-class entity with formal routing semantics, and the absence of any mechanism to sustain the motivation of the technicians who operate within these fragmented workflows. The opportunity to address this gap is also timely: serverless infrastructures and reliable event-based integration patterns have only recently matured to a point where such design knowledge can be instantiated and evaluated in an economically operated multi-tenant system, removing the technical barriers that constrained earlier proposals. Moreover, the underlying coordination problem — a single logical case fragmented across autonomous organisations that each hold only a partial view — is not unique to ITSM: it recurs in inter-organisational clinical case management, multi-firm legal matters, cross-carrier logistics, and multi-vendor security incident response, which motivates pursuing transferable design knowledge rather than an ITSM-specific point solution. This dissertation addresses both gaps.

1.2 Problem Statement

The central problem addressed by this dissertation can be stated as follows:

> *Current ITSM tools treat ticket management as an intra-organisational concern. When incidents must traverse organisational boundaries in B2B service ecosystems, three gaps emerge: no canonical ticket entity with formal routing semantics, no end-to-end visibility across organisations, and no mechanism to incentivise collaborative behaviour among the technicians who resolve cross-boundary incidents.*

This problem manifests in four forms of fragmentation synthesised in this dissertation from the reviewed literature [32, 33, 45]:

Informational fragmentation: incident data (symptoms, diagnostics, decisions, workarounds) are scattered across multiple tickets and communication channels, making it difficult to reconstruct full history or perform root-cause analysis.

Temporal fragmentation: there is no single view of the end-to-end resolution time or of waiting times between organisations; each entity sees only its own segment.

Responsibility fragmentation: at any point in time, it may be ambiguous which organisation is responsible for the next action, creating opportunities for blame-shifting and "ticket ping-pong."

Analytical fragmentation: inability to identify cross-organisational bottlenecks or measure collaboration effectiveness across the full resolution chain.

Collectively, these forms of fragmentation weaken end-to-end visibility and increase coordination effort [32, 34, 45]. Longer resolution times, duplicated coordination effort, and higher operating cost are treated here as operational implications of multi-organisation incident chains, even when each individual organisation operates efficiently in isolation.

A Systematic Literature Review conducted as part of this dissertation (Chapter 2) confirms that no production-ready framework or open-source system implements an inter-organisational ticket as a first-class entity. Multi-tenant architecture patterns [6, 22] from adjacent domains provide relevant building blocks; the SLR found no application of these patterns to inter-organisational ITSM ticket routing. Furthermore, no existing system integrates gamification mechanisms into the inter-organisational ticket lifecycle.

1.3 Research Questions

Based on the identified gaps, this dissertation addresses two research questions:

RQ1 *How can centralised ticket-management solutions be architected to support formal incident routing across organisations in B2B service ecosystems?*

RQ2 *How can gamification elements be designed and integrated into the inter-organisational ticket lifecycle?*

RQ1 targets the core architectural challenge. The existing literature proposes inter-organisational ticket correlation and fault-management information models [32, 33], but stops short of specifying how a canonical routed ticket should be designed, persisted, and governed as a production system. RQ1 asks how to close this gap: how to design a system where the ticket is a canonical entity maintained in a multi-tenant Single Source of Truth (SSOT), with explicit routing events, configurable visibility per organisation, and end-to-end metrics that span the full resolution chain. The expected answer takes the form of a domain model and system architecture, grounded in established practices for modelling complex business domains [17, 54].

RQ2 addresses the engagement dimension. Incident resolution is inherently repetitive, and the problem is compounded in inter-organisational settings where technicians must navigate unfamiliar systems and wait for responses from external partners. Gamification has been investigated as a way to support motivation and engagement in software engineering [38] and broader work-related information-system contexts [21, 30], although the evidence remains mixed and context-dependent. Its application to inter-organisational ITSM workflows remains unexplored. RQ2 therefore concerns the *design* of a gamification subsystem [16] that integrates with the inter-organisational ticket lifecycle without coupling to its core logic, and that addresses two distinctive challenges: fairness across organisations of different sizes and roles, and avoidance of perverse incentives where technicians or organisations compete on speed at the expense of resolution quality.

The behavioural impact of such a subsystem on real users requires longitudinal study and exceeds the scope of this dissertation; RQ2 is therefore answered at the design level, complemented by the qualitative signal collected from technicians in the pilot ecosystem described in Chapter 6.

1.4 Research Methodology

This dissertation follows a **Design Science Research (DSR)** methodology [23, 39]. DSR is a problem-solving paradigm appropriate when the objective is to create and evaluate an IT artefact that addresses an identified organisational problem: it produces knowledge through the iterative design of novel artefacts and the evaluation of their utility, which distinguishes it from purely empirical approaches that observe existing phenomena and from theoretical approaches that model phenomena without building concrete solutions. In DSR, the artefact is not the final contribution by itself; it is the means through which design knowledge is developed, made explicit, and evaluated in relation to a relevant organisational problem. In this dissertation, the problem is how to design solutions for incident coordination when ownership, visibility, and responsibility are distributed across multiple organisations. The Flowzao implementation therefore acts as a concrete vehicle for exploring and evaluating architectural principles, domain abstractions, and routing semantics for this class of inter-organisational ITSM problem. DSR is suitable because it supports the generation of transferable knowledge about how such solutions should be designed, rather than merely documenting the construction of a single platform.

The research is structured according to the six activities of the DSR process model proposed by Peffers et al. [39], illustrated in Figure 1.2.

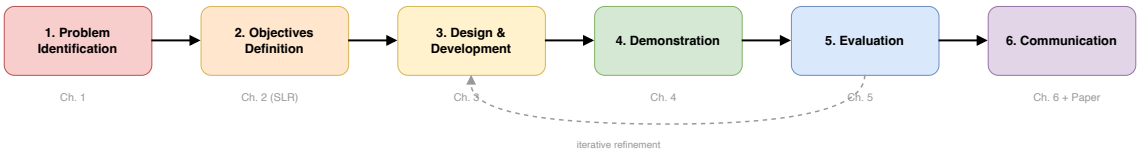


Figure 1.2: Design Science Research process model.

The model is adapted from Peffers et al. [39] and mapped to the chapters of this dissertation. The dashed arrow indicates the iterative refinement cycle between design and evaluation.

1. **Problem identification.** Chapter 1 identifies the problem of fragmented inter-organisational ticket management in B2B ecosystems, drawing on both industry practice and the academic literature; Chapter 3 corroborates it with an industry-side questionnaire administered to practitioners who have not used the platform.

2. **Definition of objectives.** The Systematic Literature Review presented in Chapter 2 derives seven requirements (R1–R7) that any multi-organisation ticket system should satisfy. These requirements act as the evaluation criteria for the artefact.

3. **Design proposal.** Chapter 4 presents the proposed architecture and domain model for inter-organisational ticket management. The design is decomposed into seven *bounded contexts*,

each owning a coherent piece of business logic. Domain events are the integration mechanism between them. The term *bounded context* is formally defined where it is first used in Chapter 4.

4. **Demonstration.** Chapter 5 describes the implementation of the proposed model as Flowzao on a cloud-native serverless infrastructure.

5. **Evaluation.** Chapter 6 evaluates the proposed design as instantiated in Flowzao through complementary instruments: a structured requirements-based assessment against R1–R4, R6, and R7; and an in-ecosystem pilot in which collaborating organisations use the platform on their own workflows. These instruments assess the implemented instantiation and generate formative evidence for the next DSR cycle; they do not claim a completed post-evaluation redesign within this dissertation. R5 (composite SLA management) was descoped to future work; the data foundations for an eventual SLA decomposition — per-organisation assignment durations and time logs — are nevertheless present in the architecture.

6. **Communication.** The results are communicated through this dissertation and an accompanying paper on the SLR.

The dashed refinement arrow in Figure 1.2 should be interpreted with this scope in mind. The implemented refinement cycle occurred primarily between requirements definition, design, and implementation: requirements emerging from the SLR shaped the domain model and evaluation criteria; Event Storming and implementation constraints refined the bounded-context decomposition, routing protocol, and guarded workflow pattern; and iterative testing corrected implementation defects before the final empirical evaluation. The in-ecosystem pilot then corroborated that the proposed design operates under real coordinated workflows, while the industry questionnaire corroborated the underlying problem; both surfaced design implications for future refinement, but they did not lead to a second fully implemented redesign cycle within the time boundaries of this dissertation.

1.5 Contributions

This dissertation makes three contributions:

C1: Systematic Literature Review. A comprehensive review of inter-organisational ticket management in B2B service ecosystems, synthesising 16 primary studies across three domains: inter-organisational fault and incident management architectures, ITSM process challenges in outsourced and multi-vendor environments, and multi-tenant platform architectures for service management. The review identifies seven requirements for a multi-organisation ticket system and catalogues seven critical research gaps, establishing that the foundational 2009–2013 cluster of work has not been followed by sustained research despite growing industry urgency. A condensed version is being prepared as an accompanying paper.

C2: Architecture, domain model, and a transferable design pattern. The dissertation's main contribution is the design knowledge captured by the *Canonical Inter-Organisational Aggregate*: a transferable architectural pattern for modelling cross-organisation coordination around a single canonical aggregate rather than replicated records. Applied to ITSM, the pattern treats the inter-organisational ticket as a first-class entity maintained in a shared platform, with responsibility transferred through formal routing events, visibility scoped per organisation, and end-to-end metrics derived from the same source of truth. The corresponding architecture is organised into seven bounded contexts with reliable event-based integration, two explicit routing mechanisms (direct transfer and transfer-request protocol), and a gamification subsystem that reacts to ticket lifecycle events without coupling to the core ticket logic. The pattern is specified with explicit forces, structure, consequences, design guidelines, and known applicabilities to adjacent inter-organisational coordination domains (clinical case management, multi-firm legal matters, cross-carrier logistics, multi-vendor incident response).

C3: Proof-of-concept instantiation and empirical validation. A functional instantiation of this architecture, Flowzao, deployed on cloud-native serverless infrastructure (Cloudflare Workers with serverless PostgreSQL) and exercised in an in-ecosystem pilot with collaborating organisations.

1.6 Dissertation Structure

The remainder of this dissertation is organised as follows:

Chapter 2: State of the Art. Presents the Systematic Literature Review, covering inter-organisational fault management, ITSM process challenges, multi-tenant architectures, and gamification in organisational contexts. The review derives seven requirements (R1–R7) and identifies seven research gaps that motivate the proposed design model.

Chapter 3: Industry Problem Validation. Reports a public, multilingual questionnaire that corroborates the fragmentation problem with industry practitioners who have not used the platform, providing an artefact-independent signal that complements the literature-derived diagnosis of Chapter 2.

Chapter 4: Proposed Architecture and Domain Model. Describes the proposed domain model and system architecture for inter-organisational ticket management, including the bounded context decomposition, aggregate design, event-driven orchestration, and gamification integration. Architectural decisions are traced back to the requirements established in Chapter 2.

Chapter 5: Implementation. Details the technology choices, deployment infrastructure, and implementation of each bounded context, with emphasis on the inter-organisational ticket lifecycle and the gamification subsystem.

Chapter 6: Evaluation. Evaluates the implemented artefact through a structured requirements-based assessment and an in-ecosystem pilot with collaborating organisations. Quantitative usage

data, qualitative technician feedback, and a standardised usability measurement are reported alongside an analysis of the extent to which each requirement is satisfied, together with the residual limitations.

Chapter 7: Conclusions and Future Work. Summarises the contributions, provides explicit answers to the research questions, and identifies directions for future research.

Appendices. Appendix A provides the domain-event catalogue used by the proof-of-concept implementation, while Appendix B reproduces the public Problem Evaluation Survey instrument. The remaining appendices document the evaluation protocol, persistence model, and supporting reproducibility material.

Chapter 2

State of the Art

This chapter establishes the theoretical and empirical foundations for the dissertation's central design contribution: an architecture, domain model, and transferable *Canonical Inter-Organisational Aggregate* pattern for inter-organisational ticket management. First, the chapter introduces the conceptual background needed to understand B2B service ecosystems and ITSM process assumptions (Section 2.1). Second, it presents the methodology and results of a Systematic Literature Review (SLR) on inter-organisational ticket management (Sections 2.2–2.3). Third, the gamification literature is reviewed to ground engagement mechanisms for the inter-organisational ticket lifecycle (Section 2.4). Fourth, the software architecture approaches that underpin the proposed architecture are examined (Section 2.5). Finally, the findings from all domains are synthesised into a set of requirements and research gaps that motivate the architecture, domain model, and instantiation choices developed in the following chapters (Section 2.6).

2.1 Conceptual Foundations

2.1.1 B2B Service Ecosystems and Distributed Service Delivery

B2B service ecosystems are networks of organisations that jointly deliver complex service offerings, supported by digital platforms and shared infrastructures [11, 50]. In such ecosystems, responsibilities for design, operation, support, and evolution of services are distributed across several entities. In IT service delivery, an end-customer incident may involve software applications, underlying infrastructure, integration layers, local processes, and user practices owned or operated by different organisations.

Although the underlying problem is singular, its management is typically decomposed into multiple tickets, one per organisation or per system. This decomposition is a direct consequence of the intra-organisational design assumptions embedded in conventional ITSM tools.

2.1.2 IT Service Management Frameworks

IT service management frameworks, particularly ITIL, have most often been studied in the literature as process implementation and improvement frameworks within organisational settings [26, 35, 36]. ITIL, now in its fourth edition [4], recognises partners and suppliers as part of service management, but it defines practices at a process level rather than prescribing a canonical cross-organisation ticket-routing data model.

Commercial ITSM tools commonly provide incident logging linked to organisational structures, queues and assignments between teams, SLAs defined from the perspective of the system-owning organisation, and reporting centred on local performance metrics. When collaboration with external providers is required, inter-organisational studies report patterns of duplicated or correlated tickets, references between external systems, and coordination outside a single authoritative lifecycle [1, 33].

The emergence of the SIAM framework reflects industry recognition of multi-supplier service-management needs [47]. However, SIAM addresses operating models, governance, process interaction, and reporting across service providers rather than providing a technical model for inter-organisational ticket routing or a canonical ticket data model.

2.2 Systematic Literature Review Methodology

2.2.1 Objective and Scope

The SLR follows a systematic approach inspired by the guidelines of Kitchenham and Charters [28], adapted to the domains of ITSM, B2B service ecosystems, and inter-organisational coordination. The objective is to identify, evaluate, and synthesise primary studies that contribute to understanding how tickets and incidents are managed across organisational boundaries. A condensed version of this review is being prepared as an accompanying paper; the present chapter extends it with deeper analysis, additional synthesis, and the integration of gamification and software architecture literature.

Three research questions guide the review:

SLR-RQ1 How are tickets or incidents that traverse multiple organisations managed and coordinated in B2B service ecosystems?

SLR-RQ2 What are the reported challenges, gaps, and requirements for enabling end-to-end ticket lifecycle management across organisational boundaries?

SLR-RQ3 Which architectural models or technical approaches address multi-organisation ticket management?

These questions are deliberately distinct from the dissertation's research questions (RQ1 and RQ2, stated in Chapter 1): the SLR questions map the existing knowledge landscape, while the dissertation questions drive the design of a new artefact.

This review sits at the boundary between a systematic literature review and a *systematic mapping study* (SMS) [40, 41]. Its research questions are broad and characterising (how cross-boundary coordination happens, what challenges and requirements exist, and which architectural approaches are used), the search string is deliberately wide, and the synthesis categorises an emerging area rather than performing a meta-analysis of comparable results — all features typical of a mapping study. It nevertheless retains elements that a purely descriptive map often omits: an explicit quality-assessment rubric (Table 2.2), structured data extraction, and a requirements-oriented synthesis (Section 2.6). It is therefore best understood as a systematic review with strong mapping characteristics; the label "SLR" is retained here for continuity, while the mapping-oriented framing is the natural basis for the condensed version prepared as an accompanying paper.

2.2.2 Search Strategy

Three academic databases were queried: Scopus, IEEE Xplore, and ACM Digital Library, covering publications from 2003 to 2025. A single comprehensive search query was designed to capture the terminological diversity of this research area:

```
("inter-organizational" OR "inter-organisational" OR
"cross-organizational" OR "cross-organisational" OR
"multi-provider" OR "multi-vendor" OR "multi-tenant" OR
"outsourc*" OR "service integration")
AND ("incident" OR "fault" OR "ticket" OR "trouble ticket" OR
"service management" OR "ITSM" OR "ITIL" OR "SLA" OR "service
delivery")
AND ("management" OR "coordination" OR "routing" OR "correlation"
OR "architecture" OR "framework" OR "model" OR "platform")
```

The query comprises three clauses. The first targets the inter-organisational dimension using terminological variants encountered in preliminary searches. The second captures the ITSM and incident management domain. The third ensures that results address substantive contributions (architectures, models, frameworks) rather than incidental mentions. The time window begins in 2003 to capture the earliest relevant work on inter-organisational workflows while avoiding obsolete technology contexts.

2.2.3 Inclusion and Exclusion Criteria

The criteria summarised in Table 2.1 guided study selection at both the title/abstract screening stage and the full-text eligibility stage. Inclusion criteria target peer-reviewed contributions that engage substantively with multi-organisational ticket management or with the underlying multi-tenant infrastructure on which such systems are built. Exclusion criteria filter out B2C-only studies, records without a substantive research method or contribution, and studies in which tickets or incidents are mentioned only in passing. The combined criteria narrow the corpus to studies that offer transferable knowledge about the inter-organisational coordination problem rather than

incidental references. The 2003 lower bound (IC3) is set to capture the earliest relevant inter-organisational workflow research — the cooperative inter-organisational workflow framework of Yan and Wang [56] — while excluding older work that predates the technological context assumed here.

Type	Code	Description
Inclusion	IC1	Peer-reviewed journal articles or conference papers
Inclusion	IC2	Addresses ticket/incident management involving multiple organisations, providers, or systems; OR addresses multi-tenant architectures for service management platforms
Inclusion	IC3	Published from 2003 onwards
Inclusion	IC4	Written in English
Exclusion	EC1	Focused exclusively on B2C consumer applications
Exclusion	EC2	Records without a substantive research method or contribution, including product descriptions
Exclusion	EC3	Tickets/incidents appear only tangentially

Table 2.1: Inclusion and exclusion criteria for study selection.

2.2.4 Selection Process and Data Extraction

The selection followed standard SLR practice [28] and is summarised in the PRISMA-inspired flow diagram in Figure 2.1.

The six-stage process comprised:

1. identification of initial references across all databases (424 records);
2. duplicate removal (312 unique records);
3. title and abstract screening against EC1–EC3 (74 candidates);
4. full-text eligibility assessment against IC1–IC4 (27 eligible);
5. quality assessment using a checklist adapted from Kitchenham and Charters, evaluating clarity of objectives, adequacy of context description, theoretical framing, method transparency, and discussion of limitations (20 studies retained — 16 primary studies and 4 secondary studies used as background);
6. systematic data extraction.

The quality assessment rubric is summarised in Table 2.2. Each criterion was scored on a 0–2 ordinal scale (0 = absent, 1 = partial, 2 = clearly addressed), and studies scoring below 5/10 were excluded.

Table 2.3 summarises the 16 primary studies analysed in depth, classified by domain and contribution type. Four further studies that passed the same screening are themselves systematic reviews; because they synthesise prior literature rather than contribute new primary data, they

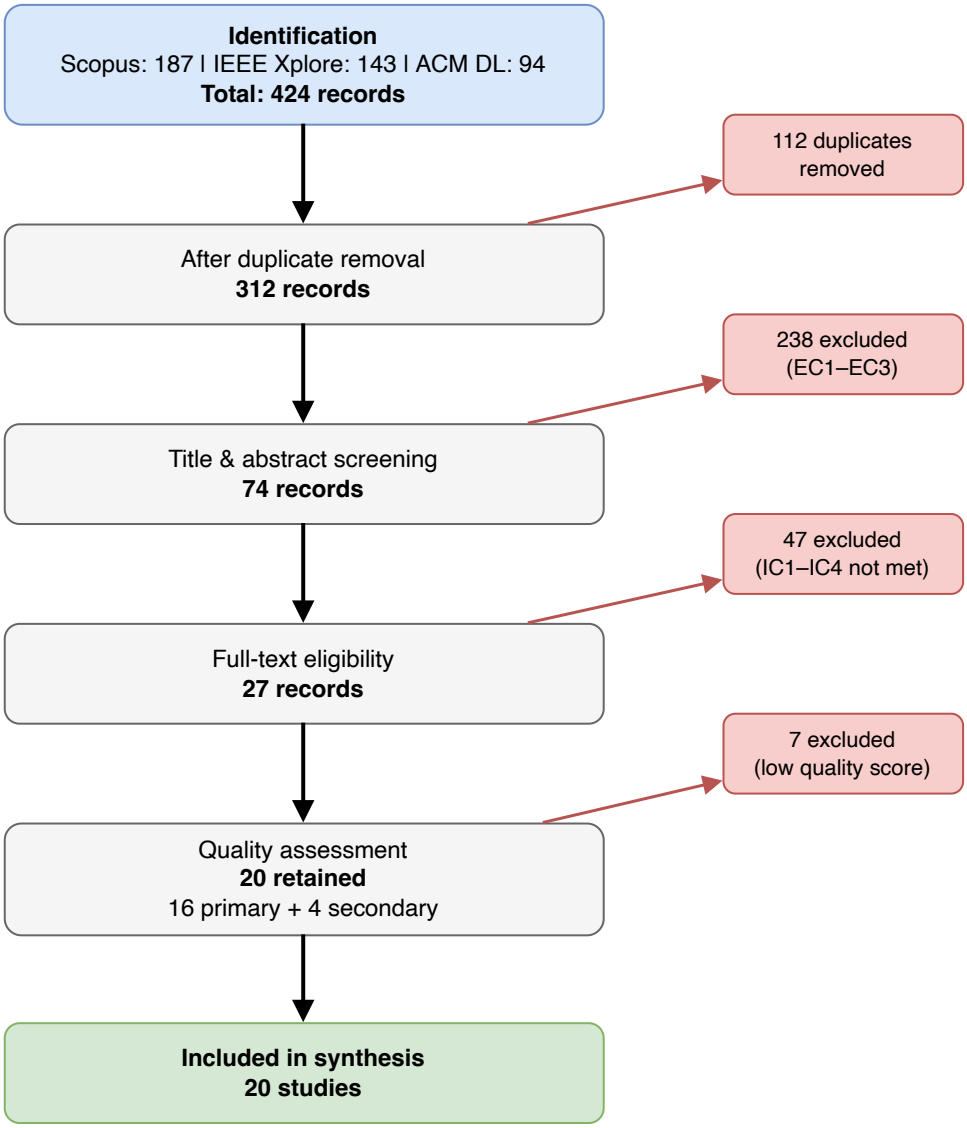


Figure 2.1: Study selection process.

Code	Criterion	Description
QA1	Clarity of objectives	Are the study's research questions or objectives explicitly stated?
QA2	Context adequacy	Is the application context (domain, organisations, scale) described in sufficient detail?
QA3	Theoretical framing	Is the contribution positioned against prior work and a defined conceptual frame?
QA4	Method transparency	Are the methods (e.g., model construction, case-study protocol, evaluation) described in a reproducible manner?
QA5	Limitations	Are threats to validity or limitations acknowledged and discussed?

Table 2.2: Quality assessment rubric.

are retained as background context rather than counted among the primary studies, and are listed separately in Table 2.4.

Study	Year	Domain	Contribution Type
Marcu et al. [33]	2009	Fault management	Correlation model
Marcu et al. [34]	2009	Service delivery	Collaboration model
Marcu & Hommel [31]	2010	Fault management	Requirements & arch.
Marcu & Schaaf [32]	2011	Fault management	Information model
Salah et al. [45]	2016	Network management	Correlation model
Marrone & Kolbe [35]	2011	ITSM frameworks	Empirical study
Adiraju [1]	2012	ITSM security	Case study
Denda & Drajić [15]	2013	Process alignment	Case study
Jäntti [27]	2011	Incident management	Empirical study
Dande et al. [12]	2024	AI in ITSM	Process review
Shwartz et al. [51]	2007	Multi-tenant ITSM	Architecture
Hattab & Belalem [22]	2023	Multi-tenant models	Formal model
Bezemer & Zaidman [6]	2010	Multi-tenant SaaS	Architecture
Theilmann et al. [52]	2010	Composite SLA	Framework
De Vleeschauwer et al. [13]	2021	SLA decomposition	Model
Yan & Wang [56]	2003	Workflow coord.	Framework

Table 2.3: Primary studies included in the systematic review.

Study	Year	Domain	Study Type
Iden & Eikebrokk [26]	2013	ITSM processes	Systematic review
Mesquida et al. [36]	2012	ITSM processes	Systematic review
Shree et al. [50]	2021	B2B platforms	Systematic review
Compagnucci et al. [11]	2025	B2B platforms	Systematic review

Table 2.4: Secondary studies (systematic reviews) retained as background context, not counted among the primary studies.

For each included study, the following data were extracted: domain and context; type of contribution (framework, model, case study, or evaluation); treatment of tickets across organisational boundaries; integration approach (replication, routing, or hybrid); architectural patterns for multi-tenant or multi-provider service management; and reported outcomes regarding efficiency, coordination, or visibility.

Figure 2.2 presents the temporal distribution of the primary studies, highlighting the concentration of foundational work in the 2009–2013 period and the notable absence of sustained research in the subsequent decade.

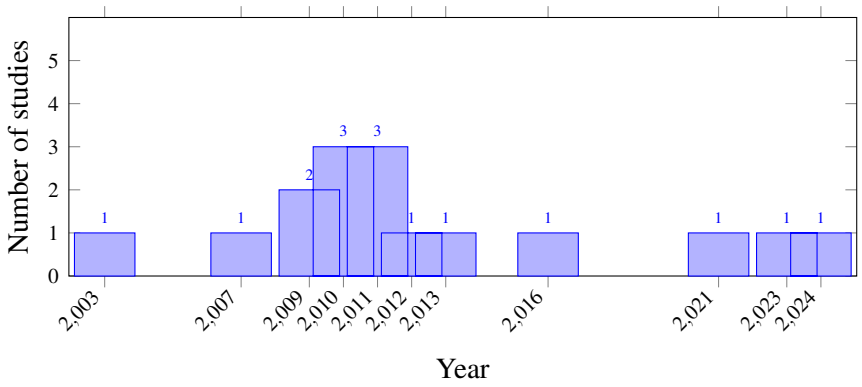


Figure 2.2: Temporal distribution of the primary studies.

2.3 Systematic Review Results

2.3.1 Inter-Organisational Fault Management Architectures

The most directly relevant body of work emerged between 2009 and 2016, when a small group of researchers recognised that provider coalitions create a fundamental problem: each provider's Incident Ticket System captures only a partial view of a service failure, yet no mechanism exists to correlate these partial views automatically [33, 34].

The earliest contribution came from Marcu et al. [33], who introduced the concept of correlating tickets across organisational boundaries by mapping service dependencies and failure propagation paths. Working in a similar direction, Marcu et al. [34] demonstrated that this partial-view problem applies to both per-request and continuous high-availability services, not just isolated incidents. Together, these two contributions framed the problem but left the solution underspecified.

The most comprehensive treatment came from Marcu et al. [31, 32], who moved beyond correlation to propose an inter-organisational fault-management architecture. Across this work, they describe organisational, functional, informational, communication, and coordination concerns; the information model formalises the fault-management information exchanged between organisations. A key insight from their work is that outsourcing erodes visibility: providers who delegate parts of their service delivery lose control over quality unless structured fault-management mechanisms exist to compensate.

Five years later, Salah et al. [45] extended ticket-correlation work to network management by modelling redundant tickets generated by the same incident in telecommunications data. Their model shared the same fundamental assumption as the earlier work: tickets remain separate entities that must be correlated after the fact, rather than a single canonical entity that is routed between organisations.

This body of work established the conceptual vocabulary of the field: ticket correlation, inter-organisational routing, service provider coalitions, and cross-boundary fault propagation. What it did not produce is a sustained research programme. Within the SLR corpus, no subsequent studies

implement these models as working systems, evaluate them with real organisations, or refine them on the basis of practical experience.

2.3.2 Integration Models: Replication versus Routing

A key conceptual distinction emerging from the literature is between *ticket replication* and *structured routing* [32, 33]:

Ticket replication: each organisation creates its own ticket, loosely linked to others representing the same underlying problem. Links are maintained through custom reference fields, free-text annotations, or external identifiers. Each ticket has an independent lifecycle governed by local semantics and local identifiers.

Structured routing: the same ticket is formally transferred between organisations, with explicit semantics of responsibility, state, and context. The ticket retains a single identity throughout its lifecycle, regardless of which organisation currently holds responsibility.

The integration approaches reported in the reviewed literature predominantly rely on replication or coordination between distinct systems. Documented mechanisms include integrations constrained by security and data-sharing concerns [1]; asynchronous message-based integrations where ticket events are published to a message bus and consumed by partner systems [32]; and process-alignment initiatives that harmonise incident-management processes across several companies [15].

While these approaches reduce manual duplication, they preserve the fundamental paradigm of one ticket per system. As a consequence, analytics and SLA reporting remain local to each organisation, and end-to-end visibility of the incident lifecycle is lost [32, 45]. Structured routing, where a ticket is formally handed over from one organisation to another with preserved identity and history, is rarely formalised in existing models or standards. The scarcity of explicit routing models represents a major conceptual gap.

2.3.3 The Inter-Organisational Ticket and Single Source of Truth

A direct result of the SLR is that the most relevant academic proposals motivate, but do not fully specify, a shared representation of incident state across organisational boundaries. The literature proposes inter-organisational ticket correlation and fault-management information models [32, 45]; this dissertation derives from those gaps the stronger architectural requirement of a canonical inter-organisational ticket that exists independently of any particular ITSM system. In this proposed view:

- the ticket has a global identifier shared across all organisations involved in its lifecycle;
- each transfer of responsibility between organisations is represented as a routing event, explicitly indicating source, destination, timestamp, and context;

- states and metrics can be defined at both segment level (per organisation) and end-to-end level (for the whole lifecycle).

A Central Ticket System, operating as a Single Source of Truth (SSOT), would store these canonical tickets and their events. Local ITSM tools in each organisation would become consumers and partial replicas of this canonical representation, instead of independent origins of truth. This design inference directly informs requirements R1, R2, R4, and R7 in Section 2.6.2.

2.3.4 ITSM Process Challenges in Multi-Vendor Environments

The fragmentation problem extends beyond technical integration to organisational and process dimensions. Several studies document these challenges from complementary perspectives, and three recurring themes emerge.

The first theme is **security as a barrier to integration**. Adiraju [1] argues that security issues and constraints can force outsourced ITSM processes and systems to work in silos across client and vendor environments. The analysis motivates an ecosystem model where client and vendor systems integrate under controlled conditions, but stops short of specifying a canonical ticket-routing mechanism.

The second theme is **process misalignment across boundaries**. Denda and Drajić [15] document the definition and implementation of a target incident-management process across three companies in three countries, using both eTOM and ITIL. This illustrates that even when organisations agree to collaborate, harmonising their internal processes is a substantial challenge. In a similar vein, Shwartz et al. [51] found that ITIL does not address the multi-customer, multi-tenant reality that service providers face, with direct consequences for data segregation, process definition, and incident handling.

The third theme is **the operational burden created by process fragmentation**. Jäntti [27] studied the improvement of incident management processes in two IT service provider companies, illustrating that incident management remains an important process-improvement target in IT service-provider environments. However, Jäntti's study focuses on ITIL-based process improvement rather than on inter-organisational ticket routing. The cross-boundary burden identified in this dissertation therefore follows primarily from the inter-organisational fault-management literature: when incident work is distributed across independent systems, technicians must reconstruct context from partial tickets, coordinate handoffs, and operate without a shared lifecycle view [32, 34, 45].

A more recent process-level perspective is offered by Dande et al. [12], who review ITIL configuration management processes in the context of AI-augmented service management. Although their primary focus is artificial intelligence, the paper is included in this review because it discusses CMDB accuracy and vendor tooling in ITIL configuration management. It is therefore used as configuration-management context rather than as direct evidence of inter-organisational ticket-routing fragmentation.

2.3.5 Multi-Tenant Architectures for Service Management Platforms

Multi-tenant architecture studies establish the technical feasibility of shared platforms, while B2B digital-platform literature provides the ecosystem context for multi-actor service delivery [6, 11, 22, 50, 51].

Hattab and Belalem [22] proposed modular models for systems based on multi-tenant services using a multi-level Petri-net approach, showing that complex multi-tenant service variants can be formally modelled and analysed. Their work provides a foundation for reasoning about configurable, tenant-specific process variants within a shared infrastructure.

Bezemer and Zaidman [6] identified the key architectural patterns and challenges of multi-tenancy in web applications, including data isolation, tenant-specific customisation, and performance isolation. Their analysis of the trade-offs between shared and isolated components is directly relevant to the design of a multi-organisation ticketing system where each organisation requires distinct workflow configurations, visibility rules, and SLA definitions while sharing a common ticket lifecycle infrastructure.

The multi-tenant platform model provides a natural architectural foundation for a centralised inter-organisational ticketing system: a shared core maintains canonical tickets and routing events, while each organisation accesses the system through views and interfaces that respect its data boundaries and workflows.

2.3.6 Composite SLA Management

The literature on composite Service Level Agreements proposes patterns relevant to multi-organisation service management. Theilmann et al. [52] provide a reference architecture for multi-level SLA management with SLA translation across service layers. De Vleeschauwer et al. [13] propose risk-based decomposition of end-to-end network-slice SLAs into domain-level targets. These approaches together suggest that composite SLA management is a relevant concern for inter-organisational ticket systems, though its implementation remains an open challenge.

2.3.7 Inter-Organisational Workflow Coordination

Research on inter-organisational workflow management provides additional conceptual foundations. Yan and Wang [56] proposed a cooperative framework for inter-organisational workflow systems, including an inter-workflow meta-model and a Remote Call Process paradigm that enables processes in one organisation to invoke services in another. This pattern is analogous to structured ticket routing, where a ticket in one organisation triggers responsibility transfer to a partner organisation with defined handoff protocols.

B2B digital-platform literature highlights adoption and success factors such as trust, governance, value creation, and actor capabilities [11, 50]; these factors are equally relevant to the adoption of a shared ticket management platform among competing or semi-cooperating service providers.

2.3.8 Commercial ITSM Tools and Multi-Organisation Support

An examination of the current commercial ITSM landscape reinforces the gap identified in the academic literature. The following examples cover widely used ITSM and customer-service platforms selected to illustrate different native collaboration mechanisms: ServiceNow, Atlassian Jira Service Management, and Zendesk. The platform-specific descriptions that follow are based on each vendor's official product documentation as consulted during the review period (2024–2026); they characterise native capabilities, not third-party plug-ins or custom integrations.

ServiceNow provides domain separation for single-instance multi-tenancy [48] and Service Bridge Remote Tasks for bidirectional task synchronisation across ServiceNow instances [49]. These mechanisms support linked or synchronised task records, but they still operate through records in each instance rather than a thesis-defined canonical ticket with formal routing events and end-to-end segment metrics.

Jira Service Management supports multi-project ticket linking and cross-site linking for Atlassian Cloud [3]. However, these links are informational references, not formal routing events. Each project maintains its own ticket identifiers, workflows, and SLAs. When a ticket must traverse organisational boundaries, the common practice is to create a linked issue in a shared project, a form of ticket replication.

Zendesk offers a "ticket sharing" feature [58] that allows tickets to be shared between Zendesk instances operated by different organisations. While this is the closest commercial approximation to inter-organisational routing among the selected tools, the receiving account gets a separate ticket ID; business rules remain separate, while status, comments, and matching custom fields can be synchronised depending on the sharing agreement.

Across the selected platforms and reviewed native documentation, the fundamental paradigm of one ticket or task record per system is preserved. Cross-organisational scenarios are addressed through replication, linking, or synchronisation, the same broad approaches documented in the academic literature [1, 32]. I did not find support for the thesis-defined combination of a single canonical cross-organisation ticket, formal routing events, end-to-end segment metrics, and Single Source of Truth semantics.

Table 2.5 synthesises this analysis by evaluating each platform against the seven requirements derived in Section 2.6.2. The assessment uses three levels: **Yes** (the requirement is natively supported), **Partial** (limited or workaround-based support exists), and **No** (no support). For context, the Flowzao proof-of-concept instantiation is included in the final column; its evaluation is detailed in Chapter 6.

Several observations emerge from this comparison:

- **Of the three platforms analysed, none natively supports R2 or R4.** Formal routing events and end-to-end metrics are absent from all three commercial tools. Cross-organisational

Requirement	ServiceNow	Jira SM	Zendesk	Flowzao
R1: Canonical inter-org. ticket model	No	No	Partial	**Yes**
R2: Formal routing events	No	No	No	**Yes**
R3: Multi-level visibility	Partial	Partial	Partial	**Yes**
R4: E2E & segment metrics	No	No	No	**Yes**
R5: Composite SLA management	No	No	No	No
R6: Multi-tenant architecture	Partial	Partial	Partial	**Yes**
R7: Single Source of Truth	No	No	No	**Yes**

Table 2.5: Commercial ITSM platform comparison.

ticket transfers are uniformly handled through replication, which inherently precludes a unified event trail and composite time measurement.

- **Zendesk's ticket sharing is the closest approximation to R1**, but the receiving instance creates a copy with its own lifecycle, breaking the canonical identity requirement. ServiceNow's domain separation isolates tenants within a single instance but does not provide a shared ticket entity across instances.
- **Visibility (R3) is partially addressed** by all three platforms through role-based access control within their respective boundaries, but none provides the fine-grained, ecosystem-scoped permission model needed for inter-organisational collaboration where competing organisations must share a workspace.
- **Multi-tenancy (R6) is partially supported** in all three platforms (ServiceNow through domain separation, Jira through multi-project structures, Zendesk through multi-brand instances), but these mechanisms serve intra-organisational segmentation rather than inter-organisational collaboration with configurable routing and visibility.
- **R5 remains a challenge for all platforms**, including Flowzao; no platform natively supports composite SLAs spanning multiple organisational segments.

This comparison reinforces the central thesis motivation: the inter-organisational ticket management gap is not merely an academic observation but a concrete deficiency in the current commercial landscape.

2.4 Gamification in Organisational and ITSM Contexts

2.4.1 Definitions and Theoretical Foundations

Gamification is defined as "the use of game design elements in non-game contexts" [16]. This definition deliberately excludes full-fledged serious games and game-based learning, focusing instead

on the selective application of individual game mechanics (such as points, badges, leaderboards, and challenges) to existing systems and workflows. This boundary between gamification and adjacent forms — serious games, game-based learning, and simulation games — is consistent with the terminology consolidated by the modelling-and-simulation community, which situates gamification alongside, but distinct from, simulation games [46].

Three complementary theoretical frameworks inform the design of gamification systems:

Self-Determination Theory (SDT). The most commonly invoked theoretical underpinning in the gamification literature is SDT [14, 43], which posits that human motivation is driven by three innate psychological needs: *autonomy* (the desire to be the causal agent of one's actions), *competence* (the desire to master tasks and learn new skills), and *relatedness* (the desire to interact with and be connected to others). When these needs are satisfied, individuals are more likely to experience intrinsic motivation, engaging in an activity for its inherent satisfaction rather than for external rewards. Sailer et al. [44] experimentally found that badges, leaderboards, and performance graphs increased competence need satisfaction, while avatars, meaningful stories, and teammates increased social relatedness; perceived decision freedom was not affected as intended.

The MDA Framework. Hunicke et al. [24] proposed the Mechanics–Dynamics–Aesthetics (MDA) framework, which decomposes game design into three layers: *mechanics* (the rules and components), *dynamics* (the run-time behaviour emerging from player interaction with the mechanics), and *aesthetics* (the emotional responses evoked in the player). The MDA framework provides a structured vocabulary for reasoning about gamification design: a designer selects mechanics (e.g., points, leaderboards), anticipates the dynamics they may produce (e.g., competition, collaboration), and evaluates whether the resulting aesthetics (e.g., sense of achievement, fellowship) align with the intended outcomes. In the context of inter-organisational ticket management, the MDA framework helps reason about whether selected mechanics may produce unintended dynamics such as destructive competition between organisations.

The Octalysis Framework. Chou [9] proposed the Octalysis framework, which identifies eight core drives of human motivation: (1) epic meaning and calling; (2) development and accomplishment; (3) empowerment of creativity and feedback; (4) ownership and possession; (5) social influence and relatedness; (6) scarcity and impatience; (7) unpredictability and curiosity; and (8) loss and avoidance. In Chou's practitioner framework, drives 1–3 are described as "white hat" and more empowering or control-supportive, while drives 6–8 are described as "black hat" because they can create urgency while potentially reducing perceived control over time. The Octalysis model provides a practical diagnostic vocabulary: by mapping a gamified system's features to the eight drives, designers can identify over-reliance on extrinsic or coercive motivators and rebalance towards more autonomy-supportive engagement. Table 2.6 maps the gamification elements later instantiated in the Flowzao proof of concept to these theoretical constructs.

Game Element	SDT Need	Octalysis Drive	MDA Aesthetic
Points	Competence	Development & accomplishment	Sensation of progress
Achievements	Competence, Autonomy	Development; Ownership	Challenge, Discovery
Leaderboards	Competence, Relatedness	Social influence	Competition, Fellowship
Rewards catalogue	Autonomy	Ownership; Empowerment	Expression, Submission
Streaks	Competence	Loss & avoidance	Challenge
Feedback ratings	Relatedness	Social influence	Fellowship

Table 2.6: Gamification elements mapped to SDT, Octalysis, and MDA.

2.4.2 Empirical Evidence on Effectiveness

Two major reviews synthesise the empirical evidence on gamification. Hamari et al. [21] conducted a literature review of empirical studies, reporting that gamification often produces positive effects on engagement and motivation, but that outcomes are context-dependent and influenced by the quality of implementation. Koivisto and Hamari [30] later extended this analysis, reviewing 819 studies and showing that gamification research has grown rapidly, with education, health, and crowdsourcing as the most studied domains.

Across these reviews, gamification appears most effective when game elements are aligned with the psychological needs of the target users [44], when the design avoids over-reliance on extrinsic rewards that may crowd out intrinsic motivation [43], and when users perceive the gamified system as fair, transparent, and voluntary rather than coercive [21].

2.4.3 Gamification in the Workplace and IT Operations

Gamification in workplace settings has been studied in the contexts of software engineering [38], crowdsourcing [37], and enterprise knowledge management. Pedreira et al. [38] conducted a systematic mapping of gamification in software engineering, identifying points, badges, and leaderboards as common mechanics across activities such as software development, requirements, testing, project management, problem resolution, collaboration, and knowledge management.

In the specific context of IT service desks and helpdesk environments, the application of gamification remains limited. While commercial ITSM vendors have introduced gamification features in recent products, academic research on their design, implementation, and impact is scarce. Within the literature reviewed for this dissertation, no study was found that applies gamification to inter-organisational ticket routing or to the cross-boundary incident lifecycle.

2.4.4 Risks and Dark Patterns in Gamification

A growing body of work cautions against naïve or coercive implementations of gamification. Hyrynsalmi et al. [25] classify dark-side concerns into limiting and harmful issues, including opti-

mising leaderboard position rather than the task, individual incentives conflicting with teamwork, reward dependence, distraction or productivity loss, and ethical concerns. Toda et al. [53] survey negative effects of gamification in education and report that leaderboards are frequently associated with negative outcomes, reinforcing the need to identify and mitigate unintended effects.

These findings have direct implications for inter-organisational ticket management. In a context where technicians from different organisations share a platform, poorly designed gamification can: (i) incentivise speed over quality, leading to premature ticket closures; (ii) create inter-organisational rivalry that undermines collaboration; (iii) penalise organisations with fewer tickets or smaller teams through unfair leaderboard dynamics; and (iv) expose commercially sensitive performance data.

The design of the gamification subsystem in this dissertation (Chapter 4) explicitly addresses these risks through configurable achievement triggers, ecosystem-scoped leaderboards, approval-based reward redemption, and opt-in participation.

2.4.5 Gamification in Inter-Organisational Contexts

The application of gamification across organisational boundaries introduces additional challenges not present in intra-organisational settings. When technicians from multiple organisations participate in a shared platform, the gamification design must address:

- **Cross-organisational fairness:** leaderboards and point systems must avoid creating perverse incentives where organisations compete on speed at the expense of quality, or where smaller organisations are systematically disadvantaged.
- **Performance data privacy:** individual and organisational performance metrics may be commercially sensitive; the gamification system must respect visibility boundaries.
- **Opt-in participation:** gamification should enhance rather than coerce, particularly when technicians from partner organisations did not choose the platform.

The review identified no existing work that applies gamification to inter-organisational ticket management. This represents a significant gap that this dissertation addresses through the design of event-driven gamification mechanisms for the inter-organisational ticket lifecycle, later implemented in the Flowzao instantiation.

2.5 Software Architecture Approaches

2.5.1 Domain Modelling, CQRS, and Event-Driven Communication

Three architectural ideas are relevant to the design knowledge developed in this dissertation. *Domain-Driven Design* (DDD) prioritises modelling complex business domains through a shared

language, bounded contexts, aggregates, and domain events [17, 54]. In a multi-organisation ticketing system, bounded contexts provide a way to separate concerns such as tickets, organisations, ecosystems, notifications, and gamification while preserving explicit integration contracts.

Command Query Responsibility Segregation (CQRS) separates commands that change state from queries that read state [18, 57]. This separation is useful for inter-organisational ticket management because the same underlying ticket lifecycle must be projected into organisation-specific views: each organisation sees data according to its role, visibility rules, and reporting needs, while the write model preserves a single authoritative lifecycle.

Event-driven communication records significant domain changes as events and delivers them to interested components [29, 42]. The relevant design value is not full event sourcing, where all state is reconstructed exclusively from events, but the use of immutable lifecycle events and reliable publication mechanisms such as the transactional outbox. This gives the architecture an auditable record of routing actions, enables reactive integrations such as notifications and gamification, and avoids tight coupling between bounded contexts.

Taken together, DDD, CQRS, and event-driven communication provide the architectural foundation for a canonical inter-organisational ticket: DDD defines the domain boundaries and aggregate structure; CQRS enables organisation-specific read models; and domain events capture routing, lifecycle, and integration semantics. Although these patterns are established in enterprise software [42, 54], the reviewed literature does not apply their combination to inter-organisational ITSM.

2.5.2 Architectural Alternatives and Trade-Off Analysis

To justify the selected architectural approach, it is instructive to consider the alternatives and their trade-offs in the context of inter-organisational ticket management.

Monolithic CRUD with shared database. The simplest approach: a single database with organisation-scoped queries, standard CRUD operations, and no event model. This is straightforward to implement but lacks an audit trail of routing events and does not naturally capture the provenance of each state change. Implementing per-organisation read views requires complex query logic, and the absence of domain events prevents reactive integration with subsystems such as gamification and notification.

Microservices without DDD. A distributed architecture where each capability (tickets, organisations, gamification) is an independent service communicating via REST or message queues. While this provides deployment independence, the lack of DDD's strategic design often leads to poorly defined service boundaries and tight coupling between services [42]. In the inter-organisational ticketing domain, where the ticket lifecycle spans multiple concerns (routing, metrics, gamification), unclear boundaries would produce either excessive inter-service communication or duplicated domain logic.

Federation / mesh model. Each organisation operates its own ITSM instance, connected through a federation protocol that synchronises ticket state. This preserves organisational autonomy but fundamentally relies on ticket replication, the paradigm identified in Section 2.3.2 as insufficient for end-to-end metrics and canonical ticket identity. Federation also introduces consistency challenges: conflicting updates in different instances require reconciliation protocols whose complexity grows with the number of participating organisations.

Modular monolith with event-driven integration (selected). A single deployable unit organised into bounded contexts with explicit contracts, domain events published through a transactional outbox, and CQRS read models. This approach preserves the conceptual clarity of microservices boundaries without the operational overhead of distributed deployment. The outbox guarantees reliable event delivery, providing a natural audit trail and a foundation for reactive integration between bounded contexts. CQRS enables per-organisation views. The trade-off is increased implementation complexity relative to CRUD, justified by the requirements for auditability (R2), multi-level visibility (R3), and end-to-end metrics (R4).

Table 2.7 summarises this comparison.

Requirement	CRUD	Microservices	Federation	DDD+CQRS+ED
R1: Canonical ticket	Partial	Yes	No	**Yes**
R2: Routing events	No	Partial	Partial	**Yes**
R3: Multi-level views	Partial	Yes	Partial	**Yes**
R4: E2E metrics	Partial	Partial	No	**Yes**
R6: Multi-tenancy	Partial	Partial	Yes	**Yes**
R7: Single Source of Truth	Yes	Partial	No	**Yes**
Audit trail	No	Partial	No	**Yes**
Impl. complexity	Low	High	Very high	Medium

Table 2.7: Comparison of architectural alternatives against key requirements.

2.6 Synthesis: Requirements and Research Gaps

2.6.1 Workflow Fragmentation as the Core Gap

The SLR and the commercial-tool analysis converge on a central gap: the absence of a shared ticket entity across organisations is associated with several recurring forms of workflow fragmentation [32, 33, 45]. Chapter 1 introduced these briefly; here they are examined in greater depth, as each has specific implications for the proposed architecture:

Informational fragmentation: incident data (symptoms, diagnostics, decisions, workarounds) are scattered across multiple tickets and communication channels. When a ticket is replicated into a partner's system, only a subset of the original context is copied, and subsequent

updates are not propagated back. Over time, the copies diverge, making it impossible to reconstruct the full incident history from any single system. This directly motivates the need for a canonical ticket entity (R1) and a Single Source of Truth (R7).

Temporal fragmentation: there is no single view of the end-to-end resolution time or of waiting times between organisations; each entity sees only its own segment. A customer who asks "how long did this take to resolve?" cannot get an answer without manually aggregating data from multiple systems. This motivates the need for end-to-end and segment metrics (R4).

Responsibility fragmentation: at any point in time, it may be ambiguous which organisation is responsible for the next action, creating opportunities for blame-shifting and "ticket ping-pong." Without formal routing events, there is no authoritative record of when responsibility was transferred and by whom. This motivates formal routing semantics (R2).

Analytical fragmentation: the inability to identify cross-organisational bottlenecks or measure collaboration effectiveness across the full resolution chain. Each organisation reports on its own performance, but no one measures the performance of the ecosystem as a whole. This motivates multi-level visibility (R3) and end-to-end metrics (R4).

In addition to these four structural forms of fragmentation, the literature identifies two further operational consequences:

Data quality degradation: manual re-entry of information across systems introduces transcription errors, information loss, and inconsistencies. The reviewed inter-organisational fault-management studies identify context loss and partial incident views as recurring consequences of independent ticket systems [32, 34, 45].

Governance and trust barriers: organisations may be reluctant to expose detailed operational data to partners, particularly when the data reveals performance weaknesses or commercially sensitive information. Adiraju [1] argues that security and access-control concerns are central barriers to integrating fragmented, outsourced ITSM processes.

The reviewed analyses in B2B support and network management environments indicate that such fragmentation is associated with weaker end-to-end visibility, higher coordination effort, and greater difficulty reconstructing the full incident lifecycle [32, 34, 45].

2.6.2 Requirements for a Multi-Organisation Ticket System

Synthesising insights across the reviewed 16 primary studies and the fragmentation gap described above, seven requirements emerge for a centralised, multi-organisation ticket system. Table 2.8 maps each requirement to its motivating literature sources; the requirements are derived in this dissertation from the gaps in those sources rather than specified verbatim by the papers. R1–R4, R6, and R7 are within the implementation scope of this dissertation; R5 (composite SLA

management) is included for completeness as it appears in the literature, but its implementation has been descoped to future work as discussed in Chapter 1 (Research Methodology).

Req.	Description	Motivating Sources
R1	Canonical inter-organisational ticket model with global identifier and explicit representation of participating organisations.	Marcu et al. [33]; Marcu & Schaaf [32]
R2	Formal routing events capturing each transfer of responsibility between organisations.	Marcu & Hommel [31]; Marcu & Schaaf [32]
R3	Multi-level visibility with fine-grained control per organisation, preserving auditability.	Adiraju [1]; Shwartz et al. [51]
R4	End-to-end and segment metrics: resolution time, time per organisation, handoff count.	Salah et al. [45]; Marcu et al. [34]
R5	Composite SLA management spanning multiple organisations.	Theilmann et al. [52]; De Vleeschauwer et al. [13]
R6	Multi-tenant architecture with logical isolation and configurable workflows.	Hattab & Belalem [22]; Bezemer & Zaidman [6]
R7	Single Source of Truth (SSOT) with canonical tickets maintained centrally.	Marcu & Schaaf [32]; B2B platform context in Shree et al. [50]

Table 2.8: Requirements for a multi-organisation ticket system and their literature sources.

2.6.3 Research Gaps

The synthesis of the SLR results, gamification literature, and software architecture review reveals seven critical research gaps:

1. **Absence of a canonical ticket metamodel.** No widely adopted model is identified in the reviewed literature for tickets that traverse multiple organisations with explicit routing events and multi-level metrics [32, 45].

2. **No sustained research programme.** The foundational 2009–2013 cluster of work, particularly that of Marcu and colleagues, has not been followed by implementation, empirical evaluation, or refinement (with Salah et al. 2016 as a notable but isolated extension), despite substantial growth in multi-vendor ecosystems over the intervening years.

3. **No end-to-end visibility solution.** Existing integration approaches (API synchronisation, message buses, intermediary portals) address data transfer but not end-to-end lifecycle management or cross-organisational analytics.

4. **Untapped multi-tenant architecture patterns.** Mature multi-tenant design patterns exist [6, 22] but have not been applied to the specific problem of inter-organisational ticket management with configurable visibility and routing.

5. **Missing governance models.** While security, trust, and data governance are recognised as barriers to inter-organisational collaboration [1, 50], no concrete governance model has been proposed for a shared ticketing platform where competing or semi-cooperating organisations must share operational data under controlled conditions.

6. **No application of gamification to inter-organisational ITSM.** Despite context-dependent evidence that gamification can support engagement and motivation in some work-related settings [21, 30], no existing work applies gamification to the inter-organisational ticket lifecycle to support technician collaboration.

7. **No integrated architecture for inter-organisational ITSM.** Domain-Driven Design, CQRS, and event-driven communication are established in enterprise software practice [42, 54], but the reviewed literature does not apply their combination to the specific problem of inter-organisational ticket management.

The absence of recent academic attention, combined with growing practical urgency evidenced by the rise of SIAM, the expansion of multi-vendor support markets, and the increasing complexity of cloud-based B2B service ecosystems, establishes a compelling and timely research gap. These literature-derived gaps are subsequently corroborated by an independent industry-side instrument: the public problem-validation questionnaire reported in the following chapter (Chapter 3), where practitioners with no exposure to the Flowzao instantiation report the same fragmentation symptoms (loss of context, unclear responsibility, absence of end-to-end metrics) on a 1–5 Likert scale, with several items clustered at or above 4.

2.6.4 Threats to Validity

Following the framework of Ampatzoglou et al. [2] for threats to validity in secondary studies, the threats to this review are organised into three categories: study selection validity, data validity, and research validity.

Study selection validity. The review queried three databases (Scopus, IEEE Xplore, ACM Digital Library). While these cover the majority of peer-reviewed computer-science and information-systems literature, relevant work in other venues (Springer, Wiley, domain-specific journals not indexed by these databases) or expressed under non-standard terminology — inter-organisational ticket management is discussed as fault management, incident correlation, trouble ticketing, or service integration — may have been missed, despite a query deliberately designed to span this terminological diversity. Because the protocol was restricted to academic databases, industry reports and vendor white papers were not searched directly, which may under-represent practitioner-driven solutions. Study

selection was performed by a single researcher, introducing a risk of subjective judgement in borderline cases; this was mitigated by defining the inclusion and exclusion criteria *a priori* (Table 2.1), applying the quality-assessment rubric systematically (Table 2.2), and discussing and reviewing borderline decisions with an experienced researcher (the supervisor). Dual-reviewer selection with inter-rater reliability measurement would nonetheless have strengthened the process further.

Data validity. Data extraction and the requirements-and-gaps synthesis were also carried out by a single researcher, following a predefined extraction procedure (Section 2.2); dual extraction with inter-rater agreement would have improved reliability. The review includes only peer-reviewed publications, which tend to report positive results, so failed inter-organisational implementations or negative multi-vendor outcomes may be under-represented (publication bias). The final set of 16 primary studies is relatively small, reflecting the genuinely underexplored nature of the area rather than an overly restrictive protocol; the temporal distribution (Figure 2.2) confirms the concentration of foundational work in 2009–2013 with limited follow-up.

Research validity. The full protocol — research questions, search string, inclusion and exclusion criteria, and quality rubric — is reported to support repeatability. As a review of a nascent area covered by the queried databases, its conclusions generalise to inter-organisational ITSM as indexed by those sources; extending the search to additional databases and grey literature is left to future replications.

2.6.5 Positioning of This Dissertation

This dissertation addresses the identified gaps by proposing a generic design approach for inter-organisational ticket management. The contribution is the architecture, domain model, and transferable *Canonical Inter-Organisational Aggregate* pattern: a way of treating the inter-organisational ticket as a first-class, shared, routable entity with explicit lifecycle events, organisation-scoped visibility, and end-to-end metrics. Chapter 4 presents the architecture and domain model, Chapter 5 describes the Flowzao implementation, and Chapter 6 evaluates the in-scope requirements (R1–R4, R6, R7), with R5 treated as future work.

2.7 Chapter Summary

This chapter established the background and evidence base for the dissertation's design contribution. The SLR shows that inter-organisational ticket management remains underexplored: existing work and commercial ITSM tools mostly rely on replication or synchronisation between independent tickets, rather than on a canonical, routable ticket entity with shared lifecycle semantics. The gamification review identifies relevant engagement mechanisms and risks, but no application to inter-organisational ITSM. The architecture review motivates DDD, CQRS, and event-driven communication as suitable building blocks. The synthesis derives seven requirements and positions

the architecture, domain model, and *Canonical Inter-Organisational Aggregate* pattern developed in Chapter 4 as a response to these gaps.

Chapter 3

Industry Problem Validation

The Systematic Literature Review in Chapter 2 derived the inter-organisational fragmentation problem, its seven requirements (R1–R7), and its research gaps from the academic literature. This chapter complements that literature-derived diagnosis with an independent industry-side reading of the same problem. A public, multilingual questionnaire was deployed to test whether the fragmentation symptoms catalogued in the SLR are recognised by professionals in their own day-to-day environments, and if so, with what frequency and intensity. The instrument is deliberately artefact-independent: respondents are asked exclusively about their experience with cross-organisational ticket coordination in their own environments. The signal it provides therefore validates the *problem* that this dissertation addresses.

3.1 Protocol and Instrument

The *Problem Evaluation Survey* is a 17-item online questionnaire available in four languages (Portuguese, English, Spanish, German) on the dissertation's landing site, at a public per-locale URL of the form `/<locale>/problem-survey`. Responses are collected through the public form and reflected in the figures reported below; the full English instrument is reproduced in Appendix B. The instrument is structured into five blocks:

Block A — background. Role, sector, years of cross-org experience, number of partner organisations coordinated per week. Used to characterise the respondent population and stratify subsequent analyses.

Block B — operational reality. Percentage of tickets crossing organisational boundaries, coordination methods used today (multi-select), and primary ITSM tool. Establishes the operational backdrop against which pain is reported.

Block C — pain validation. Six Likert items (1–5 scale), mapped onto five SLR requirements (R_1–R_4 and R_7; R_5 and R_6 are solution-side capabilities, probed in Block D rather than as first-person pain). Worded in the first person and grounded in the last six months of work to force concrete recall rather than general impression.

Block D — desirability and priorities. One overall desirability score for a canonical inter-organisational ticketing platform, plus an item-by-item importance rating of the six platform capabilities. The item-by-item form was preferred over a forced rank because the goal is to identify which capabilities matter most across the population rather than to enforce within-respondent priority distinctions.

Block E — qualitative. Two optional free-text prompts (500 characters each): a recent cross-organisational incident that went wrong, and a description of what would prevent the respondent's organisation from joining a shared platform.

A single attention-check item is interleaved into Block C (instructing respondents to select the value 3); responses that do not satisfy this check are excluded from all aggregated figures and from the means reported below.[1] Email addresses are required (a contact-channel for an optional follow-up) and are stored separately from the response analytics.

The questionnaire is intentionally short (target completion time five to seven minutes) and was reviewed for length and bias once before deployment.

3.2 Respondent Pool

The questionnaire collected 17 responses, of which 17 (100%) passed the attention check and are included in the analyses below. Although the instrument was published in four languages, the language distribution of the obtained responses (Portuguese 17, English 0, Spanish 0, German 0) is strongly skewed towards Portuguese, reflecting the predominantly Portuguese-speaking professional network through which the questionnaire circulated; the implications of this single-language concentration for generalisability are revisited among this instrument's threats to validity below. Demographic distribution is summarised in Tables 3.1–3.4. The instrument did not record respondents' employers, so the independence of responses across distinct organisations cannot be asserted directly; the spread of roles and sectors (Tables 3.1 and 3.2) indicates that the responses span multiple organisational contexts rather than a single firm, and a future deployment should capture organisational identity explicitly to confirm this.

Role	n	%
Team lead / incident coordinator	5	29%
Support technician / operator	4	24%
Architect / platform engineer	3	18%
Other	3	18%
Decision maker (CIO, CTO, Head of Operations)	1	6%
Service / ITSM manager	1	6%

Table 3.1: Respondent role distribution (attention-passed cohort).

[1] The check is purely automated; the underlying response row is preserved in the database for auditability.

Sector	n	%
IT services / Outsourcing	14	82%
Banking / Insurance	1	6%
Logistics / Transportation	1	6%
Other	1	6%

Table 3.2: Respondent sector distribution.

3.3 Operational Backdrop

Tables 3.5 to 3.7 characterise the operational reality reported by respondents, providing the context against which the pain Likert items in Block C are interpreted: what fraction of their work crosses organisational boundaries today, which ITSM tool they use, and which mechanisms they rely on to coordinate that work.

3.4 Pain Validation (SLR Cross-Check)

Table 3.8 reports the cohort means and medians on the six pain items, each annotated with the requirement it tests. Insofar as these means cluster towards the upper half of the Likert scale and the share of respondents at ≥ 4 is high, the SLR-derived hypothesis that inter-organisational ticket coordination is fragmented along the dimensions identified in Chapter 2 is supported by an industry-side signal independent of any platform demonstration.

3.5 Capability Priorities

Table 3.9 reports the cohort means on the six platform-capability importance items (Block D, Q15), sorted descending by mean importance. The ordering maps onto the seven requirements derived in Chapter 2, indicating which of them would, in the population's judgement, most justify the implementation effort. This is offered as supporting evidence for the scope decisions taken later in the dissertation (R1–R4, R6, R7 implemented; R5 deferred; Chapters 4 and 6), not as a *post-hoc* justification of those decisions.

Years of experience	n	%
4–7 years	7	41%
8+ years	5	29%
1–3 years	4	24%
< 1 year	1	6%

Table 3.3: Years of experience with inter-organisational tickets.

Partner orgs / week	**n**	**%**
1–2	6	35%
3–5	6	35%
0	4	24%
6–10	1	6%

Table 3.4: External organisations coordinated with in a typical week.

Cross-org tickets	**n**	**%**
< 10%	7	41%
> 75%	4	24%
10–25%	3	18%
26–50%	3	18%

Table 3.5: Reported share of tickets that cross at least one organisational boundary.

Primary ITSM tool	**n**	**%**
Jira Service Management	8	47%
Other	4	24%
Freshservice	3	18%
Internal / in-house	1	6%
No dedicated ITSM tool	1	6%

Table 3.6: Primary ITSM tool reported by respondents.

Coordination method	**n**	**%**
Email	15	33%
Phone / chat	13	28%
Manual ticket replication	7	15%
Shared portal	6	13%
No formal process (ad hoc)	3	7%
API integration	2	4%

Table 3.7: Coordination mechanisms in use today (multi-select; sums to more than 100%).

Pain item (mapped to)	**n**	**Mean**	**Median**	**% $\geq$ 4**
Conflicts / ping-pong over ownership of next step *(C12 / R2)*	17	4.29	4	82%
No visibility over the partner's work *(C13 / R3)*	17	4.24	4	82%
Did not know who owned the next step *(C9 / R2)*	17	4.12	4	82%
Could not measure end-to-end resolution time *(C10 / R4)*	17	3.59	4	53%
Re-entered data already present in another system *(C11 / R7)*	17	3.59	4	53%
Lost important context across systems / channels *(C8 / R1, R7)*	17	3.53	4	53%

Table 3.8: Problem-validation pain Likert results.

Capability	Maps to	n	Mean	% ≥ 4
Canonical ticket shared across organisations	R1	17	4.35	82%
End-to-end metrics (total, per-org, handoffs)	R4	17	4.35	94%
Configurable visibility per organisation	R3	17	4.35	82%
Multi-tenant workspaces with shared tickets	R6	17	4.35	82%
Formal routing events for responsibility transfer	R2	17	4.12	76%
Composite SLAs across organisations	R5	17	3.88	76%

Table 3.9: Platform capability priorities.

3.6 Qualitative Findings

The two free-text items in Block E elicited 12 narrative descriptions of recent cross-organisational incidents that went wrong, and 12 descriptions of perceived adoption barriers for a shared platform. The responses are reported in Tables 3.10 and 3.11, translated from the original Portuguese by the author and ordered by submission date; longer entries are truncated in print and marked with an ellipsis. As elsewhere in this dissertation, the qualitative material is treated as a source of concrete signals rather than a saturated thematic analysis, and a frequency-coded thematic pass is reserved for once $n \geq 30$.

3.7 Threats to Validity

Following the classification commonly used in primary empirical software-engineering studies [55], the threats to this questionnaire are grouped into construct, internal, external, and conclusion validity.

Construct validity. Block D measures *stated* desirability rather than revealed preference; intent-to-adopt does not equal adoption. Agreement-direction Likert items remain susceptible to acquiescence bias despite the attention check, so the medians and the share of ≥ 4 responses — rather than the means alone — are reported to mitigate over-reading of central tendency. The questionnaire was authored by the dissertation author, with no independent expert review of the items beyond the supervisory team, so item wording may frame responses.

Internal validity. The six-month retrospective frame constrains recall to a concrete period, but reported intensity of pain is memory-mediated rather than directly observed. The interleaved attention-check item filters inattentive responses, but cannot detect all careless answering.

External validity. The questionnaire is open and shared through the dissertation author's professional network and the public landing site, so respondents are a self-selected, non-probabilistic sample of B2B service ecosystems. Despite being offered in four languages, the obtained responses were exclusively Portuguese-language (17 of 17); whether the same fragmentation intensities hold across other linguistic, national, and regulatory environments

#	Last cross-org incident that went wrong	Respondent (role / sector)
1	The teams involved had to repeatedly ask several teams what the ticket's status was, because there was no application common to all teams.	team_lead / it_services
2	Payment-platform management.	team_lead / it_services
3	Phased communication over Slack, with no concrete update on the task's status.	architect / banking_insurance
4	Too much time between teams.	other / it_services
5	There was a project handover, and a month later tickets started appearing; as nobody was identified as responsible for the process who could answer my questions, it took much longer to resolve because I had to search within another orga...	operator / it_services
6	Tickets that end up lost, with no clear sense of whose responsibility they are.	architect / it_services
7	Lost or misinterpreted information, lack of accountability.	service_manager / it_services
8	I think it was the ticket's resolution time on the external partner's side; the lack of a quick response caused the internal ticket-resolution process to undergo several changes and medium-to-severe impacts on the organisation.	operator / logistics
9	SAP Support Package upgrade.	team_lead / it_services
10	N/A	architect / it_services
11	In one of the latest cross-organisational incidents, we carried out a critical production deployment involving the internal DevOps, development, operations and functional teams and an external partner. Although the technical deployment was completed without apparent errors, the...	team_lead / other
12	N/A	decision_maker / it_services

Table 3.10: Recent cross-organisational incident responses (translated from Portuguese).

#	What would prevent adoption of a shared platform	Respondent (role / sector)
1	Nothing.	team_lead / it_services
2	I would say it could be costs and some privacy concerns.	team_lead / it_services
3	The complexity of migrating working methods.	architect / banking_insurance
4	All new tools have high costs, both financially and then in training hours. Besides, the whole platform would need a high level of GDPR compliance, because some client matters are very sensitive and we do not want to share data...	operator / logistics
5	I don't know.	team_lead / it_services
6	Cost, security.	other / it_services
7	In my case, we have to use the clients' platforms. But within the client, I think shared use would be possible and beneficial; however, I think the impossibility may be a mix of factors. For HR tickets, data exposure is one of the factor...	service_manager / it_services
8	Perhaps implementation difficulty, financial and resource effort, and the fact that sharing internal information with other organisations is not ideal in consulting.	operator / it_services
9	Mostly cost.	architect / it_services
10	The main factors that could hinder the organisation's adoption of a platform shared with partners would be, above all, data-sovereignty and data-segregation concerns, integration with critical internal systems, contractual responsibilities, and operational governance.	team_lead / other
11	Cost.	architect / it_services
12	High cost.	decision_maker / it_services

Table 3.11: Shared-platform adoption barrier responses (translated from Portuguese).

is not established by this sample and would require targeted multi-region recruitment in a future deployment.

Conclusion validity. The attention-passed cohort is small; the results are read as a directional, industry-side cross-check of the SLR diagnosis rather than a statistically generalisable measurement, and no inferential tests are claimed.

These threats do not invalidate the cross-check the questionnaire provides: they bound it. The instrument's role is to support the SLR's diagnosis with industry-side empirical signal, not to replace the structured in-ecosystem pilot evaluation reported in Chapter 6. Future validations could complement this lightweight instrument with an expert-mediated Delphi study or an interview-based field study, trading breadth for depth and reducing the self-report and single-designer limitations noted above.

3.8 Chapter Summary

This chapter corroborated, from the industry side, the problem diagnosed in the Systematic Literature Review. Practitioners who have never used Flowzao report the same fragmentation symptoms — loss of context, unclear responsibility, absence of end-to-end metrics, and manual re-entry — at intensities clustered towards the upper half of the Likert scale, and rate the corresponding platform capabilities (mapped to R1–R4, R6, R7) as highly important. Subject to the threats to validity above — notably the Portuguese-language concentration and the open, self-selected sample — the questionnaire provides an artefact-independent signal that the problem this dissertation addresses is recognised in practice.

Chapter 4

Proposed Architecture and Domain Model

This chapter presents the proposed architecture and domain model for inter-organisational ticket management. It describes the design knowledge later instantiated in Flowzao, rather than the code-level implementation of that artefact. The proposal is guided by three architectural principles: Domain-Driven Design (DDD) for domain modelling [17, 54], Command Query Responsibility Segregation (CQRS) for separating read and write concerns [18, 57], and event-driven communication in which aggregate state changes produce immutable domain events that are delivered to other bounded contexts via a transactional outbox pattern [29, 42] (state-based persistence with an outbox, not full event sourcing).

Figure 4.1 summarises the proposal at design level. The core idea is that a shared architecture can maintain one canonical inter-organisational ticket, route responsibility through formal events, expose filtered views to each organisation, and derive metrics and reactive subsystems from the same source of truth.

The domain model was developed through an iterative process guided by Event Storming [7], a collaborative workshop technique in which domain events, commands, aggregates, and bounded contexts are discovered through visual exploration. The Event Storming sessions produced structured artefacts (event maps, aggregate boundaries, and cross-context event flows) that directly informed the bounded context decomposition and domain event catalogue presented in this chapter.

The chapter is structured as follows. Section 4.1 presents the core contribution as a transferable pattern. Section 4.2 introduces the architectural principles and patterns that support it. Section 4.3 presents the bounded context decomposition. Section 4.4 details the domain model of the core bounded contexts. Section 4.5 describes the inter-organisational ticket routing mechanism. Section 4.6 presents the gamification subsystem. Section 4.7 describes the orchestration layer. Section 4.8 states the runtime constraints that shape the proposal without specifying the concrete deployment. Section 4.9 maps each design decision to the requirements from Chapter 2.

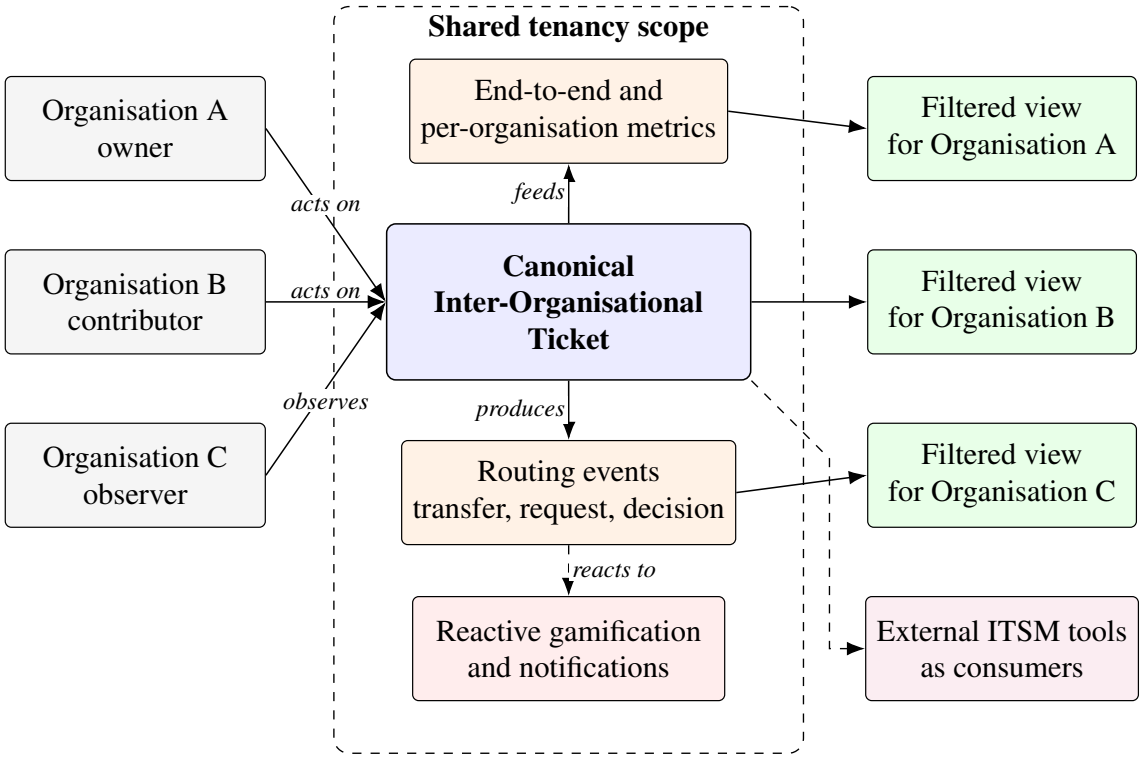


Figure 4.1: Proposed architecture at a glance.

4.1 Core Contribution: The Canonical Inter-Organisational Aggregate Pattern

The architecture presented in this chapter is anchored on a single recurring design move: when an entity must be coordinated across multiple organisations that retain operational autonomy, model it as a *canonical aggregate* owned by a shared platform, with explicit routing of responsibility rather than replication of state. This section abstracts that move from the ITSM design and articulates it as a transferable architectural pattern, enabling its reuse in adjacent inter-organisational coordination domains beyond ITSM.

4.1.1 Intent

Provide a single authoritative representation of an entity whose lifecycle spans multiple autonomous organisations, preserving its identity, history, and end-to-end metrics across organisational boundaries while allowing each organisation to participate, contribute, and observe under organisation-specific visibility rules.

4.1.2 Context and Forces

The pattern applies when the following conditions hold simultaneously:

F1. Multi-party lifecycle. The entity (a ticket, a clinical case, a legal claim, a logistics shipment) is acted upon over time by two or more autonomous organisations whose participation is sequential, overlapping, or both.

F2. Identity preservation across boundaries. Stakeholders need to refer to the entity by a stable identifier regardless of which organisation currently holds operational responsibility; renaming or duplicating the entity at each handoff destroys provenance.

F3. Asymmetric visibility requirements. Different organisations have different rights to see different facets of the entity (commercially sensitive comments, internal annotations, partner-visible status).

F4. End-to-end accountability. A meaningful claim about resolution time, segment time, handoff frequency, or chain-of-responsibility cannot be reconstructed from per-organisation siloed views; it requires a global, ordered timeline.

F5. No single-organisation dominance. No participant has the authority to host the system of record for the others; alternatively, the participants explicitly agree to operate above a shared platform.

These forces create tension with the prevailing pattern in industry, in which each organisation maintains its own system of record and integration is treated as a synchronisation problem (master-data management, message-queue-based ticket forwarding, periodic reconciliation jobs).

4.1.3 Solution Structure

The pattern is constituted by four cooperating elements (Figure 4.2 sketches their relationship):

Canonical Aggregate. A single domain aggregate, owned by a shared multi-tenant platform, that represents the entity. The aggregate has a globally unique identifier that never changes throughout its lifecycle, regardless of which organisation currently holds responsibility. All state mutations go through the aggregate root; all invariants are enforced inside it.

Participation Collection. A child collection on the aggregate that records, for each organisation that has acted upon it, the period of involvement, the role (e.g., *Owner*, *Contributor*, *Observer*), and the user(s) who acted. The collection is append-only at the participation level: ending a participation produces a new entry, never deletes a past one.

Routing Events. Two complementary mechanisms produce immutable domain events on every transfer of responsibility: a *Direct Transfer* for unilateral handoff (the sender alone decides) and a *Consent-Based Transfer Protocol* (Request, Accept/Reject) for handoffs that require the recipient's agreement. Both mechanisms share the same aggregate operations and emit events with full provenance: previous and new participation identifiers, source and target organisations, the user who initiated the transfer, and the optional reason.

Tenancy Scope. A higher-level construct groups organisations that may legally collaborate over instances of the aggregate. Authorisation, visibility, and routing operations are all bounded by tenancy scope; the canonical aggregate does not exist outside one. Permissions within a tenancy scope are fine-grained and per-user, decoupling *which organisation* from *what each user may see*.

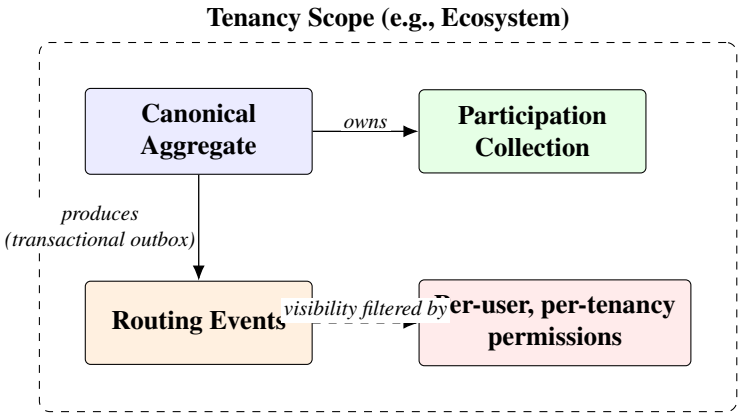


Figure 4.2: Canonical Inter-Organisational Aggregate structure.

4.1.4 Design Guidelines

Use a transactional outbox for routing events. State changes and the events that announce them must be persisted atomically; otherwise the routing log can drift from the canonical state. The outbox pattern as described by Richardson [42] is sufficient and avoids distributed transactions.

Make participations append-only. Ending a participation must record an end timestamp on the existing entry, not delete it. Past participations are part of the canonical history and feed end-to-end metrics.

Distinguish unilateral and consensual transfers. In domains where the recipient must agree (e.g., transferring a clinical case to a specialist), the consent-based protocol is mandatory; in domains where the sender has authority (e.g., emergency reassignment), the direct transfer is appropriate. Real-world domains usually need both.

Keep visibility orthogonal to ownership. Tenancy-scope membership grants the right to see the aggregate's existence; per-user permissions determine which facets are visible. This separation is what allows competing organisations to share a platform without leaking commercially sensitive information.

Resist denormalising the aggregate. CQRS read models, not the canonical aggregate, are the right place for organisation-specific views. The aggregate stays canonical; consumers receive filtered projections.

4.1.5 Consequences

Positive. (i) Identity preservation is structural rather than conventional — the aggregate has a single global identifier from its first transition. (ii) End-to-end accountability is computable from participation timestamps and routing events without cross-system joins. (iii) Adding a new participant organisation requires no schema migration in the participants' systems; they need only join the tenancy scope. (iv) Auditability is intrinsic, not bolted on, because the routing events are the system of record for responsibility. (v) Gamification, analytics, and notifications can subscribe to the routing-event stream without coupling to the canonical aggregate's internals.

Negative. (i) The pattern requires participants to operate above a shared platform; organisations that demand sovereign systems of record cannot be accommodated. (ii) A single canonical aggregate concentrates authority: the platform's availability and integrity are critical infrastructure for all participants, raising governance, security, and uptime requirements above what any single participant would face alone. (iii) Denormalisation pressure on the canonical aggregate is real (consumers want flat, organisation-specific views); resisting it requires discipline. (iv) Authorisation and visibility logic concentrate complexity: every read path must filter by tenancy scope and per-user permissions, and bugs there have higher consequences than in a single-tenant system.

4.1.6 Applicability to Adjacent Domains

This dissertation instantiates the pattern for the inter-organisational ticket domain. The same forces apply, with minor variations, in adjacent domains where similar canonical-versus-replicated debates arise — inter-organisational clinical case management, multi-firm legal matters, cross-carrier logistics, and multi-vendor incident response in cyber security — in each of which the four pattern elements (Canonical Aggregate, Participation Collection, Routing Events, Tenancy Scope) map to the domain with only the role taxonomy, visibility model, and routing-event payload specialised. The motivation for this generality — that the same cross-boundary fragmentation recurs beyond ITSM — is introduced in Chapter 1; the applicability claim is a design-level (analogical) one rather than an empirically validated result. The full pattern card, including the per-domain mapping of each element, is provided in Appendix E.

4.1.7 Anti-Patterns and Variations

Anti-pattern: Master-Slave Synchronisation. The prevailing industry alternative — treating one organisation's system as the master and replicating copies into the partners' systems — is the core gap analysed in Chapter 2 (Section 2.3.2); in the pattern's terms it violates F2 (identity preservation) and F4 (end-to-end accountability), since each replica carries its own identifier and conflict-resolution rules while timestamps and provenance are reconciled rather than authoritative. The full contrast is recorded in the pattern card in Appendix E.

Variation: Federated Canonical Aggregate. For domains where no single platform can host the canonical aggregate (e.g., regulatory or sovereignty constraints), a federated variant locates the aggregate at one of the participating organisations and uses notarised, append-only events shared via a verifiable log (blockchain or trusted notary) to maintain provenance. The pattern's core invariants (single global identifier, append-only participations, immutable routing events) are preserved; only the hosting model changes. A fully decentralised realisation of this variant is a *decentralised autonomous organisation* (DAO), in which the routing rules and participation logic are encoded as smart contracts over a shared ledger, removing the need for any single organisation to host the canonical aggregate; this trades the governance simplicity and low latency of a central host for the additional complexity, cost, and throughput constraints of on-chain coordination. This variation is outside the scope of this dissertation but provides a natural follow-up direction (Section 7.3).

Variation: Read-Side Canonical View. An intermediate option, in which each participant retains a sovereign system of record but agrees to publish a normalised event stream into a shared read-side canonical view. This is weaker than the full pattern (mutations are still per-organisation) but recovers (i) end-to-end accountability and (ii) unified analytics. It can be a stepping stone for organisations not yet ready to delegate write authority to a shared platform.

4.1.8 Pattern Summary

The Canonical Inter-Organisational Aggregate is the design knowledge crystallised by this dissertation. It is a reusable solution to a recurring problem in inter-organisational coordination. The architecture and domain model documented in this chapter express the pattern for the ITSM domain; the next chapter shows how Flowzao instantiates it as a functional proof of concept, and Chapter 6 evaluates that instantiation in real workflow conditions.

4.2 Architectural Principles and Patterns

4.2.1 Hexagonal Architecture

The proposed architecture follows a hexagonal (ports and adapters) architecture as described by Cockburn [10], in which each bounded context is organised into three concentric layers:

Domain layer: contains aggregates, entities, value objects, domain events, and domain policies. This layer has no dependencies on infrastructure or application concerns.

Application layer: contains use cases (commands and queries) and port interfaces (inbound and outbound). Use cases orchestrate domain logic and interact with the outside world exclusively through ports.

Infrastructure layer: contains adapters that implement the outbound ports (repositories, email services, file storage, queue publishers) and the inbound presentation layer (HTTP handlers).

Figure 4.3 illustrates this layering for a single bounded context.

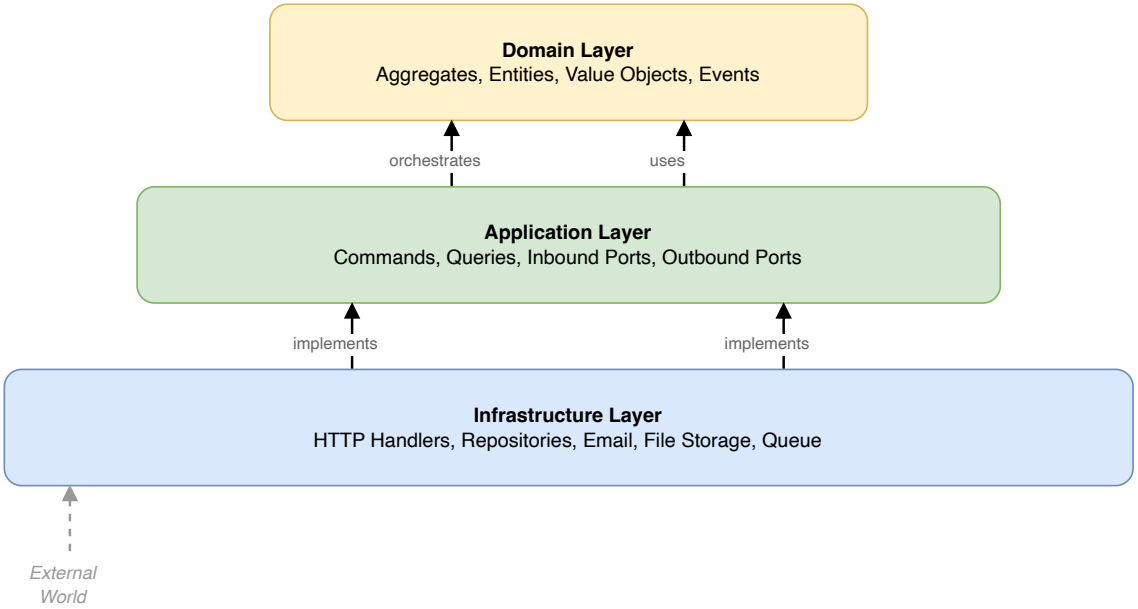


Figure 4.3: Hexagonal architecture within a bounded context.

4.2.2 Domain-Driven Design

The proposed model applies both strategic and tactical DDD patterns [17]:

- **Strategic design:** the domain is decomposed into bounded contexts, each with its own ubiquitous language, model, and consistency boundary.
- **Tactical design:** within each bounded context, the domain is modelled using aggregates (consistency boundaries), entities (objects with identity), value objects (immutable, self-validating types), and domain events (records of significant state changes).

Aggregates enforce all business invariants. No external component may modify an aggregate's state directly; all mutations occur through methods on the aggregate root, which validate invariants and produce domain events.

4.2.3 CQRS and Event-Driven Communication

The write side of each bounded context is modelled as command handlers that mutate aggregates and produce domain events. The read side is specified as projections, that is, denormalised views optimised for specific query patterns. This separation enables organisation-specific read models that present the same underlying ticket data through views tailored to each organisation's visibility rules, directly supporting requirement R3 (multi-level visibility).

Domain events serve as the primary integration mechanism between bounded contexts. When an aggregate produces an event, it is persisted atomically alongside the aggregate state change via

a transactional outbox pattern [42]. The event is then published asynchronously to subscribers in other bounded contexts, enabling loose coupling and eventual consistency.

This design deliberately adopts state-based persistence with a transactional outbox rather than full event sourcing, in which aggregate state is reconstructed entirely by replaying stored events. Full event sourcing can introduce additional operational concerns, including event-store design, replay performance, snapshot strategies, and event-version evolution [54]. The primary benefits required from domain events are an audit trail of state changes, cross-context integration, and the ability to trigger reactive workflows. The transactional outbox pattern delivers these capabilities because every domain event produced by an aggregate mutation is persisted reliably and published to consumers, without imposing the infrastructure burden of event replay. This constitutes a pragmatic trade-off: the architecture retains the option to migrate towards full event sourcing in the future, because all aggregate mutations already emit domain events as first-class artefacts, but such a migration would still require changes to persistence, aggregate rehydration, and projection code.

4.3 Bounded Context Decomposition

The domain is decomposed into seven bounded contexts, as illustrated in the context map in Figure 4.4. Solid arrows represent synchronous command invocations via the orchestration layer; dashed arrows represent asynchronous domain-event flows.

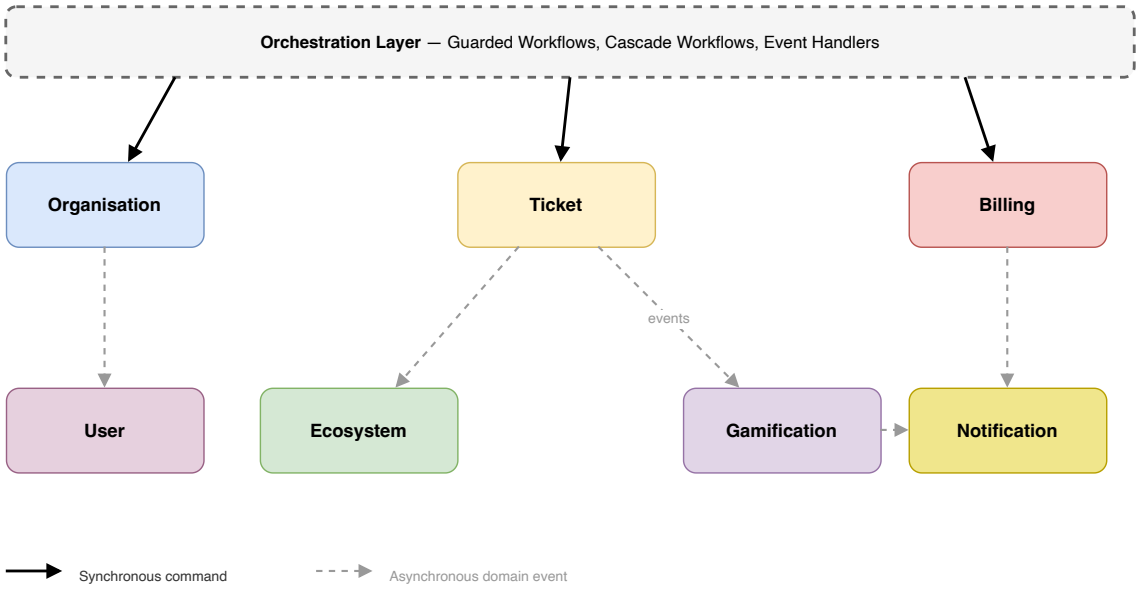


Figure 4.4: Bounded context map of the proposed architecture.

The bounded contexts relate through four integration patterns, summarised in Table 4.1:
Each bounded context is responsible for a distinct subdomain:

User: manages user identity, authentication through one or more identity providers, and user preferences. Its events capture identity, sign-in, and preference changes.

Relationship	Pattern	Mechanism
Organisation → Ecosystem	Customer-Supplier	Cascade workflows propagate membership changes
Ticket → Gamification	Event-Driven	Ticket lifecycle events trigger point and achievement evaluation
All BCs → Notification	Conformist	Notification reacts to domain events from all contexts
Billing → All BCs	Guard	Cross-cutting guard enforces subscription limits before resource creation

Table 4.1: Integration patterns between bounded contexts.

Organisation: manages organisations, memberships with four roles (Owner, Admin, Manager, Technician), and token-based invitations with configurable expiry. Enforces invariants such as single ownership and last-owner protection. Its events capture organisation creation, membership changes, role changes, invitation lifecycle, and ownership transfer.

Ecosystem: manages multi-organisation collaboration spaces. Each organisation's membership carries a role and a category-based permission grant covering ticket lifecycle operations, collaboration, routing, administration, and reporting. Table 4.2 summarises these permission categories. Its events capture ecosystem creation, organisation membership changes, and access changes.

Category	Examples	Purpose
Ticket lifecycle operations	Create, edit, assign, resolve, close	Govern mutations on the shared ticket record
Collaboration actions	Comment, attach files, track time, manage watchers	Govern collaborative work around the ticket
Routing and escalation	Transfer, request transfer, escalate	Govern cross-organisation responsibility changes
Administration and reporting	Manage services, technicians, reports	Govern workspace configuration and metrics access

Table 4.2: Ecosystem permission categories.

Ticket: the core bounded context. Manages the full ticket lifecycle, including status transitions, assignments, comments, attachments, time logs, feedback, transfer requests, and post-resolution documentation. It is intentionally agnostic of ecosystem and organisation structure. Its events form the core lifecycle and routing event catalogue.

Billing: manages the subscription lifecycle across plan tiers. Each plan defines resource limits that are enforced through cross-cutting guards in the orchestration layer (Section 4.7). The subscription aggregate models entitlement states independently of any payment provider. Its events capture subscription creation, plan updates, and payment-state changes.

Gamification: manages achievements, rewards, member profiles, and reward redemptions. Triggered reactively by ticket lifecycle events (Section 4.6). Its events capture point attribution, achievement unlocking, and reward-redemption state changes.

Notification: manages notification dispatch and user preferences. Subscribes to domain events across other bounded contexts, rendering and delivering notifications through configured channels. Users can configure per-type notification preferences. Its events capture dispatch, delivery, and failure states.

Table 4.3 summarises the domain-event responsibilities assigned to each bounded context. The complete implementation catalogue is reported separately in Appendix A.

Bounded Context	**Representative event responsibilities**
Ticket	Ticket creation, status changes, assignment changes, direct transfers, transfer requests, transfer acceptance or rejection, resolution, feedback
Billing	Subscription creation, plan changes, payment success or failure, cancellation, reactivation
Ecosystem	Ecosystem creation, organisation membership changes, invitation lifecycle, user access grants and revocations
Gamification	Point attribution, achievement unlocking, reward creation, redemption request, redemption approval or rejection
Organisation	Organisation creation, member addition or removal, role changes, invitation lifecycle, ownership transfer
User	User creation, sign-in, preference changes
Notification	Notification dispatch, delivery, failure, preference updates

Table 4.3: Domain-event responsibilities across bounded contexts.

4.4 Domain Model

Figure 4.5 presents the core domain model, focusing on the relationships between the principal aggregates. Aggregates are shown in bold, child entities are shown under their aggregate roots, and dashed references indicate cross-context relationships.

4.4.1 Organisation Aggregate

The Organisation aggregate manages the lifecycle of an organisation, its memberships, and its invitation system. The principal business invariants are summarised in Table 4.4 (rows O-1 to O-4); the aggregate produces fourteen domain events covering creation, member addition and removal, role changes, invitation lifecycle, and ownership transfer.

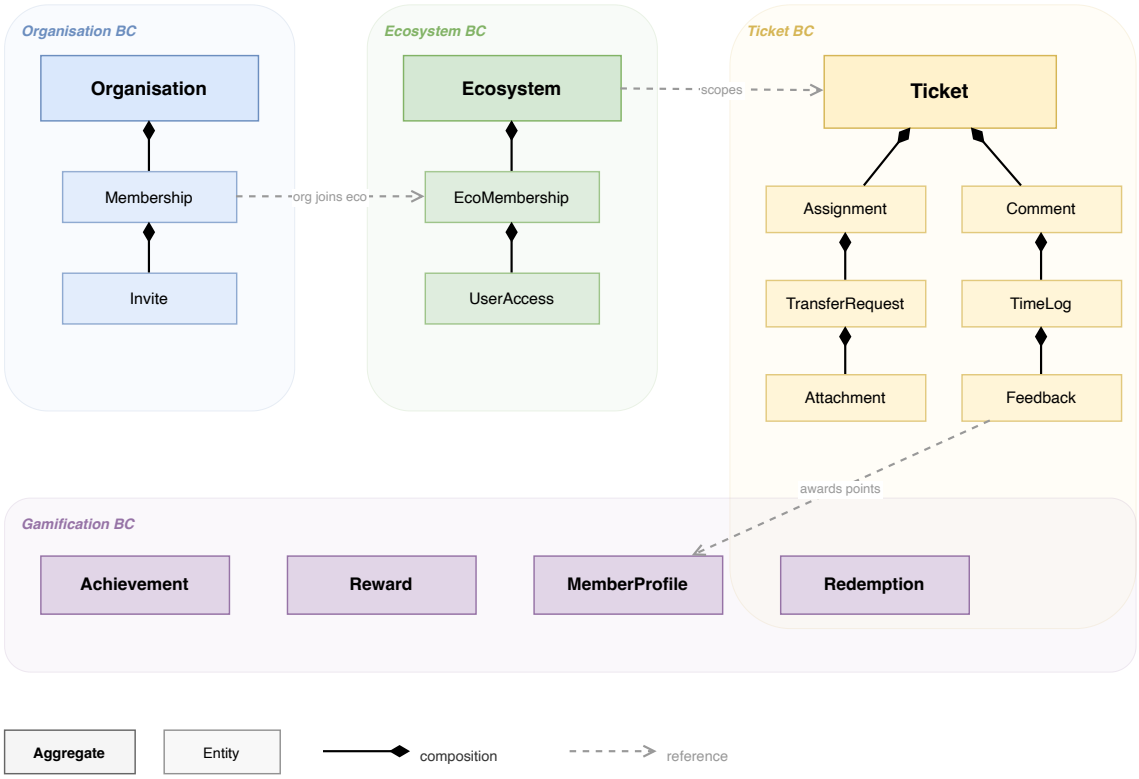


Figure 4.5: Core domain model.

4.4.2 Ecosystem Aggregate

The Ecosystem aggregate models a collaboration space in which multiple organisations interact. When an ecosystem is created, the creating organisation automatically receives an owner membership; thereafter, additional organisations join through an invitation flow analogous to the organisation-level invitation system but scoped to the ecosystem. Each member organisation may then grant user-level access to individual users, using the permission categories described in Table 4.2. The principal Ecosystem-level invariants are summarised in Table 4.4 (rows E-1 to E-3); the aggregate produces events for membership and access changes.

4.4.3 Billing Aggregates

The Billing bounded context manages plan and subscription concepts. The **Plan** aggregate defines subscription tiers and their resource limits, while the **Subscription** aggregate tracks each organisation's entitlement state over time. External payment-provider events may drive state transitions, but the domain model treats them as stimuli translated into subscription commands rather than as provider-specific concepts.

A cross-cutting plan-limit guard, specified in the orchestration layer (Section 4.7), queries the active subscription before resource creation operations such as adding members, creating ecosystems, or creating tickets. Centralising this guard keeps plan enforcement out of the core aggregates while preserving a single source of truth for entitlement-related authorisations.

Code	Invariant	Description
O-1	Single owner per organisation	At most one membership may hold the Owner role at any time.
O-2	Last-owner protection	Ownership must be explicitly transferred before the current owner can be removed.
O-3	Single membership per user	Each user may hold at most one active membership per organisation.
O-4	Invitation uniqueness and expiry	At most one pending invitation may exist per email per organisation; invitations expire after a configured period.
E-1	Founding membership	The organisation that creates an ecosystem automatically receives an Owner membership; ecosystems cannot exist without at least one member organisation.
E-2	Permission scoping	A user's effective permissions are the intersection of the membership-level grant and the user-level access entry; absence of either denies access.
E-3	Ecosystem-scoped routing	A ticket may only be transferred to an organisation that holds an active membership in the same ecosystem.

Table 4.4: Organisation and Ecosystem invariants.

4.4.4 Ticket Aggregate

The Ticket aggregate is the central domain entity of the proposed architecture and the most complex aggregate in the system, with eight child entity collections and a seven-state lifecycle. Despite this breadth, it is deliberately kept as a single consistency boundary [17]: every child collection (assignments, comments, time logs, resolution history) must be mutated under the same invariants that protect the canonical ticket, so splitting them into separate aggregates would fragment exactly the consistency the design exists to guarantee.

4.4.4.1 Status State Machine

The ticket status follows a state machine with seven states, illustrated in Figure 4.6.

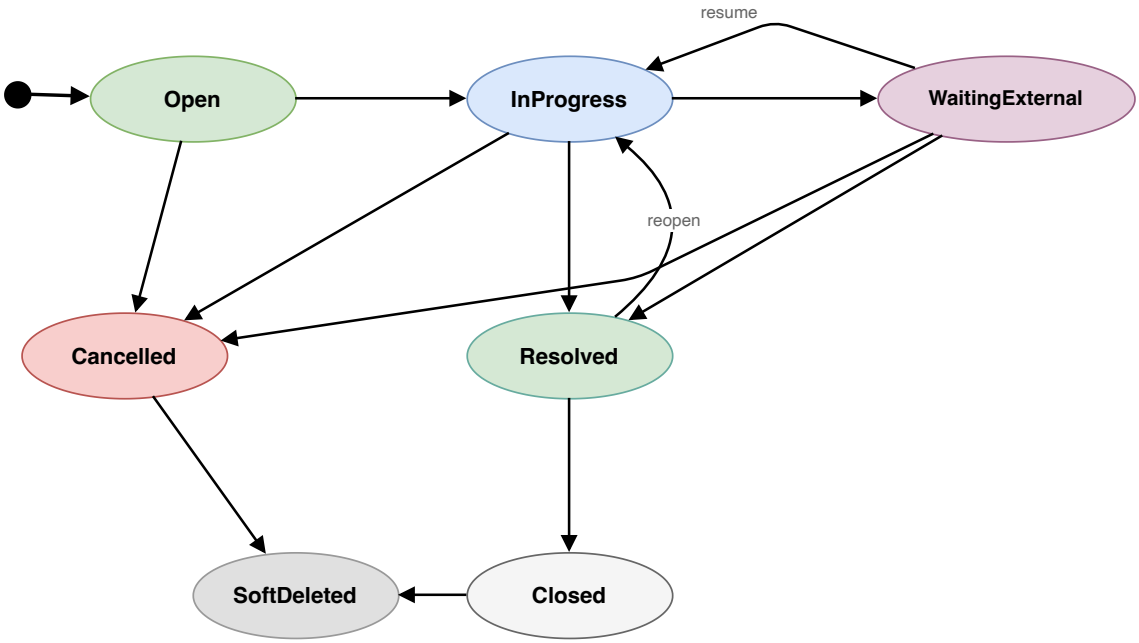


Figure 4.6: Ticket status state machine.

The `Open` state is the initial state, and `SoftDeleted` is terminal. A resolved ticket may be reopened by transitioning back to `InProgress`. The status transition rule is modelled as a value object that encodes the valid transitions. Any attempt to perform an invalid transition raises a domain exception. Resolved and closed states record lifecycle timestamps; reopening clears the resolution timestamp.

4.4.4.2 Child Entities

The Ticket aggregate owns eight child entity collections:

Assignments: represent ownership and participation. Each assignment links a ticket to an ecosystem membership with a type: *Owner* (at most one active per ticket) or *Contributor* (multiple allowed). The aggregate enforces last-owner protection.

Comments: text contributions with a visibility level (Internal, CustomerVisible, PartnerVisible). Only the author may edit or delete.

Attachments: files uploaded to cloud storage, optionally linked to a specific comment.

Time logs: time tracking entries categorised by type (Active Work, Waiting, Research, Communication), recording time per user per organisation.

Feedback: post-resolution satisfaction entries, one per assignment, each recording a rating and gamification points awarded.

Transfer requests: formal cross-organisation transfer requests with a three-state lifecycle (Pending, Accepted, Rejected). At most one may be pending at any time.

Resolution attachments: files bound specifically to the ticket's resolution (e.g., proof-of-fix screenshots, log excerpts, screen recordings). Soft-deletable so removed artefacts remain in the audit history.

Resolution history: immutable per-version snapshots of the resolution note. Each `resolve()` or post-resolution edit appends a new entry, preserving the full provenance of how the resolution was authored and refined over time.

4.4.4.3 Business Invariants

The Ticket aggregate enforces a set of business invariants that protect the consistency of the ticket throughout its lifecycle. Table 4.5 catalogues the principal invariants. Each invariant is enforced within the aggregate root; any violation raises a typed domain exception, ensuring that invalid state can never be persisted.

4.5 Inter-Organisational Ticket Routing

The inter-organisational ticket routing mechanism is the core design contribution, directly addressing requirements R1 (canonical ticket model) and R2 (formal routing events). The design materialises the concept of structured routing: a ticket retains a single identity throughout its lifecycle, regardless of which organisation currently holds responsibility.

4.5.1 Routing Model

Two routing mechanisms are available. Both operate within the scope of a single ecosystem and may move a ticket across organisational boundaries; they differ in whether the receiving organisation must explicitly consent at the moment of transfer:

Direct transfer: a unilateral, one-step operation that ends the current Owner assignment and creates a new Owner assignment for the target organisation. It is intended for scenarios in

Invariant	Description
Status state machine	Only valid transitions are permitted (e.g., Open → InProgress, but not Open → Closed).
Terminal state immutability	Tickets in terminal states (Closed, Cancelled, SoftDeleted) cannot be modified.
Last-owner protection	The sole remaining Owner assignment cannot be removed.
Single active owner	At most one active Owner assignment per ticket at any time.
Single pending transfer	At most one transfer request may be in Pending state at any time.
Author-only modification	Comments and feedback can only be edited or deleted by their author.
One feedback per assignment	Each assignment may receive at most one feedback entry.
Attachment constraints	Uploaded files must satisfy configurable size and type restrictions.

Table 4.5: Principal business invariants enforced by the Ticket aggregate.

which consent has been established a priori (for example, through ecosystem governance rules, automated dispatch policies, or trusted-partner arrangements), or for self-corrective transfers within the same organisation.

Transfer request: a multi-step consent-based protocol. The current owner creates a request, the target organisation explicitly accepts or rejects, and ownership changes only upon acceptance. This is the default mechanism for cross-organisational transfers where partner consent is required at the point of handoff.

The choice between the two is made by the initiating user based on the operating context; both produce equivalent provenance through immutable domain events, ensuring that audit and analytics treat the two paths uniformly.

Figure 4.7 illustrates the consent-based transfer request protocol, which is the more complex of the two. Both mechanisms produce explicit domain events that capture the full routing history. A direct transfer emits one transfer event. The consent-based protocol emits request and decision events, recording whether the recipient accepted or rejected the handoff.

4.5.2 Ecosystem as Routing Scope

All ticket routing occurs within the scope of an ecosystem. An organisation can only receive a ticket if it holds an active membership in the ecosystem to which the ticket belongs. Which users may initiate transfers is governed by the `TransferTickets` permission.

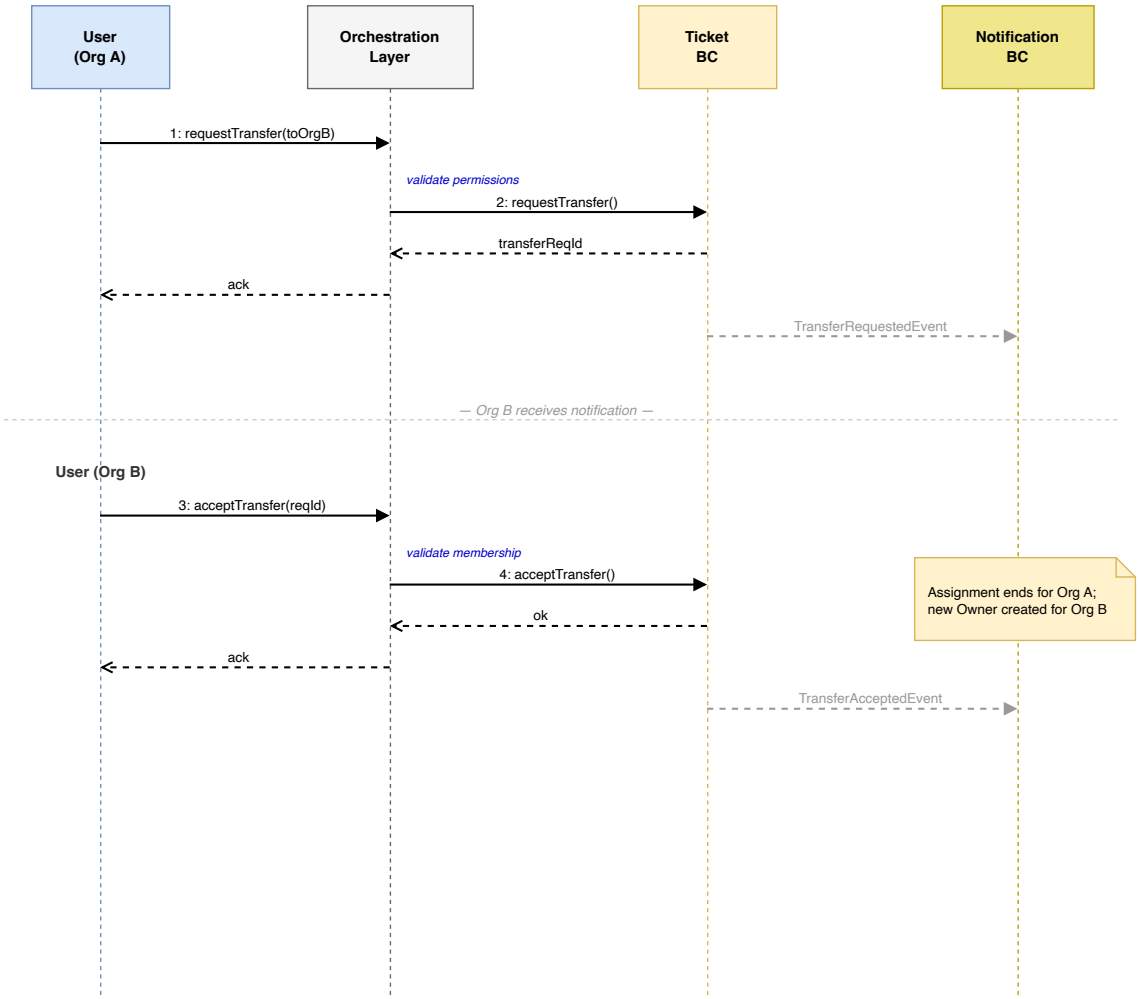


Figure 4.7: Consent-based transfer request protocol.

4.5.3 End-to-End Traceability

Because every assignment change and transfer event is captured as an immutable domain event persisted via the transactional outbox, the architecture maintains an append-only audit trail for the lifetime of each ticket. This trail supports: reconstruction of the full chain of responsibility transfers; computation of time-per-organisation metrics; identification of "ticket ping-pong" patterns; and end-to-end resolution time measurement. The proposed retention rule is that routing events are preserved for the lifetime of the parent ticket, so auditability is not lost when operational views are filtered or projected.

4.6 Gamification Subsystem Design

The gamification subsystem addresses RQ2 by integrating game-design elements into the ticket lifecycle, grounded in Self-Determination Theory [14].

4.6.1 Domain Model

The Gamification bounded context manages four aggregates:

Achievement: defines achievements with rarity (Common to Legendary), point values, and typed triggers (ResolveTickets, DayStreak, CsatScore).

Reward: defines redeemable rewards with categories (Gift Card, Time Off, Merchandise, etc.) and optional stock limits.

Member Profile: tracks points, streak, and earned achievements per ecosystem member.

Reward Redemption: manages claim lifecycle (Pending → Approved/Rejected → Fulfilled).

4.6.2 Integration with the Ticket Lifecycle

Gamification events are triggered reactively through domain events, as shown in Figure 4.8.

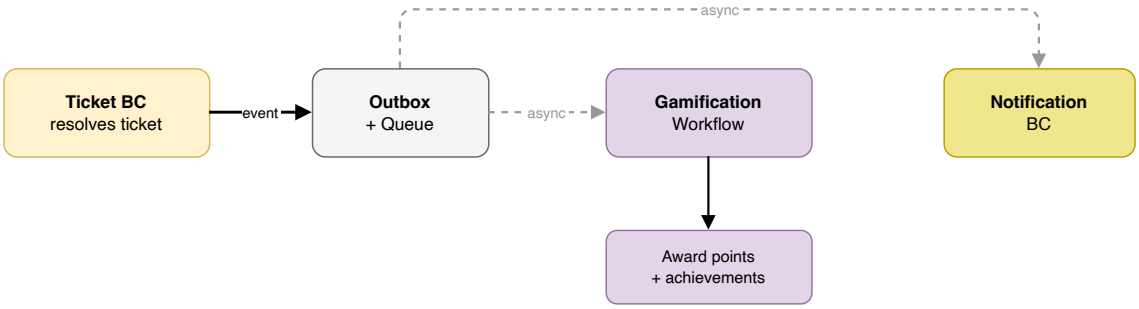


Figure 4.8: Gamification integration flow.

The Ticket bounded context remains unaware of the gamification subsystem; the coupling is purely through domain events. The feedback mechanism provides the primary point distribution: when feedback is submitted for a resolved ticket, points are calculated using a story-point Fibonacci weighting based on the ticket's estimated effort:

$$points = w(size) \times rating \tag{4.1}$$

where w maps each T-shirt size to the corresponding value in the modified Fibonacci sequence commonly used in agile story-point estimation (XS = 1, S = 2, M = 3, L = 5, XL = 8) and *rating* is the customer satisfaction score (1–5). When no rating is supplied, the multiplier defaults to 1. The product structure rewards both effort (larger tickets earn more) and quality (higher-rated resolutions earn more), counter-balancing the perverse incentive of closing many small tickets quickly at the expense of complex ones.

4.6.3 Achievement Trigger Types

The gamification subsystem defines six achievement trigger types, each evaluating a different aspect of technician performance:

`ResolveTickets`: awards an achievement after resolving a threshold number of tickets.

`ResolveInTime`: awards an achievement for resolving tickets within their due date.

`DayStreak`: awards an achievement for maintaining consecutive days of activity.

`ResolvePriority`: awards an achievement for resolving high-priority tickets.

`CsatScore`: awards an achievement for maintaining a minimum customer satisfaction rating.

`ResolveAfterHours`: awards an achievement for resolving tickets outside regular hours.

Each trigger type maps to specific SDT needs: `ResolveTickets` and `DayStreak` address *competence* through mastery progression; `CsatScore` addresses *relatedness* through quality of collaboration; and the variety of trigger types addresses *autonomy* by allowing organisations to configure which achievements are active, enabling technicians to pursue recognition through different paths.

4.6.4 Design for Fairness

Following the principles identified in Section 2.4 and the risks catalogued in the dark patterns literature [25, 53], the design incorporates explicit safeguards against coercive or unfair dynamics:

- **Configurable triggers:** achievement triggers are configurable per organisation, allowing each entity to define thresholds appropriate to its context and avoiding the imposition of one-size-fits-all targets.

- **Ecosystem-scoped leaderboards:** leaderboards are scoped to ecosystems rather than being global, preventing unfair comparisons between organisations with different workload volumes.
- **Approval-based redemption:** reward redemption requires explicit managerial approval, preventing gaming of the points system.
- **Permission-controlled visibility:** performance data visibility respects ecosystem permission boundaries via the `ViewReports` permission, ensuring that commercially sensitive performance metrics are not exposed to competitors.
- **"White hat" emphasis:** following the Octalysis framework [9] discussed in Section 2.4, the design favours white-hat core drives (development, empowerment, social influence) over black-hat drives (scarcity, loss avoidance), with streaks being the only element that involves loss aversion.

4.7 Orchestration Layer

The orchestration layer coordinates cross-context operations through three mechanisms, as depicted in Figure 4.9.

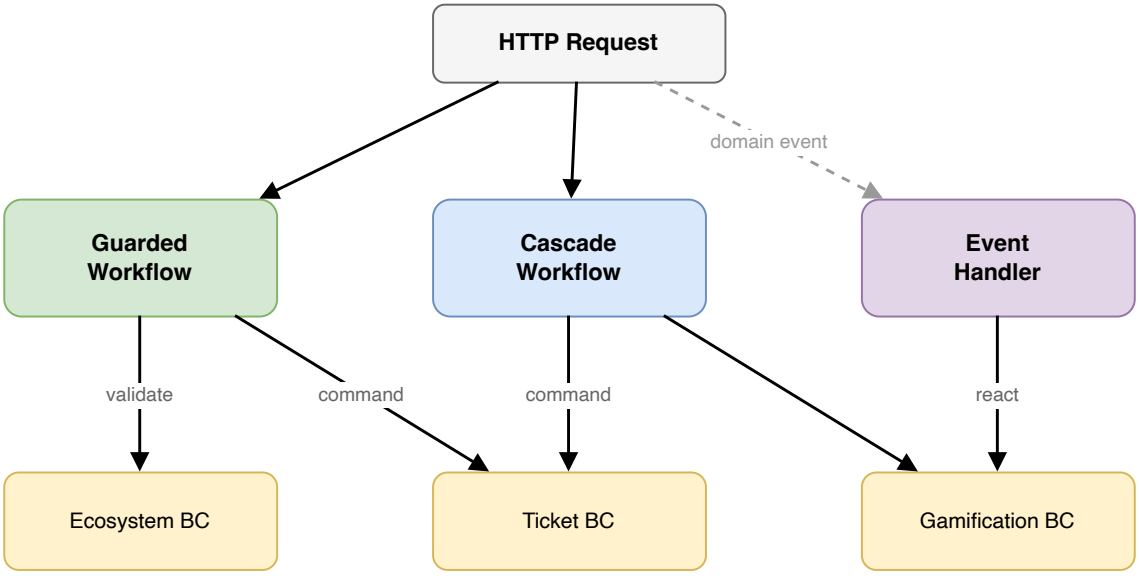


Figure 4.9: Orchestration layer mechanisms.

4.7.1 Guarded Workflows

Guarded workflows enforce invariants that span multiple bounded contexts. The key challenge is that a single user action (e.g., creating a ticket) may require validation in several contexts: the user must hold the `CreateTickets` permission in the Ecosystem context, the organisation's

subscription must not have exceeded its ticket quota in the Billing context, and the ticket data must satisfy domain invariants in the Ticket context. The guarded workflow pattern addresses this by performing all cross-context validations *before* initiating a single atomic transaction in the target bounded context. This avoids distributed transactions while preserving correctness.

4.7.2 Cascade Workflows

Cascade workflows propagate a state change in one bounded context to dependent contexts. The canonical example is organisation deactivation, which executes the following ordered steps: (i) end all active ecosystem memberships held by the organisation, (ii) revoke pending invitations issued by or addressed to the organisation, (iii) suspend gamification member profiles for users belonging to the organisation, and (iv) mark notification preferences inactive. Each step is idempotent; the cascade is executed within a single unit of work, so if any step fails, the entire operation is rolled back and the organisation remains in its pre-deactivation state. This pattern keeps cross-context state changes consistent without requiring a distributed transaction protocol.

4.7.3 Event-Driven Reactions

Asynchronous event handlers react to domain events independently of the command that produced them. When a ticket is resolved, for instance, the resolution event triggers both a gamification reaction (awarding points) and a notification reaction (notifying the requester), each operating independently. Failures in one handler do not affect others, and retry logic ensures eventual delivery through the outbox subscriber tracking mechanism.

4.7.4 Cross-Cutting Guards

A guard is a stateless decision component that intercepts an inbound command, queries one or more bounded contexts to evaluate a cross-cutting precondition, and either authorises the command or raises a typed exception that the workflow translates into a domain error. Guards are invoked by the guarded workflows defined earlier in this section, before any aggregate state is mutated, ensuring that cross-context invariants are enforced without distributed transactions and without leaking the invariants into individual aggregates.

The principal guard used in the proof-of-concept instance enforces subscription limits across the proposed system. When a workflow attempts to create a resource (a new organisation member, a new ecosystem, a new ticket), the workflow consults the guard with the requesting organisation's identifier and the resource type. The guard resolves the active subscription, retrieves the associated plan limits, and compares them against the current usage; if the limit would be exceeded, it raises a domain-level rejection. Centralising this logic in a single guard avoids duplicating subscription checks in each bounded context and ensures a single source of truth for plan-related authorisations.

4.8 Runtime Constraints

The proposed architecture assumes three runtime constraints. First, application services should be stateless, so that command and query handlers can scale independently without storing workflow state in process memory. Second, persistence must support ACID transactions, because aggregate state changes and outbox-event insertion must be committed atomically. Third, asynchronous event delivery must be durable and retryable, since notifications, gamification, and cross-context projections depend on reliable delivery without coupling them to the initiating command.

These constraints are architectural rather than vendor-specific: they shape the use of a transactional outbox, queue-backed event delivery, and CQRS projections, but they do not require a particular cloud provider. The concrete technology stack and deployment topology used by the proof-of-concept instance are implementation concerns and are reported in Chapter 5 (Section 5.1).

4.9 Design Traceability

Table 4.6 maps each requirement from Chapter 2 to the architectural mechanisms that address it. R5 is marked separately because composite SLA management was descoped from the implemented proof of concept; the architecture only provides the data foundations required for future SLA decomposition.

4.10 Architectural Decision Summary

For transparency and to support future extension or replication, Table 4.7 consolidates the principal architectural decisions taken during the design of the proposed architecture, the alternatives that were considered, and the rationale for the chosen option. Each decision is cross-referenced to the chapter section where it is discussed in depth.

4.11 Design Limitations and Trade-offs

The architecture makes deliberate trade-offs that bound its applicability; this section makes them explicit.

Single-writer contention. The canonical ticket is a single aggregate with one authoritative writer at a time. This is precisely what guarantees a consistent inter-organisational record (Section 4.1), but it concentrates write load: high-frequency concurrent updates to the same ticket serialise on the aggregate, and all of its state changes flow through a single transactional-outbox subscriber. For the coordination workloads targeted here — where a ticket is held by one organisation at a time — this is rarely a bottleneck, but a domain with many concurrent writers per aggregate would require partitioning the aggregate or relaxing its consistency boundary. The system uses state-based persistence with a transactional outbox, not full event sourcing [19]; the

Req.	Description	Design Mechanism
R1	Canonical inter-organisational ticket model	Single Ticket aggregate with global identity; assignments link tickets to organisations within ecosystems (Section 4.4.4).
R2	Formal routing events	Transfer requests and assignment changes produce immutable domain events with full provenance (Section 4.5).
R3	Multi-level visibility	CQRS read models with organisation-scoped projections; ecosystem permissions control data access (Sections 4.2, 4.3).
R4	End-to-end and segment metrics	Assignment timestamps enable per-organisation and end-to-end resolution time computation (Section 4.5).
R5*	Composite SLA management	Descoped to future work (see Chapter 1, Research Methodology). The architecture provides the data foundations: per-assignment timestamps and time logs are already captured in the Ticket aggregate (Section 4.4.4), enabling future composite SLA decomposition.
R6	Multi-tenant architecture	Bounded context decomposition with ecosystem-scoped data; shared infrastructure with logical isolation (Section 4.3).
R7	Single Source of Truth	All ticket state in a single canonical aggregate; projections are derived from the same source data (Sections 4.2, 4.4.4).

Table 4.6: Requirements-to-design traceability.

Decision	Alternatives considered	Rationale
Hexagonal architecture per BC (§4.2)	Layered *n*-tier; Clean Architecture	Explicit ports decouple domain from infrastructure, enabling test isolation and the substitution of adapters (e.g., the queue substrate) without domain change.
Modular monolith with bounded contexts (§4.3)	Microservices per BC; CRUD monolith	Preserves DDD boundary clarity without the operational and consistency costs of distributed deployment, as analysed in Chapter 2.
Transactional outbox over full event sourcing (§4.2)	Full event sourcing with replay	Delivers audit trail and reactive integration without the operational burden of an event store, snapshots, and event upcasting.
Domain events as inter-context contract (§4.2)	Direct cross-BC method calls	Loose coupling between contexts; subscribers evolve independently; failures isolated to consumer.
Single canonical Ticket aggregate vs. replication (§4.5)	Per-organisation ticket replicas with synchronisation	Directly satisfies R1 and R7; eliminates replication drift; supports end-to-end metrics.
Two routing mechanisms (direct + request) (§4.5)	Single consent-based protocol; single direct mechanism	Accommodates both pre-arranged routing rules and ad hoc consent flows while keeping a uniform event provenance.
Story-point Fibonacci weighting for points (§4.6)	Linear weighting; constant points per ticket	Rewards both effort and quality; counter-balances the speed-over-complexity perverse incentive.
Cross-cutting guards in orchestration (§4.7.4)	Per-BC duplicated checks	Single source of truth for plan limits and other cross-context preconditions.
Serverless-compatible runtime constraints (§4.8)	Stateful application services; in-process pub/sub	Stateless services, durable event delivery, and ACID-backed outbox persistence preserve scalability and auditability without binding the design to a specific deployment vendor.

Table 4.7: Architectural decision summary.

limitation is therefore write contention on the aggregate and its outbox, not the rebuild cost of an event store.

Eventual consistency across bounded contexts. Integration between bounded contexts is event-driven and therefore eventually consistent [29]. Read models and cross-context reactions — for example, the plan-limit guard reading billing usage, or the gamification context reacting to ticket events — may observe slightly stale state in the interval between an event's emission and its projection. The design accepts this in exchange for decoupling and availability; it is appropriate for the coordination domain but would be unsuitable where a strongly consistent cross-context invariant is required.

Absence of a formal saga. The orchestration layer (Section 4.7) composes multi-step workflows within a single database transaction, so a failure rolls the database changes back atomically. This guarantee does not extend to effects outside that transaction — external payment-provider calls, message dispatch, or e-mail notifications — which are not compensated automatically. A system spanning more external side effects would need an explicit saga with compensating actions [42]; the current design deliberately keeps cross-context workflows small enough to avoid that complexity, and this is identified as future work.

Modelling complexity versus CRUD. Domain-driven design, CQRS, and the transactional outbox introduce structure — aggregates, value objects, separate read models, event handlers — that a simple CRUD application would not carry (Section 4.10). The cost is justified here by the need for an auditable, canonical, multi-organisation record, but it is a real cost: the model is more demanding to learn, extend, and operate than a single relational schema with direct updates.

Centralised authority. The canonical aggregate is hosted on a shared central platform (force F5 in Section 4.1). This is precisely what makes end-to-end identity and accountability structural rather than negotiated, but it concentrates governance, availability, and integrity in a single operator, and it presupposes that the participating organisations accept a shared host. Ecosystems without a mutually trusted host — or operating under data-sovereignty constraints — may not accept this arrangement; the decentralised (DAO) and federated realisations noted in Section 4.1 relax the constraint at the cost of substantial coordination complexity, and are revisited as future work (Section 7.3).

4.12 Chapter Summary

This chapter presented the proposed architecture and domain model for inter-organisational ticket management. The proposal is decomposed into seven bounded contexts following Domain-Driven Design, with the Ticket aggregate serving as the core domain entity and as the consistency boundary that preserves canonical ticket identity across organisational handoffs.

The central design decision is the treatment of the inter-organisational ticket as a canonical entity that is routed, not replicated. Two routing mechanisms (direct transfer and transfer request protocol) produce immutable domain events that maintain a complete chain of responsibility. The gamification subsystem integrates through domain events, reacting to ticket lifecycle changes

without coupling to the core ticket logic. Fairness safeguards (ecosystem-scoped leaderboards, configurable triggers, approval-based redemption) address the risks identified in Chapter 2. A traceability analysis shows how the in-scope requirements (R1–R4, R6, R7) are addressed, while R5 is limited to data foundations for future work. Section 4.1 formalises the underlying design move as the *Canonical Inter-Organisational Aggregate* pattern, articulating its forces, structure, and applicability to adjacent inter-organisational coordination domains. The implementation of this architecture as Flowzao is presented in the following chapter.

Chapter 5

Implementation

This chapter summarises the implementation of Flowzao as the proof-of-concept instance of the architecture proposed in Chapter 4. Its purpose is not to re-present the design contribution, but to show that the proposed Canonical Inter-Organisational Aggregate, routing semantics, visibility model, and gamification integration were materialised in a working system. Detailed domain-event and persistence catalogues are provided in Appendices A and D; this chapter keeps only the implementation evidence needed to support the evaluation in Chapter 6.

5.1 Technology Stack

Table 5.1 summarises the stack used by the proof of concept. The choices are relevant only insofar as they realise the architectural constraints stated in Section 4.8: stateless request handling, transactional persistence, durable event delivery, and typed client-server contracts.

Flowzao was deployed on this stack across development, QA, and production environments, each with separate configuration and database branches, providing the operational availability required for the evaluation, with observability through structured request logs and outbox subscriber-execution records.

5.2 Implementation Boundary

The proof of concept separates backend domain logic, presentation code, shared contracts, and deployment adapters. This separation is relevant because it mirrors the hexagonal architecture described in Chapter 4: the domain model is not embedded in the user interface or in cloud-provider adapters, and implementation concerns remain secondary to the architectural proposal.

5.3 Core Platform Implementation

The backend implements the seven bounded contexts described in Chapter 4: User, Organisation, Ecosystem, Ticket, Billing, Gamification, and Notification. Each context follows the same internal

Layer	Technology	Role in the proof of concept
Runtime	Cloudflare Workers	Stateless serverless execution for the API layer.
Database	PostgreSQL (Neon) with Drizzle ORM	Transactional persistence for aggregates, projections, and the outbox.
Event delivery	Transactional outbox + Cloudflare Queues	Durable asynchronous delivery of domain events.
Authentication	BetterAuth	Multi-provider authentication and session management.
Payments	Stripe	External payment-provider integration for the Billing bounded context.
Frontend	React, Vite, TanStack Router, TanStack Query	Single-page application with typed API integration.
UI layer	Custom design system, Base UI, Tailwind CSS	Consistent application surfaces across the proof of concept.
Documentation	OpenAPI and AsyncAPI generation	Machine-readable contracts for API operations and domain events.

Table 5.1: Technology stack used by the Flowzao proof of concept.

structure: domain objects and events in the domain layer; command and query handlers in the application layer; and repositories, queue publishers, HTTP adapters, and provider integrations in the infrastructure layer.

Cross-context operations are coordinated through guarded workflows. For example, ticket creation checks ecosystem permissions and plan limits before delegating to the Ticket bounded context. Domain events are persisted through a transactional outbox in the same database transaction as the aggregate mutation, then dispatched asynchronously to subscribers such as notifications, gamification, and read-model projections. This implementation directly realises the event-driven coordination model proposed in Sections 4.2 and 4.7. The backend is covered by an automated unit-test suite that exercises aggregate invariants, command and query handlers, and read-side projections directly.

5.4 Ticket Lifecycle and Routing Implementation

The Ticket aggregate is the implementation counterpart of the canonical aggregate described in Section 4.1. It owns the ticket status, assignments, transfer requests, comments, attachments, time logs, feedback, resolution notes, and resolution history. All state changes pass through aggregate methods that validate the invariants described in Section 4.4.4; no external adapter writes ticket state directly.

5.4.1 Transfer Implementation

The two routing mechanisms proposed in Chapter 4 are implemented as two aggregate-level paths. Direct transfer ends the current Owner assignment, creates a new Owner assignment for the target organisation, and emits a transfer event with previous and new assignment identifiers. The consent-based transfer protocol creates a pending transfer request; acceptance ends the current ownership period, creates the new ownership assignment, and emits the acceptance event; rejection records the decision without changing ownership.

Both paths are invoked through guarded workflows that validate ecosystem membership, the requesting user's `TransferTickets` permission, and the target organisation's eligibility before the aggregate is mutated. The result is that a transfer changes responsibility without creating a second ticket, preserving the canonical identifier required by R1 and the formal routing history required by R2.

5.5 Gamification Engine

The gamification implementation is event-driven. The Ticket bounded context does not call the gamification subsystem directly; instead, ticket lifecycle and feedback events are consumed by gamification workflows. These workflows update member profiles, credit points, evaluate achievement triggers, and manage reward redemptions. This preserves the separation required by the design: gamification reacts to ticket work without becoming part of the core ticket invariants.

Fairness controls from Chapter 4 are implemented as configuration and visibility rules: achievement triggers can be configured per organisational context, leaderboards are ecosystem-scoped, reward redemption requires approval, and access to performance information is governed by ecosystem permissions.

5.6 Frontend Applications

The main frontend application exposes the operational surfaces needed by technicians and managers: dashboards, tickets, ecosystems, queue, gamification, rewards, analytics, billing, and settings. The administration application handles platform-level management, while the landing page is separate from the operational artefact. API access is performed through typed client hooks generated from the backend contracts, reducing drift between frontend expectations and backend schemas.

5.7 User Interface

The user interface is included only as implementation evidence for the requirements evaluated in Chapter 6. The ticket detail screen evidences the canonical ticket record (R1, R2, R7), the transfer dialogue evidences formal routing (R2), the ecosystem screens evidence the collaboration

boundary (R6), the team settings screen evidences visibility control (R3), and the analytics screen evidences reporting support (R4).

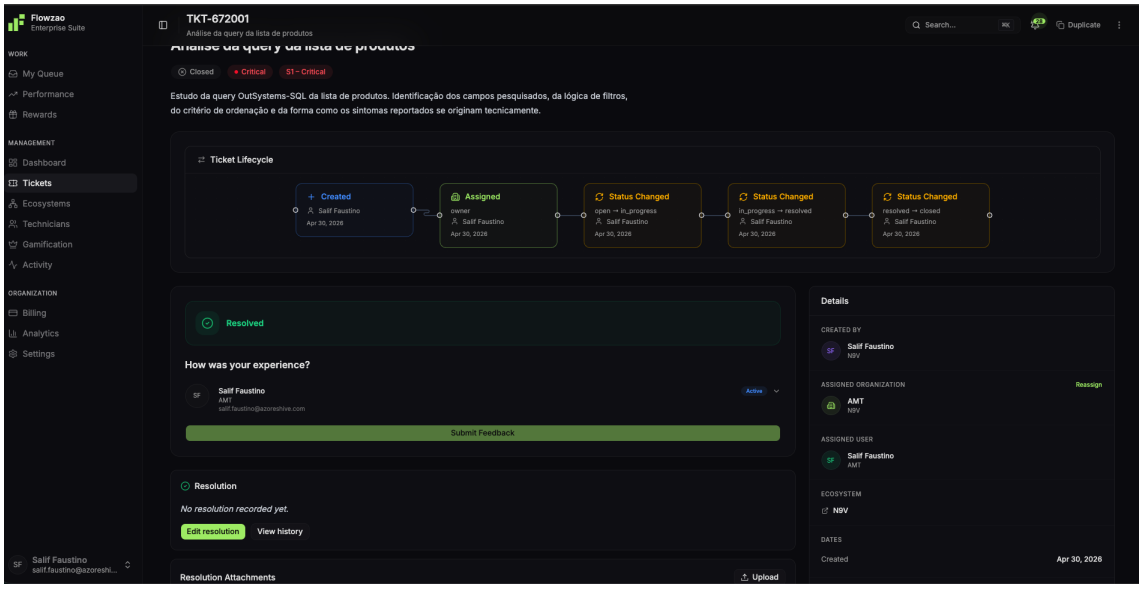


Figure 5.1: Ticket detail view.

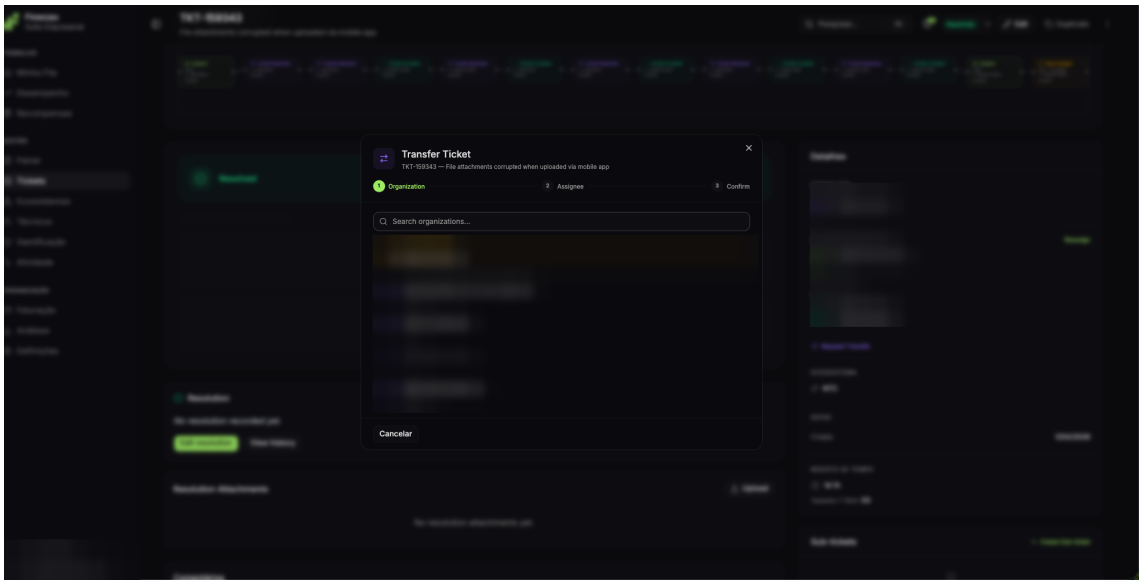


Figure 5.2: Cross-organisational transfer dialogue.

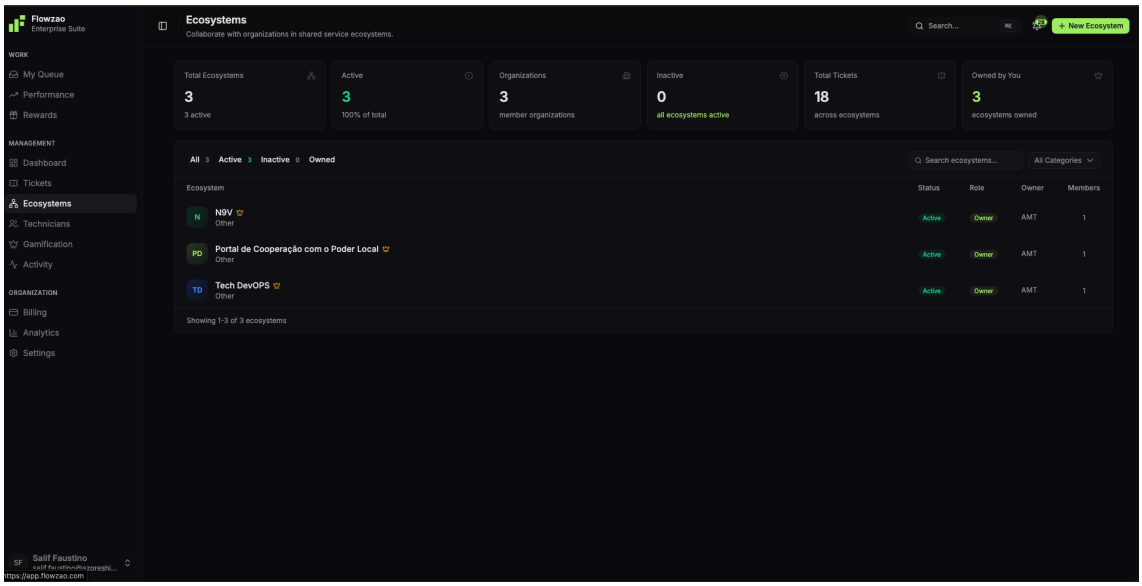


Figure 5.3: Ecosystem list.

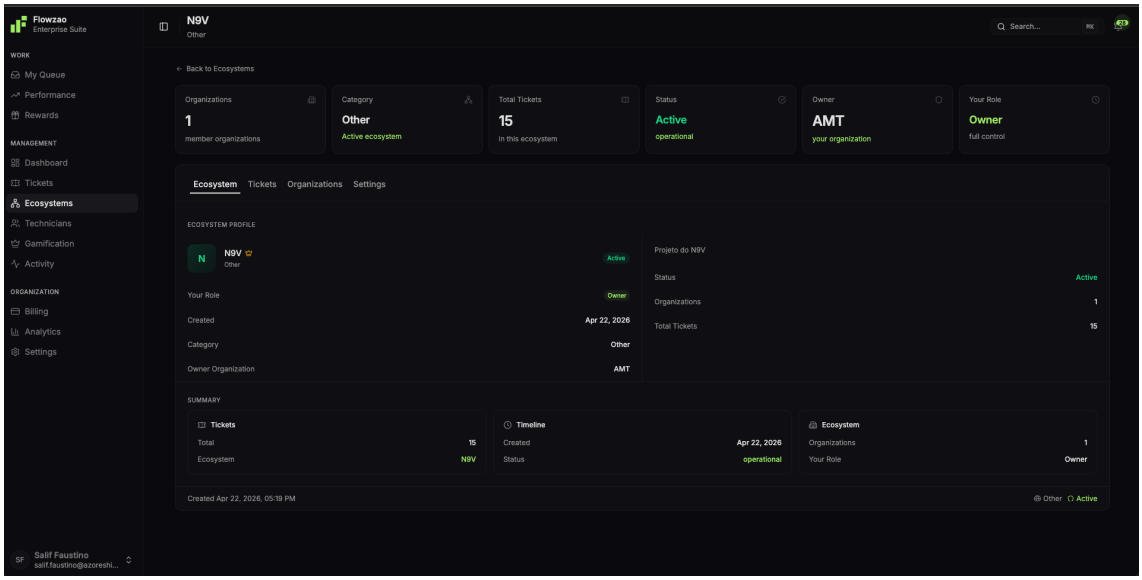


Figure 5.4: Ecosystem detail view.

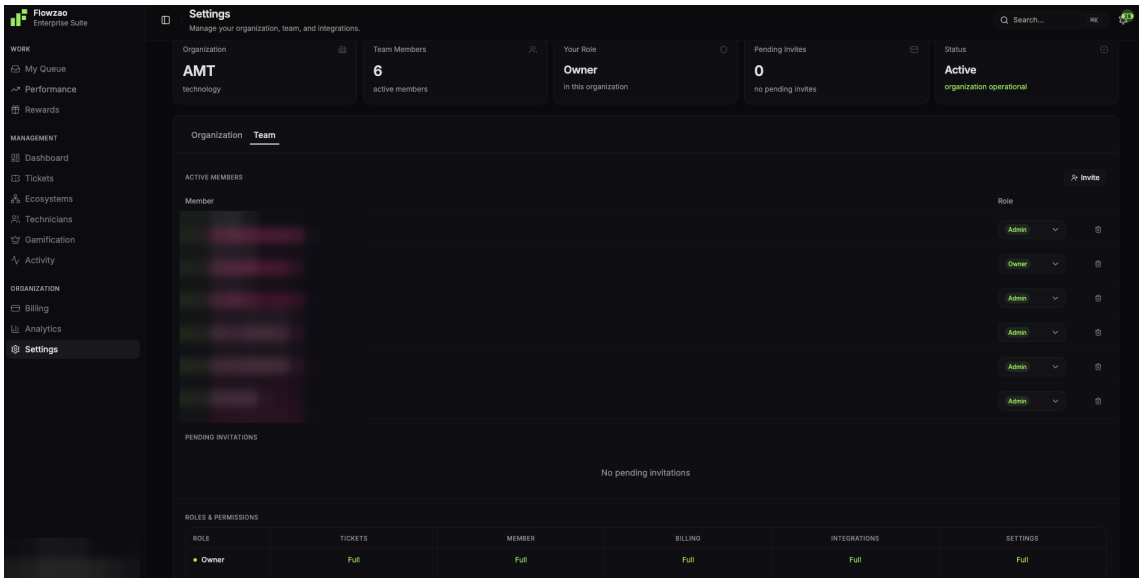


Figure 5.5: Team settings view.

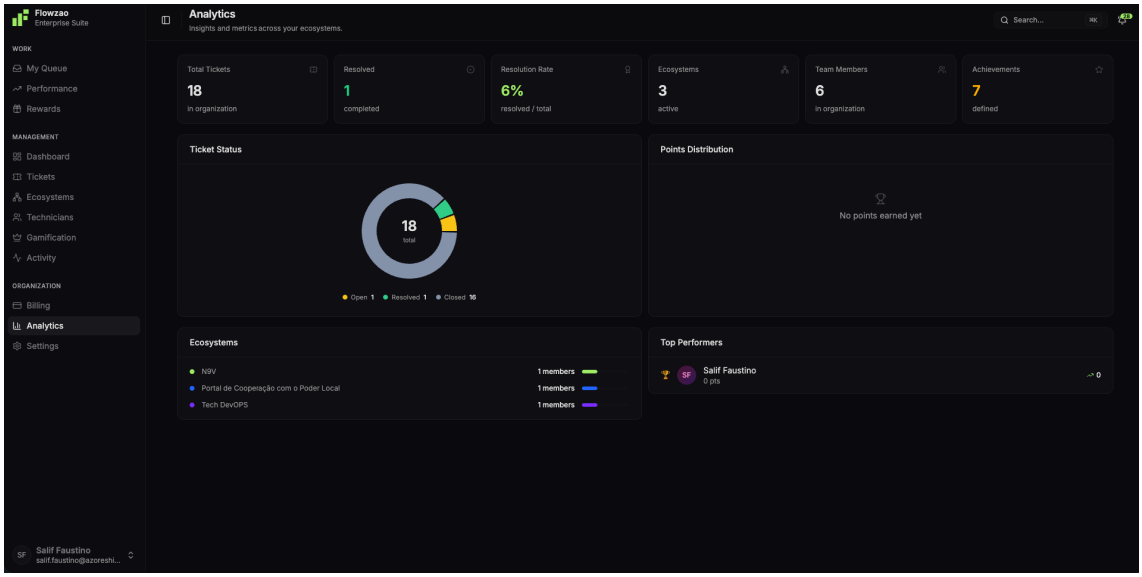


Figure 5.6: Analytics view.

5.8 Artefact Metrics

The implemented artefact is large enough to exercise the proposed architecture across realistic concerns, but its size is not treated as a contribution. At the time of evaluation, the proof of concept comprised the seven bounded contexts, a complete ticket lifecycle, two transfer mechanisms, a gamification subsystem, API documentation, and a deployed frontend. The domain-event and persistence catalogues are reported in Appendices A and D to support reproducibility.

5.9 End-to-End Walkthrough

The implemented system supports the following end-to-end scenario used in the evaluation protocol. A technician in Organisation A creates a ticket inside an ecosystem; the guarded workflow validates permissions and creates the canonical ticket. During investigation, comments and time logs are added. If the issue belongs to Organisation B, Organisation A initiates a transfer request. Organisation B accepts, ownership changes, and the ticket retains the same identifier. Organisation B resolves the ticket, after which Organisation A provides feedback. The feedback event triggers the gamification workflow, and the closed ticket remains available for end-to-end and per-organisation metrics.

Throughout the flow, there is no duplicated partner ticket. The ticket's identity, assignment history, lifecycle events, and metrics all remain attached to the same canonical aggregate.

5.10 Key Technical Challenges

The most demanding parts of the implementation were the realisation, on a stateless serverless runtime, of three cross-cutting mechanisms already motivated in the design (Chapter 4): the guarded workflows that enforce cross-context invariants before a single aggregate transaction (Section 4.7); the transactional outbox with idempotent subscriber tracking that lets notifications, gamification, and projections react without joining the initiating transaction (Section 4.2); and the ecosystem-scoped read paths that preserve multi-tenant visibility boundaries without fragmenting the canonical ticket (Section 4.3). The design rationale for each is given in Chapter 4; the contribution of this section is the evidence that the three were realisable in code on the serverless stack.

5.11 Chapter Summary

This chapter summarised Flowzao as a functional implementation of the proposed architecture. The artefact realises the canonical ticket aggregate, formal routing semantics, ecosystem-scoped visibility, event-driven gamification, and the runtime constraints required by the design. The following chapter evaluates whether this implementation satisfies the requirements and whether it provides useful evidence for the proposed design knowledge.

Chapter 6

Evaluation

This chapter presents the evaluation of the proposed design as instantiated in Flowzao, following the Design Science Research methodology [23, 39]. It evaluates the *artefact* through two complementary instruments: (i) a structured *requirements-based assessment* that examines each requirement (R1–R4, R6, R7) against the corresponding architectural mechanism (Chapter 4) and its implementation (Chapter 5); and (ii) an *in-ecosystem pilot* in which collaborating organisations exercise the platform under their own coordinated workflows, contributing quantitative usage data from real use, qualitative feedback from operating technicians, and a standardised System Usability Scale (SUS) measurement. R5 (composite SLA management) was descoped to future work, as discussed in Section 6.1.5.

Section 6.1 presents the requirements-based assessment. Section 6.2 provides a summary matrix. Section 6.3 evaluates the gamification subsystem at the design level. Section 6.4 describes the in-ecosystem pilot protocol and reports its results, showing how the artefact behaves under real coordinated workflows. Section 6.5 discusses the limitations and threats to validity of the artefact and the evaluation itself.

6.1 Requirements Evaluation

This section assesses each in-scope requirement against the architectural mechanism that satisfies it and its implementation, extending the design traceability matrix established in Chapter 4 (Table 4.6) from *design intent* to *realised evidence*. For each requirement, the evidence is presented and a conclusion assigned.

6.1.1 R1: Canonical Inter-Organisational Ticket Model

Requirement. A canonical inter-organisational ticket model with a global identifier and explicit representation of participating organisations.

Evidence. The Flowzao platform implements the Ticket aggregate as a single, canonical entity that retains its identity throughout its entire lifecycle, regardless of which organisation currently holds ownership. Each ticket receives a unique, auto-generated identifier (e.g., `TKT-A1B2C3`) that

is shared across all participating organisations, as shown in the ticket detail view (Figure 5.1). The ticket is never duplicated or replicated when transferred between organisations; the same aggregate instance is updated.

Participating organisations are explicitly represented through the Assignment child entity collection, which links the ticket to ecosystem memberships with typed roles (Owner, Contributor). The assignment history provides a complete record of every organisation that has participated in the ticket's lifecycle.

Conclusion: Fully satisfied. The platform implements a canonical ticket as a first-class entity with global identity and explicit organisational participation, addressing the inter-organisational information-model gap discussed by Marcu and Schaaf [32].

6.1.2 R2: Formal Routing Events

Requirement. Formal routing events capturing each transfer of responsibility between organisations.

Evidence. The platform implements two routing mechanisms, both producing immutable domain events; the operator initiates a transfer through a guided dialogue (Figure 5.2), and the resulting events are rendered on the ticket lifecycle timeline (Figure 5.1):

- **Direct transfer** produces a transfer event containing the previous and new assignment identifiers, source and target organisations, and timestamp.
- **Transfer request protocol** produces request and decision events. Each event captures the requesting user, target organisation, and the full context of the handoff.

All routing events are persisted via the transactional outbox pattern, ensuring that the complete routing history is permanently preserved and can be queried at any time.

Conclusion: Fully satisfied. Every transfer of responsibility is captured as an explicit, immutable domain event with full provenance. The immutable routing-event design is a Flowzao design choice aligned with the inter-organisational fault-management architecture requirements identified by Marcu and Hommel [31].

6.1.3 R3: Multi-Level Visibility

Requirement. Multi-level visibility with fine-grained control per organisation, preserving auditability.

Evidence. The platform implements visibility control at three levels:

1. **Ecosystem-level permissions:** category-based permission grants covering ticket lifecycle operations, collaboration, routing, administration, and reporting are assigned per user within each ecosystem (managed through the team settings view, Figure 5.5), controlling which operations and data each user can access.

2. **Comment visibility:** comments support three visibility levels (internal, customer-visible, partner-visible), enabling organisations to share information selectively.

3. **CQRS read models:** the separation of read and write models enables organisation-specific projections that filter data according to the requesting user's permissions, without affecting the canonical write-side representation.

Auditability is preserved through the immutable event log: every state change is recorded regardless of the visibility rules applied to read models.

Conclusion: **Satisfied within implementation scope.** The combination of ecosystem permissions, comment visibility levels, and CQRS projections responds to the security, data-isolation, and multi-tenant concerns discussed by Adiraju [1] and Shwartz et al. [51]. The evaluation establishes the mechanism at the design and implementation level; independent cross-tenant authorisation auditing is outside the scope of this dissertation.

6.1.4 R4: End-to-End and Segment Metrics

Requirement. End-to-end and segment metrics: total resolution time, time per organisation, and handoff count.

Evidence. The platform captures the data necessary for computing these metrics, which are surfaced in the analytics view (Figure 5.6):

- **End-to-end resolution time** is computed as the difference between the ticket's `createdAt` and `resolvedAt` timestamps.
- **Time per organisation** is derived from the assignment history: each assignment records a `startAt` and `endAt` timestamp, enabling per-organisation duration computation.
- **Handoff count** is the number of transfer events in the ticket's event stream.
- **Time logs** provide fine-grained time tracking categorised by type (Active Work, Waiting, Research, Communication), enabling analysis of how time is spent within each organisational segment.

Conclusion: **Fully satisfied.** The data model supports metrics motivated by the visibility and correlation problems discussed by Salah et al. [45] and Marcu et al. [34], including end-to-end resolution time, per-organisation duration, and handoff count. The assignment-based time tracking goes beyond the requirement by providing categorised time data.

6.1.5 R5: Composite SLA Management

Requirement. Composite SLA management spanning multiple organisations.

Status: **Not implemented (deferred to future work).** R5 was identified as a requirement in the SLR, drawing on the work of Theilmann et al. [52] on multi-level SLA management and

De Vleeschauwer et al. [13] on risk-based SLA decomposition. However, composite SLA management, including automatic decomposition of end-to-end SLA targets into per-organisation segments, was descoped from the implementation; proactive breach prediction is treated as a separate future extension. The proof of concept prioritised the canonical ticket model, formal routing semantics (R1, R2), and the supporting infrastructure (R3, R4, R6, R7). Composite SLA management is deferred to future work.

6.1.6 R6: Multi-Tenant Architecture

Requirement. Multi-tenant architecture with logical isolation and configurable workflows.

Evidence. The platform implements multi-tenancy through the Ecosystem abstraction (Figures 5.3 and 5.4):

- Each ecosystem provides a logically isolated collaboration space where multiple organisations interact.
- Data isolation is enforced at query level: every database query filters by the requesting user's ecosystem memberships.
- Workflow configurability is achieved through the category-based ecosystem permission model, enabling each organisation to define which operations its users can perform within each ecosystem.
- The billing subsystem (Free, Standard, Enterprise plans) enforces resource limits per organisation (ticket quotas, member limits).

Physical database separation is not implemented; isolation relies on logical scoping. This is a deliberate trade-off: logical isolation simplifies cross-organisation ticket operations (the core use case) while providing adequate security for a proof of concept.

Conclusion: Satisfied within implementation scope. The ecosystem-based multi-tenancy aligns with the multi-tenant characteristics and challenges discussed by Hattab and Belalem [22] and Bezemer and Zaidman [6], particularly logical scoping and tenant-specific configurability. The claim is limited to the implemented logical-isolation model; physical tenant isolation and independent security auditing are outside the scope of this dissertation.

6.1.7 R7: Single Source of Truth (SSOT)

Requirement. Canonical tickets maintained centrally, with local ITSM tools operating as consumers and partial replicas.

Evidence. The platform maintains all ticket state in a single canonical aggregate, surfaced as one authoritative record in the ticket detail view (Figure 5.1). The aggregate state is the authoritative source of truth, and all read models (projections) are derived from the same canonical data. Domain events are published via the transactional outbox for integration with other bounded contexts. There is no scenario in which conflicting ticket state exists across multiple systems.

The platform exposes OpenAPI and AsyncAPI specifications, which could serve as integration points for external ITSM tools to consume ticket data. However, pre-built connectors for specific ITSM tools (e.g., ServiceNow, Jira) are not implemented.

Conclusion: Satisfied within platform scope. The SSOT principle is realised within the Flowzao platform: all ticket state has one authoritative source and projections derive from it. External ITSM integration via pre-built connectors is outside the scope of the proof of concept, so this requirement is satisfied for the implemented platform but not yet for a full hybrid deployment with existing ITSM tools.

6.2 Evaluation Summary

Table 6.1 summarises the evaluation results across all seven requirements.

Req.	Description	Conclusion	Key evidence
R1	Canonical ticket model	Fully satisfied	Ticket aggregate with global ID; assignment entities
R2	Formal routing events	Fully satisfied	2 transfer mechanisms; 4 routing events; outbox
R3	Multi-level visibility	Satisfied in scope	Permission model; comment visibility; CQRS projections
R4	E2E and segment metrics	Fully satisfied	Assignment timestamps; time logs; event stream
R5	Composite SLA	Not implemented	Descoped; deferred to future work
R6	Multi-tenant architecture	Satisfied in scope	Logical ecosystem isolation; permission model; billing plans
R7	Single Source of Truth	Satisfied in platform scope	Canonical aggregate; derived projections; API specs

Table 6.1: Summary of the requirements evaluation.

Of the seven requirements identified in the SLR, six (R1–R4, R6, R7) were within the implementation scope and are satisfied within that scope. R5 (composite SLA management) was descoped from the implementation: the proof of concept prioritised the canonical ticket model and routing mechanisms (R1, R2) as the core evaluated mechanisms, with composite SLA management deferred to future work.

6.3 Gamification Evaluation (RQ2)

Beyond the seven functional requirements, the platform's gamification subsystem is evaluated against the design principles established in Chapter 2.

Integration with ticket lifecycle. The gamification engine is fully integrated: points are awarded through the feedback mechanism after ticket resolution, and achievement triggers evaluate six distinct aspects of technician performance (`ResolveTickets`, `ResolveInTime`, `DayStreak`, `ResolvePriority`, `CsatScore`, `ResolveAfterHours`). The integration is purely event-driven: the Ticket bounded context is unaware of the gamification subsystem.

SDT alignment. The gamification elements address all three SDT needs: *competence* through achievements and point progression; *relatedness* through feedback ratings and ecosystem-scoped leaderboards; and *autonomy* through configurable achievement triggers and a rewards catalogue with user-initiated redemption.

Fairness safeguards. The design incorporates the safeguards identified in Section 4.6: scoped leaderboards reduce unfair comparisons; approval-based redemption mitigates gaming; permission-controlled visibility protects sensitive performance data; and configurable triggers allow per-organisation customisation.

Empirical signal. The in-ecosystem pilot reported in Section 6.4 provides initial qualitative evidence on the perceived integration, fairness, voluntariness, and motivational potential of the subsystem (items G1–G5 of the questionnaire). The signal is interpretable as confirmatory or critical at the design level; the long-term *behavioural* impact (sustained motivation, habituation effects, distortion of resolution priorities) cannot be determined from a six-week pilot and remains a deliberate scope boundary.

6.4 Empirical Evaluation with B2B Partner Organisations

The industry-side cross-check in Chapter 3 established that the fragmentation symptoms targeted by the artefact are recognised by industry professionals who have not used Flowzao. This section turns to the artefact itself: an in-ecosystem pilot was conducted with collaborating organisations using the platform within a shared B2B service ecosystem. The section describes the protocol, summarises the quantitative usage data from real use, reports the standardised System Usability Scale (SUS) measurement, and synthesises the qualitative findings from operating technicians' free-text responses alongside the structured Likert items.

6.4.1 Protocol Design

The evaluation protocol comprises three parts, included in full in Appendix C:

1. **Part A (Participant Profile):** collects demographic data including the participant's role, years of professional experience, familiarity with existing ITSM tools, and prior exposure to the inter-organisational ticket coordination problem.

2. **Part B (Task-Based Walkthrough):** guides the participant through a complete inter-organisational ticket lifecycle in six sequential tasks: (i) create a ticket within a pre-configured multi-organisation ecosystem; (ii) add a comment and log time against the ticket; (iii) initiate a cross-organisation transfer to a partner organisation; (iv) verify ticket identity preservation and history continuity after transfer; (v) resolve the ticket and submit feedback with ratings; (vi) view gamification elements (points, achievements, leaderboard, rewards catalogue). This walkthrough instantiates the end-to-end inter-organisational scenario described in Section 5.9 of Chapter 5. Each task records completion status, perceived difficulty, and qualitative observations.

3. **Part C (Structured Questionnaire):** collects quantitative data across four dimensions:

    - **Requirements satisfaction (C1):** six Likert items mapped to the requirements within scope (R1–R4, R6, R7), aligned with the canonical R-code definitions in Section 2.6.2 of Chapter 2.
    - **Gamification assessment (C2):** five Likert items evaluating integration quality, perceived motivation, cross-organisation fairness, voluntariness, and reward meaningfulness.
    - **System Usability Scale (C3):** the standard 10-item SUS instrument [8], scored using Brooke's algorithm of odd-item ($raw - 1$) and even-item ($5 - raw$) contributions multiplied by 2.5. Scores are interpreted against later published SUS benchmarks, where approximately 68 is commonly treated as average [5].
    - **Comparison with current tools (C4):** comparative self-report against participants' existing ITSM tools, identification of the most valuable feature, identification of missing or unsatisfactory features, and free-text remarks.

The protocol was designed following guidelines for expert evaluation in Design Science Research [23]. The questionnaire was implemented as a web form on the project's landing site to ensure consistent administration across sites.

In addition to the questionnaire responses, the platform's own usage over the pilot window provided a complementary, objective source of evidence: tickets created per organisation, cross-organisation transfer counts and outcomes, time-to-resolution distributions, per-organisation ticket-hold durations, handoff distributions, gamification engagement, and feedback ratings.

6.4.2 Setup and Participants

The platform was deployed and exercised within a shared B2B service ecosystem formed by three collaborating organisations that participate in the same coordinated workflow. The participating organisations are anonymised throughout the dissertation; sectoral profile and size band are described qualitatively rather than tabulated, on the explicit request of the participants during the consent phase. The workflow instantiates the inter-organisational fragmentation pattern characterised in Chapter 1: tickets are created in one organisation, transferred to a partner for action,

and then closed or followed up within the same shared ecosystem. The pilot data window opened on 2026-04-27 and remains active at the time of this writing; the figures reported in this chapter reflect the data collected up to 2026-06-03.

A total of 7 technicians and operations leaders from the participating organisations submitted the in-ecosystem questionnaire over the pilot window. The empirical evaluation is treated as ongoing: the numerical results in Tables 6.2, 6.3, and 6.4 reflect the responses and usage data collected up to the snapshot date. The pilot is intentionally narrow because its purpose is to provide evidence that the canonical-ticket paradigm can operate under real coordinated workflows, not to establish industry-wide prevalence. That broader claim was addressed by the industry problem-validation questionnaire reported in Chapter 3.

6.4.3 Quantitative Findings from Pilot Telemetry

Across the pilot period, the participating organisations created 38 tickets on the platform. Of these, 14 (36.8%) involved at least two organisations through the inter-organisational routing mechanism, providing direct empirical evidence that the canonical-ticket pattern (R1) and formal routing events (R2) operate at the level of real workflow rather than as theoretical mechanisms only.

Metric	**Value**	**Interpretation / R-code**
Tickets created (pilot period)	38	Volume of usage.
Tickets crossing ≥ 2 organisations	14 (36.8%)	Direct evidence for R1, R2, R7.
Median time-to-resolution	10 h	R4 (end-to-end metric).
P90 time-to-resolution	11.7 h	R4 (tail behaviour).
Tickets resolved	20	Resolution throughput.
Mean feedback rating (1–5)	4.3	Quality signal.
Members with ≥ 1 earned achievement	4 (80%)	RQ2 engagement signal.
Reward redemptions requested / approved / fulfilled	1 / 1 / 1	RQ2 engagement signal.

Table 6.2: Quantitative findings from the platform's usage over the pilot window.

The full distributions for time-to-resolution, per-organisation hold time, and handoff counts are not reproduced as separate figures: at the current sample size, the aggregate measures in Table 6.2 already convey the headline shape of each distribution.

6.4.4 System Usability Scale Results

The cohort returned a mean SUS score of 61.8 (SD 20.1, $n = 7$, range 50–92.5, median 50). Per the benchmark reported by Bangor et al. [5], a SUS score around 68–70 is commonly treated as approximately average; values above the low 70s are interpreted as moving into above-average usability. The platform's measured value places its perceived usability in the below-average band relative to that benchmark.

Given the small, demographically homogeneous cohort — predominantly software engineers with two to five years of experience — inherent in a six-week pilot of this scale, this result is interpreted as a directional signal rather than a generalisable measurement. The per-respondent distribution (Table 6.3) is moreover bimodal: the five earliest respondents returned the exact scale midpoint, while the two most recent returned scores in the excellent band. The mean and the median should therefore be read together, and the SUS and Likert results treated as formative rather than summative evidence.

Respondent	Submitted	SUS	Band
P1	2026-04-27	50	Poor
P2	2026-04-27	50	Poor
P3	2026-04-27	50	Poor
P4	2026-04-27	50	Poor
P5	2026-04-28	50	Poor
P6	2026-06-03	92.5	Excellent
P7	2026-06-03	90	Excellent
Mean	—	**61.8**	—

Table 6.3: Per-respondent SUS scores.

6.4.5 Per-Requirement and Per-Gamification Likert Results

Table 6.4 reports the cohort means on the C1 requirements-satisfaction items and the C2 gamification items, on a 1–5 Likert scale. These items mirror the canonical requirement definitions and the design fairness goals stated in Chapters 2 and 4.

Code	Item	Mean	SD	n
R1	Canonical ticket with global identity	4.7	0.5	7
R2	Formal routing events on every transfer	4.9	0.4	7
R3	Tailored views with controlled visibility	4.9	0.4	7
R4	End-to-end and per-organisation metrics	5	0	7
R6	Multi-tenant coexistence without interference	5	0	7
R7	Single authoritative ticket location	4.9	0.4	7
G1	Gamification well integrated into the workflow	5	0	7
G2	Mechanics seem likely to motivate technicians	4.9	0.4	7
G3	Fair across organisations of different sizes	4.9	0.4	7
G4	Voluntary and non-coercive	4.9	0.4	7
G5	Rewards catalogue appears meaningful	4.7	0.5	7

Table 6.4: Pilot requirements and gamification Likert results.

6.4.6 Qualitative Findings from Free-Text Responses

The questionnaire's three free-text fields (*most valuable feature*, *missing or unsatisfactory feature*, and *additional comments*) provide the qualitative complement to the Likert and SUS scales. As of

the data extraction date, 1 respondent(s) provided substantive answers in the *most valuable* field and 1 in the *missing feature* field; the others left these fields blank. As discussed in the protocol above, the cohort is small and the qualitative material reported here is an early signal rather than a saturated thematic analysis. All free-text responses are reported in full, translated from the original Portuguese by the author; a thematic clustering pass will be applied once $n \geq 10$ to enable a frequency-ranked list of themes.

Most valuable feature (positive signal). Table 6.5 reports the free-text responses to the *most valuable feature* prompt. Recurrent terms in the early responses centre on adaptability of the platform to different needs and on perceived intuitiveness of the user experience.

#	Feedback (free text)	Submitted
1	The product's robust flexibility stands out for its ability to adapt to different needs, combined with evident care for usability. The user experience is extremely intuitive, giving the user a simple, efficient, and pleas...	2026-04-28

Table 6.5: Most valuable feature responses (translated from Portuguese).

Missing or unsatisfactory features (improvement signal). Table 6.6 reports the responses to the *missing or unsatisfactory feature* prompt. With $n = 1$, no statistical claims are made; each response is a directional improvement candidate.

#	Feedback (free text)	Submitted
1	Optimise the typographic sizes of several elements, with the aim of improving legibility. This adjustment allows for more comfortable and clearer reading, contributing to a more accessible and efficient user experience.	2026-04-28

Table 6.6: Missing or unsatisfactory feature responses (translated from Portuguese).

Cross-mapping with requirements. The per-requirement qualitative interpretation reported in Section 6.1 relies on (i) the corresponding C1 Likert score and its dispersion, (ii) the pilot-telemetry signals reported earlier in this section (e.g., the share of tickets routed across organisations as evidence for R1, R2, R7), and (iii) the free-text material above. As the cohort grows, future iterations of the evaluation will enable a frequency-coded thematic analysis of the questionnaire free-text fields, coded against each R1–R7 requirement plus the gamification G1–G5 dimensions.

6.4.7 Comparative Assessment

In the C4 comparative item, participants rated Flowzao's handling of cross-organisation tickets relative to their existing ITSM tools on a six-point scale (Much worse / Worse / Similar / Better / Much better / N/A). The distribution is reported in Table 6.7.

Response	Count
Better	2
Much better	5

Table 6.7: Comparative assessment against existing ITSM tools.

6.5 Limitations and Threats to Validity

The evaluation in this chapter was deliberately scoped to be feasible within a master's dissertation. Following the classification commonly used in primary empirical software-engineering studies [55], its limitations and the threats to the validity of the inferences drawn from it are grouped into construct, internal, external, and conclusion validity, followed by a research-ethics note and a scope note, so that the strength of those inferences can be judged accurately:

Construct validity. All Likert and comparative measures rely on self-report; observable workflow improvements (e.g., reductions in resolution time) are inferred from the platform's usage data and from participant accounts, not from a controlled before/after experiment. The requirements assessment was conducted by the artefact's designer, which may introduce a positive-response bias; anonymous questionnaire submission, full quotation of the free-text responses, and the use of objective pilot telemetry mitigate but do not eliminate this risk.

Internal validity. The pilot covered a six-week period, which is too short to rule out novelty effects or to assess sustained behavioural effects such as gamification habituation [30]; behavioural conclusions are restricted accordingly. The dissertation author also had operational contact with the participating organisations during the pilot, a proximity that may have influenced responses.

External validity. The participating organisations volunteered, are part of a single ecosystem, and may differ from a representative population of B2B service operators in their motivation to adopt new tools; one pilot ecosystem with a small number of technicians is below the threshold at which statistical generalisation is meaningful. The industry problem-validation questionnaire (Chapter 3) was designed to widen the empirical reach beyond the pilot, providing a complementary industry-side cross-check rather than a probabilistic industry-wide measurement; larger-scale or multi-ecosystem studies remain future work.

Conclusion validity. Given the sample size, the findings are reported as directional signals rather than generalisable measurements, and no inferential statistical tests are claimed.

Research ethics. The study involved human participants (the pilot technicians and the questionnaire respondents). Participation was voluntary and based on informed consent, with anonymisation of organisations and individuals and separation of contact details from the response analytics. The study was not, however, submitted to a formal institutional ethical review board; obtaining such review is a prerequisite for the larger-scale, multi-ecosystem studies identified as future work.

Scope — SLA management not implemented. As discussed in Section 6.1.5, R5 (composite SLA management) was descoped and is deferred to future work.

These limitations and threats to validity do not invalidate the in-ecosystem pilot but bound its claims: it reports what the participating organisations and their technicians experienced during the period, not what the wider population of B2B service ecosystems would experience in aggregate. The complementary industry-side cross-check was reported in Chapter 3.

6.6 Chapter Summary

This chapter evaluated the proposed design as instantiated in Flowzao through a multi-pronged approach. The structured requirements-based assessment found that the six requirements within scope (R1–R4, R6, R7) are satisfied within the implemented platform scope; R5 (composite SLA management) was descoped to future work.

The in-ecosystem pilot, conducted with collaborating organisations using the platform within a single B2B service ecosystem, provided three complementary forms of evidence: pilot telemetry quantifying real cross-organisation usage, a standardised System Usability Scale measurement alongside requirement- and gamification-oriented Likert scales, and the free-text responses collected through the in-ecosystem questionnaire. The empirical signals corroborate the requirements-based findings while surfacing early, directional pain points that the requirements-based assessment alone could not reveal. The SUS result in particular should be read as a mixed usability signal rather than a summative endorsement.

The gamification subsystem demonstrates technical feasibility, theoretical alignment with Self-Determination Theory, and initial qualitative signal from the in-ecosystem pilot cohort; assessing its long-term behavioural impact remains future work, as the six-week pilot is too short to observe habituation effects.

Taken together, the evaluation supports the central thesis: it is technically feasible to design and implement a multi-tenant platform that maintains a canonical inter-organisational ticket as a Single Source of Truth with formal routing semantics, end-to-end metrics, and integrated gamification mechanisms. The limitations acknowledged in this chapter, notably the limited scale of the empirical evaluation, are identified as priorities for future work.

Chapter 7

Conclusions and Future Work

7.1 Summary of Contributions

This dissertation addressed the problem of fragmented ticket management in B2B service ecosystems, where incidents that traverse organisational boundaries are managed through disconnected tickets, informal coordination channels, and manual data replication. The work followed a Design Science Research methodology and produced three contributions, whose significance is summarised below.

C1: Systematic Literature Review. The SLR revealed that inter-organisational ticket management is not merely under-researched but effectively abandoned: the foundational 2009–2013 cluster of work was never followed by a comparably sustained programme, despite growing industry reliance on multi-vendor service delivery. The seven requirements and seven research gaps derived from the review — which screened 424 initial records down to 16 primary studies, with four secondary reviews retained as background — provide a structured research agenda that did not previously exist for this domain. A condensed version of the review is being prepared as an accompanying paper.

C2: Architecture, domain model, and a transferable design pattern. The central contribution is the design knowledge captured by the *Canonical Inter-Organisational Aggregate*: a transferable architectural pattern for treating the inter-organisational ticket as a *Single Source of Truth* in a shared platform, not as a synchronisation problem between independent systems. The pattern catalogues the forces (multi-party lifecycle, identity preservation, asymmetric visibility, end-to-end accountability), the four cooperating elements (canonical aggregate, participation collection, routing events, tenancy scope), the consequences, the design guidelines, and the known applicabilities (inter-organisational clinical case management, multi-firm legal matters, cross-carrier logistics, multi-vendor incident response in cyber security). In the ITSM instantiation, the bounded context decomposition, event-driven orchestration, and guarded workflow patterns provide reusable architectural knowledge for any multi-organisation coordination domain, not only for Flowzao.

C3: Proof-of-concept instantiation and empirical validation. Flowzao is the functional instantiation of the architecture (C2). It has been exercised in an in-ecosystem pilot in which collaborating organisations coordinate real workflows on the platform within a shared B2B service ecosystem. The artefact is evaluated against the six requirements within scope (R1–R4, R6, R7) through a structured requirements-based assessment and the in-ecosystem pilot itself, combining pilot telemetry, technician feedback (free-text responses and Likert items), and a standardised System Usability Scale measurement (Chapter 6). The empirical work is complemented by a public, multilingual *Problem Evaluation Survey* (Chapter 3) that probes the SLR's pain hypotheses with respondents who have not used the platform, providing an artefact-independent industry-side cross-check beyond the pilot ecosystem.

7.2 Answers to the Research Questions

7.2.1 RQ1: Architecture for Inter-Organisational Ticket Routing

How can centralised ticket-management solutions be architected to support formal incident routing across organisations in B2B service ecosystems?

The answer to RQ1 is the architectural model instantiated in Flowzao. It shows how a modular monolith organised into bounded contexts, with a Ticket aggregate that publishes domain events via a transactional outbox as its core, can satisfy this requirement through four mechanisms:

1. **Canonical ticket entity:** a single Ticket aggregate with a global identifier, never duplicated across organisations. The assignment model (Owner and Contributor roles) explicitly tracks which organisations participate in each ticket's lifecycle.

2. **Formal routing events:** two routing mechanisms (direct transfer and transfer request protocol) produce immutable domain events that capture every transfer of responsibility with full provenance. Because all transfer events are persisted via the transactional outbox, routing history is permanently preserved and fully queryable.

3. **Multi-level visibility:** CQRS read models combined with ecosystem-scoped permission categories enable organisation-specific views of the same canonical ticket data, balancing transparency with commercial sensitivity.

4. **End-to-end metrics:** assignment timestamps, time logs, and the event stream provide the data foundation for computing total resolution time, per-organisation duration, handoff count, and collaboration effectiveness indicators.

The key architectural insight is that the inter-organisational ticket should be modelled as a *Single Source of Truth* maintained in a centralised, multi-tenant platform, not as a synchronisation problem between independent systems. This proposes and demonstrates an alternative paradigm to the prevailing approach identified in the SLR, where ticket integration is treated as data replication

between systems that each maintain their own version of the truth. The empirical evaluation reported in Chapter 6 provides initial evidence that this paradigm can operate in real workflow conditions across the pilot ecosystem, including across organisational boundaries via the formal routing protocol.

7.2.2 RQ2: Gamification in Inter-Organisational Ticket Management

How can gamification elements be designed and integrated into the inter-organisational ticket lifecycle?

The answer to RQ2 is a reactive, event-driven gamification subsystem that integrates with the ticket lifecycle through domain events, without coupling the core ticket management logic to gamification concerns. The design demonstrates that:

1. **Points and achievements** can be awarded based on ticket lifecycle events (resolution, feedback, timely completion) through an event-driven pipeline that evaluates six trigger types against member profiles. The story-point Fibonacci weighting of points by ticket effort, multiplied by feedback rating, is designed to mitigate the perverse incentive of closing many small tickets quickly at the expense of complex ones.

2. **Fairness in inter-organisational contexts** can be supported through ecosystem-scoped leaderboards, configurable achievement triggers, approval-based reward redemption, and permission-controlled visibility of performance data. The questionnaire administered to technicians at the pilot ecosystem included a dedicated fairness item (G3) and a voluntariness item (G4) intended to surface coercive or unfair dynamics if they exist.

3. **Theoretical grounding** in Self-Determination Theory [14], the MDA framework [24], and the Octalysis model [9] provides a principled basis for selecting game elements that favour intrinsic motivation ("white hat" drives) over extrinsic pressure.

RQ2 is therefore answered at the design level. The in-ecosystem pilot reported in Chapter 6 provides initial qualitative signal on perceived fairness, voluntariness, and motivational potential, but the long-term *behavioural* impact (sustained motivation, habituation, and avoidance of subtle distortions in resolution priorities) cannot be determined from a six-week pilot of this scale and remains a deliberate scope boundary, identified explicitly in Section 7.3.

7.3 Future Work

Based on the contributions and limitations of this dissertation, the following directions for future work are identified:

Cross-domain validation of the pattern. The transferability of the *Canonical Inter-Organisational Aggregate* to adjacent coordination domains — inter-organisational clinical case management, multi-firm legal matters, cross-carrier logistics, and multi-vendor security incident

response — is, in this dissertation, a design-level (analogical) claim rather than an empirically validated result. The natural research step is to instantiate the pattern in at least one such domain and exercise it under analogous forces, testing whether its four cooperating elements (canonical aggregate, participation collection, routing events, tenancy scope) hold when the role taxonomy, visibility model, and routing-event payload differ. Confirming or refuting transferability would move the pattern from a hypothesis grounded in a single worked example towards a tested design theory.

Controlled comparison of routing and replication. The dissertation argues on architectural grounds that a canonical routed ticket avoids the informational, temporal, and responsibility fragmentation inherent in the prevailing replication paradigm. A controlled study — comparing canonical routing against replication-based coordination on matched incident scenarios, and measuring context retention, coordination effort, handoff latency, and end-to-end visibility — would test that claim directly and quantify the benefit.

Scaling the empirical evaluation. The in-ecosystem pilot reported in Chapter 6 provides evidence that the platform operates within a single coordinated B2B ecosystem; the public problem-validation questionnaire (Chapter 3) is the lighter-weight instrument designed to broaden the empirical reach. The natural next step is a multi-ecosystem, multi-sector study spanning additional pilot ecosystems and a longer observation window, sufficient to support statistical generalisation and to capture seasonal or workload-dependent effects that the six-week pilot cannot.

Gamification effectiveness study. A longitudinal empirical study, complementary to the qualitative signal already collected, measuring the impact of the gamification subsystem on technician engagement, resolution time, and collaboration quality across organisations. The study should compare gamified and non-gamified conditions and assess both quantitative metrics and qualitative experience over a sufficiently long observation window to surface habituation effects.

AI-assisted inter-organisational routing. The routing-event history produced by the canonical aggregate is a structured dataset for studying how incidents move across organisations. Open questions include whether routing decisions, ticket "ping-pong" patterns, and end-to-end resolution times can be predicted from this history, and whether learned models can recommend handoffs that reduce coordination overhead. The complete event log produced by the transactional outbox provides a rare, ground-truth dataset for this line of inquiry.

Decentralised realisation of the canonical aggregate. The architecture assumes a centrally hosted canonical aggregate (force F5). Where no participant can host the system of record, a decentralised realisation — for example a *decentralised autonomous organisation* (DAO) with routing and participation encoded as smart contracts over a verifiable ledger — could preserve the pattern's invariants (single global identifier, append-only participations, immutable routing events) without a central operator. Investigating whether such a design

sustains end-to-end accountability at acceptable latency and cost, and how its governance compares with the centralised model, is a natural extension of the Federated Canonical Aggregate variation (Section 4.1).

Governance and trust models. Developing formal governance models for the shared platform, addressing questions of data ownership, dispute resolution, and accountability in contexts where competing organisations must collaborate. This is the principal non-technical research gap identified in the SLR (Gap 5) and remains open after this dissertation.

7.4 Final Remarks

The ITSM industry is at an inflection point. As organisations increasingly depend on networks of specialised service providers, the inability to manage incidents across organisational boundaries is no longer a minor inconvenience; it is a structural bottleneck that degrades service quality, obscures accountability, and frustrates every technician caught in a chain of forwarded emails and duplicated tickets. The SLR conducted in this dissertation confirms that the research community recognised this problem over fifteen years ago, yet the dominant commercial tools still treat inter-organisational ticket management as an integration afterthought rather than a first-class concern.

This dissertation offers a different starting point. By treating the inter-organisational ticket as a Single Source of Truth maintained in a shared platform, rather than as a synchronisation problem between independent systems, the Flowzao architecture shows how the informational, temporal, and responsibility fragmentation described in the problem statement can be addressed at the architectural level. The patterns developed here — canonical ticket identity, formal routing events, ecosystem-scoped visibility, and event-driven gamification — are not specific to the Flowzao platform; they constitute transferable design knowledge for anyone building collaborative service management systems.

The Flowzao platform is a proof of concept, not a finished product, but it is no longer only a whiteboard idea: it has been deployed and exercised in an in-ecosystem pilot within a B2B service ecosystem, providing the empirical foundation reported in Chapter 6. Much remains to be done — larger and longer empirical studies, additional pilot ecosystems, validation of the pattern in adjacent domains, and the governance models needed for competing organisations to trust a shared platform — but the architecture and domain model are grounded in both literature and practice, and the implementation provides initial evidence for the viability of the approach in coordinated workflow conditions. If multi-vendor service delivery is to move from ad hoc coordination to structured collaboration, systems based on this kind of canonical inter-organisational ticket model offer one defensible path forward.

References

[1] Seshu Kumar Adiraju. "Security Considerations in Integrating the Fragmented, Outsourced, ITSM Processes". In: *Proc. 3rd International Conference on Services in Emerging Markets (ICSEM)*. 2012, pp. 175–182. DOI: 10.1109/ICSEM.2012.33.

[2] Apostolos Ampatzoglou et al. "Identifying, Categorizing and Mitigating Threats to Validity in Software Engineering Secondary Studies". In: *Information and Software Technology* 106 (2019), pp. 201–230. DOI: 10.1016/j.infsof.2018.10.006.

[3] Atlassian. *Create and Link Work Items Between Different Jira Cloud Sites*. Atlassian Support. 2024. URL: https://support.atlassian.com/jira/kb/create-and-link-work-items-between-different-jira-cloud-sites/ (visited on 03/01/2025).

[4] AXELOS. *ITIL Foundation: ITIL 4 Edition*. TSO (The Stationery Office), 2019. ISBN: 978-0-11-331607-6.

[5] Aaron Bangor, Philip Kortum, and James Miller. "Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale". In: *Journal of Usability Studies* 4.3 (2009), pp. 114–123. URL: https://uxpajournal.org/wp-content/uploads/sites/7/pdf/JUS_Bangor_May2009.pdf.

[6] Cor-Paul Bezemer and Andy Zaidman. "Multi-Tenant SaaS Applications: Maintenance Dream or Nightmare?" In: *Proc. Joint ERCIM Workshop on Software Evolution and Int. Workshop on Principles of Software Evolution (IWPSE-EVOL)*. 2010, pp. 88–92. DOI: 10.1145/1862372.1862393.

[7] Alberto Brandolini. *Introducing EventStorming*. Available at https://www.eventstorming.com/book/. Leanpub, 2021.

[8] John Brooke. "SUS: A "Quick and Dirty" Usability Scale". In: *Usability Evaluation in Industry*. Ed. by Patrick W. Jordan et al. Taylor & Francis, 1996, pp. 189–194.

[9] Yu-kai Chou. *Actionable Gamification: Beyond Points, Badges, and Leaderboards*. Octalysis Media, 2015. ISBN: 978-1511744041.

[10] Alistair Cockburn. *Hexagonal Architecture*. 2005. URL: https://alistair.cockburn.us/hexagonal-architecture (visited on 06/05/2026).

[11] Lorenzo Compagnucci et al. "A Systematic Literature Review of Business-to-Business Platforms for the Digital Transformation of the Manufacturing Industry: Taking Stock and Advancing Through Research". In: *Technovation* 148 (2025), p. 103330. DOI: 10.1016/j.technovation.2025.103330.

[12] Fredrick Dande et al. "Artificial Intelligence Integration in IT Service Management: An ITIL Configuration Management Process Review". In: *Proc. 1st World Congress on Industrial Engineering and Operations Management*. Detroit, MI, USA, 2024. DOI: 10.46254/wc01.20240164.

[13] Danny De Vleeschauwer, Chrysa Papagianni, and Anwar Walid. "Decomposing SLAs for Network Slicing". In: *IEEE Communications Letters* 25.3 (2021), pp. 950–954. DOI: 10.1109/LCOMM.2020.3033042.

[14] Edward L. Deci and Richard M. Ryan. "Self-Determination Theory and the Facilitation of Intrinsic Motivation, Social Development, and Well-Being". In: *American Psychologist* 55.1 (2000), pp. 68–78. DOI: 10.1037/0003-066X.55.1.68.

[15] Aleksandra Denda and Dejan Drajić. "Aligned Software and Process Models with Both the eTOM Framework and the ITIL Processes". In: *Proc. IEEE Int. Conf. on Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS)*. 2013, pp. 647–650. DOI: 10.1109/TELSKS.2013.6704461.

[16] Sebastian Deterding et al. "From Game Design Elements to Gamefulness: Defining "Gamification"". In: *Proc. 15th Int. Academic MindTrek Conf.: Envisioning Future Media Environments*. 2011, pp. 9–15. DOI: 10.1145/2181037.2181040.

[17] Eric Evans. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley, 2004. ISBN: 0-321-12521-5.

[18] Martin Fowler. *CQRS*. Available at https://martinfowler.com/bliki/CQRS.html. 2011.

[19] Martin Fowler. *Event Sourcing*. Available at https://martinfowler.com/eaaDev/EventSourcing.html. 2005.

[20] Grand View Research. *IT Service Management Market Size, Share & Trends Analysis Report, 2025–2030*. Grand View Research. 2024. URL: https://www.grandviewresearch.com/industry-analysis/it-service-management-market-report (visited on 06/03/2026).

[21] Juho Hamari, Jonna Koivisto, and Harri Sarsa. "Does Gamification Work? — A Literature Review of Empirical Studies on Gamification". In: *Proc. 47th Hawaii Int. Conf. on System Sciences (HICSS)*. 2014, pp. 3025–3034. DOI: 10.1109/HICSS.2014.377.

[22] Nabil Hattab and Ghalem Belalem. "Modular Models for Systems Based on Multi-Tenant Services: A Multi-Level Petri-Net-Based Approach". In: *Journal of King Saud University – Computer and Information Sciences* 35.8 (2023), p. 101671. DOI: 10.1016/j.jksuci.2023.101671.

[23] Alan R. Hevner et al. "Design Science in Information Systems Research". In: *MIS Quarterly* 28.1 (2004), pp. 75–105. DOI: 10.2307/25148625.

[24] Robin Hunicke, Marc LeBlanc, and Robert Zubek. "MDA: A Formal Approach to Game Design and Game Research". In: *Proc. AAAI Workshop on Challenges in Game AI*. Vol. 4. 1. 2004, p. 1722.

[25] Sami Hyrynsalmi, Jouni Smed, and Kai K. Kimppa. "The Dark Side of Gamification: How We Should Stop Worrying and Study also the Negative Outcomes of Bringing Game Design Elements to Everywhere". In: *Proc. 1st Int. GamiFIN Conf.* 2017, pp. 96–104.

[26] Jon Iden and Tom Roar Eikebrokk. "Implementing IT Service Management: A Systematic Literature Review". In: *International Journal of Information Management* 33.3 (2013), pp. 512–523. DOI: 10.1016/j.ijinfomgt.2013.01.004.

[27] Marko Jäntti. "Improving Incident Management Processes in Two IT Service Provider Companies". In: *Proc. 22nd International Workshop on Database and Expert Systems Applications (DEXA Workshops)*. 2011, pp. 26–30. DOI: 10.1109/DEXA.2011.42.

[28] Barbara Kitchenham and Stuart Charters. *Guidelines for Performing Systematic Literature Reviews in Software Engineering*. Tech. rep. EBSE-2007-001. Keele University, 2007.

[29] Martin Kleppmann. *Designing Data-Intensive Applications*. O'Reilly Media, 2017. ISBN: 978-1-449-37332-0.

[30] Jonna Koivisto and Juho Hamari. "The Rise of Motivational Information Systems: A Review of Gamification Research". In: *International Journal of Information Management* 45 (2019), pp. 191–210. DOI: 10.1016/j.ijinfomgt.2018.10.013.

[31] Patricia Marcu and Wolfgang Hommel. "Requirements and Concepts for an Inter-Organizational Fault Management Architecture". In: *Proc. 9th RoEduNet International Conference*. Sibiu, Romania: IEEE Computer Society, 2010, pp. 322–327.

[32] Patricia Marcu and Thomas Schaaf. "An Information Model for Inter-Organizational Fault Management". In: *Proc. IEEE/IFIP Int. Symp. on Integrated Network Management (IM)*. 2011, pp. 1043–1049. DOI: 10.1109/INM.2011.5990502.

[33] Patricia Marcu, Larisa Shwartz, and Genady Grabarnik. "Model for Incident Ticket Correlation for Inter-Organizational Service Delivery". In: *Proc. eChallenges e-2009 Conference*. Istanbul, Turkey: IIMC International Information Management Corporation, 2009, pp. 1–8.

[34] Patricia Marcu et al. "Managing Faults in the Service Delivery Process of Service Provider Coalitions". In: *Proc. IEEE International Conference on Services Computing (SCC)*. 2009, pp. 65–72. DOI: 10.1109/SCC.2009.41.

[35] Mauricio Marrone and Lutz M. Kolbe. "Impact of IT Service Management Frameworks on the IT Organization: An Empirical Study on Benefits, Challenges and Processes". In: *Business & Information Systems Engineering* 3.1 (2011), pp. 5–18. DOI: 10.1007/s12599-010-0141-5.

[36] Antoni Lluís Mesquida et al. "IT Service Management Process Improvement Based on ISO/IEC 15504: A Systematic Review". In: *Information and Software Technology* 54.3 (2012), pp. 239–247. DOI: 10.1016/j.infsof.2011.11.002.

[37] Benedikt Morschheuser et al. "Gamified Crowdsourcing: Conceptualization, Literature Review, and Future Agenda". In: *International Journal of Human-Computer Studies* 106 (2017), pp. 26–43. DOI: 10.1016/j.ijhcs.2017.04.005.

[38] Oscar Pedreira et al. "Gamification in Software Engineering — A Systematic Mapping". In: *Information and Software Technology* 57 (2015), pp. 157–168. DOI: 10.1016/j.infsof.2014.08.007.

[39] Ken Peffers et al. "A Design Science Research Methodology for Information Systems Research". In: *Journal of Management Information Systems* 24.3 (2007), pp. 45–77. DOI: 10.2753/MIS0742-1222240302.

[40] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. "Guidelines for Conducting Systematic Mapping Studies in Software Engineering: An Update". In: *Information and Software Technology* 64 (2015), pp. 1–18. DOI: 10.1016/j.infsof.2015.03.007.

[41] Kai Petersen et al. "Systematic Mapping Studies in Software Engineering". In: *Proc. 12th Int. Conf. on Evaluation and Assessment in Software Engineering (EASE)*. BCS Learning & Development, 2008, pp. 68–77.

[42] Chris Richardson. *Microservices Patterns: With Examples in Java*. Manning Publications, 2018. ISBN: 978-1617294549.

[43] Richard M. Ryan and Edward L. Deci. “Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions”. In: *Contemporary Educational Psychology* 25.1 (2000), pp. 54–67. DOI: 10.1006/ceps.1999.1020.

[44] Michael Sailer et al. “How Gamification Motivates: An Experimental Study of the Effects of Specific Game Design Elements on Psychological Need Satisfaction”. In: *Computers in Human Behavior* 69 (2017), pp. 371–380. DOI: 10.1016/j.chb.2016.12.033.

[45] Saeed Salah et al. “A Model for Incident Tickets Correlation in Network Management”. In: *Journal of Network and Systems Management* 24.1 (2016), pp. 57–91. DOI: 10.1007/s10922-014-9340-6.

[46] Rodrigo Pereira dos Santos and Esteban Walter Gonzalez Clua. “Simulation Games”. In: *Body of Knowledge for Modeling and Simulation*. Ed. by Tuncer Ören, Bernard P. Zeigler, and Andreas Tolk. Simulation Foundations, Methods and Applications. Cham: Springer, 2023. DOI: 10.1007/978-3-031-11085-6_5.

[47] Scopism. *SIAM Foundation Body of Knowledge*. Scopism. 2020. URL: https://www.scopism.com/wp-content/uploads/2021/03/SIAM-Foundation-Body-of-Knowledge-2nd-edition.pdf (visited on 06/05/2026).

[48] ServiceNow. *Domain Separation*. ServiceNow Product Documentation (Zurich release). 2025. URL: https://servicenow.com/docs/bundle/zurich-platform-security/page/administer/company-and-domain-separation/concept/c_DomainSeparation.html (visited on 03/01/2025).

[49] ServiceNow. *Using Remote Tasks to Fulfill Consumer Requests*. ServiceNow Product Documentation. 2024. URL: https://www.servicenow.com/docs/r/washingtondc/service-bridge/service-bridge-v2-remote-task-overview.html (visited on 06/05/2026).

[50] Deepika Shree et al. “Digital Platforms for Business-to-Business Markets: A Systematic Review and Future Research Agenda”. In: *Journal of Business Research* 137 (2021), pp. 354–365. DOI: 10.1016/j.jbusres.2021.08.031.

[51] Larisa Shwartz et al. “IT Service Provider’s Multi-Customer and Multi-Tenant Environments”. In: *Proc. 9th IEEE Int. Conf. on E-Commerce Technology and 4th IEEE Int. Conf. on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE)*. 2007, pp. 559–566. DOI: 10.1109/CEC-EEE.2007.20.

[52] Wolfgang Theilmann et al. “A Reference Architecture for Multi-Level SLA Management”. In: *Journal of Internet Engineering* 4.1 (2010), pp. 289–298. DOI: 10.21256/zhaw-1757.

[53] Armando M. Toda, Pedro Henrique Dias Valle, and Seiji Isotani. “The Dark Side of Gamification: An Overview of Negative Effects of Gamification in Education”. In: *Higher Education for All. From Challenges to Novel Technology-Enhanced Solutions*. Vol. 832. Communications in Computer and Information Science. Springer, 2018, pp. 143–156. DOI: 10.1007/978-3-319-97934-2_9.

[54] Vaughn Vernon. *Implementing Domain-Driven Design*. Addison-Wesley, 2013. ISBN: 978-0-321-83457-7.

[55] Claes Wohlin et al. *Experimentation in Software Engineering*. Springer, 2012. ISBN: 978-3-642-29043-5. DOI: 10.1007/978-3-642-29044-2.

[56] Shung-Bin Yan and Feng-Jian Wang. “A Cooperative Framework for Inter-Organizational Workflow System”. In: *Proc. 27th Annual International Computer Software and Applications Conference (COMPSAC)*. 2003, pp. 64–71. DOI: 10.1109/CMPSAC.2003.1245323.

[57] Greg Young. “CQRS Documents”. In: (2010). Available at https://cqrs.files.wordpress.com/2010/11/cqrs_documents.pdf.

[58] Zendesk. *Setting Up Ticket Sharing with Other Zendesk Support Accounts*. Zendesk Help. 2024. URL: https://support.zendesk.com/hc/en-us/articles/4408893967514-Setting-up-ticket-sharing-with-other-Zendesk-Support-accounts (visited on 03/01/2025).

Appendix A

Domain Event Catalogue

This appendix enumerates every domain event defined across the bounded contexts of the Flowzao platform. Domain events represent state transitions within aggregates and serve as the primary integration mechanism between bounded contexts in the event-driven architecture described in Chapter 4. A total of **107 domain events** are distributed across seven bounded contexts, as catalogued in Table A.1. The list is generated directly from the event files present in the codebase (`*.event.ts`), one event per file; the class name is rendered in PascalCase, while the source file uses the equivalent kebab-case form.

Table A.1: Complete domain event catalogue organised by bounded context.

Event Name	BC	Description
`TicketCreated`	Ticket	A new ticket is created in an ecosystem.
`TicketUpdated`	Ticket	A ticket's mutable fields are modified.
`TicketStatusChanged`	Ticket	The ticket transitions between lifecycle states.
`TicketResolved`	Ticket	The ticket is marked as resolved with a resolution note.
`TicketReopened`	Ticket	A previously resolved ticket is reopened.
`TicketClosed`	Ticket	The ticket is closed after resolution.
`TicketCancelled`	Ticket	The ticket is cancelled before resolution.
`TicketDeleted`	Ticket	The ticket is permanently deleted.
`TicketAssigned`	Ticket	A user is assigned to the ticket.
`TicketUnassigned`	Ticket	A user assignment is removed from the ticket.

Event Name	**BC**	**Description**
`TicketTransferred`	Ticket	The ticket is transferred between organisations (direct transfer).
`TransferRequested`	Ticket	A formal cross-organisation transfer request is initiated.
`TransferAccepted`	Ticket	A transfer request is accepted by the recipient organisation.
`TransferRejected`	Ticket	A transfer request is rejected by the recipient organisation.
`AssignmentEnded`	Ticket	An ownership assignment ends (with `endAt` timestamp).
`AssignmentUpdated`	Ticket	An assignment's metadata is updated.
`UserReassigned`	Ticket	The user holding an assignment changes within the same organisation.
`CommentAdded`	Ticket	A comment is added to the ticket.
`CommentUpdated`	Ticket	An existing comment is edited.
`CommentDeleted`	Ticket	A comment is removed.
`AttachmentUploaded`	Ticket	A file is attached to the ticket or to a comment.
`AttachmentDeleted`	Ticket	An attachment is removed from the ticket.
`TimeLogged`	Ticket	A time-tracking entry is recorded against the ticket.
`TimeLogUpdated`	Ticket	An existing time-log entry is modified.
`TimeLogDeleted`	Ticket	A time-log entry is removed.
`FeedbackSubmitted`	Ticket	Resolution feedback is submitted for the ticket.
`FeedbackUpdated`	Ticket	A feedback entry is modified after submission.
`FeedbackDeleted`	Ticket	A feedback entry is removed.
`ResolutionEdited`	Ticket	The resolution note is amended after initial resolution.
`ResolutionAttachmentAdded`	Ticket	A file is bound to the ticket's resolution.
`ResolutionAttachment-Removed`	Ticket	A resolution attachment is soft-deleted.
`PlanCreated`	Billing	A new subscription plan is defined.
`PlanUpdated`	Billing	An existing plan's parameters are modified.

Event Name	**BC**	**Description**
`PlanPublished`	Billing	A plan becomes available for new subscriptions.
`PlanUnpublished`	Billing	A plan is hidden from new subscriptions.
`PlanDeactivated`	Billing	A plan is permanently deactivated.
`PlanReactivated`	Billing	A deactivated plan is reactivated.
`SubscriptionCreated`	Billing	An organisation acquires a subscription.
`SubscriptionUpgraded`	Billing	The subscription moves to a higher tier.
`SubscriptionDowngraded`	Billing	The subscription moves to a lower tier.
`SubscriptionCanceled`	Billing	A subscription cancellation is initiated.
`SubscriptionResumed`	Billing	A paused subscription is resumed.
`SubscriptionReactivated`	Billing	A cancelled subscription is reactivated.
`SubscriptionTerminated`	Billing	A subscription is permanently terminated.
`PaymentSucceeded`	Billing	A payment is successfully processed.
`PaymentFailed`	Billing	A payment attempt fails.
`ChargeDisputed`	Billing	A customer disputes a charge.
`ChargeRefunded`	Billing	A charge is refunded to the customer.
`DisputeWon`	Billing	A charge dispute is resolved in favour of the merchant.
`DisputeLost`	Billing	A charge dispute is resolved in favour of the customer.
`EcosystemCreated`	Ecosystem	A new ecosystem is created.
`EcosystemUpdated`	Ecosystem	An ecosystem's metadata is modified.
`EcosystemDeactivated`	Ecosystem	An ecosystem is deactivated.
`EcosystemReactivated`	Ecosystem	A deactivated ecosystem is reactivated.
`EcosystemSoftDeleted`	Ecosystem	An ecosystem is soft-deleted (preserved in history).
`EcosystemJoined`	Ecosystem	An organisation completes the ecosystem-join flow.

Event Name	**BC**	**Description**
`EcosystemMemberAdded`	Ecosystem	An organisation is added as an ecosystem member.
`EcosystemMemberRemoved`	Ecosystem	An organisation is removed from an ecosystem.
`EcosystemMemberLeft`	Ecosystem	An organisation voluntarily leaves an ecosystem.
`EcosystemMembership-Updated`	Ecosystem	A member organisation's role or permissions change.
`EcosystemOwnership-Transferred`	Ecosystem	Ecosystem ownership is transferred to another member.
`EcosystemInviteSent`	Ecosystem	An invitation to join the ecosystem is sent.
`EcosystemInviteResent`	Ecosystem	An invitation is re-sent.
`EcosystemInviteAccepted`	Ecosystem	An invitation is accepted.
`EcosystemInviteRejected`	Ecosystem	An invitation is rejected.
`MemberUserAccessGranted`	Ecosystem	A user is granted access within a member organisation.
`MemberUserAccessRevoked`	Ecosystem	A user's ecosystem access is revoked.
`MemberUserAccessUpdated`	Ecosystem	A user's ecosystem access permissions are updated.
`AchievementCreated`	Gamification	A new achievement is defined.
`AchievementUpdated`	Gamification	An achievement's parameters are modified.
`AchievementActivated`	Gamification	An achievement is made available to members.
`AchievementDeactivated`	Gamification	An achievement is no longer available.
`AchievementUnlocked`	Gamification	A member earns an achievement.
`ActivityRecorded`	Gamification	Member activity is logged for streak and trigger evaluation.
`PointsCredited`	Gamification	Points are added to a member's balance.
`RewardCreated`	Gamification	A new redeemable reward is defined.
`RewardUpdated`	Gamification	A reward's parameters are modified.
`RewardActivated`	Gamification	A reward becomes available for redemption.
`RewardDeactivated`	Gamification	A reward is hidden from the catalogue.

Event Name	**BC**	**Description**
`RedemptionRequested`	Gamification	A member requests a reward redemption.
`RedemptionApproved`	Gamification	A redemption request is approved by a manager.
`RedemptionRejected`	Gamification	A redemption request is rejected.
`RedemptionFulfilled`	Gamification	An approved redemption is delivered to the member.
`OrganizationCreated`	Organisation	A new organisation is created.
`OrganizationUpdated`	Organisation	An organisation's metadata is modified.
`OrganizationDeactivated`	Organisation	An organisation is deactivated.
`OrganizationReactivated`	Organisation	A deactivated organisation is reactivated.
`OrganizationSoftDeleted`	Organisation	An organisation is soft-deleted.
`MemberInvited`	Organisation	A user is invited to join an organisation.
`MemberAdded`	Organisation	A user is added as an organisation member.
`MemberRemoved`	Organisation	A user is removed from an organisation.
`MemberRoleUpdated`	Organisation	A member's role is updated (Owner, Admin, Manager, Technician).
`OwnershipTransferred`	Organisation	Organisation ownership is transferred to another member.
`InviteResent`	Organisation	A pending invitation is re-sent.
`InviteAccepted`	Organisation	A pending invitation is accepted.
`InviteRejected`	Organisation	A pending invitation is rejected.
`InviteRevoked`	Organisation	A pending invitation is revoked by the issuer.
`UserCreated`	User	A new user account is provisioned.
`UserUpdated`	User	A user's profile is modified.
`UserSignedIn`	User	A user successfully authenticates.
`UserSoftDeleted`	User	A user account is soft-deleted.
`UserPreferencesUpdated`	User	A user's preferences (theme, locale, etc.) are updated.
`UserPlatformRoleUpdated`	User	A user's platform-level role is changed.
`NotificationSent`	Notification	A notification is queued for delivery.

Event Name	**BC**	**Description**
`NotificationDelivered`	Notification	A notification is confirmed as delivered.
`NotificationFailed`	Notification	A notification delivery attempt fails.
`NotificationPreferences-Updated`	Notification	A user's notification preferences are updated.

Appendix B

Problem Evaluation Survey Instrument

This appendix reproduces the full instrument deployed as the public *Problem Evaluation Survey* reported in Chapter 3. The questionnaire is administered online at the per-locale URL `/<locale>/problem-survey` on the dissertation's landing site, in four languages (Portuguese, English, Spanish, German). The English version is reproduced here for reference. The Portuguese, Spanish, and German translations preserve the same item structure, option codes, and ordering, and are bundled with the source code of the landing site. The data-side option codes (`operator`, `telco`, `lt_1`, etc.) are language-stable and persisted in the database; the labels rendered to respondents are the localised strings below.

The instrument is organised into five thematic blocks plus a single attention-check item and a mandatory identification field, for a total of 17 substantive items. Estimated completion time is five to seven minutes. Responses that do not satisfy the attention check are excluded from the aggregated analyses but preserved in the database for audit.

Introduction shown to respondents

> *You are invited to take part in this survey as part of a master's dissertation (FEUP, University of Porto). The goal is to understand, based on your professional experience, the reality of managing tickets/incidents that involve multiple organisations. It takes 5 to 7 minutes. Responses are confidential.*

Block A — Background

A1. What is your current role?

- Support technician / operator (`operator`)
- Team lead / incident coordinator (`team_lead`)
- Service / ITSM manager (`service_manager`)
- Architect / platform engineer (`architect`)

- Decision maker: CIO, CTO, Head of Operations (`decision_maker`)
- Other (`other`)

A2. Which sector does your organisation operate in?

- Telecommunications (`telco`)
- Banking / Insurance (`banking_insurance`)
- Healthcare (`healthcare`)
- Logistics / Transportation (`logistics`)
- IT services / Outsourcing (`it_services`)
- Industry (`industry`)
- Public sector (`public_sector`)
- Other (`other`)

A3. For how many years have you dealt with tickets or incidents that involve other organisations?

- Less than 1 year (`lt_1`)
- 1 to 3 years (`1_3`)
- 4 to 7 years (`4_7`)
- 8 or more years (`8_plus`)

A4. In a typical week, with how many external organisations do you coordinate tickets?

- 0 — I do not coordinate with external organisations (`zero`)
- 1 to 2 (`1_2`)
- 3 to 5 (`3_5`)
- 6 to 10 (`6_10`)
- More than 10 (`gt_10`)

Block B — Operational reality

B5. What share of the tickets you handle crosses at least one organisational boundary?

- Less than 10% (`lt_10`)
- 10% to 25% (`10_25`)
- 26% to 50% (`26_50`)
- 51% to 75% (`51_75`)
- More than 75% (`gt_75`)

B6. How is inter-organisational coordination handled today? *(multiple choice)*

- Email (`email`)
- Phone, chat — Teams, Slack, WhatsApp, etc. (`phone_chat`)
- Manual ticket replication between ITSM tools (`manual_replication`)
- API integration between ITSM tools (`api_integration`)
- Shared portal (`shared_portal`)
- No formal process — handled case by case (`ad_hoc`)

B7. What is the primary ITSM tool you use?

- ServiceNow (`servicenow`)
- Jira Service Management (`jira_sm`)
- Zendesk (`zendesk`)
- Freshservice (`freshservice`)
- GLPI (`glpi`)
- Internal / in-house solution (`internal`)
- We do not use a dedicated ITSM tool (`none`)
- Other (`other`)

Block C — Pain validation

Five-point Likert scale: 1 = Strongly disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly agree. *Header shown to respondents: "Thinking about the last 6 months, indicate your level of agreement with each statement."*

C8. I lost important context because information was scattered across multiple systems or channels.
Maps to R1, R7

C9. There were times when I did not know which organisation was responsible for the next step.
Maps to R2

C10. I could not measure the total end-to-end resolution time because each organisation sees only its own segment.
Maps to R4

C11. I had to re-enter manually data that already existed in another system (copy-paste, transcription).
Maps to R7

C12. Conflicts or *ping-pong* occurred between organisations over who was responsible for the next step.
Maps to R2

C13. I had no visibility over the partner's work (whether it was being handled, blocked, or forgotten).
Maps to R3

Attention check (interleaved within Block C)

The following item is shown between two pain items within Block C:

> *Attention check: to confirm your attention, please select option 3 (Neutral) on this line.*

Responses with a value other than 3 are flagged with `attention_check_passed=false` and excluded from the aggregated figures reported in Chapter 3.

Block D — Desirability and priorities

D14. On a scale of 1 to 5, how valuable would it be for you to have a platform where the ticket exists as a single entity shared across all organisations involved, with unified history and end-to-end metrics? *(1 = no value; 5 = would transform my day-to-day work)*

D15. Rate the importance of each of the following capabilities, from 1 (not important) to 5 (very important):

- Canonical ticket shared across organisations *(maps to R1)*
- Formal events for transfers of responsibility *(maps to R2)*
- End-to-end metrics — total time, time per organisation, number of handoffs *(maps to R4)*
- Configurable visibility per organisation — each controls what it shares *(maps to R3)*
- Multi-tenant — each organisation has its own workspace but shares tickets *(maps to R6)*
- Composite SLAs across organisations *(maps to R5)*

Block E — Qualitative (optional free text, 500 characters each)

E16. Briefly describe the last *cross-organisational* incident that went wrong and what failed in the coordination.

E17. What would prevent your organisation from joining a shared platform with partners (technical, contractual, data sovereignty, cost, etc.)?

Identification

Email. Professional email address. Mandatory. Used only for a possible follow-up contact; stored separately from the analytical responses and not publicly associated with them.

Persistence schema (data side)

The instrument writes one row to the `problem_evaluation_response` table (in the `evaluation` schema) per submission. Each row records:

- the language code used (`pt-PT`, `en-US`, `es-ES`, or `de-DE`) and the email address;
- the categorical answers to A1–A4 (background) and B5–B7 (operational reality);
- the multi-select coordination methods as a JSONB array;
- the six pain Likert values (one column per item C8–C13);
- the attention-check value and the boolean outcome flag;
- the desirability score (D14) and the six capability importance scores (D15);
- the two optional free-text responses (E16, E17);
- the creation timestamp, plus SHA-256 hashes of the client IP and user-agent for soft de-duplication without storing identifying data.

Every figure reported in Chapter 3 is computed directly from this set of responses.

Appendix C

Evaluation Protocol

This appendix presents the evaluation protocol designed for expert evaluation of the Flowzao platform with industry practitioners. The protocol was designed following established guidelines for expert evaluation in Design Science Research [23] and targets practitioners with direct experience in IT service management or multi-organisation support coordination. Its design and rationale are discussed in Section 6.4.

The protocol comprises three parts: participant profiling (Part A), a task-based walkthrough of a complete inter-organisational ticket lifecycle (Part B, six tasks), and a structured questionnaire covering requirements satisfaction, gamification assessment, the System Usability Scale [8], and comparison with current ITSM tools (Part C).

The full protocol document follows on the next page.

Flowzao Platform Evaluation Protocol

Expert Evaluation of a Multi-Organisation Ticket Management Platform
for B2B Service Ecosystems

Salif Henrique Faustino
Faculty of Engineering, University of Porto
Master in Software Engineering

2026

About This Evaluation

Thank you for participating in this evaluation. You are being asked to evaluate **Flowzao**, a multi-tenant platform for managing support tickets across multiple organisations in B2B service ecosystems. The platform was developed as part of a master's dissertation at FEUP.

The evaluation consists of three parts:

1. **Part A** — Participant profile (2 minutes)
2. **Part B** — Task-based evaluation with the platform (15–20 minutes)
3. **Part C** — Questionnaire (10 minutes)

Estimated total time: 25–30 minutes.
Your responses are anonymous and will be used exclusively for academic purposes.

Part A — Participant Profile

A1. **What is your current role?**

◯ Software Engineer/Developer ◯ IT Support/Helpdesk ◯ IT Manager
◯ Product Manager ◯ Student ◯ Other: _____

A2. **How many years of professional experience do you have?**

◯ 0–1 ◯ 2–5 ◯ 6–10 ◯ 10+

A3. **How familiar are you with IT Service Management (ITSM) tools?**
(e.g., Jira Service Management, ServiceNow, Zendesk, Freshdesk)

◯ Never used ◯ Basic familiarity ◯ Regular user ◯ Expert/Administrator

A4. **Have you experienced the problem of coordinating tickets across multiple organisations?**

◯ Yes, frequently ◯ Yes, occasionally ◯ No, but I understand the problem ◯ No

Part B — Task-Based Evaluation

You will be given access to the Flowzao platform with a pre-configured scenario. Please complete the following tasks and answer the questions after each task.

Scenario Setup

The platform is pre-configured with:

- **Ecosystem:** "Healthcare Support"
- **Organisation A:** "TechIntegrate" (systems integrator) — you are logged in as a technician here
- **Organisation B:** "CloudCore" (infrastructure provider)
- **Organisation C:** "DataSpecialists" (database vendor)

Task 1: Create a Ticket

Create a new ticket in the "Healthcare Support" ecosystem with:
- Subject: "Database connection timeout on patient portal"
- Priority: High
- Type: Incident
- Assign it to yourself

Were you able to complete this task? ◯ Yes ◯ Partially ◯ No

How easy was it? (1=Very Difficult, 5=Very Easy) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

Comments: _____

Task 2: Add a Comment and Log Time

On the ticket you created:
- Add a comment: "Initial investigation: the timeout occurs during peak hours"
- Log 30 minutes of Active Work time

Were you able to complete this task? ◯ Yes ◯ Partially ◯ No

How easy was it? (1=Very Difficult, 5=Very Easy) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

Task 3: Transfer the Ticket to Another Organisation

The root cause is in the infrastructure layer. Transfer the ticket to "CloudCore" (Organisation B):
- Initiate a transfer request to Organisation B
- (The evaluator will accept the transfer on behalf of Org B)

Were you able to complete this task? ◯ Yes ◯ Partially ◯ No

How easy was it? (1=Very Difficult, 5=Very Easy) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

Comments: _____

Task 4: View the Ticket After Transfer

After the transfer is accepted:

- View the ticket's assignment history
- Verify that the ticket retains its original ID and full history (comments, time logs)
- Check who is now the owner

Were you able to see the full history? ◯ Yes ◯ Partially ◯ No

Is it clear which organisation currently owns the ticket? ◯ Yes ◯ No

Task 5: Resolve the Ticket and Submit Feedback

- Resolve the ticket
- Submit feedback with a rating for each organisation that participated

Were you able to complete this task? ◯ Yes ◯ Partially ◯ No

How easy was it? (1=Very Difficult, 5=Very Easy) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

Task 6: View Gamification Elements

- Check your points/achievements after resolving the ticket
- View the leaderboard for the ecosystem
- Browse the rewards catalogue

Were you able to find the gamification features? ◯ Yes ◯ Partially ◯ No

Are the gamification elements motivating? (1=Not at all, 5=Very) ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

Part C — Questionnaire

C1. Requirements Satisfaction

Please rate how well the platform addresses each of the following requirements.
(1 = Strongly Disagree 5 = Strongly Agree)

Req.	Statement	Rating
R1	The platform provides a single, canonical ticket that is shared across all organisations (not duplicated in each system).	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
R2	Every transfer of responsibility between organisations is explicitly captured as an event with full details (who, when, why).	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
R3	Each organisation can only see the data it is authorised to access, while the platform maintains overall auditability.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
R4	The platform enables measurement of end-to-end resolution time and time spent per organisation.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
R5	The platform supports tracking SLA compliance across multiple organisations involved in resolving a ticket.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
R6	Multiple organisations can coexist on the platform with their own workflows, roles, and configurations.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
R7	The platform serves as a Single Source of Truth — there is one authoritative version of each ticket, not fragmented copies.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

C2. Gamification Assessment

(1 = Strongly Disagree 5 = Strongly Agree)

Code	Statement	Rating				
G1	The gamification elements (points, achievements, leaderboards) are well integrated into the ticket workflow.	◯ 1	◯ 2	◯ 3	◯ 4	◯ 5
G2	The gamification mechanics would motivate me to resolve tickets more efficiently.	◯ 1	◯ 2	◯ 3	◯ 4	◯ 5
G3	The gamification system feels fair across organisations of different sizes.	◯ 1	◯ 2	◯ 3	◯ 4	◯ 5
G4	The gamification elements feel voluntary and non-coercive (I would not feel forced to participate).	◯ 1	◯ 2	◯ 3	◯ 4	◯ 5
G5	The rewards catalogue provides meaningful incentives.	◯ 1	◯ 2	◯ 3	◯ 4	◯ 5

C3. System Usability Scale (SUS)

Please rate each statement based on your experience with the platform.
(1 = Strongly Disagree 5 = Strongly Agree)

Code	Statement	Rating
S1	I think that I would like to use this system frequently.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S2	I found the system unnecessarily complex.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S3	I thought the system was easy to use.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S4	I think that I would need the support of a technical person to be able to use this system.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S5	I found the various functions in this system were well integrated.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S6	I thought there was too much inconsistency in this system.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S7	I would imagine that most people would learn to use this system very quickly.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S8	I found the system very cumbersome to use.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S9	I felt very confident using the system.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5
S10	I needed to learn a lot of things before I could get going with this system.	◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ 5

C4. Comparison with Current Tools

C4.1. Compared to the ITSM tools you currently use (if any), how does Flowzao handle **cross-organisation ticket management**?

◯ Much worse ◯ Worse ◯ Similar ◯ Better ◯ Much better ◯ N/A

C4.2. What is the **single most valuable feature** of the platform?

C4.3. What is the **biggest limitation or missing feature**?

C4.4. Would you recommend this platform to an organisation that manages tickets across multiple partner companies?

◯ Definitely not ◯ Probably not ◯ Neutral ◯ Probably yes ◯ Definitely yes

C4.5. Any additional comments or suggestions?

Thank you for your participation!
Your feedback is invaluable for this research.

Appendix D

Data Model

This appendix consolidates the persistence-side view of the platform: the ten PostgreSQL schemas, the 36 tables they contain, and the principal foreign-key relationships between them. The data model implements the domain model described in Chapter 4; one schema per bounded context plus the cross-cutting outbox, address, and evaluation schemas.

Bounded-context overview

Figure D.1 shows the seven bounded contexts at the persistence level, with the principal aggregates each context owns. Solid arrows denote cross-context foreign-key references (carried as identifiers, not as physical joins, in keeping with the loose-coupling principle of Chapter 4); dashed arrows denote asynchronous domain-event flows mediated by the transactional outbox. The `better_auth` schema is excluded from the count because it is owned by the authentication library rather than by the domain model.

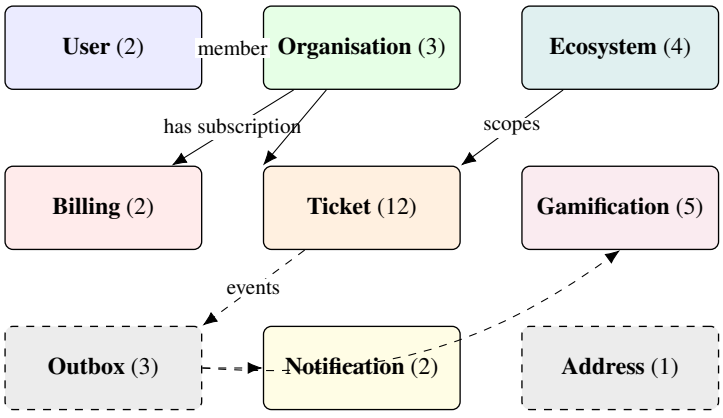


Figure D.1: Persistence-layer bounded-context map.

Schema-by-schema catalogue

Schema `user` (2 tables)

`user` Core user record (identity, display name, audit timestamps).

`user_preference` Per-user UI preferences (theme, locale).

Schema `organization` (3 tables)

`organization` Tenant root; owns memberships and invitations.

`organization_membership` Maps users to organisations with a role (Owner, Admin, Manager, Technician); enforces single-owner and last-owner-protection invariants.

`organization_invitation` Pending join invitations with token, expiry, and status.

Schema `ecosystem` (4 tables)

`ecosystem` Multi-organisation collaboration space.

`ecosystem_member` Membership of a member organisation in an ecosystem (with category-based permission grants).

`ecosystem_user_access` User-level access entries that intersect with the membership grant to determine effective permissions.

`ecosystem_invitation` Pending invitations to join an ecosystem.

Schema `ticket` (12 tables)

`ticket` Canonical ticket aggregate root: status, priority, subject, identifiers, audit timestamps.

`ticket_assignment` Per-organisation participation (Owner / Contributor) with `startAt` and `endAt` timestamps; the substrate for per-organisation time metrics.

`ticket_comment` Comments with a visibility level (internal, customer-visible, partner-visible).

`ticket_attachment` Files attached to comments or to the ticket directly.

`ticket_time_log` Time-tracking entries (Active Work, Waiting, Research, Communication).

`ticket_feedback` Resolution feedback container (one row per ticket).

`feedback_entry` Per-assignment feedback rating and points awarded.

`ticket_event` Lifecycle audit log (status transitions, assignments, transfers).

`ticket_lifecycle` Resolution and closure timestamps used for end-to-end metrics.

`transfer_request` Formal cross-organisation transfer request (Pending → Accepted / Rejected).

`ticket_resolution_attachment` Files bound specifically to the resolution (soft-deletable).

`ticket_resolution_history` Immutable per-version snapshots of the resolution note, preserving edit provenance.

Schema `billing` (2 tables)

`plan` Subscription tier definitions (Free, Standard, Enterprise) with resource limits.

`subscription` Per-organisation subscription with state machine (Trialing, Active, PastDue, Unpaid, Cancelled, Terminated) and Stripe identifiers.

Schema `gamification` (5 tables)

`achievement` Achievement definitions (rarity, points, trigger type, parameters).

`reward` Redeemable rewards (category, point cost, optional stock limit).

`member_profile` Per-ecosystem-member balance (points, streak, last activity, earned achievements).

`reward_redemption` Redemption lifecycle (Requested → Approved / Rejected → Fulfilled).

`member_achievement` Many-to-many join recording each unlock event.

Schema `notification` (2 tables)

`notification` Dispatch record (event source, recipient, channel, delivery status).

`notification_preference` Per-user opt-in/out per notification type.

Schema `evaluation` (2 tables)

`evaluation_response` In-ecosystem pilot questionnaire submissions (see Appendix C).

`problem_evaluation_response` Public Problem Evaluation Survey submissions (see Appendix B).

Schema `outbox` (3 tables)

`outbox_event` Domain events with JSONB payload, status, and ordering metadata.

`outbox_event_subscriber` Per-handler idempotency tracking to guarantee exactly-once delivery semantics.

`outbox_event_subscriber_execution` Per-attempt execution log (success, failed, retry).

Schema `address` (1 table)

`address` Normalised address entries reused across organisations and billing.

Cross-cutting design decisions

Identifier-only cross-context references. Foreign keys cross schema boundaries only as identifiers; aggregate root invariants are enforced *inside* the owning bounded context, never by joins from another context. This preserves the autonomy of each bounded context and aligns with the modular-monolith stance defended in Chapter 2, Section 2.5.2.

UUID primary keys. All primary keys are UUIDs to avoid coordination on sequence generation across schemas and to keep ticket identifiers stable across organisational handoffs (the core mechanism behind R1).

Soft-delete where audit matters. Resolution attachments, comments, and other artefacts that affect history use soft-delete (`deletedAt`, `deletedByUserId`) rather than hard deletion, preserving the audit trail required by R3.

Outbox-mediated integration. No bounded context queries another's tables directly. Every cross-context reaction is mediated by the outbox, producing the event-driven topology described in Chapter 4.

Appendix E

Canonical Inter-Organisational Aggregate: Pattern Card

This appendix presents the *Canonical Inter-Organisational Aggregate* (CIOA) pattern in standalone pattern-card form, following the structural conventions established by the design-pattern community (Intent, Context, Forces, Solution, Implementation, Consequences, Known Uses, See Also). The card is intended as a self-contained reference for practitioners and researchers who wish to apply the pattern in domains beyond IT Service Management, without having to read the full architecture chapter. The full discussion — with figures, code, and the rationale that led to each pattern element — is in Chapter 4, Section 4.1.

Pattern name. *Canonical Inter-Organisational Aggregate* (CIOA).

Category. Architectural pattern; domain modelling.

Also known as. Shared canonical entity; single source of truth across tenants.

Intent

Provide a single authoritative representation of an entity whose lifecycle spans multiple autonomous organisations, preserving its identity, history, and end-to-end metrics across organisational boundaries while allowing each organisation to participate, contribute, and observe under organisation-specific visibility rules.

Context

A long-lived entity (a ticket, a clinical case, a legal matter, a logistics shipment, a security incident) is acted upon over time by two or more autonomous organisations. The dominant industry practice replicates the entity into each organisation's system of record and treats integration as a synchronisation problem, with the consequence that identity is lost at every handoff and end-to-end metrics cannot be computed without cross-system reconciliation.

Forces

F1. Multi-party lifecycle. The entity is acted upon by two or more autonomous organisations whose participation is sequential, overlapping, or both.

F2. Identity preservation across boundaries. Stakeholders need to refer to the entity by a stable identifier regardless of which organisation currently holds operational responsibility.

F3. Asymmetric visibility requirements. Different organisations have different rights to see different facets of the entity (commercially sensitive comments, internal annotations, partner-visible status).

F4. End-to-end accountability. Claims about resolution time, segment time, handoff frequency, and chain-of-responsibility cannot be reconstructed from per-organisation siloed views; they require a global, ordered timeline.

F5. No single-organisation dominance. No participant has the authority to host the system of record for the others; participants either operate above a shared platform or agree to a federated host.

Solution

Model the entity as a single domain aggregate, owned by a shared multi-tenant platform, with four cooperating elements:

Canonical Aggregate. One aggregate root with a globally unique identifier that never changes through the entity's lifecycle. All state mutations go through the aggregate root; all invariants are enforced inside it.

Participation Collection. A child collection on the aggregate recording, for each organisation that has acted, the period of involvement (`startAt`, `endAt`), the role (e.g., Owner, Contributor, Observer), and the user(s) who acted. The collection is append-only at the participation level.

Routing Events. Two complementary mechanisms produce immutable domain events on every transfer of responsibility: a *Direct Transfer* for unilateral handoff and a *Consent-Based Transfer Protocol* (Request, Accept/Reject) for handoffs that require the recipient's agreement. Both emit events with full provenance.

Tenancy Scope. A higher-level construct (an ecosystem, a coalition, a programme) groups organisations that may collaborate over instances of the aggregate. Authorisation, visibility, and routing are all bounded by tenancy scope; the canonical aggregate does not exist outside one.

Implementation guidelines

- Use a transactional outbox for routing events; state changes and event emission must be atomic.
- Make participations append-only; ending a participation records an end timestamp rather than deleting the entry.
- Distinguish unilateral and consensual transfers; provide both mechanisms.
- Keep visibility orthogonal to ownership: tenancy-scope membership grants the right to see the entity exists; per-user permissions determine which facets are visible.
- Resist denormalising the canonical aggregate; place organisation-specific views in CQRS read models.

Consequences

Positive. (i) Identity preservation is structural rather than conventional. (ii) End-to-end accountability is computable from participation timestamps and routing events without cross-system joins. (iii) Adding a new participant organisation requires no schema migration in the participants' systems; they need only join the tenancy scope. (iv) Auditability is intrinsic, not bolted on. (v) Gamification, analytics, and notifications can subscribe to the routing-event stream without coupling to the canonical aggregate's internals.

Negative. (i) Participants must operate above a shared platform; organisations that demand sovereign systems of record cannot be accommodated by the canonical variant. (ii) The canonical aggregate concentrates authority: the platform's availability and integrity are critical infrastructure for all participants. (iii) Denormalisation pressure on the canonical aggregate is real and requires discipline to resist. (iv) Authorisation and visibility logic concentrate complexity; every read path must filter by tenancy scope and per-user permissions.

Known uses

Inter-organisational ITSM (this dissertation). Flowzao models the cross-organisational ticket as the canonical aggregate. Participations are organisations' Owner or Contributor assignments, routing events capture every transfer between organisations, and the tenancy scope is the *Ecosystem*.

Inter-organisational clinical case management. A patient case that traverses primary care, specialist consultation, and hospital admission. Canonical aggregate is the case; participations are providers' episodes of involvement; routing events are formal referrals and admissions.

Multi-firm legal matters. A legal case shared across firms (lead counsel, local counsel, expert witnesses). Canonical aggregate is the matter; routing events are formal substitutions and transfers of representation.

Cross-carrier logistics shipments. A shipment traversing road, ocean, and last-mile carriers. Canonical aggregate is the shipment with its global tracking identifier; routing events are intermodal handoffs.

Multi-vendor incident response in cyber security. A security incident involving the affected enterprise, a managed-detection vendor, an incident-response firm, and law enforcement. Routing events are formal escalations and containment handoffs.

Anti-patterns and variations

Anti-pattern: Master-Slave Synchronisation. The dominant industry approach treats one organisation's system as the master and replicates copies into the partners' systems. This violates F2 (each replica has its own identifier) and F4 (timestamps are reconciled, not authoritative).

Variation: Federated Canonical Aggregate. For domains where no single platform can host the canonical aggregate (e.g., regulatory or sovereignty constraints), a federated variant locates the aggregate at one of the participating organisations and uses notarised, append-only events shared via a verifiable log (blockchain or trusted notary). The pattern's core invariants are preserved; only the hosting model changes.

Variation: Read-Side Canonical View. Each participant retains a sovereign system of record but agrees to publish a normalised event stream into a shared read-side canonical view. Weaker than the full pattern (mutations remain per-organisation) but recovers end-to-end accountability and unified analytics. A stepping stone for organisations not yet ready to delegate write authority.

See also

Aggregate [17]. Provides the tactical building block for the canonical entity.

Bounded Context [17]. Defines the boundary inside which the canonical aggregate is the single source of truth.

Transactional Outbox [42]. Operational mechanism for the atomic state-change and event emission required by Implementation guideline 1.

CQRS [57]. Pairs naturally with the pattern: the canonical aggregate is the write model; per-organisation projections are the read models.

Tenant [6]. Related to Tenancy Scope; CIOA adapts the multi-tenant idea to multi-organisation collaboration over a shared aggregate.

Appendix F

Detailed Review Selection and Supporting Data

This appendix documents the detailed study-selection process of the Systematic Literature Review (Chapter 2) and the availability and anonymisation of the data supporting the empirical evaluation (Chapters 3 and 6).

Detailed study selection

Table F.1 expands the PRISMA-inspired flow of Figure 2.1 into per-stage record counts and the criterion applied at each stage. The identification stage aggregates three databases (Scopus: 187; IEEE Xplore: 143; ACM Digital Library: 94). The inclusion and exclusion criteria (IC1–IC4, EC1–EC3) are defined in Table 2.1, and the quality-assessment rubric (QA1–QA5, scored 0–2, threshold $5/10$) in Table 2.2.

Stage	**Retained**	**Excluded**	**Criterion**
Identification (Scopus + IEEE + ACM)	424	—	Search string, 2003–2025
Duplicate removal	312	112	Duplicate records
Title & abstract screening	74	238	EC1–EC3
Full-text eligibility	27	47	IC1–IC4
Quality assessment	20	7	Score $< 5/10$ (QA1–QA5)
Classification	16 + 4	—	16 primary, 4 secondary

Table F.1: Detailed study-selection funnel (per-stage counts and criteria).

Of the 20 studies that passed quality assessment, 16 are primary studies analysed in depth (Table 2.3) and 4 are secondary studies (systematic reviews) retained as background context (Table 2.4). The per-study quality-assessment scores (QA1–QA5 for each retained study) and the list of full-text-excluded records with their exclusion reasons are available from the author on request, so that the selection can be audited independently.

Supporting data and anonymisation

The datasets underlying the empirical results of Chapters 3 and 6 — the responses to the industry problem survey and to the in-ecosystem pilot questionnaire — are available from the author on reasonable request, in de-identified form.

In line with the consent obtained from participants, direct identifiers are removed before any dataset is shared: contact details, collected solely as an optional follow-up channel and kept separate from the analysis, are stripped; organisations are not named; and each response is referenced only by a pseudonymous code. No free-text field that could re-identify a participant or a named organisation is disclosed beyond the quotations already reported (and translated) in the body of the dissertation. Given the homogeneity of the pilot cohort and the removal of direct identifiers, the residual re-identification risk of any shared dataset is low.

The study relied on human participants; as noted in Section 6.5, participation was voluntary and based on informed consent, and no submission to a formal institutional ethical review board was made — a step identified as a prerequisite for the larger-scale studies proposed as future work.