DPP Project Tasking Outline

Structured Implementation Task List

Nathan Lunceford

2025-02-25

Table of contents

1 Project Overview

This document outlines the implementation tasks for the enhanced Distributor Data Extraction Ingestion project. Each task is designed to be completed within 1-3 days and includes clear objectives, completion criteria, and time estimates.

2 Phase 1: Foundation and Core Infrastructure (Weeks 1-2)

2.1 Task 1.1: Set Up Project Structure and Development Environment

Objective: Create the initial project structure and configure the development environment.

Completion Criteria:

- Repository initialized with proper structure and README
- Solution structure created with defined projects and namespaces
- Docker Compose configuration for local development with HashiCorp Vault, FerretDB, and minio (S3 alternative)
- CI/CD pipeline templates defined

Estimated Time: 2 days

2.2 Task 1.2: Implement Basic Dependency Injection Framework

Objective: Set up the DI container and register core services.

Completion Criteria:

- Program.cs and CreateHostBuilder implemented
- Core service interfaces defined
- Basic DI container configuration with service registration

Estimated Time: 1 day

2.3 Task 1.3: Implement HashiCorp Vault Integration

Objective: Create the Vault service for credential management.

Completion Criteria:

- VaultService class implemented with basic operations
- Authentication methods supported (AppRole, Token)
- Unit tests for Vault interaction
- Configuration model for Vault settings

Estimated Time: 2 days

2.4 Task 1.4: Implement FerretDB Integration

Objective: Create the FerretDB service for configuration management.

Completion Criteria:

- FerretDBService class implemented with MongoDB driver
- Basic CRUD operations for configuration management
- Unit tests for FerretDB interaction
- Configuration model for FerretDB settings

Estimated Time: 2 days

2.5 Task 1.5: Implement AWS S3 Integration

Objective: Set up the S3 client for data upload.

Completion Criteria:

- S3DataUploader class implemented
- Upload functionality with metadata support
- Unit tests for S3 interaction
- Configuration model for S3 settings

Estimated Time: 1 day

2.6 Task 1.6: Implement Core Models and Interfaces

Objective: Define the core domain models and interfaces for the system.

Completion Criteria:

- ERPConfiguration model implemented
- ERPCredentials model implemented
- UploadConfiguration model implemented
- Core interfaces (IExtractor, ITransformer, IUploader) defined

Estimated Time: 1 day

3 Phase 2: Core Functionality Implementation (Weeks 3-4)

3.1 Task 2.1: Implement Registry Pattern Components

Objective: Create the registry classes for dynamic component resolution.

Completion Criteria:

- ERPRegistry implemented
- ExtractorRegistry implemented
- TransformationRegistry implemented
- UploaderRegistry implemented
- Unit tests for registry functionality

Estimated Time: 2 days

3.2 Task 2.2: Implement API Mode Extraction Components

Objective: Create the components for API-based extraction.

Completion Criteria:

- APIRequestBuilder implemented with fluent interface
- AuthenticationBuilder implemented
- API-based extractor implementation
- Unit tests for API extraction

Estimated Time: 3 days

3.3 Task 2.3: Implement Database Mode Extraction Components

Objective: Create the components for database-based extraction.

Completion Criteria:

- DatabaseQueryBuilder implemented with fluent interface
- DatabaseQuery class with SQL generation
- Database-based extractor implementation
- Unit tests for database extraction

Estimated Time: 3 days

3.4 Task 2.4: Implement Basic Transformation Logic

Objective: Create the transformation components for column standardization.

Completion Criteria:

- Basic transformer implementation
- Column mapping functionality
- Parquet conversion logic
- Unit tests for transformation

Estimated Time: 2 days

3.5 Task 2.5: Implement ERPService Core Logic

Objective: Create the main orchestration logic for the extraction ingestion process.

Completion Criteria:

- ProcessERPData method implemented with basic flow
- Integration with registries and components
- Support for both API and Database modes
- Error handling for basic scenarios

Estimated Time: 3 days

3.6 Task 2.6: Implement Command-Line Interface

Objective: Create the command-line interface for the application.

Completion Criteria:

- System.CommandLine integration
- Command-line argument parsing
- Help documentation
- Exit code handling

Estimated Time: 1 day

4 Phase 3: Resilience and Error Handling (Weeks 5-6)

4.1 Task 3.1: Implement Circuit Breaker Pattern

Objective: Add circuit breaker protection for external dependencies.

Completion Criteria:

- Circuit breaker implementation for Vault
- Circuit breaker implementation for FerretDB
- Circuit breaker implementation for S3
- Circuit breaker implementation for API calls
- Unit tests for circuit breaker functionality

Estimated Time: 2 days

4.2 Task 3.2: Implement Bulkhead Pattern

Objective: Add resource isolation using bulkheads.

Completion Criteria:

- Connection pool isolation for different ERP types
- Resource allocation for critical vs. non-critical operations
- Bulkhead configuration model
- Unit tests for bulkhead functionality

Estimated Time: 2 days

4.3 Task 3.3: Implement Retry Strategies with Exponential Backoff

Objective: Enhance retry logic with exponential backoff and jitter.

Completion Criteria:

- RetryPolicyBuilder implemented
- Exponential backoff strategy with jitter
- Integration with HTTP client factory
- Unit tests for retry policies

Estimated Time: 1 day

4.4 Task 3.4: Implement Self-Healing Procedures

Objective: Add automatic recovery procedures for common failure scenarios.

Completion Criteria:

- Token renewal for expired credentials
- Connection pool refresh for stale connections
- Automatic cleanup for temporary resources
- Unit tests for self-healing functionality

Estimated Time: 3 days

4.5 Task 3.5: Enhance Error Handling and Logging

Objective: Improve error handling with classification and structured logging.

Completion Criteria:

- Error classification (transient vs. persistent)
- Structured logging with correlation IDs
- Context-enriched log entries
- Integration with Serilog

Estimated Time: 2 days

4.6 Task 3.6: Implement Health Checks

Objective: Create health check endpoints for system monitoring.

Completion Criteria:

- Health check implementation for Vault
- Health check implementation for FerretDB
- Health check implementation for S3
- Health check API endpoint
- Integration with monitoring system

Estimated Time: 1 day

5 Phase 4: Security Enhancements (Weeks 7-8)

5.1 Task 4.1: Implement Credential Caching

Objective: Add caching to credential retrieval for improved performance.

Completion Criteria:

- CachedCredentialProviderDecorator implemented
- TTL-based cache invalidation
- Thread-safe caching mechanism
- Unit tests for caching functionality

Estimated Time: 1 day

5.2 Task 4.2: Implement Least Privilege Access

Objective: Enhance security with dynamic, operation-specific credentials.

Completion Criteria:

- LeastPrivilegeCredentialProvider implemented
- Operation-specific credential generation
- Credential scoping based on context
- Integration with Vault dynamic secrets

Estimated Time: 2 days

5.3 Task 4.3: Implement Field-Level Encryption

Objective: Add encryption for sensitive fields.

Completion Criteria:

- EncryptionService implemented
- Field-level encryption/decryption
- Key management integration
- Unit tests for encryption functionality

Estimated Time: 3 days

5.4 Task 4.4: Implement Data Masking

Objective: Create data masking functionality for non-production environments.

Completion Criteria:

- DataMaskingService implemented
- Various masking techniques (hashing, tokenization)
- Configuration model for masking rules
- Unit tests for masking functionality

Estimated Time: 2 days

5.5 Task 4.5: Implement Mutual TLS

Objective: Enhance secure communication with mutual TLS.

Completion Criteria:

- mTLS configuration for HTTP clients
- Certificate management integration
- mTLS support in API request builder
- Unit tests for mTLS functionality

Estimated Time: 2 days

5.6 Task 4.6: Implement Audit Logging

Objective: Add comprehensive audit logging for security compliance.

Completion Criteria:

- AuditLogger implemented
- Integration with Vault audit backend
- Operational audit events defined
- Compliance reporting capabilities

Estimated Time: 1 day

6 Phase 5: Data Quality and Governance (Weeks 9-10)

6.1 Task 5.1: Implement Data Contract Validator

Objective: Create validation logic for data contracts.

Completion Criteria:

- DataContractValidator implemented
- Schema validation functionality
- Value validation against business rules
- Validation result model with severity levels
- Unit tests for validation functionality

Estimated Time: 3 days

6.2 Task 5.2: Implement Data Lineage Tracking

Objective: Add data lineage capabilities for audit and compliance.

Completion Criteria:

- DataLineageService implemented
- Lineage record creation and management
- Transformation tracking
- Integration with metadata services
- Unit tests for lineage functionality

Estimated Time: 2 days

6.3 Task 5.3: Implement Data Catalog Integration

Objective: Create integration with data catalog for metadata management.

Completion Criteria:

- CatalogService implemented
- Dataset metadata management
- Schema versioning support
- Unit tests for catalog functionality

Estimated Time: 2 days

6.4 Task 5.4: Implement Schema Evolution Support

Objective: Add capabilities for handling schema changes.

Completion Criteria:

- SchemaVersionManager implemented
- Backward compatibility handling
- Schema migration support
- Unit tests for schema evolution

Estimated Time: 3 days

6.5 Task 5.5: Implement Data Quality Metrics Collection

Objective: Add collection of data quality metrics.

Completion Criteria:

- DataQualityMetricsCollector implemented
- Quality dimension measurements
- Integration with metrics service
- Unit tests for metrics collection

Estimated Time: 2 days

6.6 Task 5.6: Implement Data Quality Reporting

Objective: Create reporting capabilities for data quality.

Completion Criteria:

- DataQualityReportGenerator implemented
- Quality issue summarization
- Trend analysis for quality metrics
- Integration with notification system

Estimated Time: 2 days

7 Phase 6: Performance Optimization (Weeks 11-12)

7.1 Task 6.1: Implement Incremental Extraction

Objective: Add support for incremental data extraction.

Completion Criteria:

- ExtractConfigBuilder implemented
- Change Data Capture (CDC) support
- Watermark management
- Unit tests for incremental extraction

Estimated Time: 3 days

7.2 Task 6.2: Implement Batch Processing

Objective: Enhance performance with batch processing capabilities.

Completion Criteria:

- BatchProcessor implemented
- Memory-efficient processing
- Progress tracking
- Unit tests for batch processing

Estimated Time: 2 days

7.3 Task 6.3: Implement Data Compression

Objective: Add compression support for improved efficiency.

Completion Criteria:

- CompressionService implemented
- Multiple compression algorithm support
- Compression level configuration
- Unit tests for compression functionality

Estimated Time: 1 day

7.4 Task 6.4: Implement Metrics Collection

Objective: Add comprehensive metrics collection for performance monitoring.

Completion Criteria:

- MetricsService implemented
- Integration with Prometheus
- Custom dimensions and labels
- Unit tests for metrics functionality

Estimated Time: 2 days

7.5 Task 6.5: Implement Performance Benchmarking

Objective: Create benchmarking capabilities for performance testing.

Completion Criteria:

- BenchmarkRunner implemented
- Standard performance scenarios
- Result comparison and reporting
- Integration with CI/CD pipeline

Estimated Time: 2 days

7.6 Task 6.6: Optimize Resource Usage

Objective: Tune system for optimal resource utilization.

Completion Criteria:

- Memory usage optimization
- Thread pool configuration
- Connection pool tuning
- Performance test results showing improvement

Estimated Time: 3 days

8 Phase 7: Operational Excellence (Weeks 13-14)

8.1 Task 7.1: Implement Feature Flag Management

Objective: Add feature flag capabilities for gradual rollout.

Completion Criteria:

- FeatureFlagService implemented
- Flag configuration management
- Context-based flag evaluation
- Unit tests for feature flags

Estimated Time: 2 days

8.2 Task 7.2: Implement Distributed Tracing

Objective: Add distributed tracing for end-to-end visibility.

Completion Criteria:

- TracingService implemented
- Integration with OpenTelemetry
- Trace correlation across components
- Trace sampling configuration

Estimated Time: 2 days

8.3 Task 7.3: Implement OpenAPI Documentation

Objective: Create API documentation for service interfaces.

Completion Criteria:

- OpenAPI configuration
- API endpoint documentation
- Model documentation
- Swagger UI integration

Estimated Time: 1 day

8.4 Task 7.4: Create Operational Runbooks

Objective: Develop runbooks for common operational procedures.

Completion Criteria:

- Installation and deployment guide
- Troubleshooting procedures
- Monitoring guidelines
- Disaster recovery procedures

Estimated Time: 3 days

8.5 Task 7.5: Implement GitOps Configuration

Objective: Set up GitOps-based configuration management.

Completion Criteria:

- Infrastructure as Code templates
- Configuration as Code approach
- Deployment pipeline integration
- Change approval workflow

Estimated Time: 2 days

8.6 Task 7.6: Implement Blue-Green Deployment Support

Objective: Add support for zero-downtime deployments.

Completion Criteria:

- Deployment strategy implementation
- Version compatibility verification
- Rollback procedures
- Load balancer integration

Estimated Time: 3 days

9 Phase 8: Integration and Testing (Weeks 15-16)

9.1 Task 8.1: Implement Integration Test Harness

Objective: Create test infrastructure for comprehensive integration testing.

Completion Criteria:

- Test harness framework
- Mock ERP implementations
- Test data generation
- Test execution automation

Estimated Time: 3 days

9.2 Task 8.2: Implement End-to-End Tests

Objective: Create end-to-end tests for critical workflows.

Completion Criteria:

- E2E test scenarios for API mode
- E2E test scenarios for Database mode
- Test assertions and verification
- Test reporting

Estimated Time: 3 days

9.3 Task 8.3: Implement Performance Tests

Objective: Create performance tests for system evaluation.

Completion Criteria:

- Load test scenarios
- Stress test scenarios
- Performance metrics collection
- Test result analysis

Estimated Time: 2 days

9.4 Task 8.4: Implement Security Tests

Objective: Create security tests to validate protection measures.

Completion Criteria:

- Authentication and authorization tests
- Encryption verification
- Secure communication tests
- Audit log verification

Estimated Time: 2 days

9.5 Task 8.5: Implement Fault Injection Tests

Objective: Create tests to validate resilience capabilities.

Completion Criteria:

- Dependency failure simulations
- Network degradation tests
- Resource exhaustion tests
- Recovery verification

Estimated Time: 2 days

9.6 Task 8.6: Conduct System Integration Testing

Objective: Perform comprehensive integration testing with all components.

Completion Criteria:

- All subsystems tested together
- Edge cases and boundary conditions tested
- Performance under load verified
- Documentation of test results

Estimated Time: 3 days

10 Phase 9: Finalization and Deployment (Weeks 17-18)

10.1 Task 9.1: Finalize Documentation

Objective: Complete all system documentation.

Completion Criteria:

- Architecture documentation finalized
- API documentation completed
- Development guide updated
- Operational procedures documented

Estimated Time: 3 days

10.2 Task 9.2: Conduct Code Review and Cleanup

Objective: Perform comprehensive code review and cleanup.

Completion Criteria:

- Code review completed
- Code quality issues addressed
- Technical debt remediated
- Code documentation updated

Estimated Time: 2 days

10.3 Task 9.3: Prepare Release Artifacts

Objective: Create release artifacts for deployment.

Completion Criteria:

- Release notes prepared
- Versioned artifacts created
- Deployment packages assembled
- Release verification checklist completed

Estimated Time: 1 day

10.4 Task 9.4: Set Up Monitoring and Alerting

Objective: Configure production monitoring and alerting.

Completion Criteria:

- Metrics dashboards created
- Alert rules configured
- On-call procedures defined
- Monitoring documentation completed

Estimated Time: 2 days

10.5 Task 9.5: Conduct Deployment Rehearsal

Objective: Perform a rehearsal of the production deployment.

Completion Criteria:

- Staging environment deployment completed
- Verification procedures executed
- Rollback procedures tested
- Deployment timing measured

Estimated Time: 1 day

10.6 Task 9.6: Execute Production Deployment

Objective: Deploy the system to production.

Completion Criteria:

- Production deployment completed
- Post-deployment verification performed
- Stakeholder sign-off obtained
- Transition to operational support

Estimated Time: 1 day

11 Project Timeline Summary

- **Phase 1: Foundation and Core Infrastructure** - Weeks 1-2
- **Phase 2: Core Functionality Implementation** - Weeks 3-4
- **Phase 3: Resilience and Error Handling** - Weeks 5-6
- **Phase 4: Security Enhancements** - Weeks 7-8
- **Phase 5: Data Quality and Governance** - Weeks 9-10
- **Phase 6: Performance Optimization** - Weeks 11-12
- **Phase 7: Operational Excellence** - Weeks 13-14
- **Phase 8: Integration and Testing** - Weeks 15-16
- **Phase 9: Finalization and Deployment** - Weeks 17-18

Total estimated project duration: **18 weeks**