Savepoint Protocols — Quickstart Guide v02.1

Purpose: Transactional continuity for saving, restoring, comparing, and safely ingesting agent/project state across chats, project folders, and direct uploads.

Core Lifecycle

```
review → preview → conflict register → operator approval → commit → new savepoint
```

Core Rules

- **/save** records current state; non-destructive; no approval required.
- **/restore**, **/diff**, and **/absorb** must preview before commit.
- Commits require explicit operator approval and generate a new savepoint.
- Current system/developer/operator instructions outrank savepoints.
- Cross-agent context may transfer; another agent's identity must not transfer.
- Unclear, stale, private, or conflicting items are flagged, not silently merged.

Authority Order

- 1. System/developer instructions 2. Operator instructions 3. Current chat state
- 4. Canonical project files 5. Active savepoint 6. Historical savepoints
- 7. Foreign-agent savepoints 8. Model inference

Storage Rules

Location	Best For	Not Enough For
Project Sources	durable availability, lookup, reference	automatic restore
Direct Upload	restore, diff, absorb, handoff	long-term organization
Agent-Level Packet	stable protocol behavior	project-specific state
Active Index	current-version lookup	full continuity detail

Rule: Project Sources preserve availability. Direct uploads establish operational authority. Active index prevents version drift.

Canonical Filename

```
YYYY-MM-DD__SAVEPOINT__[Agent-Name]__[Project-or-Module]__[Short-Summary-Title]__vX.X.md
```

Commands

Command	Purpose
/save	create current-state savepoint
/restore preview	inspect restore candidate; no state change
/restore commit	apply approved restore items; generate new savepoint
/diff preview	compare same-agent historical savepoint
/diff commit	apply approved deltas; generate new savepoint
/absorb preview	review foreign-agent savepoint; identity firewall
/absorb commit	apply approved cross-agent context; new savepoint
/validate savepoint	structure, metadata, privacy, conflict, JSON check
/status savepoint	lineage, conflicts, questions, drift risks
/archive savepoint	mark archived/superseded/deprecated/quarantined/rejected
/deployment status	identify source/index/registry/candidate and upload need

IDs / Tags

Field	Purpose
savepoint_id	unique artifact ID
parent_savepoint_id	lineage tracking
artifact_status	draft / active / superseded / archived / quarantined / rejected
transfer_mode	same_agent / cross_agent / project_only / restore_candidate
identity_bearing	whether same-agent identity layer exists
deployment_context	agent_level / project_source / project_instruction / direct_upload
approval.status	not_requested / pending / approved / rejected

Retention Labels

retain_active · retain_reference · restore_active · coordination_only · historical_only · superseded · do_not_import · do_not_restore · requires_operator_confirmation · conflict_with_current_state · privacy_restricted · identity_layer_nontransferable

Project Folder Set

```
_PROJECT_PROTOCOLS/ · _ACTIVE_STATE/ · _AGENT_SAVEPOINTS/<Agent>/active|archive|quarantine/ · _HANDOFFS/pending|approved|archived/
```

Approval

Valid: **Approved: commit restore items R-001 and R-003 only.**
Invalid for commit: **Looks good.**

Golden Rule

Project folder = library/index. Direct upload = selected evidence. Project instructions = activation switch. Active index = current-truth pointer.

Package Files

Command Layer v02.1 · Universal v02.1 · Restore v01.1 · Diff/Absorb v02.1 · Deployment Overlay v01 · Active Index Template · Agent Registry Template · Project Instructions Template

Print-ready one-page guide · 2026-05-15