RRS-002

Modern SaaS Architecture for Operational Platforms

Enterprise Software | Multi-Tenant Systems | Product Architecture

Author	Axel Vargas - Founder, RADPR
Edition	Publication-ready working edition - July 2026
Series	RADPR Research Series

Abstract

A practical architecture for building multiple industry platforms from a reusable SaaS core, covering shared services, vertical templates, tenant isolation, configuration strategy, deployment operations, and product governance.

Executive Summary

Operational software is frequently rebuilt as isolated applications even when authentication, roles, files, notifications, dashboards, and reporting are substantially the same. A modular multi-tenant architecture reduces duplicated work while preserving vertical specialization.

- Shared core services.
- Strict tenant isolation.
- Configurable modules and workflow states.
- Industry templates.

1. Architecture Principles

The platform should keep shared infrastructure stable while allowing vertical modules to evolve independently. Configuration should handle ordinary variation; custom code should be reserved for true operational differences.

- Modularity over duplication.
- Observable actions and audit trails.
- Secure defaults.
- Data portability.

2. Core Platform Services

Core services form the reusable layer available to every vertical.

- Authentication and session security.
- Tenant, user, role, and permission management.
- Files, notes, tasks, notifications, and activity logs.
- Reports, dashboards, APIs, imports, and exports.

3. Vertical Templates

A vertical template translates the common platform into the language and workflow of an industry.

- Healthcare: residents, meals, restrictions, inventory, family access.
- Legal: cases, hearings, readiness, deadlines, documents, staff.
- Repair: customers, tickets, parts, payments, closeout, recharges.

4. Multi-Tenant Security

Tenant isolation must exist in database queries, file paths, background jobs, caches, and reporting. Every access decision should consider user permission and tenant ownership.

- Central authorization helpers.

- Tenant-scoped queries by default.
- Audit of cross-tenant administrative access.
- Separate external integration credentials.

5. Deployment and Operations

A SaaS platform needs repeatable deployment, backups, monitoring, migrations, incident response, and support processes.

- Restoration-tested backups.
- Environment separation.
- Schema migration controls.
- Error monitoring and audit review.

6. Product Strategy

The economic advantage comes from reusing proven modules while selling and supporting outcomes for specific industries.

- Validate one operational problem at a time.
- Use pilots to refine the template.
- Measure adoption and benefit.
- Move repeatable requests into the core.

Conclusion

A reusable core does not eliminate customization. It makes customization more disciplined. The strongest architecture is stable at the foundation, flexible at the workflow layer, and clear about which capabilities belong to the shared platform.

References and Source Context

- RADPR RADCore architecture and prototype experience across healthcare, legal, and repair-service verticals.
- General secure software design and multi-tenant application principles.