PIRANHA CMS

André Oliveira, Alexandre Cotorobai,
Hugo Correia, Joaquim Rosa

Scenario 2

Table of contents

01.
System Analysis & Project Vision

Current State Analysis

- Piranha CMS is a lightweight, flexible CMS built on ASP.NET Core
- Strong in simplicity and efficiency for basic content management
- Supports traditional CMS and headless implementations

Current limitations:

- Basic content states (New, Draft, Unpublished, Published)
- No multi-stage approval workflows
- Lacks collaborative editing features
- Basic permission model without state-specific permissions

Strategic Goals & Quality Attributes

Vision: Transform Piranha CMS into an enterprise-grade editorial platform orchestrating complex content approval workflows while maintaining performance excellence.

Strategic Goals:

- Advanced Multi-stage Content States and Intelligent Permissions
- Seamless, Context-aware Content Handoffs
- Enterprise-grade Scalability and Performance

02. Architectural Design Methodology

Architectural Design Methodology

Selection of Attribute-Driven Design (ADD)

Why ADD was chosen:

- Explicitly prioritizes quality attributes from the outset
- Provides iterative approach for progressive evolution
- Integrates stakeholders throughout the design process
- Preserves integrity of existing architectural layers

Comparison with alternatives:

- Architecture-Centric Design Method (ACDM) - Less explicit focus on quality attributes
- TOGAF ADM - Excessive formality and documentation for our medium-scale project

Implementation Process:

1. Reviewing inputs
2. Iteration goal & Input Selection
3. Choosing Elements to Refine
4. Choosing Design Concepts
5. Instantiate Architectural Elements
6. Sketching Views & Recording Design Decisions
7. Performing Analysis & Review

03. Domain-Driven Design Approach

Strategic Domain Analysis

Core Domain:

- Editorial Workflow

Supporting Domains:

- Content Management
- User & Permissions
- Audit & History

Generic Domain:

- Notification

Domain Models

Core Domain:

Editorial Workflow: most significant business value and competitive differentiation

- Workflow definition and configuration;
- State management and transitions;
- Approval processes and validation rules;
- Content assignment and task management.

Domain Models

Supporting Domain:

Content Management

- Uses existing Piranha capabilities
- Adds support to workflow integration

User & Permissions

- Authentication and workflow authorization for secure operations
- Implemented this domain with a combination of existing framework capabilities and custom extensions

Audit & History

- Tracking content changes and workflow activities
- Enables compliance and traceability

Domain Models

Generic Domain:

Notification

- Manages communication about workflow events

04.

Cross-Cutting Concerns

Cross-Cutting Concerns

Logging and Error Handling

- Standardized log formats and structured errors
- Centralized aggregation for traceability
- Uniform capture of failures and state changes

Security

- Role-based access control (RBAC) across services
- Encrypted communication and secure APIs
- Centralized authentication and authorization policies

Performance

- Caching strategies and minimal synchronous dependencies
- Shared performance metrics across modules
- Resource contention reduction for system responsiveness

Scalability

- Stateless processing and asynchronous communication
- Horizontal scalability at service boundaries
- Load balancing for demand-driven growth

Observability

- Unified collection of metrics, logs, and traces
- Integration with Prometheus, Jaeger, and Grafana
- End-to-end visibility for monitoring and root cause analysis

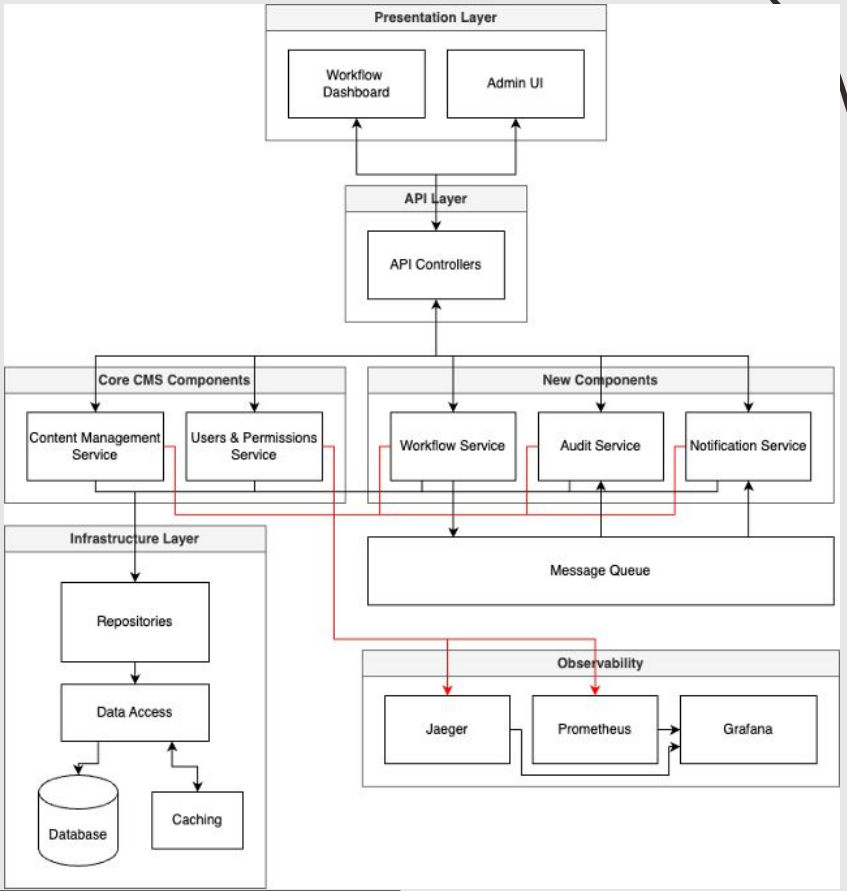
05.

Architecture & Roadmap

Architecture

High-Level Architecture

- **Presentation Layer:** Workflow Dashboard, Admin UI
- **API Layer:** RESTful endpoints for all operations
- **Application Services Layer:** Core CMS Components and New Components
- **Infrastructure Layer:** Data access, caching, repositories
- **Message Queue**
- **Observability**



Architecture

Key Architectural Decisions

1. Clean Separation of Concerns
2. Event-Driven Communication between domains
3. Stateless Services for scalability

Roadmap

Week 1: Project Setup and Domain Modeling
- Establish foundation and model core workflow structures

Week 2: Workflow Engine and Services
- Develop state machine logic and supporting application services

Week 3: Permissions and Audit Features
- Implement role-based access and content traceability

Week 4: UI Integration and Finalization
- Integrate with CMS UI and deliver templates and documentation

Outcome: Deliver an enterprise-ready workflow module with state management, permissions, and full audit logging

Thanks

Software Architecture 2025