



Document History & Cleanup Log

This document tracks all documentation changes, consolidations, and deletions to maintain historical record.

New Documentation - January 2025

WSL Windows Setup Guide

Created: docs/deployment/WSL_WINDOWS_SETUP_GUIDE.md

Purpose: Comprehensive guide for setting up WSL, Docker, and Cursor on Windows without admin privileges for ODRAS development

Size: 600+ lines

Status: Active - Essential setup guide for Windows developers

Key Features:

- WSL installation without admin privileges
- Docker Engine setup in WSL
- Cursor IDE installation and configuration
- Complete ODRAS setup instructions
- Comprehensive troubleshooting section

New Documentation - January 4, 2025

Publishing and Domain Architecture

Created: docs/architecture/PUBLISHING_ARCHITECTURE.md

Purpose: Comprehensive publishing mechanism for project artifacts and network collaboration

Size: 500+ lines

Status: Active - Core publishing system design

Created: docs/architecture/DOMAIN_DRIVEN_ARCHITECTURE.md

Purpose: Domain-driven design framework for ODRAS business logic organization

Size: 400+ lines

Status: Active - Domain architecture foundation

Created: docs/workbenches/publishing-workbench/CURRENT_STATUS.md

Purpose: Publishing workbench implementation status and roadmap

Size: 200+ lines

Status: Active - Publishing workbench development

Updated: docs/WORKBENCH_OVERVIEW.md

Changes: Added Publishing workbench, updated workbench count to 14

Status: Active - Updated workbench overview

Updated: docs/architecture/event-architecture/CURRENT_STATUS.md

Changes: Added publishing events, subscription events, network events

Status: Active - Enhanced event architecture

New Documentation - October 12, 2025

Workbench and Architecture Organization

Created: docs/WORKBENCH_OVERVIEW.md

Purpose: Comprehensive overview of reorganized workbench and architecture documentation

Branch: cursor/organize-odras-documentation-folders-d83a

Size: 200+ lines

Status: Active - Primary reference for workbench organization

New Structure Created:

- docs/workbenches/ - Individual workbench documentation
 - cqmt-workbench/ - Conceptual Query Management Tool
 - ontology-workbench/ - Ontology management and editing
 - requirements-workbench/ - Requirements analysis and management
 - das-workbench/ - Distributed Autonomous System management

- `knowledge-management-workbench/` - Knowledge management and retrieval
- `conceptualizer-workbench/` - AI-powered system conceptualization
- `configurator-workbench/` - Manual configuration capabilities
- `tabularizer-workbench/` - Transform individuals into structured tables
- `thread-manager-workbench/` - DAS conversation thread management
- `event-management-workbench/` - Event flow management and monitoring
- `data-management-workbench/` - Central data orchestration and integration
- `pubsub-workbench/` - Publish/subscribe messaging management
- `process-workbench/` - BPMN process management and execution
- `docs/architecture/` - Architecture component documentation
 - `core-architecture/` - Fundamental system structure
 - `database-architecture/` - Data storage and management
 - `rag-architecture/` - Retrieval-Augmented Generation
 - `event-architecture/` - Event-driven communication
 - `integration-architecture/` - External system integration
 - `REFACTOR_ARCHITECTURE.md` - Critical refactoring strategy
 - `PLUGGABLE_ARCHITECTURE.md` - Plugin-based architecture design

Key Benefits:

- Clear separation of workbench and architecture concerns
- Status tracking with `CURRENT_STATUS.md` files for each component
- Priority management with defined next steps
- Dependency mapping between components
- Testing status visibility

Files Moved:

- CQMT documentation → `workbenches/cqmt-workbench/`

- Ontology documentation → `workbenches/ontology-workbench/`
- DAS documentation → `workbenches/das-workbench/`
- Thread Manager documentation → `workbenches/thread-manager-workbench/`
- RAG documentation → `architecture/rag-architecture/`
- Database documentation → `architecture/database-architecture/`

Status Documents Created:

- Each workbench and architecture component now has a `CURRENT_STATUS.md` file
- Tracks implementation status, completed features, in-progress work, and pending features
- Includes technical debt, next priorities, dependencies, and testing status
- Provides clear visibility into what needs to be completed
- **Total Workbenches:** 9 (5 core + 4 specialized)
- **Total Architecture Components:** 5

New Documentation - October 12, 2025

Ontology Imports Persistence Issues - Lessons Learned

Created: `docs/development/ONTOLOGY_IMPORTS_PERSISTENCE_ISSUES.md`

Purpose: Document failed attempt to fix ontology imports persistence for future reference

Branch: `fix/ontology_imports` (16 commits, October 11, 2025)

Outcome: Branch abandoned without merge - architectural refactor needed instead

Size: 450+ lines

Status: Reference documentation - Do not merge branch code

Problems Documented:

- Import disappearance on browser refresh
- LocalStorage vs Fuseki synchronization issues

- Data corruption during save operations
- Imported node position persistence failures
- Three-way state management conflicts (Canvas ↔ LocalStorage ↔ Fuseki)

Attempted Solutions (All Failed):

1. Backend synchronization - Fuseki as source of truth
2. Import system overhaul - atomic imports only
3. LocalStorage restoration fixes
4. Imported node position persistence changes
5. Data corruption prevention measures

Diagnostic Tools Created:

- Debug functions: `debugImports()` , `checkDataIntegrity()` , `emergencyReload()`
- Extensive logging with emoji prefixes
- Import synchronization verification
- Position persistence testing

Root Causes Identified:

- No clear source of truth between three storage layers
- Missing import lifecycle state machine
- Race conditions between async operations
- 15,000+ line monolithic frontend file (architectural debt)

Recommended Future Approach:

- Full architectural refactor with proper state management library
- Single source of truth (Fuseki) with localStorage as cache only
- Break up monolithic app.html into testable modules
- Test-driven development for import lifecycle

Key Lesson: Incremental fixes on architectural problems don't work - need planned refactor

New Documentation - October 11, 2025

Ontology Inheritance System Implementation

Created: docs/features/ONTOLOGY_INHERITANCE_SYSTEM.md

Purpose: Comprehensive documentation of successful ontology inheritance system implementation

Issue: Need for class inheritance with "is_a" relationships where child classes inherit parent properties in individuals tables

Solution: Complete inheritance system with multiple parents, cross-project support, conflict resolution

Size: 350+ lines

Status: Active - Core inheritance system operational

Key Features Implemented:

- Multiple parent inheritance with conflict resolution
- Cross-project inheritance from reference ontologies
- Abstract class support with UI validation
- Enhanced properties panel with parent selection dropdown
- Individuals tables show inherited properties with visual indicators (↑)
- Property range conflict resolution (xsd:float > string)
- URI mapping between display names and RDF storage

Test Case Confirmed Working:

- Object class (ID property) → PhysicalObject class (Mass, Length properties) → Aircraft class (vendor property)
- Aircraft individuals table correctly displays: Name, ID↑, Mass↑, Length↑, vendor, Actions
- All inherited properties properly marked with inheritance indicators

Modified Files:

- backend/services/ontology_manager.py - Core inheritance engine with recursive resolution
- backend/api/ontology.py - New inheritance API endpoints

- `frontend/app.html` - Multi-select parent UI, RDF conversion fixes, inheritance table generation

API Endpoints Added:

- `GET /api/ontology/classes/{class_name}/all-properties` - Returns inherited properties
- `GET /api/ontology/available-parents` - Lists available parent classes
- `GET /api/ontology/classes/{class_name}/hierarchy` - Returns inheritance tree
- `PUT /api/ontology/classes/{class_name}` - Updates class with inheritance data

Remaining Minor Issues Identified:

- Property duplication in RDF (workaround: conflict resolution active)
- UI field duplication in properties panel (cosmetic)
- Abstract class checkbox persistence refinement needed
- Parent selection persistence timing improvements

New Documentation - October 7, 2025

RAG System Stabilization Guide

Created: `docs/development/RAG_STABILIZATION_GUIDE.md`

Purpose: Comprehensive documentation of RAG system fixes and improvements

Issue: UAS names query only returned 2 platforms instead of 9, source attribution failures

Solution: Enhanced chunk retrieval, fixed deduplication logic, improved source attribution

Size: 400+ lines

Status: Active - Primary reference for RAG system troubleshooting

Key Changes Documented:

- Increased chunk limits from 25 to 50 in DAS2 core engine
- Lowered similarity threshold from 0.15 to 0.1 for better coverage

- Modified deduplication to allow 3 chunks per document instead of 1
- Fixed source attribution by adding `asset_id` and `document_type` to chunk payloads
- Enhanced external task worker context from 3 to 10-15 chunks
- Created comprehensive test script for RAG validation

Results Achieved:

- UAS names query now returns all 9 platforms (was 2)
- Source attribution shows correct document titles (was "Unknown Document")
- Chunk retrieval increased from 3 to 9 chunks per query
- Response quality significantly improved with comprehensive details

Modified Files:

- `backend/services/das2_core_engine.py` - Chunk limits and thresholds
- `backend/services/rag_service.py` - Source attribution and deduplication
- `backend/services/store.py` - Chunk payload fields
- `backend/services/external_task_worker.py` - Context chunks
- `scripts/single_query_test.py` - NEW - Comprehensive test script
- `scripts/ci_rag_test.py` - NEW - Automated CI test script
- `.github/workflows/ci.yml` - Added RAG testing to CI pipeline

New Documentation - October 6, 2025

Database Connection Pool Troubleshooting Guide

Created: `docs/development/DATABASE_CONNECTION_POOL_TROUBLESHOOTING.md`

Purpose: Comprehensive troubleshooting guide for database connection pool issues

Issue: Users unable to login after idle periods due to stale database connections (Oct 2025)

Solution: TCP keepalive, connection validation, pool size reduction, background monitoring

Size: 518 lines

Status: Active - Primary reference for connection pool debugging

Key Changes Documented:

- Added TCP keepalive to PostgreSQL connections (keepalives=1, idle=30s, interval=10s, count=5)
- Implemented connection health checks in `_conn()` method
- Reduced connection pool from 5-50 to 2-20 connections
- Created background monitor task (`backend/services/db_monitor.py`)
- Improved connection return logic with validation

Modified Files:

- `backend/services/db.py` - Core connection pool changes
- `backend/services/config.py` - Pool size configuration
- `backend/services/db_monitor.py` - NEW - Background monitoring
- `backend/main.py` - Startup event integration

Document Cleanup - September 23, 2025

Branch: `cleanup/repo-cleanup`

Commit: [Commit hash will be added after commit]

Total Documents Before: 74

Total Documents After: 27 (63% reduction)

Additional Cleanup: Moved all top-level .md files to docs/ folder

? Consolidation Strategy

1. DAS Documentation →

`docs/architecture/DAS_COMPREHENSIVE_GUIDE.md`

Consolidated from 6 documents:

- `DAS_Architecture_and_Implementation_Plan.md`
- `DAS_Current_Architecture_Documentation.md`
- `DAS_MVP_Implementation_Summary.md`
- `DAS_Tomorrow_Goals_Knowledge_and_API_Execution.md`

- `Digital_Assistance_System_(DAS)_MVP_Specification.md`
- `Session_Intelligence_and_Event_Capture_for_DAS.md`

2. Testing Documentation → `docs/development/TESTING_GUIDE.md`

Consolidated from 5 documents:

- `TESTING_AND_VALIDATION_GUIDE.md`
- `TESTING_ENFORCEMENT_GUIDE.md`
- `TESTING_ENFORCEMENT_SUMMARY.md`
- `TESTING_IMPLEMENTATION_SUMMARY.md`
- `architecture/DATABASE_SCHEMA_TESTING_GUIDE.md`

3. Ontology Workbench → `docs/features/ONTOLOGY_WORKBENCH_GUIDE.md`

Consolidated from 4 documents:

- `features/ontology_workbench_mvp.md`
- `features/ontology_workbench_post_mvp.md`
- `features/ontology-workbench-upgrade-plan.md`
- `ontology_import_equivalence_system.md`

4. File Management → `docs/features/FILE_MANAGEMENT_GUIDE.md`

Consolidated from 3 documents:

- `features/file_management_workbench_mvp.md`
- `features/file_management_status_2024.md`
- `features/data_manager_workbench_spec.md`

5. Namespace Management → `docs/features/NAMESPACE_MANAGEMENT_GUIDE.md`

Consolidated from 5 documents:

- `namespace/Namespace_White_Paper.md`
- `namespace/namespace_mvp.md`
- `namespace/namespace_implementation_plan.md`

- namespace/installation_examples.md
- namespace-organization-uri-design.md

? Documents Deleted

Historical/Planning Documents (Completed Work)

- archive/mvp_week2_deliverables.md - Historical planning document
- archive/mvp_week2_executive_summary.md - Historical planning document
- archive/odras_mvp_updates_week2.md - Historical planning document
- archive/todo_morning_next_steps.md - Outdated todo list
- ODRAS_Heilmeier_Catechism.md - Historical planning document
- ODRAS_Heilmeier_Executive_Summary.md - Historical planning document
- project_todos.md - Outdated project todos
- phase2_quick_reference.md - Outdated phase planning
- REFACTORING_SUMMARY.md - Work completed
- REVIEW_INTERFACE_IMPLEMENTATION.md - Implementation completed

Implementation Status Documents (Work Complete)

- archive/USER_TASK_IMPLEMENTATION_STATUS.md - Implementation completed
- archive/USER_TASK_INTEGRATION.md - Implementation completed
- archive/DEVELOPMENT_STATUS.md - Outdated status document
- FEATURE_IMPLEMENTATION.md - Work completed

Experimental/Unused Features

- Prok0S/Prok0S.md - Experimental feature not implemented
- Prok0S/build_prompt.md - Experimental feature not implemented
- Prok0S/procos_spec.md - Experimental feature not implemented
- Prok0S/why_this_works.md - Experimental feature not implemented
- shac1_discussion.md - Feature not implemented
- shac1_example.md - Feature not implemented

Concept Documents (Superseded by Implementation)

- archive/concept_need_reviewed.md - Concept superseded

- `archive/concept_needs.md` - Concept superseded
- `archive/concept_tool_spec.md` - Concept superseded

Testing Setup (Superseded by Current Testing)

- `archive/test_data_setup_guide.md` - Superseded by current test framework
- `archive/HOW_TO_TEST_CHANGES.md` - Superseded by current testing guide

Miscellaneous

- `archive/LABELS.md` - GitHub labels (not needed in docs)
- `archive/OLLAMA_GPU_SETUP.md` - Specific setup guide (moved to deployment if needed)
- `NDIA/meia_spec.md` - Specific client document (can be archived separately)

? Documents Preserved

Core Documentation (Kept as-is)

- `README.md` - Main project documentation
- `ROADMAP.md` - Future planning
- `AUTHENTICATION_SYSTEM.md` - Current system documentation
- `IRI_SYSTEM_OVERVIEW.md` - Current system documentation
- `USER_MANAGEMENT_UI.md` - Current system documentation

Architecture (Consolidated + Key Docs)

- `architecture/DAS_COMPREHENSIVE_GUIDE.md` - **NEW CONSOLIDATED**
- `architecture/DATABASE_SCHEMA_MANAGEMENT_SUMMARY.md` - Current system

Deployment

- `deployment/INSTALLATION_AND_IRI_SETUP.md` - Current deployment guide
- `deployment/INSTALLATION_SPECIFIC_IRI_CONFIG.md` - Current configuration
- `deployment/FEDERATED_ACCESS_QUICK_REFERENCE.md` - Current system

Development (Consolidated + Key Docs)

- `development/TESTING_GUIDE.md` - **NEW CONSOLIDATED**
- `development/BPMN_LLM_Integration_Guide.md` - Current integration guide
- `development/BPMN_WORKFLOWS.md` - Current workflow documentation

Features (All Consolidated)

- `features/ONTOLOGY_WORKBENCH_GUIDE.md` - **NEW CONSOLIDATED**
- `features/FILE_MANAGEMENT_GUIDE.md` - **NEW CONSOLIDATED**
- `features/NAMESPACE_MANAGEMENT_GUIDE.md` - **NEW CONSOLIDATED**
- `features/knowledge_management_mvp.md` - Current feature

Specifications

- `ODRAS_Advanced_Features_Specification.md` - Current specification
- `ODRAS_Use_Cases.md` - Current use cases
- `Ontology-Driven Requirements Analysis System (ODRAS).md` - Core specification

Implementation Guides

- `rag_query_implementation.md` - Current implementation
 - `ontology_state_persistence.md` - Current implementation
 - `reference_ontologies_feature.md` - Current feature
 - `Project_Thread_Intelligence_Architecture.md` - Current architecture
 - `PERSONA_PROMPT_README.md` - Current system
-

Future Document Management

Guidelines for New Documents

1. **Consolidate similar topics** into comprehensive guides
2. **Archive completed work** rather than keeping implementation docs
3. **Use clear naming conventions** with categories
4. **Regular cleanup** every 3-6 months

5. **Historical tracking** in this document

Document Categories

- **Architecture** - System design and technical specifications
- **Features** - User-facing feature documentation
- **Deployment** - Installation and configuration
- **Development** - Developer guides and processes
- **Archive** - Historical documents (if needed for reference)

Naming Conventions

- Use UPPERCASE for consolidated guides: `DAS_COMPREHENSIVE_GUIDE.md`
- Use descriptive names: `ONTOLOGY_WORKBENCH_GUIDE.md`
- Include version/date for specifications: `API_SPEC_v2.md`