

Phase 1: Core Git Engine & Workpad System

Duration: October 18-20, 2025 (3 days)

Status: 🟡 In Progress

Goal

Complete Git operations and workpad lifecycle, including repository initialization, workpad management, checkpoint system, test orchestration, and basic merge operations.

Objectives

1. Git Engine ✅ (Now)

- Repository initialization from zip files
- Repository initialization from Git URLs
- Workpad creation and management
- Checkpoint system for saving state
- Fast-forward merge operations
- Branch management (trunk + workpads)

2. Core Abstractions ✅ (Now)

- `sologit/core/repository.py` - Repository class
- `sologit/core/workpad.py` - Workpad class
- Metadata management
- State persistence

3. Test Orchestrator ⌚

- Docker sandbox integration
- Test configuration parsing (evogit.yaml)
- Test execution with timeout
- Result collection and reporting
- Parallel test execution

4. Patch Engine ⌚

- Apply diffs/patches to workpads
- Conflict detection
- Patch validation
- Rollback capability

5. CLI Commands ⌚

- `evogitctl repo init --zip <file>`
- `evogitctl repo init --git <url>`
- `evogitctl pad create <name>`
- `evogitctl pad list`
- `evogitctl pad promote <id>`

- `evogitctl test run`

6. Dependencies (Now)







- `gitpython>=3.1.40`
- `docker>=7.0.0`

Architecture

Git Engine Design

```
GitEngine
├─ init_from_zip(zip_buffer, name) → repo_id
├─ init_from_git(git_url, name) → repo_id
├─ create_workpad(repo_id, title) → pad_id
├─ apply_patch(pad_id, patch) → checkpoint_id
├─ promote_workpad(pad_id) → commit_hash
├─ revert_last_commit(repo_id) → void
├─ get_diff(pad_id) → diff_string
├─ get_repo_map(repo_id) → file_tree
```

Workpad Lifecycle

1. CREATE
`evogitctl pad create "add-login"`

 Workpad: `pads/add-login-20251016-1423`
2. DEVELOP
 Apply patches  Checkpoint t1
 Apply patches  Checkpoint t2
Run tests  Green
3. PROMOTE
`evogitctl pad promote <pad_id>`

 Fast-**forward** merge **to** main

 Delete workpad branch

Test Orchestration

```
TestOrchestrator
├─ run_tests(repo_path, tests[], parallel) → results[]
├─ run_single_test(repo_path, test_config) → result
├─ build_dependency_graph(tests[]) → graph
├─ execute_dependency_graph(...) → void
```

Deliverables

Code Components

- `[] sologit/core/repository.py` - Repository abstraction
- `[] sologit/core/workpad.py` - Workpad abstraction
- `[] sologit/engines/git_engine.py` - Git operations

- [] `sologit/engines/patch_engine.py` - Patch application
- [] `sologit/engines/test_orchestrator.py` - Test execution
- [] `sologit/cli/commands.py` - Updated with Phase 1 commands

Tests

- [] `tests/test_git_engine.py` - Git engine tests
- [] `tests/test_workpad.py` - Workpad lifecycle tests
- [] `tests/test_patch_engine.py` - Patch application tests
- [] `tests/test_test_orchestrator.py` - Test orchestration tests

Documentation

- [x] Phase 1 Overview (this document)
- [] Git Engine Architecture
- [] Test Orchestrator Design
- [] CLI Command Reference

Validation Criteria

Must Pass

- [x] Repository can be initialized from zip
- [] Repository can be initialized from Git URL
- [] Workpad can be created from repository
- [] Patches can be applied to workpad
- [] Tests can run in Docker sandbox
- [] Workpad can be promoted to trunk (fast-forward merge)
- [] CLI commands work end-to-end

Performance Targets

- Repository initialization: < 30 seconds
- Workpad creation: < 2 seconds
- Patch application: < 5 seconds
- Test execution: Varies by test suite
- Promote operation: < 3 seconds

Dependencies Added

```
# Git operations
gitpython>=3.1.40

# Docker operations
docker>=7.0.0
```

Next Phase

Phase 2: AI Integration - Model routing, planning, patch generation, cost tracking.

Related Documents

- [Git Engine Architecture](#) (../architecture/git-engine.md)
 - [Test Orchestrator Design](#) (../architecture/test-orchestrator.md)
 - [CLI Reference](#) (../guides/cli-reference.md)
-

Started: October 16, 2025

Target Completion: October 20, 2025