PLAN_TRADER_MERGE_PAPER_LIVE

Status: planned (plan only — awaits operator GO) **Priority:** P2 (refactor — kein Bug-Fix, kein Trading-Effekt) **Erstellt:** 2026-05-18 (post P3-Bundle) **Roadmap-ID:** TRADER-MERGE-PAPER-LIVE **Voraussetzung:** Stabiles Bot+Worker-Setup. Sample-Trades nach Bundle-1/2a/2b/COMPOSE/P3.

---

1. Befund

1.1 Aktuelle Architektur

```
trading/execution/paper_trade.py   ── 1164 LoC, 19 methods
  └ class PaperTrader              ── Basisklasse, Geschäftslogik:
      __init__, _load_state, _save_state, get_portfolio,
      baseline_reader, _baseline_*, _is_base_baseline_frozen,
      execute_buy, execute_sell,
      execute_tier3_buy, execute_partial_sell, check_tier3_exits,
      _reload_state_if_externally_changed, update_prices

trading/execution/live_trade.py    ── 1228 LoC, 23 methods
  └ class LiveTrader(PaperTrader)  ── Echte Binance-Order-Calls:
      __init__ (mit ccxt-Bindings), _label, _call_read, _call_order,
      _feed_circuit_breaker, _fire_alert,
      _track_sell_failure, _reset_sell_failure, _check_phantom_and_cleanup,
      _split_symbol, _sync_balance, _ensure_markets_loaded,
      _is_tradable, _parse_filled, _extract_fee_in_quote,
      _resolve_order, _verify_via_balance_diff,
      execute_buy (override), execute_sell (override),
      execute_tier3_buy (override), execute_partial_sell (override)
```

Inheritance-Aufteilung: - `PaperTrader` hält Portfolio-State, JSON-Persistenz, Geschäftslogik (PnL, DCA-Trigger-Loop in `update_prices`, Tier3-Budget). - `LiveTrader` ergänzt Binance-Exchange-Bindings, Circuit-Breaker, Outage-Resilience, Order-Lifecycle-Forensik. - `LiveTrader` überschreibt nur die 4 execute_*-Methoden — alles andere ist inherited.

1.2 Aktive vs. dormante Pfade

Pfad	Aktiv heute?
`LiveTrader.execute_buy` / `execute_sell`	**JA** (LIVE_TRADING_ENABLED=true, BINANCE_TESTNET=true)
`PaperTrader.execute_buy` / `execute_sell`	**NEIN** (seit Phase L 2026-05-10)
`PaperTrader.update_prices` / DCA-Loop / Tier3-Logik	**JA** — inherited durch LiveTrader
`PaperTrader.get_portfolio`, `_load_state`, `_save_state`, `_reload_state_if_externally_changed`	**JA** — inherited
`PaperTrader.execute_tier3_buy` / `execute_partial_sell`	**dormant** — LiveTrader override genutzt

→ **Die Inheritance trägt zwei Lasten gleichzeitig**: gemeinsame Geschäftslogik (sinnvoll) + dormante Paper-Pfade (unnötig).

1.3 Call-Site-Inventar (read-only Grep-Verifikation)

Datei	Verwendung
`main.py`	`from execution.live_trade import LiveTrader; live_trader = LiveTrader()` (Line 1410)
`command_worker.py`	`from trading.execution.live_trade import LiveTrader` (Line 376) lazy
`baseline_bootstrap.py`	Reader hängt sich an `live_trader.state` an
19 Test-Dateien	mix aus direkter PaperTrader-Instanziierung + LiveTrader-via-inheritance
`dca_manager.py`, `position_manager.py`	nehmen ein Trader-Objekt entgegen (duck-typing)
GUI / reporter.py	nutzen nur `get_portfolio()` Lese-API

1.4 Heutige Schmerzen (aus aktuellen Sessions)

1. **LABEL-1-LIVETRADER-FIX (gefixt heute)**: `_label1_started` musste in **beiden** Klassen separat gepflegt werden — Inheritance-Drift verursacht latente Bugs.
2. `COMMAND-WORKER-DCA-CLEANUP-HOOK` **(Bundle 2b)**: nötig weil unklar war, ob LiveTrader-execute_sell die paper-side cleanup übernimmt (tat sie nicht direkt — main.py-Loop cleant, Worker-Path nicht).
3. **PAPER-vs-TESTNET-Wording-Doppel** (LEGACY-PAPER-BUY-LOG-LABELS, Bundle 2a-safe): 5 Stellen in main.py mit "PAPER BUY"-Strings musten umbenannt werden, weil das Wrapper-Layer (Aufruf von LiveTrader) immer noch "PAPER" sagte.
4. **paper_trade.py:529 vs live_trade.py:819**: zwei Stellen, die `_new_pos` dict bauen. Zukünftige Felder (z.B. `initial_entry_price` aus ENTRY-PRICE-SEMANTIC) müssten in beiden gepflegt werden.
5. **dca_manager state-cleanup-Hooks** (DCA-LOG-ORPHANS-CLEANUP) sind über 3-4 Stellen verstreut.

1.5 Was die Inheritance heute richtig macht

- Klare Separation "WAS passiert bei einem Buy" (Paper) vs. "WIE wird der Buy auf Binance ausgeführt" (Live)
- LiveTrader-Tests können auf PaperTrader-Basis aufsetzen (state-only behavior)
- Eine zukünftige PaperTrader-Reaktivierung (z.B. für Backtesting / Strategy-Validation in pure-paper-mode) wäre möglich

---

2. Motivation für Merge

Pro: - ein einziger Code-Pfad für Trading-Logik → keine Inheritance-Drift mehr - klare Code-Lokalisierung: alles in `live_trade.py` (oder besser: `trader.py`) - weniger kognitive Last beim Onboarding ("warum gibt es ZWEI Trader-Files?") - 1-Spot-Patching für Felder die heute zwei Stellen brauchen - Wording-Konsistenz: keine "Paper"-Reste mehr

Contra: - PaperTrader als reine Bookkeeping-Klasse hat semantischen Wert (Geschäftslogik vs. I/O) - Backtesting / Pure-Paper-Mode (z.B. für Strategy-Audit) ist heute architektonisch möglich; nach Merge nicht mehr trivial - ~ 50-100 Test-Edits sind zu erwarten (19 Dateien × geschätzt 2-5 Anpassungen) - Bot-Recreate nötig (Image-Rebuild) - Refactor-Risiko: ein subtiler Logik-Drift beim Mergen kann Verhalten ändern

---

3. Drei Architektur-Optionen

Option A — Hard-Merge in `live_trade.py` (operator-Wunsch wörtlich)

Vorgehen: 1. Alle PaperTrader-Methoden in `LiveTrader` einklappen (kein super-Inheritance mehr). 2. PaperTrader-Class komplett löschen, `paper_trade.py` löschen. 3. Tests umstellen: alle `from execution.paper_trade import PaperTrader` → entfernen, durch LiveTrader-Stub ersetzen. 4. Wording "PAPER" überall raus.

Vorteile: - Maximale Vereinfachung. - Operator-Wunsch wörtlich erfüllt.

Nachteile: - Kein PaperTrader-Class mehr → keine isolierte Bookkeeping-Klasse für Tests. - Pure-Paper-Backtest-Pfad architektonisch tot. - LiveTrader bekommt ~ 2400 LoC monolithisch. - 19 Test-Dateien müssen umgestellt werden (große Diff-Welle). - Risikoreich, weil Test-Recoveries die Original-PaperTrader-Semantik nutzen.

Option B — Rename + Cleanup (`paper_trade.py` → `trader_core.py`, behalte Inheritance)

Vorgehen: 1. `paper_trade.py` → `trader_core.py`, Klasse `PaperTrader` → `TraderCore`. 2. `LiveTrader(TraderCore)` bleibt. 3. Alle "PAPER"-Strings → mode-aware oder neutral. 4. Dormante paper-spezifische Methoden (z.B. dormant paper-`execute_buy`) entweder löschen oder mit Comment "REMOVED with TRADER-MERGE Phase 1" markieren. 5. Tests: 1 Such-/Ersetz für `PaperTrader` → `TraderCore`.

Vorteile: - Bewahrt die saubere Separation. - Lower-Risk: kein Logik-Rewrite, nur Namens-Refactor. - Operator sieht ein klares `trader_core.py` + `live_trade.py` (eindeutig benannt). - Backtest-Pfad bleibt theoretisch möglich. - Diff klein: ~ 100-150 Zeilen Ersetz + 1 file-rename.

Nachteile: - Operator-Wunsch ("in eine Datei") nicht wörtlich erfüllt. - 2 Files bleiben.

Option C — Behutsame Cleanup-Phase (kleinste Bewegung)

Vorgehen: 1. `paper_trade.py` behalten. 2. Dormante Methoden löschen (z.B. das `execute_buy` Logger-String "PAPER BUY", die "PAPER DCA/PYRAMID"-Strings, dormant `execute_partial_sell` falls Live-Override es ersetzt). 3. Test-Files: nur die ändern die echt PaperTrader-Instanz brauchen. 4. Wording in Strings nur dort fixen wo es operator-sichtbar ist.

Vorteile: - Minimal-Risk. - Kein Bot-Recreate nötig (außer für Wording-Strings).

Nachteile: - Operator-Wunsch deutlich verfehlt. - Inheritance-Drift bleibt. - Doppelt-Pflegen ENTRY-PRICE etc. bleibt.

---

4. Empfohlene Option

Option B ist das beste Kosten-Nutzen-Profil: - Operator bekommt einen sauberen, einzelnen aktiven Trader (`LiveTrader`). - Die Basisklasse heißt nicht mehr „Paper", sondern semantisch korrekt `TraderCore` . - Refactor-Risiko ist niedrig (rein Rename + Wording-Cleanup, keine Logik-Verschiebung). - 19 Test-Dateien sind 1× sed-replace betroffen, kein Logik-Edit. - Backtest-Pfad bleibt theoretisch erhalten (für T2-SOLANA-SHADOW oder T3-COPY).

Falls Operator den vollen Merge (Option A) explizit will, muss er die ~ 2400-LoC-Datei-Größe und das Test-Rewriting akzeptieren — und die architektonische Backtest-Option opfern.

---

5. Scope Option B (empfohlen, detailliert)

5.1 Datei-Rename

```
trading/execution/paper_trade.py   →  trading/execution/trader_core.py
```

5.2 Klassen-Rename

```
# trader_core.py
class TraderCore:           # was: PaperTrader
    ...

# live_trade.py
class LiveTrader(TraderCore):    # was: LiveTrader(PaperTrader)
    ...
```

5.3 Wording-Cleanup

Datei	Strings
`trader_core.py` (ex-paper_trade.py)	`"📈 PAPER BUY:"` → `"📈 BUY:"` (dormant-Pfad-Log)
`trader_core.py`	`"📈 PAPER DCA/PYRAMID:"` → `"📈 DCA/PYRAMID:"`
`trader_core.py`	`"📈 PAPER SELL:"` → `"📈 SELL:"`
`trader_core.py` Kommentare	„Paper"-Wording → „Core" / „state-only"
`live_trade.py` Kommentare	„PaperTrader" → „TraderCore"
`command_worker.py` Kommentare	analog
`trading/docs/architecture_*.md`	Architekturdiagramm aktualisieren

5.4 Import-Updates

Aufrufer	Vorher	Nachher
`main.py`	`from execution.paper_trade import PaperTrader`	(entfällt — nur LiveTrader import bleibt)
`command_worker.py` Kommentar	`PaperTrader` → `TraderCore`	
Tests (19 files)	`from execution.paper_trade import PaperTrader` → `from execution.trader_core import TraderCore`	
Tests	`PaperTrader()` → `TraderCore()`	

5.5 Dead-Code-Cleanup (innerhalb des Refactors)

Im `trader_core.py` :

- `execute_buy` dormant-path (Line 364-560): wird nie aktiv ausgeführt, weil LiveTrader übersteuert. Empfehlung: behalten als „state-only fallback" mit klarem Comment `# DORMANT — LiveTrader.execute_buy overrides this. Kept for explicit state-only invocation (tests, dry-run).`

- `execute_partial_sell` dormant-path (Line 800+): analog, behalten als Fallback.
- `execute_tier3_buy` dormant-path: analog.
- `update_prices` (Line 1024): **KEEP** — diese Methode ist aktiv (via LiveTrader-Inheritance).
- `_load_state` / `_save_state` / `_reload_state_if_externally_changed`: **KEEP** — aktiv via Inheritance.

5.6 Tests

- Refactor-Konsistenz-Test (neu): `test_trader_core_rename.py` AST-Guard, `class TraderCore` existiert, `PaperTrader` nicht.
- Bestehende 19 Test-Dateien: 1× sed `s/PaperTrader/TraderCore/g; s/from execution.paper_trade/from execution.trader_core/g`.
- Smoke-Test: Bot-Boot post-Refactor, alle 994 Tests-Suite-Lauf, mindestens dieselbe pre-existing-fail-Quote (7/44).

5.7 GUI / Docs

- `gui/docs/roadmap/ROADMAP.php` Roadmap-ID `TRADER-MERGE-PAPER-LIVE` → done nach Cutover.
- `trading/docs/architecture_*.md`: Diagramm-Box `PaperTrader (Basisklasse)` → `TraderCore (Basisklasse)`.
- Memory-Datei MEMORY.md: Update auf Trader-Architektur.

---

6. Cutover-Plan (Option B, SOT-1d)

```
P0  Operator GO TRADER-MERGE-PAPER-LIVE (Option B).
P1  Read-only Re-Recon (Test-Inventar, Confirm-Counter).
P2  Watchdog freeze (bot + worker crontab).
P3  Backup state-files + bot-data Volume nach
    /root/state-backups/trader-merge-<utc>/.
P4  Stop clawbot + clawbot-worker (gui-db/gui/searxng untouched).
P5  Code-Refactor:
    - git mv paper_trade.py trader_core.py
    - sed-replace `class PaperTrader` → `class TraderCore`
    - sed-replace imports + class-uses in 19+2 files
    - Wording-Strings cleanup in trader_core.py
    - live_trade.py imports + base-class-Name update
P6  AST + Unit-Tests offline:
    - python3 -c "import ast; ast.parse(trader_core.py)"
    - python3 -m unittest discover -s trading/tests
P7  docker compose build (bot+worker image rebuild).
P8  Run alle Bundle-Tests + cumulative im candidate-image.
P9  3-Way MD5 verify trader_core.py + live_trade.py.
P10 Recreate worker, dann bot.
P11 Watchdog-Tick spawn main.py + Health verify.
P12 Live-verify: ein BUY oder SELL nach Refactor läuft normal.
P13 Watchdog re-enable.
P14 Roadmap-Update + Architecture-Doc Update + Commit.
```

Geschätzte Cutover-Dauer: 90-120 min (Refactor groß).

6.1 Rollback-Plan

```
git revert <commit-hash>
docker compose build
docker rm -f clawbot clawbot-worker
docker compose up -d clawbot-worker
docker compose up -d clawbot
```

State-Backup ist intakt — kein Datenverlust.

---

7. Boundaries

- **Bot-Touch ja** (Recreate + Image-Rebuild).
- **Worker-Touch ja** (Recreate + Image-Rebuild).
- **GUI-Touch**: nur Roadmap.php + Architecture-Doc.
- 0× DB-Migration.

- 0× DB-Mass-Mutation.
- 0× Mainnet.
- 0× CommandBus-Version-Bump.
- 0× Strategy-Parameter-Change.
- 0× State-Mutation außer file-rename.
- 0× Push ohne separates GO.

---

8. Risk

Risiko	Severity	Mitigation
sed-replace übersieht Edge-Case (z.B. Kommentar mit `PaperTrader`)	mittel	grep nach `PaperTrader|paper_trade` post-refactor; alle Treffer reviewen
Test-Refactor bricht 1+ Test	mittel	Pre-Cutover 994/994-Snapshot, danach selbe oder bessere Pass-Quote
Import-Order-Issue post-rename	klein	python3 -c "from execution.live_trade import LiveTrader" smoke pre-build
Subtiler Logik-Drift weil ein Cleanup-Comment Code falsch interpretiert	hoch	`git diff` auf trader_core.py vs paper_trade.py akzeptiert nur Rename + Wording, NULL Logik-Diff
Memory-Tasks brechen weil sie `PaperTrader` referenzieren	klein	grep MEMORY.md, Update gewährt
Subagents / IDE referenzieren `paper_trade.py`	klein	Doc-Update
Architecture-PDFs in /srv/shares/planung referenzieren PaperTrader	klein	nicht-blockierend, Doc-Update follow-up

---

9. Acceptance Criteria

- `paper_trade.py` existiert nicht mehr (oder ist ein 3-Zeilen-Shim für Backwards-Compat).
- `trading/execution/trader_core.py` existiert mit Klasse `TraderCore`.
- `LiveTrader(TraderCore)` in `live_trade.py`.
- 0 Vorkommen von `PaperTrader` in der reachable Codebase (außer evtl. in Migrations-Kommentaren oder MEMORY.md historisch).
- 0 Vorkommen von `"PAPER BUY"` / `"PAPER SELL"` / `"PAPER DCA"` etc. in reachable Strings.
- Trading-Suite Pass-Quote ≥ vor Refactor (pre-existing 7/44 unchanged).
- Bot+Worker healthy post-Cutover.
- LiveTrader BUY/SELL nach Cutover funktioniert (mindestens 1 echter Buy + 1 echter Sell beobachtet).
- 3-Way MD5 Repo == Image == Container für `trader_core.py` + `live_trade.py`.

---

10. Out-of-Scope (separate Pläne, nicht in diesem Refactor)

- ENTRY-PRICE-SEMANTIC (eigenes Bundle nach Refactor — wird LEICHTER weil nur eine Stelle zu pflegen).
- Pure-Paper-Backtest-Pfad (falls jemals gewünscht, separates Design).
- Tier3-Cleanup (separate Phase).
- Strategy-Parameter-Tuning.

---

11. STOP

Kein Code-Change vor: - Operator-GO TRADER-MERGE-PAPER-LIVE (Option A, B oder C). - Backup-Pfad steht. - Cutover-Fenster gesetzt (~ 60-90 min Bot-Downtime). - Optional: vorab `git log paper_trade.py` für historische Last-Commits-Check.

Bis dahin: - Aktuelle Inheritance bleibt funktional. - ENTRY-PRICE-SEMANTIC-Phase wäre zweimal zu patchen. - Wording-Drift bleibt.

---

12. Klare Empfehlung

GO Option B (Rename + Cleanup) als idealer Mittelweg zwischen Operator-Wunsch und Risk-Vermeidung.

Falls Operator den Hard-Merge wirklich will: **GO Option A** mit klarem Verständnis der zusätzlichen Test-Edits + Logik-Risiko.

Falls Operator unsicher: **GO Option C** als nicht-invasiver erster Schritt, danach Option B beim nächsten Bedarf.