

D06 账户状态与准入域 · 设计文档

本域总览见《账号计费中心 · 结构大纲》。本文为该域完整设计：定位与边界 / 角色与依赖 / 数据模型 / 核心流程 / 接口契约 / 关键约束与验收标准。

1. 定位与边界

定位：全仓「这个客户此刻能不能用」的唯一判定者与唯一下发者。本域回答三件事——客户处于五态中的哪一态；状态键里放什么（载荷契约）；读侧怎么判、何时放行。它是全仓契约最敏感的一个域：键名、载荷、版本语义与读侧规则是跨服务契约，写侧改了字段而读侧不认，会以「放行欠费客户」或「误封已充值客户」两种相反形式暴露。

核心概念

概念	承载	语义
状态键	状态存储中的客户级键	对外实时准入媒介，唯一生产者为本域、消费方只读；键缺失或过期一律放行
派生态	state	active / grace / overdue / banned / closed 五值，判定生命周期优先
封锁位	blocked 与 block_reason	拒绝语义的载体：quota 按额度拒绝，status 按状态拒绝
宽限窗	grace_until	仅 grace 态透出；窗是否走完由消费方按自身时钟判定
准入真相	available	唯一公式 = 现金 + Σ(在营未过期赠金剩余) + 信用额度；大于 0 即放行
状态版本	state_version	客户级单调版本，库层在行锁内取全局序列赋值；写侧去旧专用
欠费标记	overdue_since	派生量，供报表、清单与欠款分级使用；不是准入真相

关键口径

- 准入读实时可用额度，不读派生的欠费标记：标记只在钱包变更事务末尾重算，赠金过期属被动衰减、不落事务，故标记会滞后；拿它做准入会把现金为正、仅赠金刚过期的客户拦在门外。
- 状态含时间条件，故判定的一部分必须留在读侧：宽限窗内服务、窗外停服无法在写侧固化成布尔位，消费方必须自己比时钟。
- 载荷扩展向后兼容：新增字段只服务写侧去旧，消费方忽略即可——这是版本字段能安全加入的原因。
- 可容忍偏差，但方向必须是「宁可放行」：键缺失、过期、状态存储不可达一律放行，敞口有界。

边界

- 不管钱：现金余额、赠金剩余、信用额度的记账与重算在钱包与账务域；本域只读其派生出的可用额度。
- 不管准入执行：拦截动作在数据面网关；本域只负责判定、下发与状态查询。
- 不管计量与计价（计量与计费域）、不管订单与退款（订购与支付域、退款与对账域）、不管风控分级台账（风控与合规域）、不管通知投递（通知与触达域）。

- 运营面写入口的语义分属：状态标记、信用额度、调整、宽限四类写入口的资金语义在钱包与账务域，「它们如何改变派生态」属本域；端点索引统一登记在 D11。

2. 角色与依赖

能力矩阵

能力	本域	计费路由	数据面网关	运营面	门户用户
判定派生态	✓（唯一实现）	✗	✗	✗	✗
写状态键	✓（唯一生产者）	✗	✗	✗	✗
读状态键	✗	✓	✓	仅投影	仅投影
状态查询端点	✓（提供）	✓	✓	✓	✗
状态标记与宽限设置	语义在本域	✗	✗	✓	✗
派生态展示	✗	✗	✗	✓	✓（本人）

- 状态键的读写权属严格单向：本域写，其余角色只读；命名空间内出现非本域写入的内容属安全事件。
- 准入执行与计费扣费各自独立，但依赖本域下发的同一个判定，不存在第二处判定实现。

上游依赖

上游	依赖内容	语义
钱包与账务域	现金余额、赠金剩余、信用额度、宽限截止	可用额度公式的唯一输入
钱包与账务域	欠费标记重算结果	派生量，供清单与账龄分级
账号与身份域	客户标识与工作区归属	状态键的键主体
订购与支付域	客户创建点	创建即写键的触发来源

下游被引用

引用方	引用内容	语义
数据面网关	状态键与封锁语义	每请求前置准入，按原因分流到额度拒绝或状态拒绝
计费路由	状态查询端点	扣费前判定整批拒绝，在途损失入台账
门户与自助	派生态与宽限截止	欠费告知与还款入口的展示口径（详见 D12）
风控与合规域	欠款账龄与封锁分级	提醒 / 催收 / 冻结的事件输入
通知与触达域	变更事件	欠费与解封告知的触发面（投递不在本域）
运营管理	状态与宽限的只读投影	客户台账与审计索引

3. 数据模型

本域无自有表。事实载体有两个：状态存储中的客户级状态键（键名、载荷与存活期共同构成对外契约）与客户表上的状态与版本列（`status` / `grace_until` / `grace_granted_at` / `overdue_since` / `credit_limit` / `state_version`）。前者是派生态的对外投影，后者是判定输入与版本序的落点；两者都不由本域独立拥有，本节只描述读取关系与派生依赖。

3.1 域内关联总览

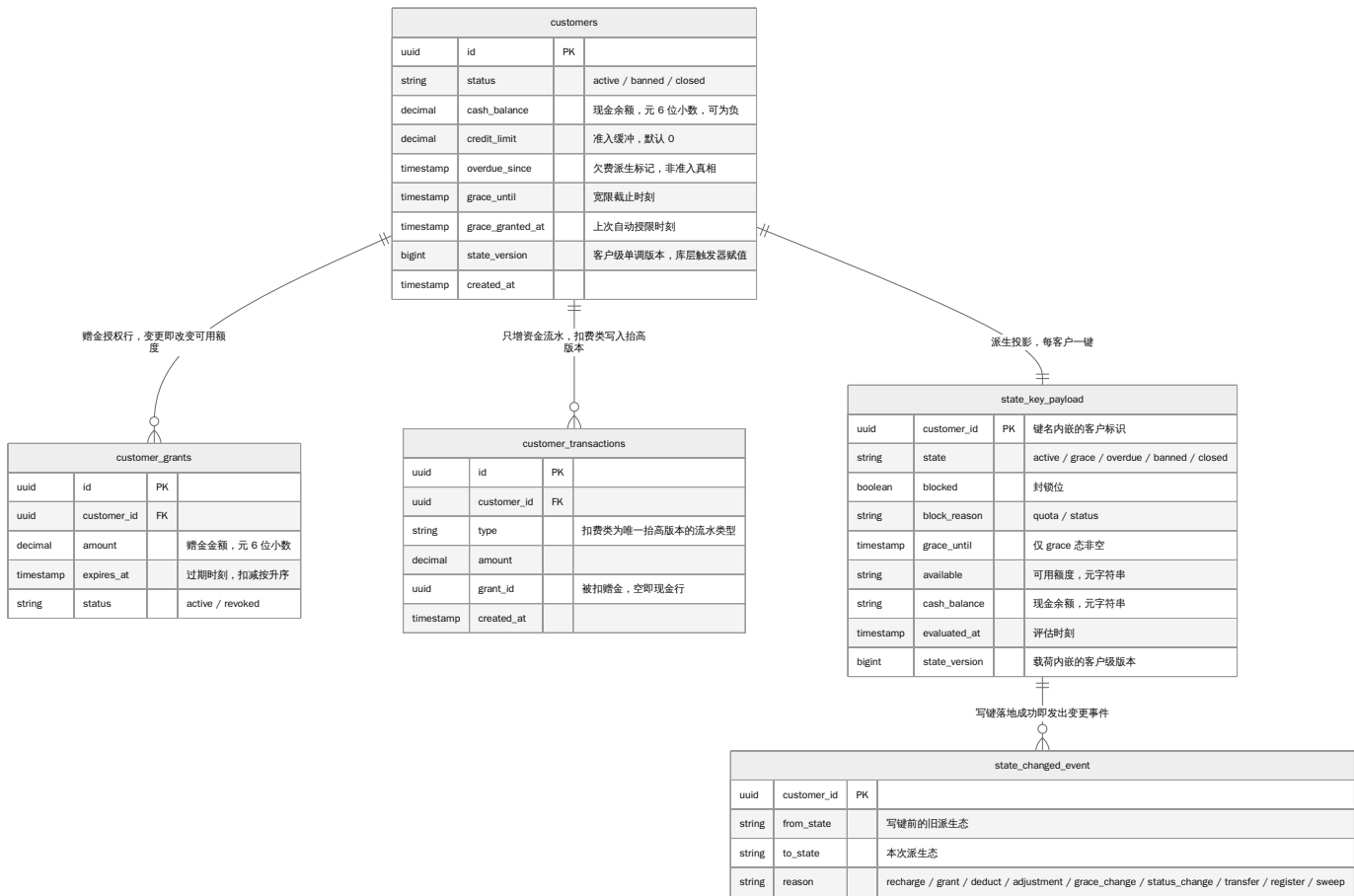


图 1

3.2 域间引用

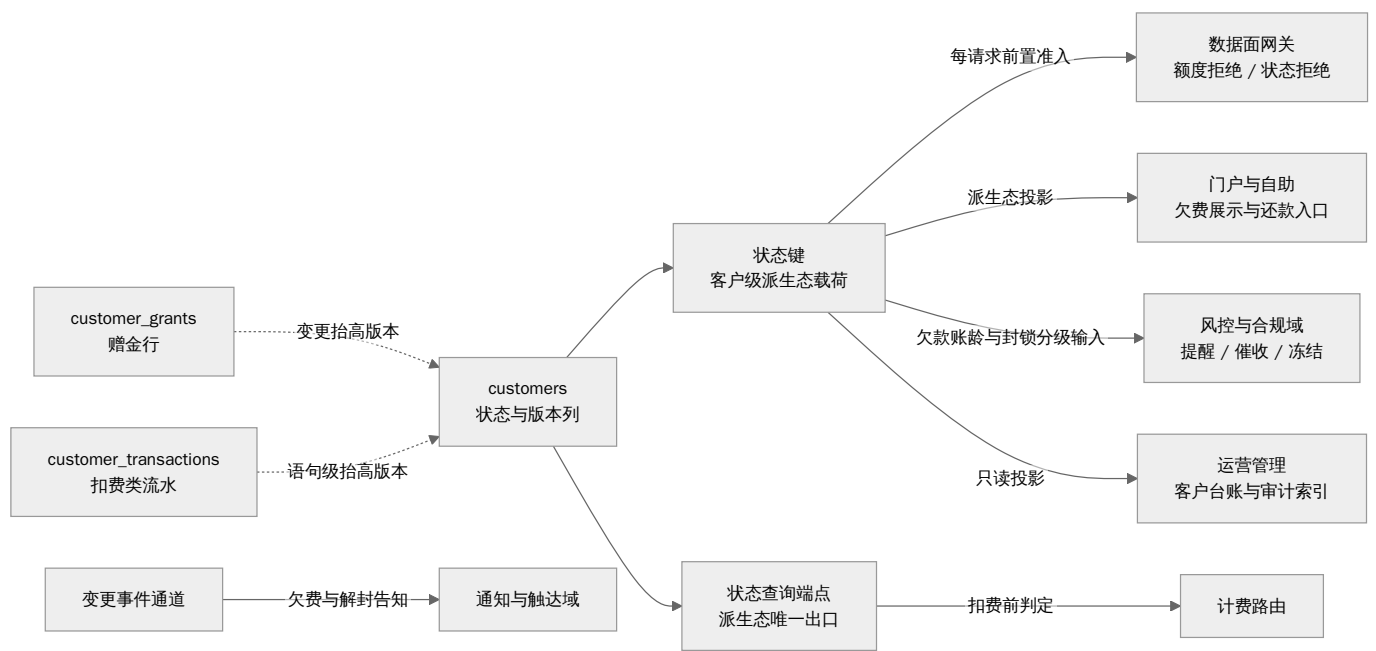


图 2

- 赠金行与流水对客户只落弱引用：版本抬高由库层触发器完成，本域不持有其写路径。
- 键与事件是两个面：键是判定面（含时间条件，读侧要算），事件是通知面（只告知，不承载判定）。

3.3 表清单（本域无自有表，引用 3 表）

表	职责（本域视角）	关键字段
customers	状态与版本的落点：判定输入的来源，版本序的载体	status / cash_balance / credit_limit / overdue_since / grace_until / grace_granted_at / state_version / created_at
customer_grants	赠金剩余参与可用额度；授权行变更抬高客户版本	customer_id / amount / expires_at / status
customer_transactions	扣费类流水的写入抬高客户版本；赠金已用量聚合	customer_id / type / amount / grant_id / created_at

状态存储中的状态键与变更事件通道不是关系库表，故不入本表清单，其键名与载荷见 4.2 与 4.3。

3.4 设计约束

1. 键名唯一形态：键名形态为 STATE_KEY_PREFIX:{customer_id}:state——STATE_KEY_PREFIX 是部署内共享的客户状态命名空间常量，取值为两段（平台共享前缀 + 固定客户段 cust）；中段为客户标识，尾段为固定后缀 state。变更事件通道常量 STATE_CHANNEL 同属该命名空间（尾段 state-changed）。两者均不得按环境改写，否则消费方读到的是另一片命名空间。
2. 载荷字段固定九项：customer_id、state、blocked、block_reason、grace_until、available、cash_balance、evaluated_at、state_version；金额一律元字符串、6 位小数。
3. 宽限截止的透出条件：grace_until 仅当派生态为 grace 时非空，其余态一律为空——消费方据此把「窗内服务」写成一次时钟比较。

4. 写入必须原子比对：写键走「读旧值 → 比对版本 → 写键」三步的原子脚本，在状态存储的单线程执行模型内完成、无并发窗口；等价的「先读后写」会被并发发布插队，去旧形同虚设。
 5. 版本由库层赋值：`state_version` 由客户表上的触发器在行锁内取全局序列（`customer_state_version_seq`）赋值，应用层只读不写；行锁把同客户的并发写串行化，后取到锁的事务必然取到更大序列值，故版本序 = 提交序。
 6. 版本采样先于内容采样：派生必须「先版本、后内容」，使「版本不大于内容新鲜度」恒成立；反序在已提交读下会产出「旧内容 + 新版本」并被当最新值接受，修复失效。
 7. 该次序由构造保证：状态派生的唯一入口不接受调用方传入的可用额度，两值一律由函数内部按固定次序采样——调用方无从「先算好内容再进来读版本」。
 8. 键的存活期双档：放行态用短存活期 `state_key_ttl_seconds`（默认 600 秒），封锁态用长存活期 `state_blocked_key_ttl_seconds`（默认 7200 秒）；后者不得低于一个加固周期 `state_sweep_interval_seconds`（默认 300 秒）。
 9. 命名空间唯一生产者：本域是该命名空间唯一写者；键已存在但内容不是合法对象时属「外力写入」，写入照旧（fail-safe）但必须落告警。
 10. 状态值域封闭：派生态仅取五个值，客户经营状态仅取 `active` / `banned` / `closed` 三值；`closed` 为终态，无退出路径。
 11. 不落任何自有数据：本域不建表、不建列，全部事实读自客户行与赠金行、流水行；可用额度公式只在钱包与账务域一处维护，本域只调用不复制。
-

4. 核心流程

4.1 状态派生与判定（生命周期优先）

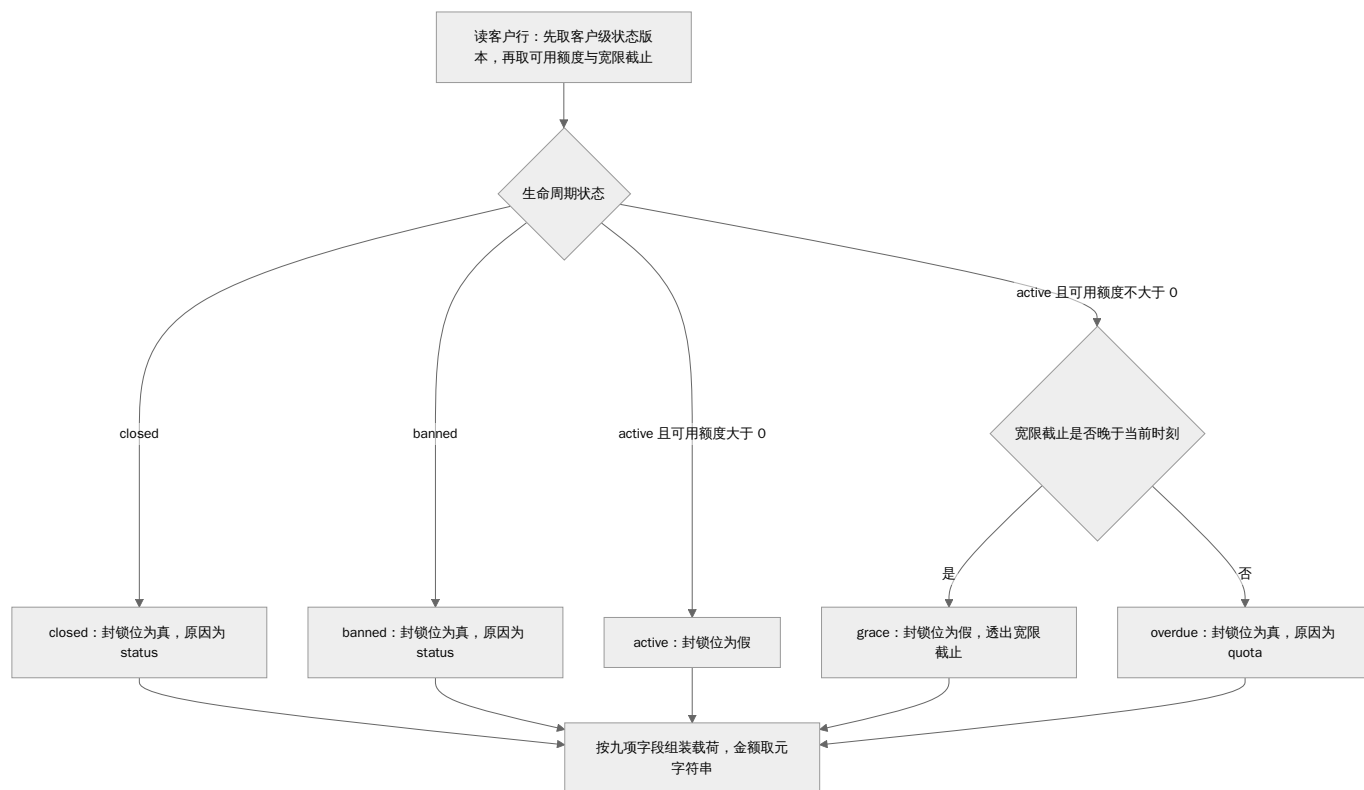


图 3

- 判定次序不可调换：生命周期优先——已注销或已封禁客户即便额度充裕也不得放行。
- 门槛是严格大于 0：`available > 0` 即在营态；不大于 0 才进入宽限判断。
- 宽限关闭时（`overdue_grace_days` 默认 0）`grace` 态不出现，等价于额度耗尽即停服。

4.2 发布、版本比对与去旧

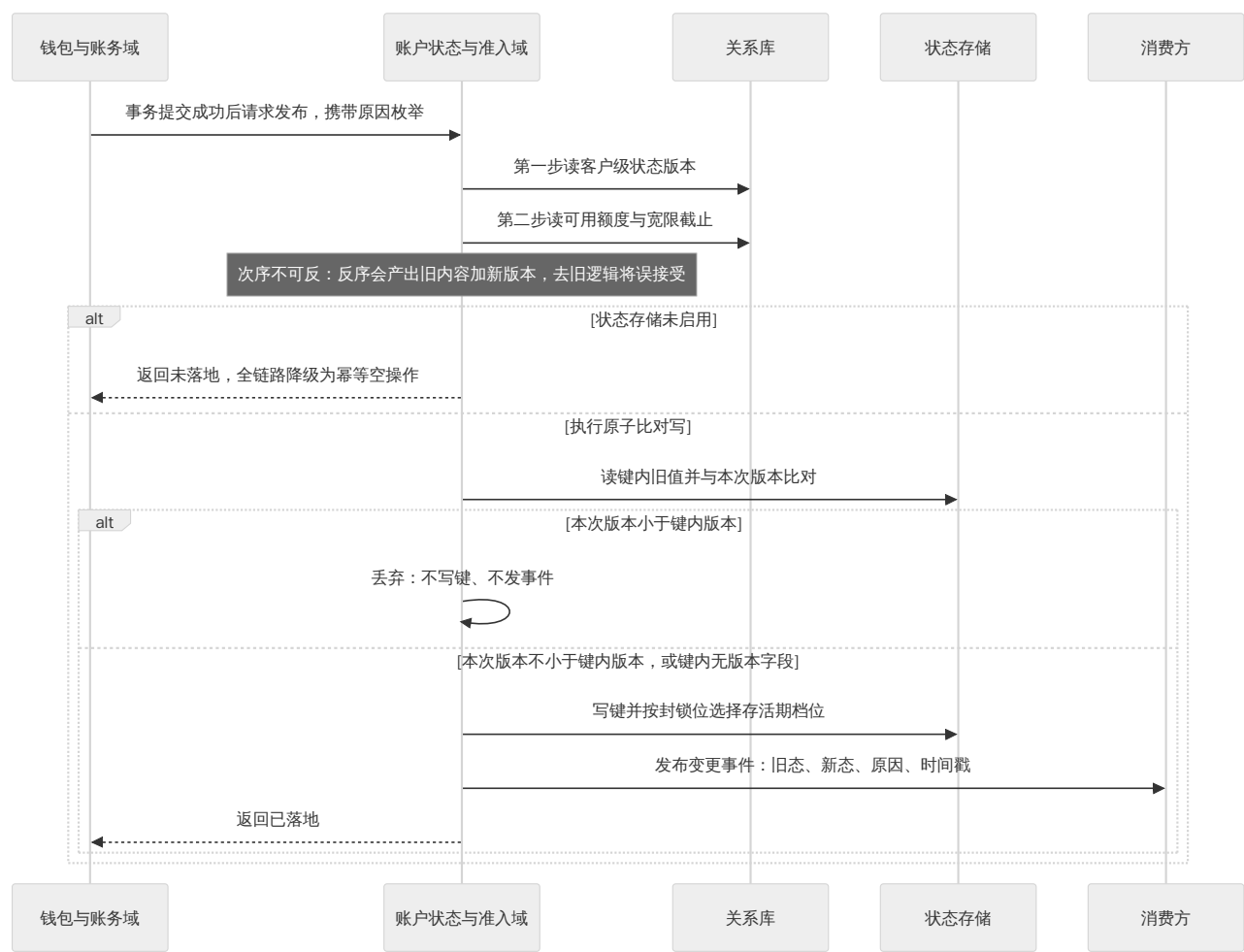


图 4

- 比较规则三支：新版本小于键内版本即丢弃；相等允许写入（重复发布，如加固续期与多副本同值）；键内无版本字段视为 0，任何新版本都可覆盖——滚动升级期的兼容通道。
- 异常一律告警不抛出：写键与发事件失败不得改变业务事务的成败。
- 键已存在但内容不是合法对象时覆盖行为照旧，同时落安全告警（非本域写入只可能是命名空间被外部进程污染）。
- 事件与键同源同刻：只在键真正落地后发出，旧态取自键内旧值；丢弃分支不发事件。

4.3 消费方读侧判定

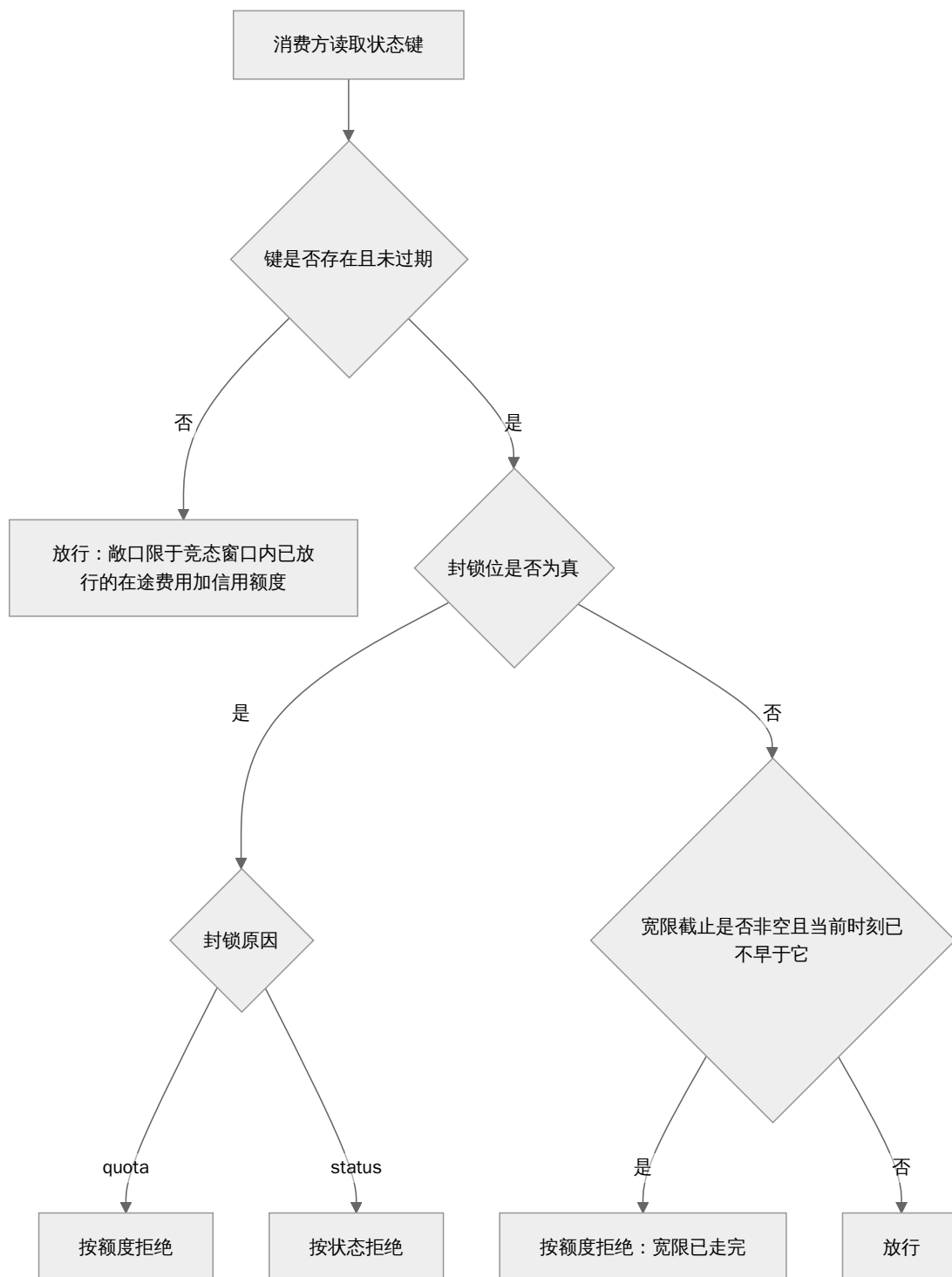


图 5

- 读侧只做三件事：看封锁位、比宽限时钟、缺键放行；不需要感知版本字段——版本只服务写侧去旧。
- 宽限态键内封锁位为假，故消费方必须独立完成时钟判定；这是「写侧无法把时间条件固化成布尔位」的直接结果。
- 键缺失或过期一律放行。敞口上界 = 在途未落账请求费用合计 + 信用额度；串行调用下约为单笔最大费用 + 信用额度。
- 信用额度为 0 不等于零敞口：准入只要求 `available > 0`，单笔费用可完全透支。

4.4 双档存活期与误封上界

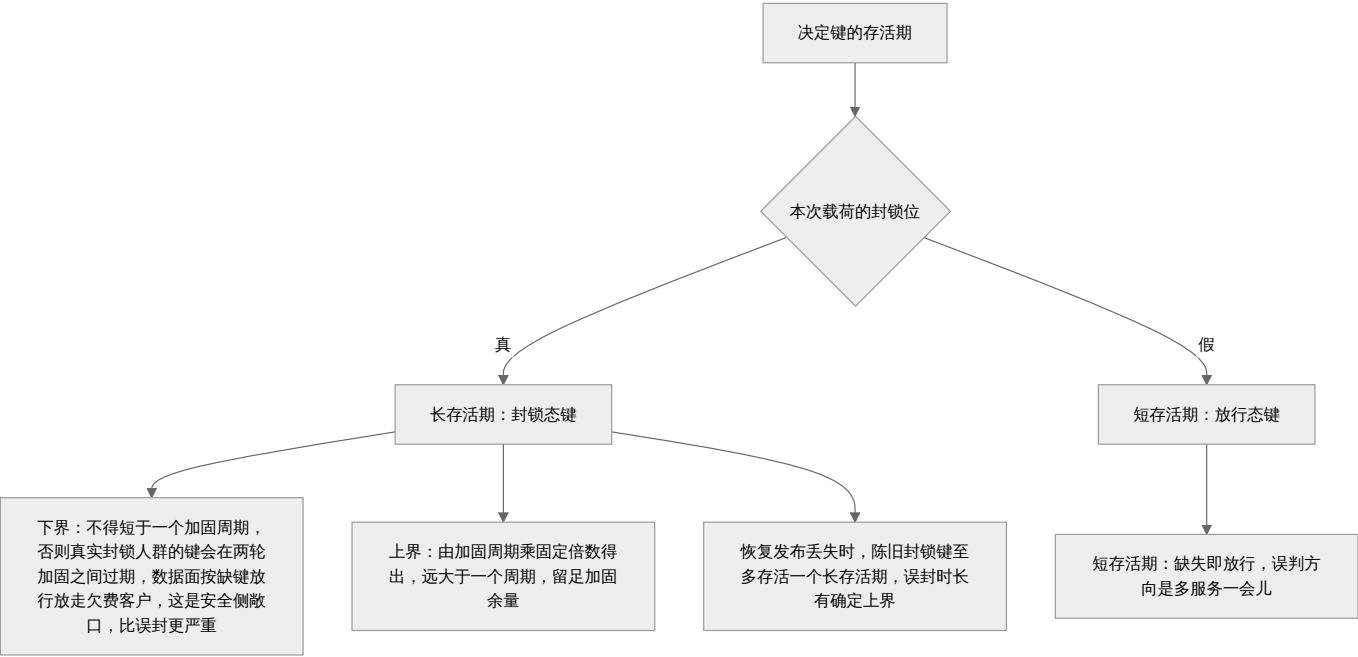


图 6

- 长存活期由双向夹逼得出：下界不低于一个加固周期，上界取周期固定倍数（默认 24 倍即 2 小时）；两条须同时满足。
- 若封锁态键改为不过期，「解封已提交但恢复发布丢失」会让陈旧封锁键永不过期，该客户持续被拒直至全量轮转扫到——这正是引入长存活期的原因。
- 两档存活期不可互换：放行态过长只是延长误判窗口，封锁态过短则直接放走欠费人群。

4.5 解封路径的同步写键

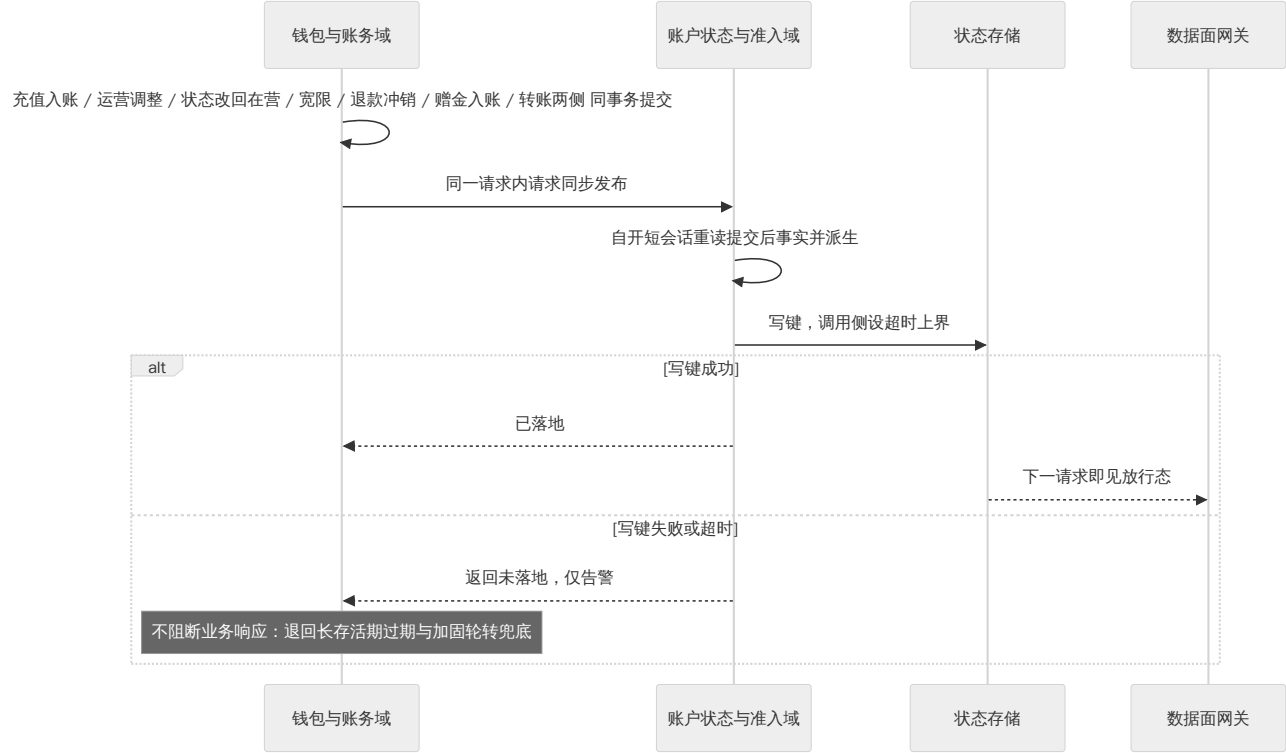


图 7

- 解封类路径必须同步写键：只建异步任务时，「已调度但未执行即进程终止」会留下陈旧封锁键，用户侧表现为持续的额度或状态拒绝。
- 同步发布在业务事务提交之后执行，且自开短会话重读事实——调用方会话可能持有陈旧属性，自开会话读到的才是提交后事实。
- 超时上界由 `state_publish_sync_timeout_seconds`（默认 2 秒）给出：状态存储挂起不得无限拖住响应，超时即放弃同步写、退回兜底链路。
- 同步与异步可并存，重复发布无害——去旧逻辑保证同值重复写入与乱序旧值写入都被正确处理。

4.6 加固轮转的两段扫描

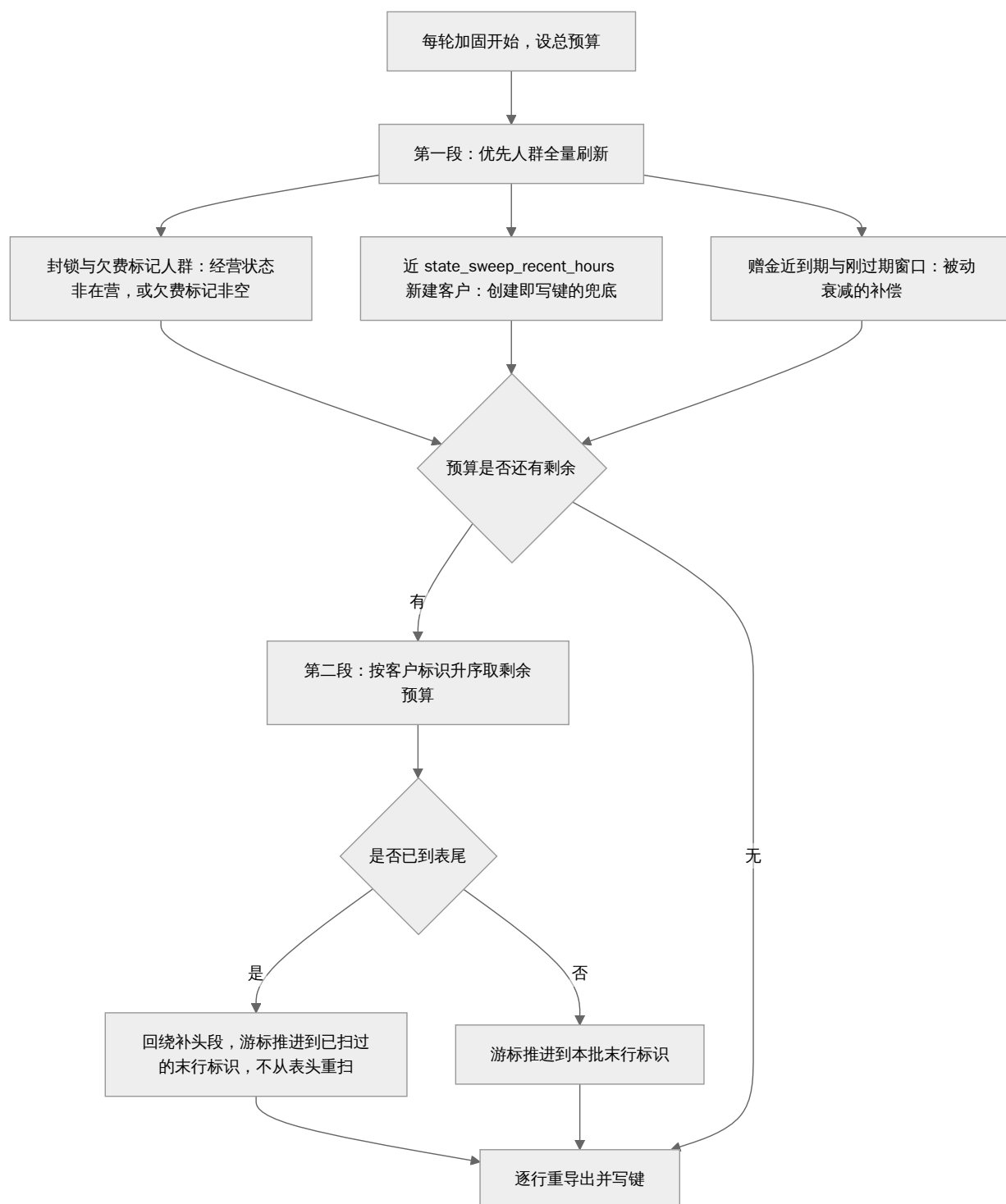


图 8

- 轮转周期公式： $[\text{客户数} \div \text{每轮预算}] \times \text{加固间隔}$ （预算默认 1000、间隔默认 300 秒）；「等一个轮转周期」对全部客户成立，靠的就是第二段游标推进。
- 游标必须推进到本批末行的标识而非置空：置空会让下一轮从表头重扫，头段被重复刷新而中段要等完整周期，周期退化为两倍。
- 优先人群刷新间隔不大于一个存活期，与封锁键长存活期配合，使真实封锁人群的键在任意时刻都至少被加固过一次。
- 该循环读连接与业务请求连接隔离，避免全量扫描侵蚀扣费接口的连接预算；多副本并发幂等安全。

4.7 创建即写键

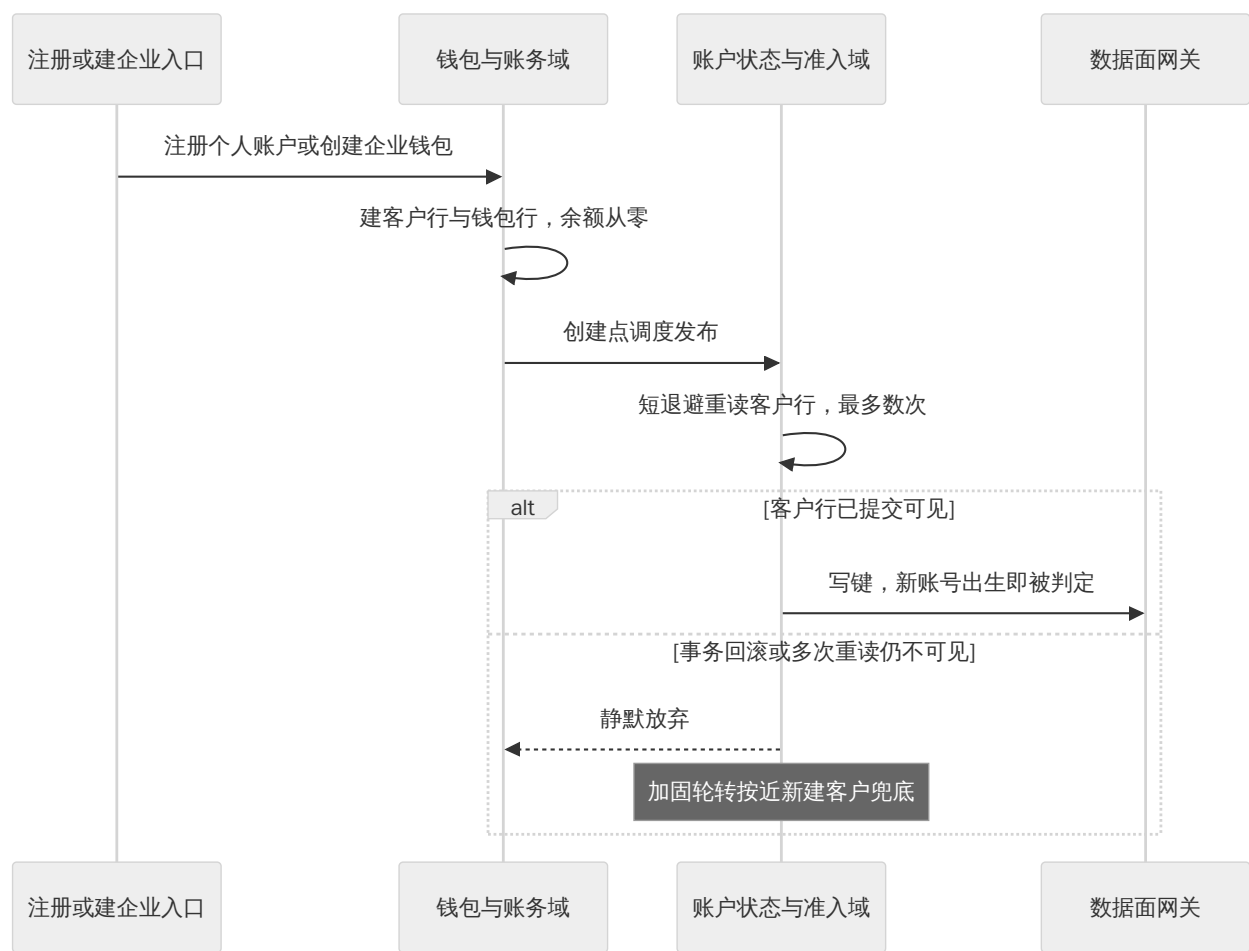


图 9

- 创建点调度的原因是无赠金的新账号一出生派生态即为欠费（可用额度为 0）；不写键则数据面按缺键放行一次消费，批量注册即成系统性白嫖。
- 钱包行在业务层创建、由接口层稍后提交，创建点拿不到提交后时机，故须退避重读；可见即写键，回滚则放弃（不存在的客户本就不该有键）。
- 兜底是双重的：加固轮转的近新建客户段，加上键的存活期本身。

4.8 欠费派生标记的重算与宽限授限

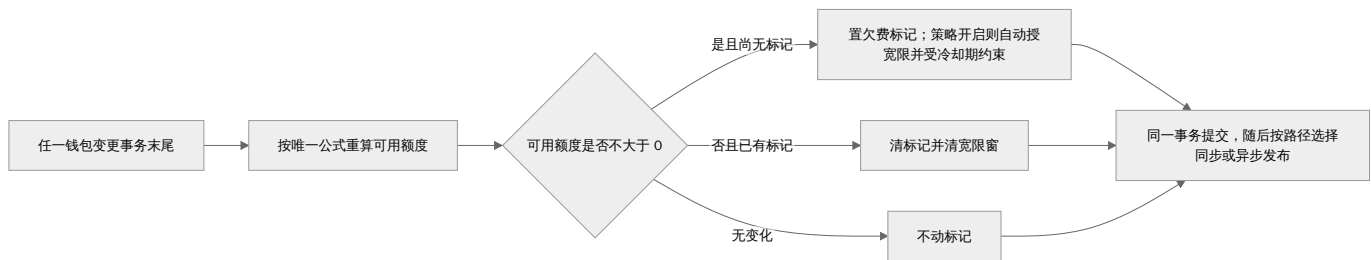


图 10

- 该处是欠费标记的唯一置位与清除点，取代两条旧路径：「扣费成功即清标记」（在途小额扣费会错误解封）与「入账无条件清标记」（不足额充值会错误解封）。

- 自动授信三条件缺一不可：策略天数大于 0、当前不在宽限窗内（运营预设优先）、距上次自动授信已超过冷却期 `grace_regrant_cooldown_days`（默认 30 天，防「最小充值循环刷宽限」）。
- 恢复（可用额度转正）即清宽限窗，下次欠费按规则重新授信；标记是派生量，赠金过期属被动衰减、不落事务，会滞后到下一次钱包变更，故准入与生命周期出口都不读它。

4.9 欠款生命周期分级与运营杠杆

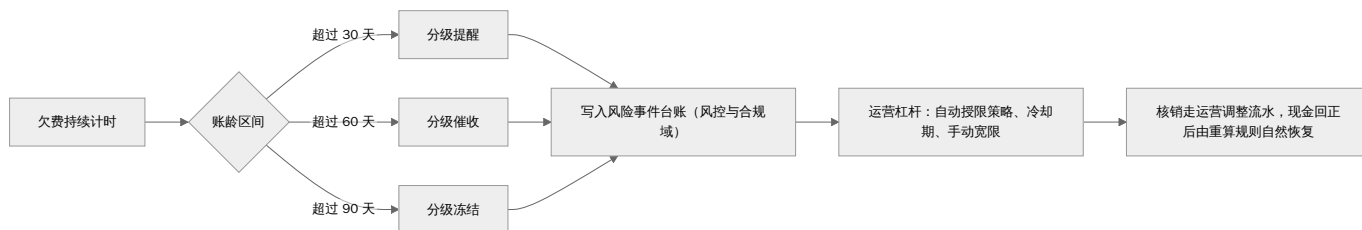


图 11

- 门槛 30 / 60 / 90 天按欠费持续时长递进：提醒 → 催收 → 冻结；分级判定与事件落库在风控与合规域。
- 运营杠杆是三个正交旋钮：自动授信策略（天数）、冷却期、手动宽限（延长或清零）；宽限设置只改宽限截止，清零即欠费即时停服。
- 核销是运营调整流水（正方向），现金回正后由重算规则自然恢复；解封方向必须走同步写键。

4.10 状态存储不可用时的降级

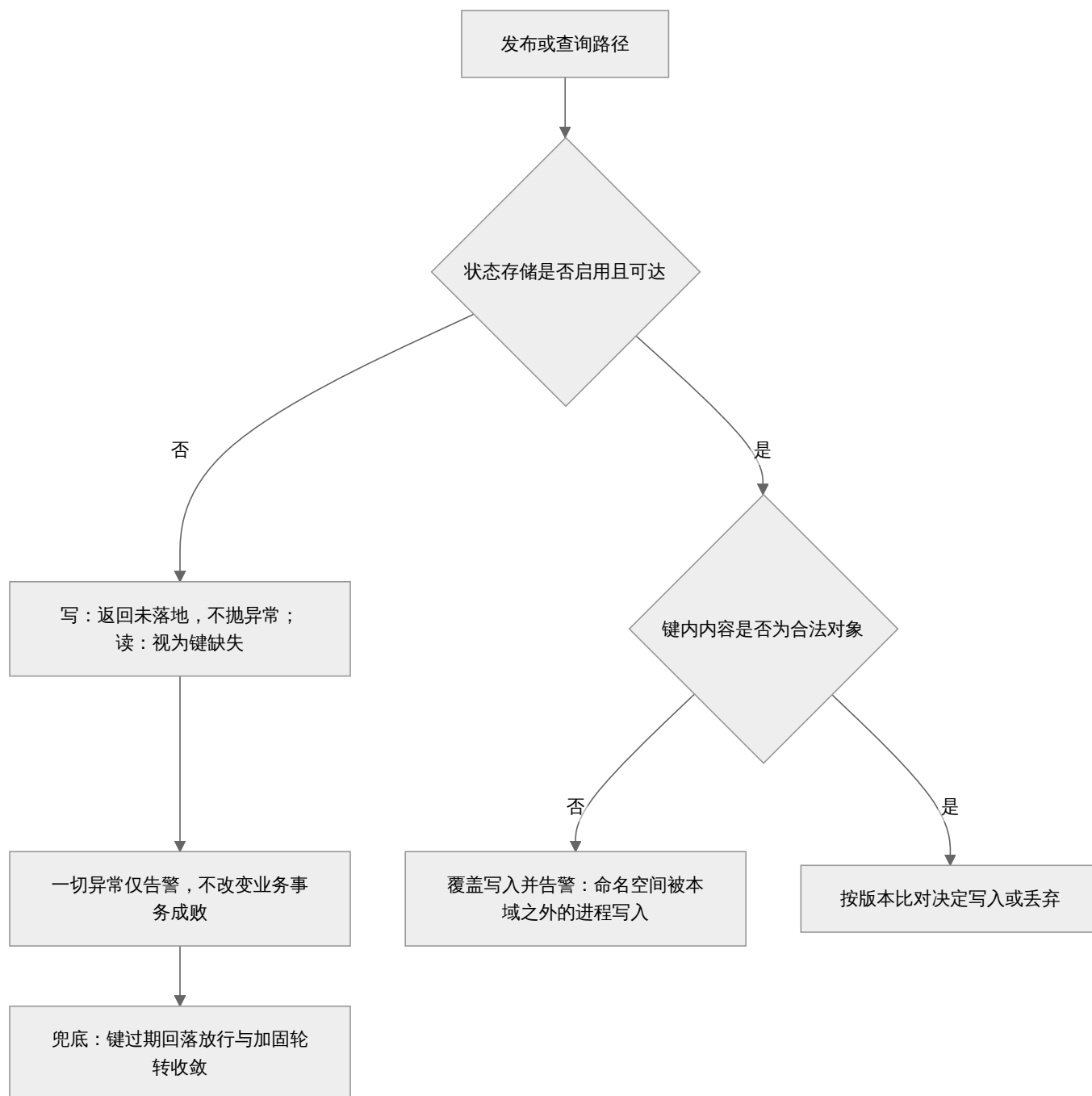


图 12

- 连接配置项为空即全链路幂等空操作降级（发布返回未落地、查询视为缺键），本地环境零外部依赖。
- 降级语义与读侧规则一致：读侧本就把缺键当放行，故状态存储不可用的净效果是「准入退回放行」，而非拒绝全体用户。
- 多副本并发扫描的幂等安全由两件事保证：进程内游标各自推进、重复刷新同值无害；写键按版本去旧。

5. 接口契约

5.1 准入状态查询（`/internal`，计量链令牌）

方法	路径	语义
GET	<code>/internal/customers/{customer_id}/state</code>	派生态唯一出口：返回 <code>state</code> / <code>blocked</code> / <code>block_reason</code> / <code>grace_until</code> / <code>available</code> / <code>cash_balance</code> ，并附经营状态与两个旧字段 <code>allowed</code> 与 <code>reason</code> （封锁位取反，原因取 <code>customer_disabled</code> 或 <code>insufficient_available</code> ）

- 该端点不写键、只做派生与回显；可用额度由本域统一计算，调用方不得自行加总组件。
- 消费方据此判定：封锁位为真即拒（按原因分流），宽限态由自身时钟比对 `grace_until`；载荷与写键同源，不存在第二处判定。

5.2 状态发布所依附的写入口（语义归属见对应域）

方法	路径	语义
POST	<code>/internal/customers/{customer_id}/deduct-batch</code>	批量扣费，提交后异步发布（详见 D04）
POST	<code>/internal/customers/{customer_id}/deduct</code>	单笔扣费兼容入口，同上（详见 D04）
POST	<code>/internal/customers/{customer_id}/credit</code>	赠金入账，解封路径同步写键（详见 D04）
PATCH	<code>/internal/customers/{customer_id}/status</code>	状态标记，同步写键（详见 D04）
PATCH	<code>/api/account/ops/customers/{customer_id}/status</code>	运营面状态标记，同步写键（索引见 D11）
PATCH	<code>/api/account/ops/customers/{customer_id}/credit-limit</code>	信用额度配置，抬高额度可解封，同步写键（详见 D04）
POST	<code>/api/account/ops/customers/{customer_id}/adjustment</code>	运营调整含坏账核销，同步写键（详见 D04）
PUT	<code>/api/account/ops/customers/{customer_id}/grace</code>	宽限设置，同步写键（详见 D04）

- 本表只登记状态发布所依附的写入口，各行语义归属相邻域，此处不重复展开。
- 充值入账、退款冲销、查单自愈入账、转账两侧、关停与注销、注册与建企业同样在提交后发布；本域只承接「发布」这一步。

6. 关键约束与验收标准

6.1 约束

- 状态键单一生产者：命名空间内只有本域写键；写键失败不阻断业务，读侧一律只读。
- 判定唯一实现：五态判定只有一处实现，状态键与状态查询端点共用同一产出；调用方不得自算可用额度或自行推断派生态。

3. 生命周期优先：`closed` / `banned` 直判封锁，`closed` 为终态；账务态只在经营状态为在营时参与。
4. 版本序即提交序：版本由库层触发器在行锁内取全局序列赋值，应用层只读不写；绕过库层赋值即令去旧失效。
5. 写侧去旧三支：新版本小于键内版本丢弃且不发事件；相等允许；键内无版本字段视为 0 允许覆盖（滚动升级兼容）。
6. 采样次序先版本后内容：派生入口不接受调用方传入可用额度，次序由构造保证，新增调用点无法破坏它。
7. 读侧规则封闭：封锁位为真即拒（按原因分流）；封锁位为假且宽限截止已过按额度拒；键缺失或过期一律放行。消费方不感知版本字段。
8. 宽限截止仅 `grace` 态透出：其余四态该字段为空，「窗内放行」只能由读侧时钟判定完成。
9. 存活期双向夹逼：放行态短存活期（默认 600 秒），封锁态长存活期（默认 7200 秒）；后者不低于一个加固周期，上界取周期固定倍数。
10. 解封路径同步写键：充值、运营调整、状态改回在营、宽限、退款冲销、赠金入账、转账两侧须在同一请求内完成写键；失败或超时仅告警。
11. 创建即写键：注册与建企业须在创建点调度发布，杜绝新账号因缺键被放行一次消费。
12. 加固轮转两段且游标不回退到表头：优先人群每周期全量刷新，剩余预算按标识升序推进并回绕；周期 = $\lceil \text{客户数} \div \text{每轮预算} \rceil \times \text{加固间隔}$ 。
13. 标记是派生量：唯一置位与清除点在钱包变更事务末尾；准入与生命周期出口都不读它。
14. 降级为放行且敞口有界：状态存储未启用或不可达按缺键处理（放行），敞口 = 在途未落账费用 + 信用额度，由存活期与加固轮转收敛。

6.2 验收标准

#	验收口径
1	经营状态为封禁或注销时，无论可用额度多高，派生态均为对应封锁态且封锁原因为状态
2	可用额度恰为 0 且宽限关闭时判定为欠费态、原因为额度；最小正数时为在营态
3	宽限窗内为宽限态、封锁位假、截止非空；窗走完后同一客户转为欠费态
4	两笔并发扣费后键内 <code>available</code> 与最后提交那笔一致，不存在旧快照覆盖新值
5	伪造比键内版本更小的发布：键与事件通道均无变化
6	与键内版本相等的重复发布：键被重写（续期）且不报错
7	键内无版本字段的存量键：任何新版本都能覆盖
8	键内被外部写入非合法内容：发布照常覆盖并产生安全告警
9	扣费仅命中赠金、现金差额为 0 时客户版本仍被抬高（语句级生效）
10	入账与赠金撞唯一约束时收敛为幂等，不因版本写入变成序列化错误，不返回 5xx
11	充值解封后同一请求返回时封锁位已为假；同步写键失败后键至多存活一个长存活期即回落放行
12	长存活期低于一个加固周期时被验收拒绝（封锁人群会在加固间隙失效）

#	验收口径
13	跑满一个轮转周期后全库每个客户的键都被刷新过至少一次；到表尾后不出现头段重复刷新而中段被跳过
14	新注册且无赠金的账号：创建后立即读键为欠费态，无被放行一次消费的窗口
15	宽限窗内客户在加固轮转中被按当下时钟重新求值：未到期续宽限、已到期固化为欠费态
16	赠金自然过期（无任何钱包事务）后，一个加固周期内键内 <code>available</code> 与派生态被重新导出
17	状态存储连接配置为空时，扣费与充值接口行为不变（发布为零等空操作），无异常抛出
18	状态查询端点与状态键读出的 <code>state</code> / <code>blocked</code> / <code>available</code> 完全一致（同源判定）
19	消费方仅凭封锁位、宽限截止与键存在性即可完成判定，忽略版本字段不影响结果

6.3 待后续设计细化的开放项

1. 载荷的显式契约版本化：现以「新增字段向后兼容 + 消费方忽略」替代协议版本号；若出现破坏性变更，需要引入键名或字段层面的版本协商。
2. 准入收紧档（可用额度低于单笔均值若干倍即拒）与单笔上限的默认值：现默认关闭，仅在敞口需收紧时启用。
3. 高敏租户收紧开关：默认放行语义写死，收紧为断路的口径与生效范围待定。
4. 每轮预算与加固间隔的容量模型：客户量级增长后两者的相对关系需重新标定，以维持「优先人群间隔不大于一个存活期」。
5. 变更事件的订阅方清单与消费方式（谁订阅、如何幂等去重）由各消费方自定，尚未收敛为统一约定。
6. 账龄分级的事件落库形态与提醒触达的唯一化口径（避免同一账龄段重复提醒）。