

Practical Software Engineering & Project Management Guidelines

Enterprise-grade Development Framework

Emmanuel BRUNO
Université de Toulon, LIS UMR CNRS 7020
emmanuel.bruno@univ-tln.fr

2026-10-03

Table des matières

1	Project Framework Overview	1
1.1	Purpose & Benefits	1
1.2	Framework Phases & Deliverables	2
1.2.a	1. Project Initiation	2
1.2.b	2. Requirements & Planning	2
1.2.c	3. Development & Quality	2
1.2.d	4. Release & Operations	2
1.3	Quick Navigation	2
1.3.a	1. Project Management	2
1.3.b	2. Engineering Practices	2
1.4	Core Principles	2
1.4.a	Engineering Excellence	2
1.4.b	Agile Values	3
1.5	Required Documentation	3
1.5.a	By Project Phase	3
1.6	Process Workflows	3
1.6.a	1. Requirements Process	3
1.6.b	Documentation Types	3
1.6.c	Traceability Framework	3
1.6.d	2. Development Process	3
1.6.e	Agile Workflow States	3
1.6.f	3. Quality Assurance	4
1.6.g	4. Release Management Process	4
1.7	Team Organization	4
1.7.a	Internal Roles	4
1.8	Tools & Infrastructure	4

1.8.a	Required Tools	4
1.9	Maintenance & Support	4
1.9.a	Post-Release Activities	4
1.10	Appendices	5
1.10.a	A. Templates Index	5
1.10.b	B. Process Checklists	5
1.10.c	C. Metrics Dashboard	5

1 Project Framework Overview

This practical framework provides tested patterns for software engineering and agile project management. Whether you're starting a new project or improving an existing one, you'll find actionable guidelines and concrete examples.

1.1 Purpose & Benefits

- **For Teams:** Standardize development practices and improve collaboration
- **For Leaders:** Guide technical decisions and project governance
- **For Organizations:** Ensure quality, maintainability, and scalability

1.2 Framework Phases & Deliverables

1.2.a 1. Project Initiation

- Guides:
 - Project Initiation
 - Participation Roles
- Deliverables:
 - Project Charter template
 - Team Structure guide
- Metrics:
 - Team onboarding time
 - Environment setup time
- Owner: Project Manager
- Review Triggers:
 - Major scope changes
 - Team composition changes
 - Business strategy shifts

1.2.b 2. Requirements & Planning

- Deliverables:
 - System Requirements template
 - Technical Design template
- Metrics:
 - Requirements clarity score
 - Design review completion
- Owner: Product Owner
- Review Triggers:
 - New feature additions
 - Architecture changes
 - External dependency changes

1.2.c 3. Development & Quality

- Deliverables:
 - Development Guidelines template
 - Quality Standards template
- Metrics:
 - Code coverage
 - Technical debt ratio
 - Build success rate
- Owner: Tech Lead
- Review Triggers:
 - New technology adoption

- Quality metric changes
- Build pipeline updates

1.2.d 4. Release & Operations

- Deliverables:
 - Release Plan template
 - Deployment Guide template
- Metrics:
 - Deployment frequency
 - Mean time to recovery
 - Change failure rate
- Owner: Release Manager
- Review Triggers:
 - New deployment target
 - Infrastructure changes
 - Release process changes

1.3 Quick Navigation

1.3.a 1. Project Management

- Project Initiation
- Requirements Engineering
- Agile Process
- Team Organization
- Performance Metrics

1.3.b 2. Engineering Practices

- Solution Architecture
- Engineering Standards
- Quality Engineering
- Delivery & Deployment

1.4 Core Principles

1.4.a Engineering Excellence

1. **Quality Engineering**
 - Test-driven development (TDD)
 - Continuous integration practices
 - Automated testing strategy
 - Quality gates and metrics
2. **Architecture & Design**
 - Domain-driven design
 - Clean architecture principles
 - Microservices patterns

- API-first development
3. **Development Standards**
 - Coding standards
 - Code review practices
 - Documentation requirements
 - Security practices
 4. **Release Management**
 - Deployment strategies
 - Version control
 - Release planning
 - Post-deployment monitoring

1.4.b Agile Values

Agile methodologies focus on iterative progress, adaptive planning, and delivering continuous value while fostering collaboration between self-organizing teams and stakeholders.

Key principles include:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

1.5 Required Documentation

1.5.a By Project Phase

1. **Initiation Phase**
 - Required: Project Charter
 - Required: Team Structure
2. **Requirements Phase**
 - Required: System Requirements Specification
 - Required: Technical Architecture
 - Supporting: Functional Requirements
3. **Development Phase**
 - Required: Development Guidelines
 - Required: Quality Standards
4. **Release Phase**
 - Required: Release Plan
 - Required: Deployment Guide

1.6 Process Workflows

1.6.a 1. Requirements Process

See Requirements Phase for deliverables.

1.6.b Documentation Types

1. **Core Documentation**
 - System Requirements Specification
 - System Overview
 - Requirements Glossary
2. **Requirement Types**
 - Functional Requirements
 - Non-functional Requirements
 - User Stories
 - Use Cases
 - Scenarios and Tests

1.6.c Traceability Framework

Re- qui- re- ment ID	Sour- ce	User Sto- ry	De- sign Docu- ment	Im- ple- men- ta- tion	Test Case	Sta- tus
E01- F01- S01	Client Meeting #2	E01- F01- US01	DB Sche- ma v1.2	UserControl	FullMe	F01- ri- TA01 fied

1.6.d 2. Development Process

See Development Phase for metrics.

1.6.e Agile Workflow States

1. **Backlog**
 - Grooming → Estimation Process
 - Refinement → Quality Gates
2. **Sprint Planning**
 - Planning → Ceremony Guide
 - Estimation → Story Points
3. **Development**
 - Daily Updates → Standup Guide
 - Tracking → Sprint Metrics
4. **Review & Testing**
 - Code Review

- Quality Assurance
- User Acceptance

1.6.f 3. Quality Assurance

1.6.f.a Testing Pyramid

1. **Unit Tests ([Target]%)**
 - Focus: Component-level testing
 - Tools: JUnit, Jest, PyTest
 - Coverage Target: 80%
2. **Integration Tests ([Target]%)**
 - Focus: Service integration
 - Tools: Postman, RestAssured
 - Coverage: Critical paths
3. **UI Tests ([Target]%)**
 - Focus: User workflows
 - Tools: Selenium, Cypress
 - Coverage: Core user journeys

1.6.g 4. Release Management Process

1.6.g.a Release Steps

1. **Build Preparation**
 - Version updates
 - Environment configuration
 - Feature flag verification
2. **Testing & Validation**
 - Regression testing
 - Performance testing
 - Security validation
3. **Release Candidate**
 - Final builds
 - Release tagging
 - Artifact archival
4. **Deployment**
 - Store submission
 - Staged rollout
 - Post-deployment monitoring

1.6.g.b Deployment Checklist

- ☐ Version numbers updated
- ☐ Production API endpoints configured
- ☐ Feature flags verified
- ☐ Localization files complete

- ☐ Clean builds generated successfully
- ☐ Release builds signed

1.7 Team Organization

1.7.a Internal Roles

1. **Product Owner**
 - Represents stakeholders' interests
 - Defines and prioritizes product backlog
 - Makes final decisions on product features
2. **Scrum Master**
 - Facilitates Agile processes
 - Removes impediments
 - Coaches team on Agile practices
3. **Development Team**
 - Software Developers
 - Quality Assurance Engineers
 - UI/UX Designers
 - DevOps Engineers
 - Database Administrators

1.8 Tools & Infrastructure

1.8.a Required Tools

Development:

- Git & GitHub/GitLab
- CI/CD Pipeline
- Code Quality Tools
- Test Automation

Project Management:

- Agile Board (JIRA/Azure DevOps)
 - Documentation (Confluence/SharePoint)
- Communication (Slack/Teams)
- Metrics Dashboard

1.9 Maintenance & Support

1.9.a Post-Release Activities

1. **Monitoring**
 - Performance metrics
 - Error tracking
 - User analytics

2. Support

- Issue resolution
- User feedback
- Documentation updates

3. Continuous Improvement

- Sprint retrospectives
- Process refinement
- Technical debt management

1.10 Appendices

1.10.a A. Templates Index

// ...consolidate all template references...

1.10.b B. Process Checklists

// ...consolidate all checklists...

1.10.c C. Metrics Dashboard

// ...consolidate all metrics...