Página 3

Continuing the doc truth-check, here’s the verification rundown for both guides plus the concrete deltas you’ll want to edit next.

Findings for GUIA_DESARROLLO.md

Still accurate

- Package layout, system summaries (mutations JSON loader, tiered loot pipeline, anti-farm hooks, synergy architecture) all match the current classes (MutationDataLoader, TieredLootDataLoader, AntiFarmManager, SynergyManager, etc.).
- Runtime flow (MobSpawnHandler → TierManager → MutationAssigner/SynergyManager, TierLoot interception, mutation tick/on-hit handling) matches the real implementations.
- Mutations budget table (EPIC 25 / LEG 50 / MYTHIC 75 / DEF 100) reflects `MutationAssigner#getBudgetForTier`.
- Telemetry/event hooks cited (TierLootTelemetryEvents, AntiFarmEvents.*, AntiFarmDashboardEvents, TierEvents) are live and wired exactly as described.

Needs correction

Topic	Doc statement	Reality & evidence
Phase 1 lifecycle order	Lists SynergyModule.initializeDetections() first and omits the config load/save cycle.	`Legacycreaturescorey#onInitialize` begi with CoreyConfig.INSTANCE.loadO then registers data loaders; Synerg later through SynergyManager.bootstrap
Registered data loaders	Guide only mentions MutationDataLoader, TieredLootDataLoader, MobAttributeDataLoader.	registerData() also wires MobTierRuleDataLoader, Bion and AntiFarmExclusionDataLoader, calls MutationRuntime.register(), M and CoreyLootModifiers.register().
Spawn rule source	Says TierManager.determineAllowedTiers uses tags like `tier_leve`.	The current implementation pulls from MobTierRuleDataLoader (data and falls back to EnumSet.noneOf i no longer drive the decision.
Difficulty formula	States `effective = ceil(avg((global + player)/2))` and flags config-based probabilities/client sync as “pending.”	EffectiveDifficultyCalculator applies averages, biome multipliers, and an penalties; TierProbabilityCalculator multipliers (epic/legendary/mythic/ and DifficultyManager syncs snaps via DifficultySyncPayload.
Command / manager registration	Guide omits MutationCommand, CoreyHudCommand, and CoreyHealthMonitor from the lifecycle section.	registerEvents() now registers thos previously listed hooks.

Findings for API_REFERENCE.md

Confirmed accurate

- CoreyAPI method descriptions, behavior, and failure semantics match the actual methods (including forced-mutation handling and optional returns in `CoreyAPI#spawnCategorizedMob`/getTier/getMutations/registerSynergyProvider/getSynergyStatuses).
- LegacyCreaturesTierApi and AntiFarmDashboardApi sections describe the real signatures and return types verbatim.
- ClientEffectPayload registration instructions align with ModNetworking.init() and the client-side receiver in `ClientEffectHandler`.
- Event listings (TierEvents, TierLootEvents, TieredLootTableEvents, AntiFarmEvents, AntiFarmDashboardEvents) match the actual Fabric events, names, and intent.
- Example synergy provider code mirrors the real SynergyProvider contract (module detection, validate(), loot hook usage).

Minor clarifications to consider

- SynergyProvider now declares default implementations for validate, onRegister, onMobTiered, and onLootGenerated. The sample interface in the doc still works, but adding a parenthetical "(methods have default no-op bodies; override the ones you need)" would reduce confusion.
- When describing LegacyCreaturesTierApi.getTier, note that it intentionally returns empty for mobs still in NORMAL, matching the implementation.

Quality gates

- **Build:** PASS (latest `./gradlew build` run in this workspace exited with code 0; no code changes were made afterward).
- **Lint / Tests:** Not rerun for this documentation-only audit; existing status unchanged.

Recommended next steps

1. Update GUIA_DESARROLLO.md with the corrected lifecycle ordering, the additional loaders/commands, the current tier-rule source (`mob_tier_rules.json`), and the modern effective-difficulty formula.
2. Optionally tweak API_REFERENCE.md to mention that SynergyProvider methods are defaulted, so modders only need to override the hooks they care about.

Once those edits land, both documents will reflect the current codebase faithfully. Let me know if you'd like me to draft the wording updates.