火龙果・整理
uml.org.cn

知乎Feed流架构演进

火龙果・整理
uml.org.cn

提 纲

A. Feed 流的需求和特点
B. 老 Feed 流的构架遇到的问题
C. 新构架 Redis module 技术方案
D. Redis module 方案遇到的问题
E. 新的问题与挑战

知乎 Feed 流的需求和特点

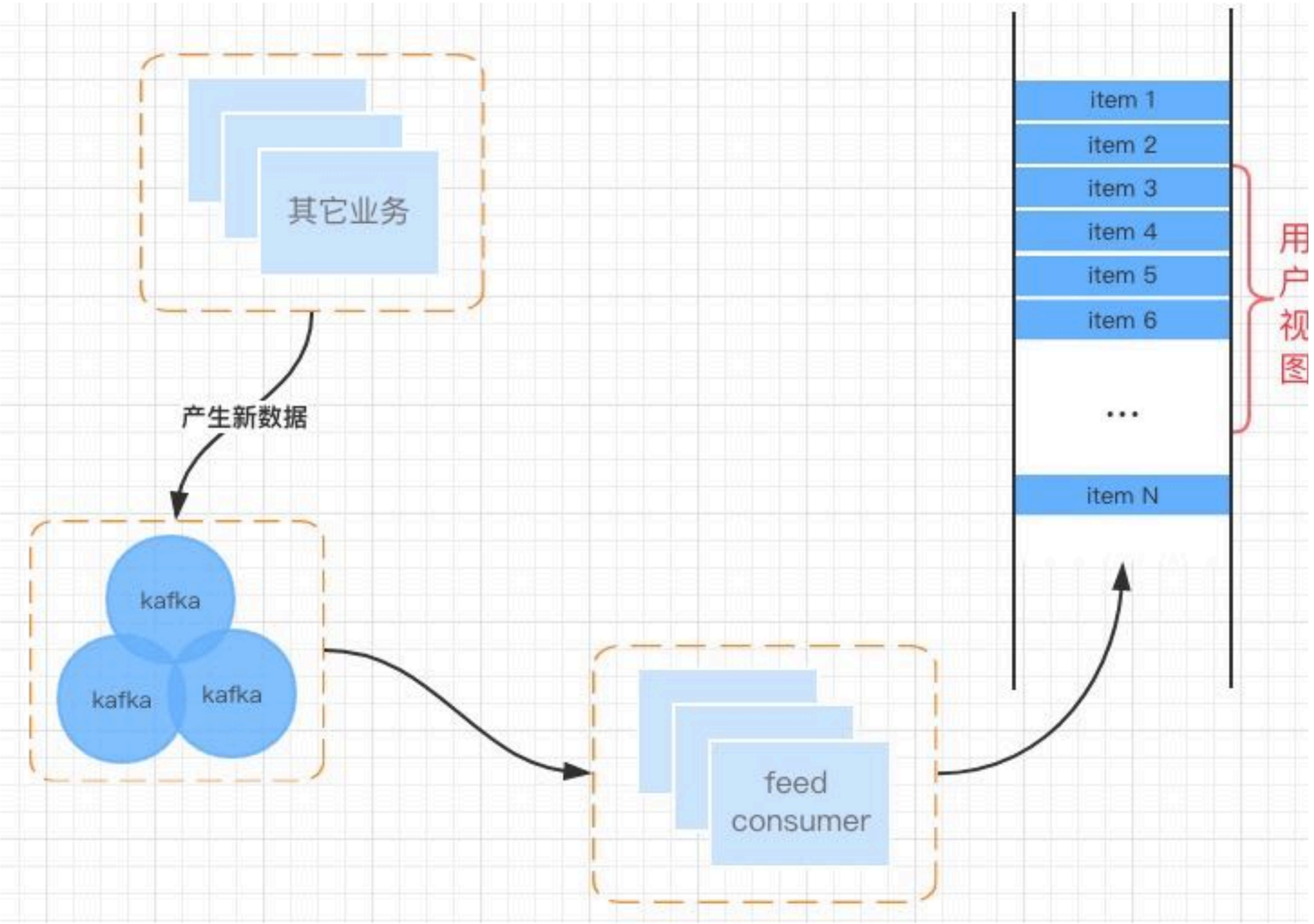
A. DAU 2600万（2017.9）

B. 个性化推荐，每次请求返回内容不同（与搜索不同， cache 难做）

C. 用户动态准实时分发

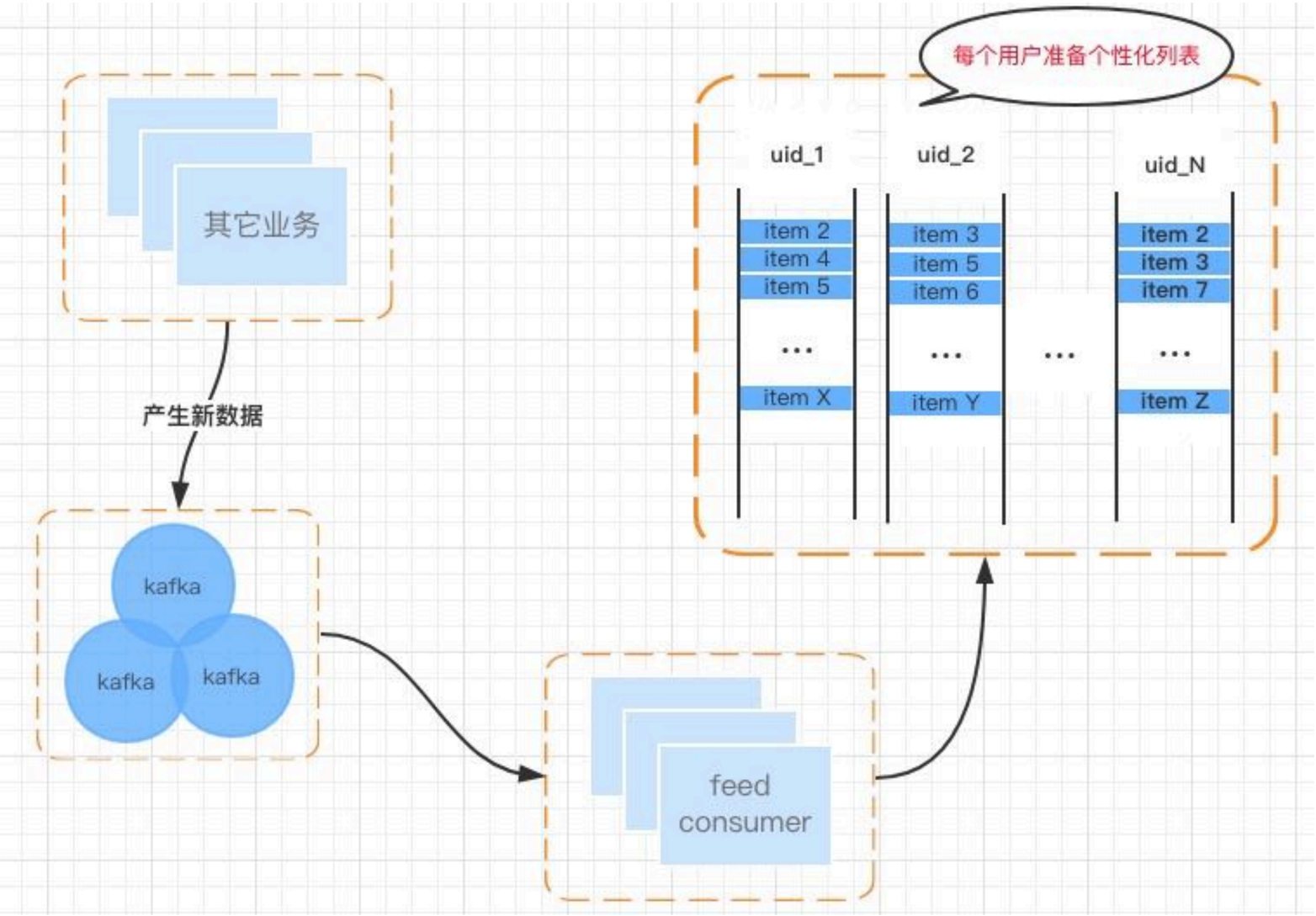
无个性化（热门榜单）

火龙果·整理
uml.org.cn



用户个性化 push

火龙果·整理
uml.org.cn

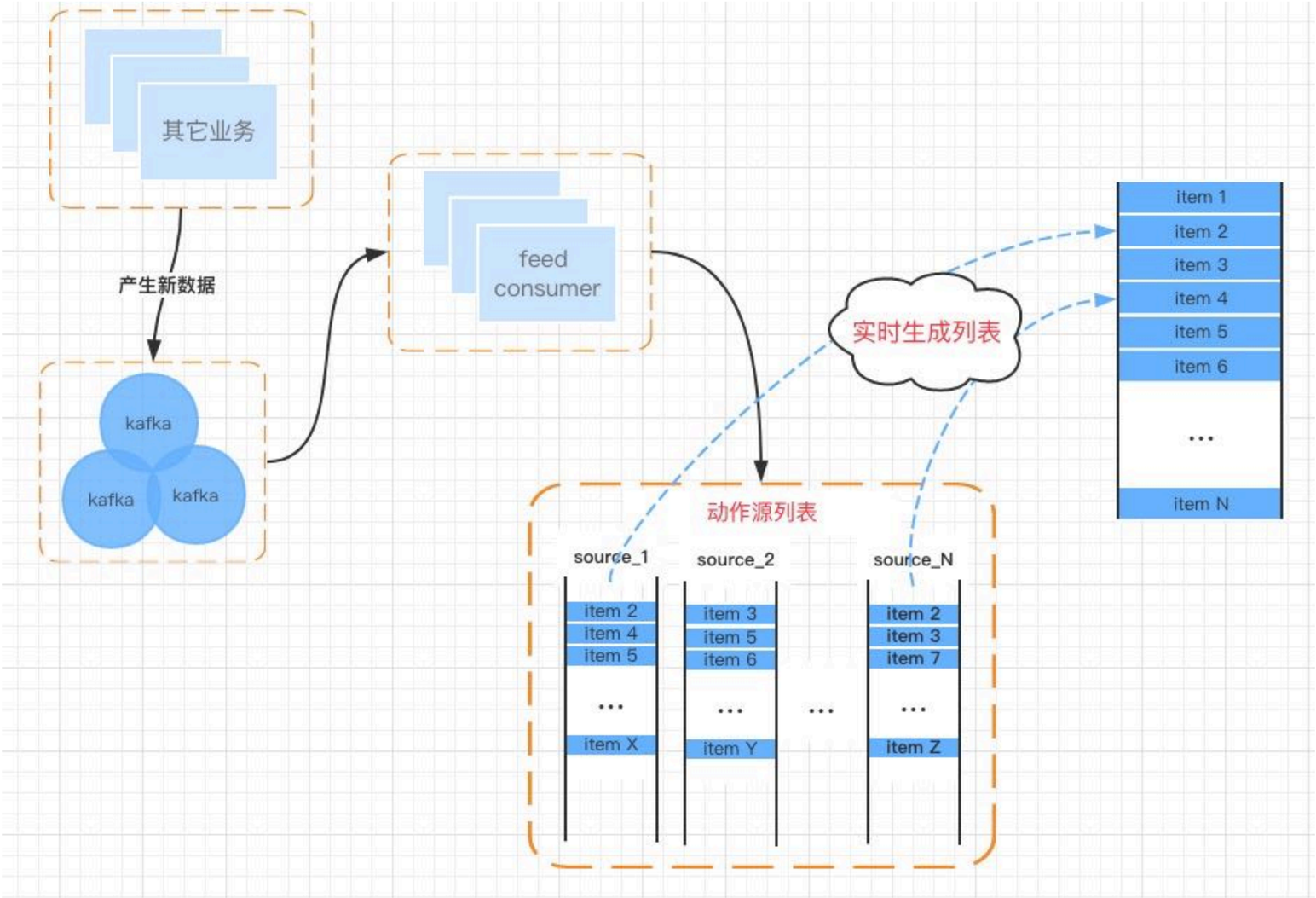


push 存在的问题

A. 资源消耗严重，计算量大，存储量大
B. 智能推荐，排序难以实时调整
C. 过滤比较难做（关注 or 被删除）
D. 动态准实时分发难以达到 （高粉丝用户）

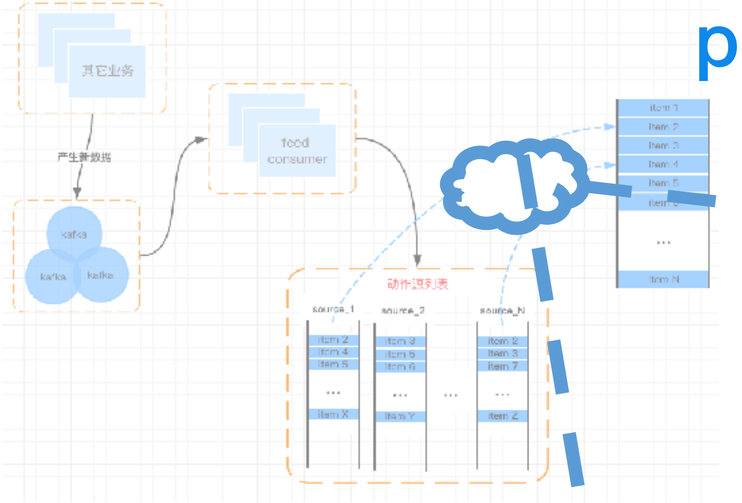
实时 pull

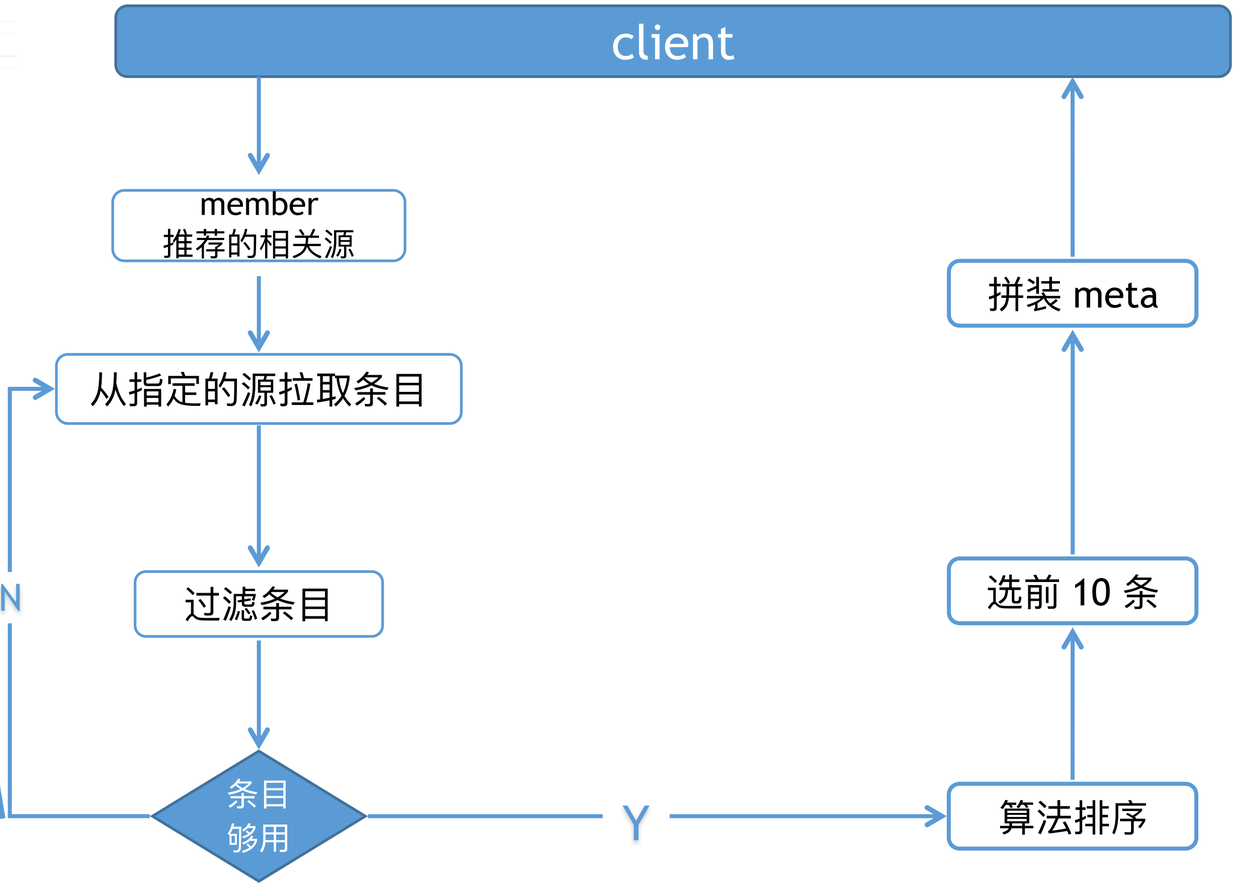
火龙果·整理
uml.org.cn



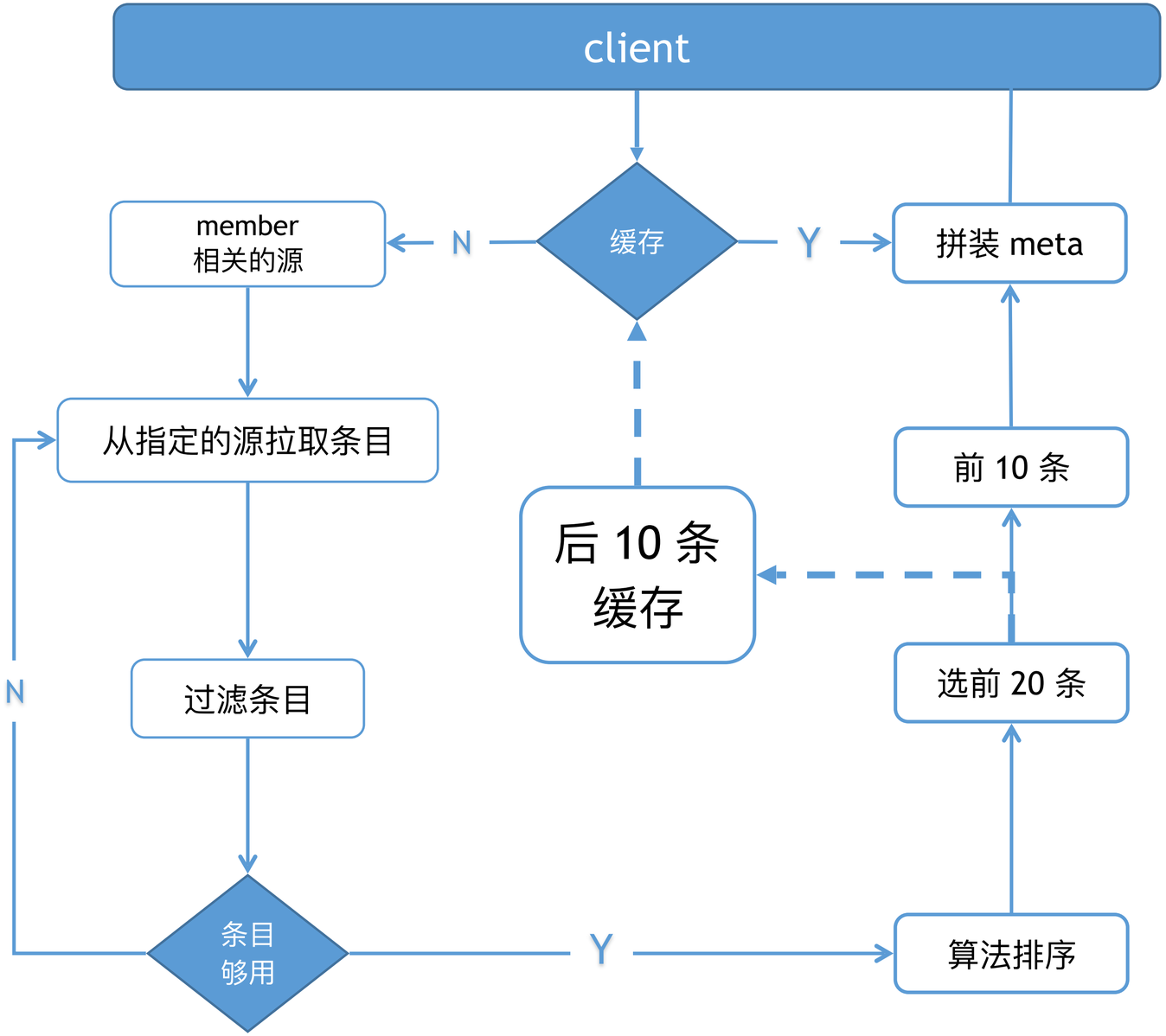
pull 计算流程（慢）

火龙果·整理
uml.org.cn





提前计算，做缓存



火龙果・整理
uml.org.cn

提前计算的问题

- 存在冗余计算，占用资源多
- 冷启动 P95 响应时间长
- 用户行为分发延迟，体验差
- 离线计算策略复杂，难以维护
- 推荐算法难以实时调整

火龙果・整理
uml.org.cn

如何优化，难点儿在哪儿？

- 依赖服务响应慢
  - redis cache + local cache
  - gevent 并发
  - 超时做降级
- Python 计算太慢
  - Cython 模块替换
- 由于条目不够，反复访问底层源的存储 feesource

火龙果・整理
uml.org.cn

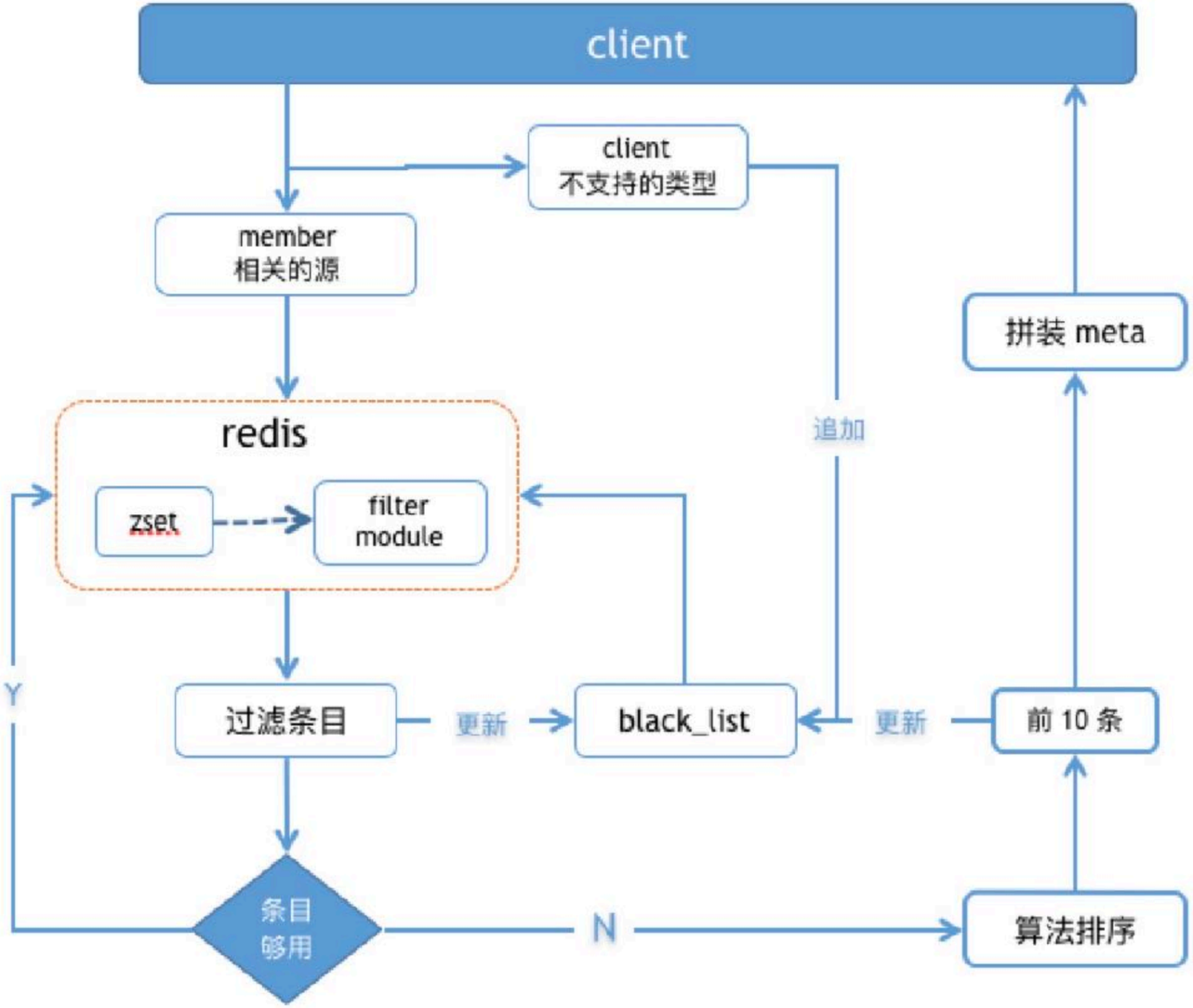
可能的解决方案

- 拉取出更多的条目，防止被过滤掉?
  - 拉取更多的条目也会浪费时间，过滤压力大
- 根据算法拉取出更精准的条目?
  - 算法期望召回池越大越好

计算下推，接近存储

火龙果·整理
uml.org.cn

新 feed 计算逻辑



火龙果・整理
uml.org.cn

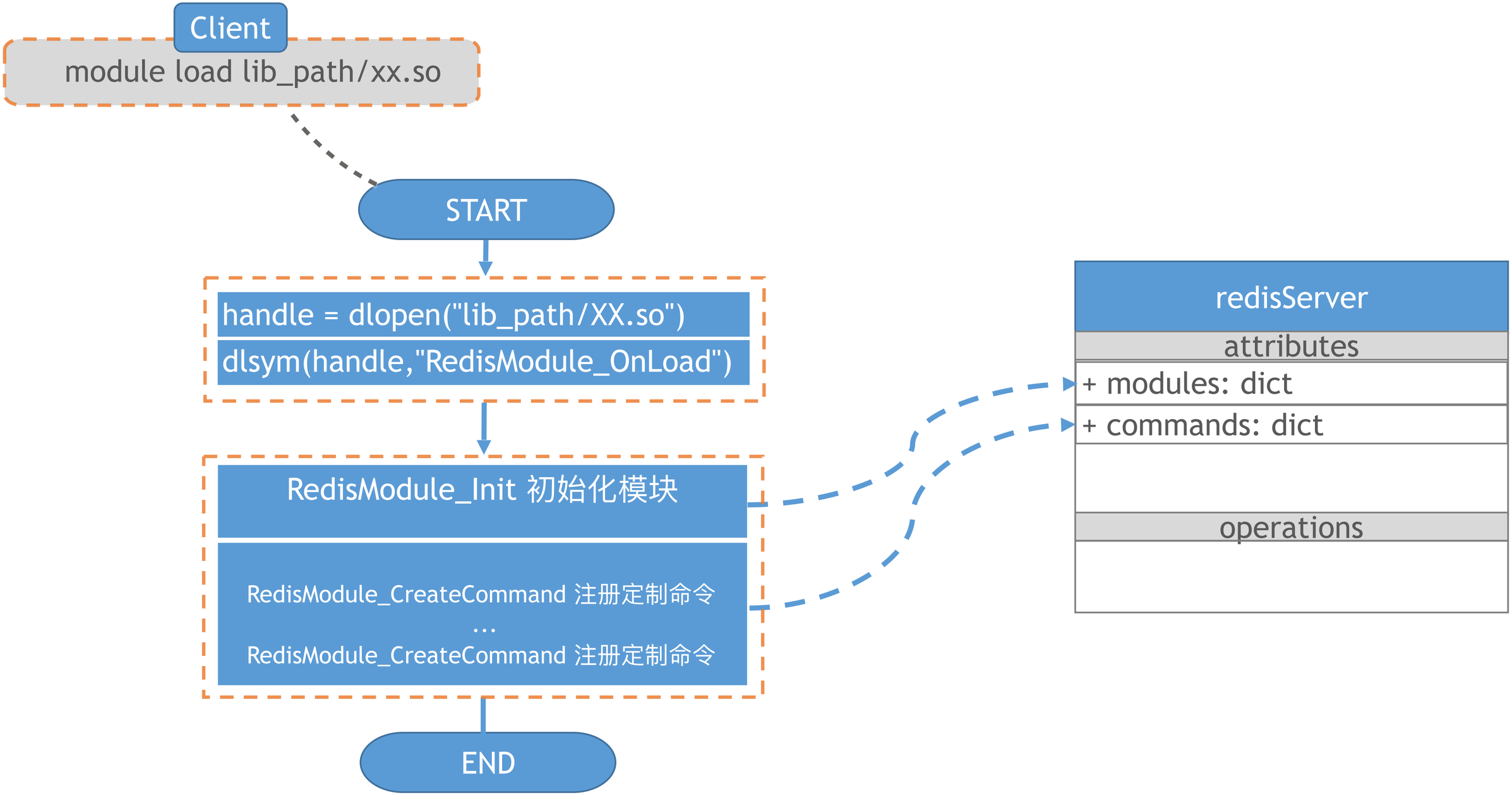
Redis Module

Redis modules make possible to extend Redis functionality using external modules, implementing new Redis commands at a speed and with features similar to what can be done inside the core itself.

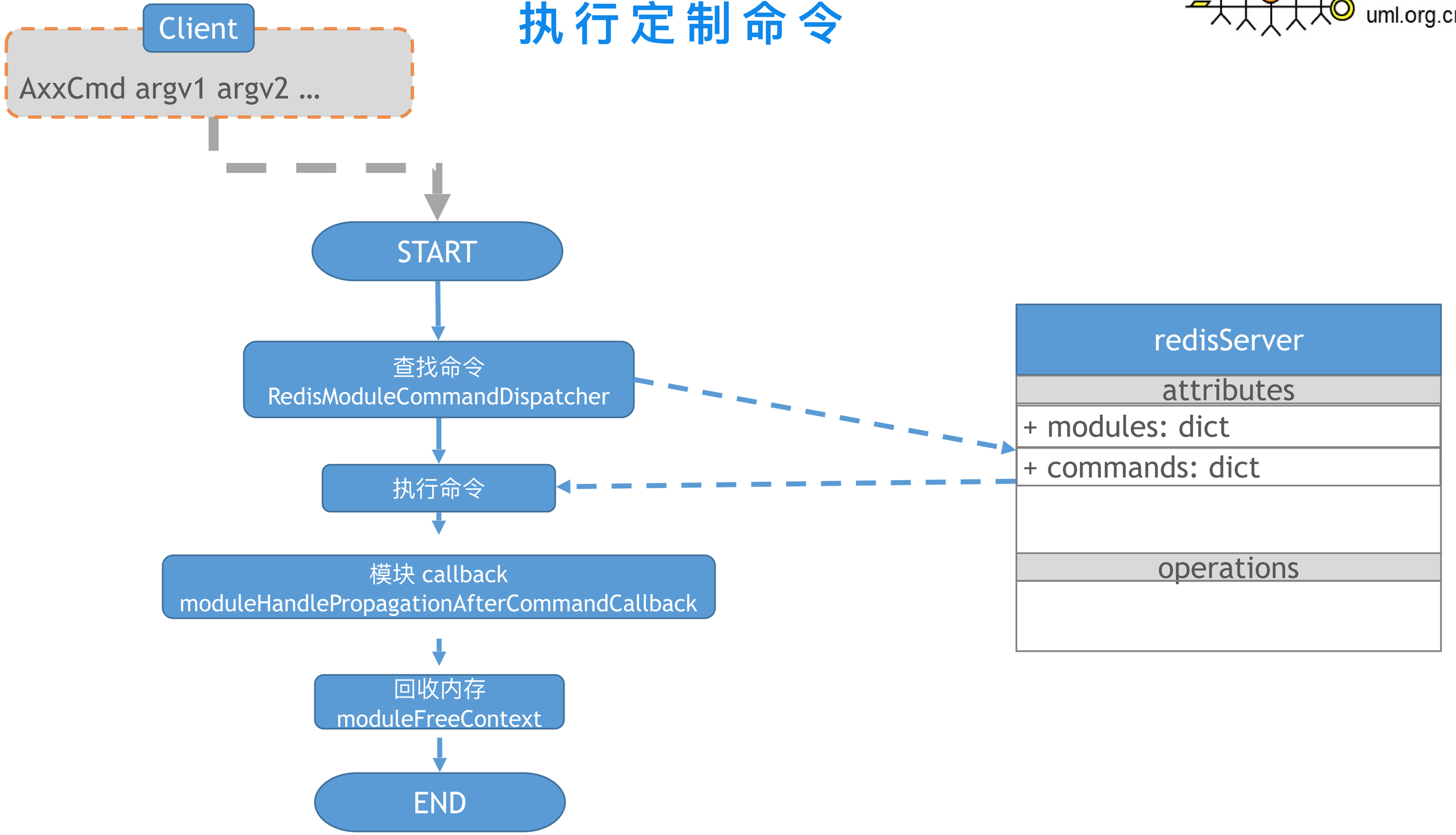
Redis Module

- 加载定制命令（MODULE LOAD module load lib_path/xx.so）
- 执行定制命令
- 卸载定制命令（MODULE UNLOAD mymodule）

加载定制命令

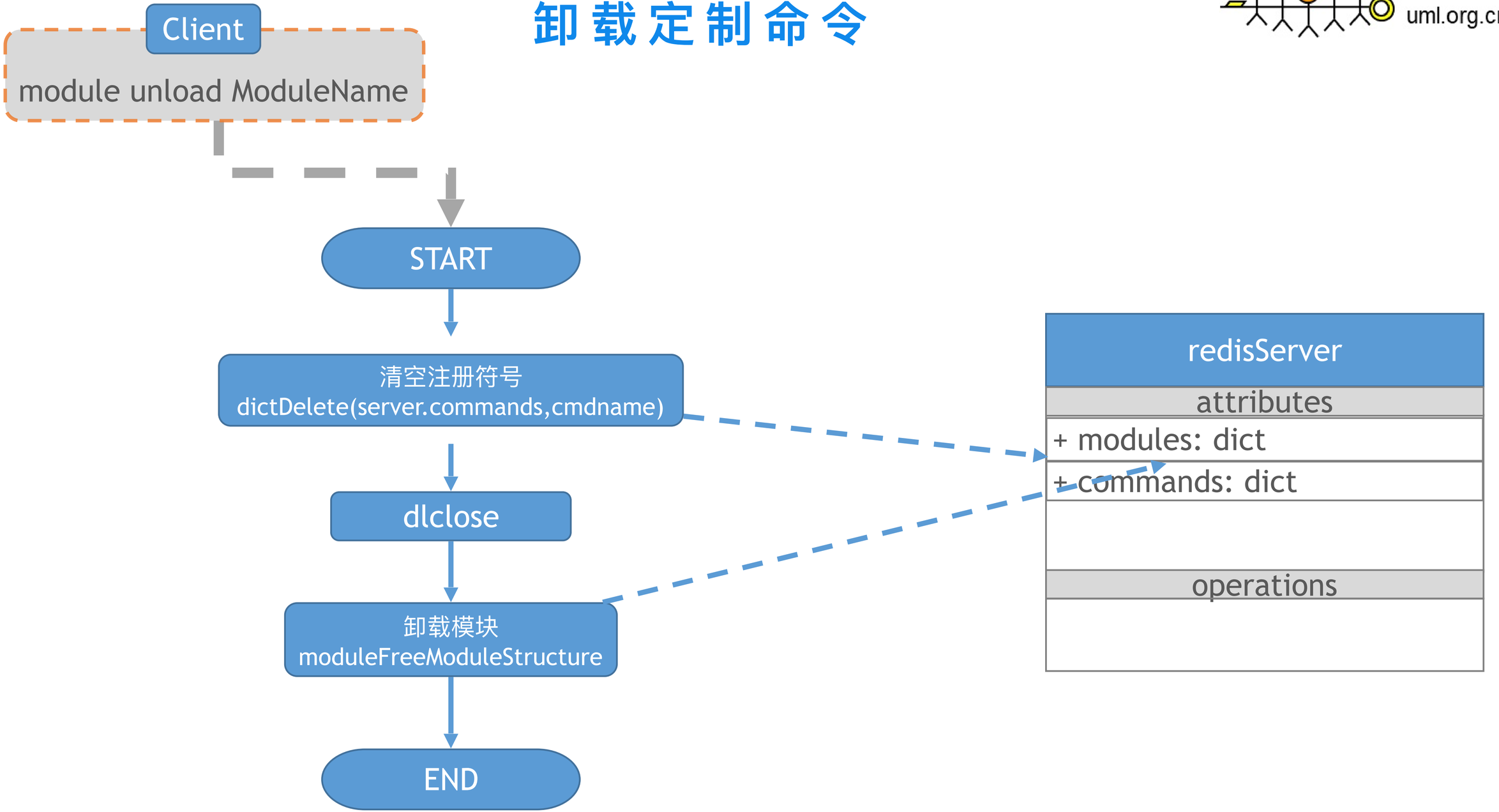

Client
module load lib_path/xx.so
START
handle = dlopen("lib_path/XX.so")
dlsym(handle,"RedisModule_OnLoad")
RedisModule_Init 初始化模块
RedisModule_CreateCommand 注册定制命令
...
RedisModule_CreateCommand 注册定制命令
END
redisServer
attributes
+ modules: dict
+ commands: dict
operations

执行定制命令



火龙果・整理
uml.org.cn

卸载定制命令



火龙果·整理
uml.org.cn

火龙果・整理
uml.org.cn

定制的命令

- open_core, close_core
  - Redis module 更新时打开 core dump，如果 crash 方便分析
- ts_query
  - Redis 内部的过滤，归并流程

ts_query 接口设计

- Request
  - [source_type, source_id]: 需要拉取 source
  - black_items: 需要过滤的黑名单条目
  - merge_strategy: 合并策略
  - required_number: 需要返回的条目数量
- Response
  - [item_type, item_id, item_action, item_score]: 返回需要数目的 feed 条目

带来的收益

- 响应时间 P95 降低 300ms
- 去掉了离线计算，节省 60% 的计算资源
- 内容候选池更大，为算法提供了更大的空间
- 用户动态实时分发，算法实时调整

火龙果・整理
uml.org.cn

遇到的问题

- Module 更新不生效，继续调用老 Module
- Redis 单线程 CPU 瓶颈，高可用
- Redis 的内存浪费

Module 更新不生效

- 加载新的 so 后，发现调用的还是老逻辑
  - GDB 发现存在符号被标记 DF_1_NODELETE
  - dlcose 仅仅声明 so 不在被系统使用，so 内存占用依旧存在
  - gcc 编译使用 STB_GNU_UNIQUE ，防止符号被标记为 DF_1_NODELETE
  - 按照依赖路径加载 so，不直接加载定制的 so

火龙果・整理
uml.org.cn

Redis 单线程 CPU 瓶颈，高可用

- 采用 Redis Sentinel 部署集群，保证高可用
- 一致性哈希 Redis shard，每个 shard 采用 master slave 的方式部署

火龙果・整理
uml.org.cn

Redis 内存浪费

- 采用 protobuf 和 Redis ziplist 数据压缩，减少 shard
- 有多个 slave，还是浪费内存，没有根本解决问题
  - Redis 仅仅作为任务队列，任务分派给其他进程处理?
  - 同一台机器伴随 Redis 部署计算节点?

Feed 架构的历程

- Feed 都一样
- Feed 个性化，推模型
- Feed 个性化，拉模型 + 离线计算
- Feed 个性化，拉模型，采用 Redis Module，计算接近存储

火龙果・整理
uml.org.cn

THANKS