PIKO RESEARCH 技术报告 · PIKO-RESEARCH-2026-01

PESA：个人生态超级智能体

面向跨设备个人生态的系统架构、形式化模型与注意力经济学。

作者：Ruiming Zhao ORCID
Prysai Lab

首发：2026-11-15 版本：1.2 · 修订 2026-11-24 阅读：约 25 分钟

ZENODO · CC BY 4.0：DOI：10.5281/zenodo.24429625

摘要

单任务语言模型智能体已经能够调用外部工具、编写代码并操作图形界面，独立完成越来越复杂的单点任务。然而在日常个人使用中，底座模型变强并没有自然换来人的轻松。由于现有个人 AI 产品仍被困在孤立的会话框或单台设备的屏幕里，用户依然不得不亲自留意新消息、在邮箱、日历、浏览器与多个聊天窗口之间反复搬运上下文，并在智能体接管桌面操作时全程盯屏看护。

本文提出并形式化定义 **PESA（Personal Ecosystem Super Agent，个人生态超级智能体）**：一种服务于单一个人主权、以个人全部设备生态（手机、平板、电脑与手表）为工作环境、以“事项（Thing）”而非“对话（Session）”为基本工作单位、在每台设备上按平台原生习惯协同执行、且在任何对外写操作前默认等待本人显式确认的常驻超级智能体。在继承常驻双层 Super-agent 架构 [1] 的基础上，本文围绕设备、事件流、线索、事项、后台执行器与用户策略建立形式化模型，推导出四分量人类注意力成本模型、面向高噪个人事件流的 Horvitz 混合主动投机阈值、闭合八态生命周期与跨端乐观并发提交规则，并提出七条可验证的运行时系统不变量。在此基础上，我们证明了对外副作用安全性与跨端幂等性（命题 1）、单桌面焦点租约的无死锁与有限等待特性（命题 2），以及主权约束下人类注意力的不可压缩下界（命题 3）。

工程层面，本文给出完整的端到端参考架构——涵盖单一常驻前台协调器、任务级最小权限隔离的后台执行器、跨设备三层**事项档案库（Thing Archive）**、端云两级降噪过滤、单桌面互斥租约（`ComputerHandsLease`）、独立自愈看门狗，以及三项确定性优先调度算法（`route`、`choose_device`、`choose_surface`）——并构建了抵御间接提示注入与隐蔽 URL 参数外泄的纵深防御模型。通过在四类高频跨设备场景上进行可逐跳核对的解析注意力核算，我们揭示了仅停留在单端主动（L3）仍会残留平均每事项 0.50 次跨设备手动搬运，而完整生态架构（L4 PESA）可将人工介入从 L0 的 3.75 步降至 1.25 步（降低 66.7%），并将手动上下文搬运彻底归零（降低 100%）。最后，本文提出合成回放基准 **PESA-Bench**，报告参考实现 **Piko** 生产运行时中六项核心机制的确定性单元测试验证，并系统讨论了方法论局限性、有效性威胁与开放研究问题。

关键词：个人生态超级智能体（PESA）；个人 AI 智能体；以人为中心的 AI；多设备协同系统；注意力经济学；混合主动交互；投机执行；乐观并发控制；桌面租约仲裁；间接提示注入

1. 引言

今天的大模型已经能读懂长文档、写出可运行的代码，也能模拟鼠标键盘操作软件。但有一个令人意外的现象：模型越聪明，个人用户往往反而越忙。为了办妥一件稍微复杂的日常事务——比如协调一次客户改期、比价后填写公司采购单，或是追踪一场持续三周的项目谈判——用户常常不得不在手机和电脑上同时开四五个互不相通的聊天窗口。每个窗口都是一个孤立的沙盒，不知道你在另一个标签页里聊过什么，更不知道你十分钟前在另一台设备上做过什么。于是，人成了系统之间唯一的粘合剂：亲自发现事务、把散落的上下文复制拼成提示词、在工具运行期间守着屏幕盯防、做完后再把结果搬回邮件或业务系统。

DeepSeek Harness 团队在 Super-agent 技术报告中指出，受制于工作记忆和任务切换开销，一个人同一时段通常只能有效跟进 3 到 5 个并行会话 [1]。在以“单次对话”为中心的架构下，**底座模型决定了单轮回答的质量上限，但人的注意力决定了每天能推进多少件事的吞吐上限**。人机交互领域的经典实证早已说明了这种碎片化打断的隐性代价：Mark、Gudith 与 Klocke 在 CHI 2008 的实验室研究中发现，当知识工作者被频繁打断并被迫在多任务间切换时，人们会本能地加快操作节奏来弥补损失的时间，结果总耗时看起来虽未明显拉长，却付出了显著上升的认知负荷、时间压力、挫败感与错误率 [2]。在邮件、日历、浏览器与多个 AI 窗口之间来回切换，本质上就是一种高强度的持续自我打断。

> 直观理解。 更强的大模型解决了“你把背景交代清楚后，它怎么把这一步干好”的问题；但它没有解决“谁替你留意日常动静、谁把上下文跨设备带过去、怎么防止后台任务抢你的鼠标，以及怎么在真正发出去之前稳稳停住让你拍板”的问题。PESA 正是为补齐这一缺失的系统层而设计的架构。

1.1 当前个人智能体落地的四重结构性瓶颈

2024 至 2026 年间，个人 AI 工具迅速从纯文本问答演进到工具调用、桌面与浏览器操控（Computer-Use Agents）以及独立 AI 硬件。然而进入高频日常使用后，现有路线反复撞上四面结构性高墙：

1. **提示词冷启动与上下文搬运税。** 真实生活中的事务几乎从不始于一个空白输入框，而是始于一封客户改期邮件、一条航班延误提醒、一张手机拍下的发票，或群聊里的一句日程变动。只要用户必须手动收集这些碎片、组织成提示词再把产出搬回原应用，协调本身的麻烦就常常超过亲手办完这件事的成本，导致日常留存难以维系。
2. **单桌面焦点争抢与“盯屏监工”疲劳。** 桌面操控智能体能点击按钮、填写表单，但一台个人电脑只有一套物理鼠标键盘和一个前台活动窗口。智能体操控桌面时，用户不仅无法继续用电脑工作，还会因担心误点、误删或提前提交而不得不全程盯着屏幕看护。同步独占屏幕的交互方式，把人的注意力与执行时长死死绑定在了一起。
3. **独立 AI 硬件孤岛与既有设备生态的割裂。** 试图绕开现有操作系统的独立挂件或掌上硬件屡屡遇冷，原因很朴素：人们已经拥有一套分工清晰、全天相伴的设备群——口袋里的手机、手腕上的手表、用于阅读与对比的平板、桌上的电脑。用户缺的不是一块没有历史数据和生态应用的新屏幕，而是一套能让同一个持续在线的智能体进驻既有设备群的系统架构。
4. **主动性与信任感的两难。** 纯被动助手在合上电脑、锁上手机的瞬间便停止运转；可一旦盲目开启主动推送，每天涌入的数百条促销邮件、群聊闲聊与应用通知就会化作令人烦躁的弹窗噪音——更危险的是，如果系统在未经本人核对的情况下以用户真实身份对外发信、提交表单或发起支付，任何一次失误都会造成不可挽回的信任崩塌。

1.2 研究问题与主要贡献

本文是一篇系统架构与愿景论文，围绕三个核心研究问题展开：

- **RQ1（抽象模型与注意力）：** 应当用怎样的系统抽象与能力分级，将跨设备个人生态智能体与会话型聊天工具、单端自动化工具区分开来？每一级架构演进究竟消除了哪一部分人类注意力成本？
- **RQ2（系统协同与并发）：** 单一常驻智能体如何把持续在线的云端推理与间歇休眠的本地设备协同起来，在单台电脑上防止多个后台任务争抢桌面焦点，并在多块屏幕间避免重复触发对外操作？
- **RQ3（信任边界与降噪纪律）：** 主动式个人智能体如何在高噪日常事件流中既不空耗云端算力、又不频繁打扰用户，同时抵御来自外部网页与邮件的间接提示注入与隐蔽数据外泄？

针对上述问题，本文做出五项主要贡献：

1. **概念框架与 L0 – L4 能力分级（第 3 节）。** 正式定义 PESA（Personal Ecosystem Super Agent），基于以人为中心的 AI 双维原则 [3] 阐明三条设计命题与六项系统特征，建立与模型参数量解耦、按系统替人分担的协调范围划分的 L0 – L4 分级标准。
2. **形式化模型、混合主动决策边界与理论保证（第 4 节）。** 给出将设备能力正交拆分为执行能力与送达能力 $\kappa(d) = \langle \kappa_{\mathsf{exec}}(d), \kappa_{\mathsf{surf}}(d) \rangle$ 的形式化定义；构建四分量注意力成本方程与高噪个人环境下的 Horvitz 投机阈值；设计闭合八态生命周期与基于乐观并发控制 [4] 的版本化原子提交规则；证明了对外副作用安全性、桌面租约无死锁性与主权注意力下界三条核心命题。
3. **跨设备参考架构与确定性优先调度算法（第 5 – 6 节）。** 设计单一常驻前台、任务级隔离后台执行器、跨设备三层事项档案库、断线检查点续接、有界时间互斥租约（`ComputerHandsLease`）[5] 及独立自愈看门狗，并给出事件路由（`route`）、执行设备选择（`choose_device`）与情境送达面选择（`choose_surface`）三项低开销调度算法及复杂度分析。
4. **生态原生交互、容错场景与纵深防御体系（第 7 – 9 节）。** 阐述四类设备形态的人体工学分工，剖析四类高频跨设备场景的正常闭环与异常恢复路径，说明八态如何折叠为极简三态视图，并建立抵御间接提示注入与隐蔽 URL 外泄的三道架构防线 [6]。

5. **评估指标、L0 – L4 解析核算与参考实现验证（第 10 – 12 节）。** 提出八项正交评估指标与合成回放基准 PESA-Bench；在四类场景上逐跳核算 L0 至 L4 的人工介入与跨端搬运（量化说明为何 L3 单端主动仍会残留跨端搬运）；报告参考系统 Piko 六项生产运行时机制的确定性单元测试结果，并坦诚剖析本文的方法论局限与有效性威胁。

2. 背景与相关工作

PESA 汇聚了智能体系统、操作系统、分布式系统与人机交互六条脉络的研究成果。梳理每一条脉络解决了什么、又在个人日常生活中留下了什么空白，有助于看清 PESA 的系统定位。

2.1 从对话助手到单任务工具与桌面智能体

早期对话助手采用受限于单一窗口的无状态轮询交互。ReAct [7] 通过交替生成推理轨迹与环境动作，使语言模型能依据外部观测修正计划；Toolformer [8] 证明模型可通过自监督学习掌握外部 API 调用时机。综述研究系统梳理了这一范式如何演进为编码智能体、浏览器智能体以及 MCP 等标准化工具连接生态 [9]。

但进入真实个人环境后，独立单任务智能体仍面临显著可靠性鸿沟。在包含 812 个真实网站长程任务的 WebArena 基准中 [10]，人类完成率为 78.24%，而当时 GPT-4 智能体仅为 14.41%；在涵盖 Ubuntu、Windows 与 macOS 共 369 个真实桌面任务的 OSWorld 基准中 [11]，人类完成率为 72.36%，而发布时最优多模态智能体仅为 12.24%。即便后续模型在视觉定位与长链规划上持续提升，日常个人任务仍频繁遇到三道并非靠单纯扩大模型规模就能消除的高墙：跨设备登录态过期与二次验证、关键上下文散落在另一台设备或另一款应用中，以及不可逆的末端外部动作（发邮件、提单、付款）绝不容许系统静默犯错。

2.2 多智能体协作、分层记忆与常驻 Super-agent

AutoGen [12] 等多智能体框架允许专职角色通过可编程对话分工协作，但协同生命周期大多仍绑定于单次会话。Generative Agents [13] 为虚拟社区角色构建了按近期性、重要性与相关性检索的连续记忆流及反思树，但沙盒环境不存在真实资金责任与外部攻击者。MemGPT [14] 借鉴操作系统虚拟内存思想，将上下文窗口视为“主存”，通过显式调用在主存与外部归档存储之间换入换出历史记录。PESA 的事项档案库直接继承了这一分层换页洞见，但将索引与召回的基本单位从松散的聊天片段升级为带因果关联链的跨设备事项（`#T-id`）。

DeepSeek Harness 团队提出的 Super-agent 架构 [1] 迈出了打破单次会话束缚的关键一步：引入常驻路由层（Router）、后台并发执行层（Worker）、事件驱动触发、存算分离与外部看门狗，并催生了 `deepseek-super-agent` [15]、`opensuperagent` [16] 等社区开源实现。然而 Super-agent 的原生目标场景是**软件研发团队**——依托企业群聊、代码托管平台与 24 小时常开的云端开发机。将其迁移至**单一个人的日常生活生态**时，三类全新系统约束立刻凸显：① **个人噪音放大效应**：日常手机通知、群聊与营销邮件中超过 80% 无需任何动作；② **设备间歇休眠与单桌面焦点冲突**：个人笔记本随时会合盖放进背包，且一台电脑只有一套物理键鼠；③ **个人主权与法律责任**：发出的每一封信、提交的每一张表都以本人实名承担后果，绝不允许越权自治。

2.3 分布式系统原语与平静人机交互基础

PESA 的核心协调机制并非拍脑袋的经验规则，而是直接扎根于分布式系统与 HCI 领域的经典理论：

- **有界时间租约（Leases）。** Gray 与 Cheriton 在 SOSP 1989 提出“租约”机制 [5]：通过授予带超时自动回收的排他控制权，确保持有者宕机或失联时系统能无需人工干预地自动恢复活性。PESA 通过 `ComputerHandsLease` 将这一经典原语精准用于个人电脑物理桌面焦点的互斥仲裁。
- **乐观并发控制（OCC）。** Kung 与 Robinson 在 ACM TODS 1981 指出 [4]，当写冲突远少于读操作时，在提交阶段校验版本号快照可避免沉重的分布式锁开销。PESA 据此实现跨手机、手表、平板与电脑的版本化原子确认，彻底杜绝快速连点造成的重复发信或重复扣款。
- **平静技术、情境感知与以人为中心的 AI。** Weiser [17] 及其与 Brown 合著的平静技术理论 [18] 主张，日常计算任务应安居于注意力的边缘，仅在必要时平滑进入中心；Dey [19] 对“情境（Context）”给出了经典形式化界定，奠定了感知用户当前状态并匹配送达面的理论基础；Horvitz [20] 用贝叶斯期望效用刻画了混合主动交互，在主动协助收益与误

打扰代价之间做定量权衡；Shneiderman [3] 进一步论证，“高程度机器自动化”与“高程度人类控制权”是两条彼此正交的坐标轴，而非非此即彼的一维滑块——这正是 PESA“后台充分投机准备、末端由本人拍板确认”的理论基石。

- **投机执行与系统原生框架。** 现代超标量处理器在分支条件未决前先行在重排序缓冲区中投机执行，仅在命中后原子提交架构状态 [21]。与此同时，Siri、App Intents、快捷指令、Handoff 与 ActivityKit 锁屏实时活动 [22][23] 等系统框架提供了得天独厚的跨端原生交互面，但仍以短链路同步指令为主，缺少跨周存活的云端异步事项状态机。PESA 并不与操作系统原生入口竞争，而是将其整合为感知与送达能力层。

表 1　PESA 与主流智能体架构的多维度定位对比

对比维度	对话式助手	单任务工具 / 操控智能体	团队级 Super-agent	PESA（个人生态超级智能体）
基本工作单位	单次独立对话（Session）	单次指令任务（Task）	研发事件线索（Thread）	**具体个人事项（Thing，`#T-id`）**
运行与感知环境	单一聊天窗口	单台电脑或浏览器沙盒	团队群聊、代码库与云端主机	**个人全部设备（手机/平板/电脑/手表）与应用**
触发与主动性	纯被动等待输入	人工发起、同步执行	团队成员与研发流水线事件	**用户意图 + 经端云两级过滤的日常事件流**
并发与响应性	单窗口阻塞	执行时常独占屏幕焦点	常驻路由 + 后台并发执行器	**轻快常驻前台 + 隔离并发后台执行器**
离线韧性与焦点互斥	关标签页即停	合盖中断、多任务抢鼠标	云端常开、无单桌面焦点冲突	**云端续跑 + 静默唤醒 + `ComputerHandsLease` 单租约互斥**
记忆与跨端召回	困在单段聊天记录	局限于当次运行上下文	单平台内按编号检索	**三层记忆 + 跨设备事项档案库（`#T-id` 逐跳召回）**
送达与交互面	统一对话列表	终端日志或悬浮操控框	群聊消息与代码评审（PR）	**按 κ_{surf} 匹配情境：灵动岛、Watch、iPad 分栏、卡片**
对外安全与凭据隔离	人工手动复制外发	凭据常暴露于上下文	依团队 CI/CD 权限自动执行	**凭据保险库封存 + 最小工牌隔离 + 默认版本化确认关卡**

来自 Super-agent 的系统实测启示与适用边界。 DeepSeek Harness 报告中的三组系统实测直接指导了 PESA 的流水线设计 [1]：① 前后台双层并发将任务完成尾延迟（p95）从单会话中性基线的 29.5 分钟压低至 15.7 分钟（缩短 46.8%，中位数 12.4 分钟）；② 开启线索订阅直通避免了对同一线索重复做全量路由，防止读取缓存 token 从 2323 万暴涨至 3177 万（+36.8%）并减少约 30% 的路由决策；③ 入站速率从每分钟 1 条升至 40 条时，积压合批使平均批大小从 1.0 自适应增至 2.8，单次请求耗时仅从 2.5 秒微增至 3.9 秒，均摊每条决策耗时从 6.4 秒降至 1.6 秒（缩短 75%）。需要审慎指出，上述数据测得于研发流水线负载，个人生活环境的噪音密度远高于代码仓库，因此 PESA 在保留订阅直通与积压合批的同时，必须在云端模型之前前置零 token 的端侧规则预筛与每日投机配额门控（见第 4.2 节）。

3. 概念框架与能力分级

3.1 形式化定义

> **定义 1（PESA）。** **个人生态超级智能体（Personal Ecosystem Super Agent, PESA）** 是一种服务于单一个人主权、以该个体全部设备与应用生态为运行环境的常驻超级智能体。它在所有设备上保持单一连续身份与统一上下文；以具体事项（Thing）而非对话（Session）为基本工作单位，在隔离的后台沙盒中并发推进多项事务；通过端云两级过滤从日常事件流中识别高价值事务并提前完成只读调查与草稿准备；在每台设备上遵循该平台原生的交互习惯进行感知、执行与送达；除用户在策略白名单中显式预先授权的特定低风险动作外，任何改变外部世界状态的写操作默认必须先获得本人携带版本号的显式确认。

> **直观理解。** 换句大白话说，PESA 不再是一个停在浏览器标签页里等你敲字的聊天框，而更像一位安静、守规矩、贯通你所有设备的专属助理：只读取你亲手打开的信息来源，在后台默默查清资料、拟好草稿，拿不准价值时绝不出声，只在真正需要你拍板的那一刻，在你此刻最方便看的那块屏幕上轻点一下。

名称中的四个词各自划定了清晰的工程边界：

- **Personal（单一个人主权边界）：** 记忆、日历、通讯录与账号凭据严格归属于这一个人，不是在多方利益间做折中的多租户团队机器人。
- **Ecosystem（异构多端运行环境）：** 运行载体是用户已有的互补硬件群（如手机、平板、电脑、手表）及其原生系统服务，而非单一网页或孤立单机。
- **Super（常驻双层与超会话能力）：** 沿袭并扩展 Super-agent [1] 的系统内涵，具备跨会话连续性、前后台全双工并发、事件驱动主动性与断线自愈持久性。
- **Agent（有界闭环行动体）：** 不仅能生成文本，还能在严格权限边界内调度云端工具与本地端侧执行器，把事务推进到“万事俱备、只待确认”的交付态。

3.2 三条核心设计命题

PESA 的整体设计建立在三条相互支撑的命题之上，呼应了 Shneiderman 关于“高自动化与高人类控制并行不悖”的二维人本 AI 框架 [3]：

1. **命题一：工作环境是跨设备生态，而不是单一聊天窗口。** 个人的数字上下文天然按设备物理形态分片存放：随身消息在手机，深度文档与桌面软件在电脑，长文对比在平板，行进中的快速反馈在手腕。局限于单一窗口的智能体在架构上是“半盲”的，必然把跨端整合的成本转嫁给人。
2. **命题二：面向用户的基本单位是“事项”，而不是“对话”。** “新建对话”是早期无状态模型接口遗留的交互产物，并非人类组织事务的心智模型。人在日常中关心的是“客户改期回复发出去了吗”“采购比价表填好了吗”“两周前合同里违约金条款的原话是什么”。提示词组装、会话轮次与后台执行器调度都应被封装在系统内部。
3. **命题三：高程度后台自动化与严格的人类最终拍板权相辅相成，而非此消彼长。** 检索、跨端同步、表单预填与异常重试应当彻底交给机器在幕后消化；人的注意力只需保留在两个最具价值的节点——**提出意图**，以及**在对外动作前做出最终确认**。

3.3 六项核心系统特征

在继承 Super-agent 四项基础系统特征（连续、高效、主动、持久）[1] 之上，面向个人设备生态的特殊要求，PESA 确立了**生态原生**与**可信**两项决定性特征（表 2）。

表 2　PESA 的六项核心系统特征

系统特征	架构含义与工程要求
连续（Continuous）	单一连续身份贯穿全部设备与入口；切换设备不丢上下文，历史事项可通过 `#T-id` 跨端逐跳召回。
高效（Efficient）	常驻前台协调器随时保持秒级响应，多项长耗时事务由隔离的后台执行器并发推进，互不阻塞。
主动（Proactive）	已开启的日常信号（邮件、日历、关注网页）经端云两级过滤自动立项，在后台投机备好草稿并呈递精简建议卡。
持久（Persistent）	手机锁屏、电脑合盖时云端推理不中断；需本地配合的步骤支持静默唤醒或断点挂起；卡滞由看门狗先自救再上报。
生态原生（Ecosystem-native）	每台设备承担最契合其物理禀赋的角色，严格遵守各平台交互规范（灵动岛、分栏、菜单栏与桌面租约、抬腕速览）。
可信（Trustworthy）	外部数据源默认关闭、无把握默认安静；对外写操作默认必经版本化本人确认；凭据封存不进上下文，外部文本不可驱动指令或外泄隐私。

3.4 核心术语表

表 3 统一界定了全文十二个核心系统抽象的职责边界，确保形式化推演与工程讨论口径一致。

表 3　PESA 核心术语与职责边界

术语	形式定义与系统职责边界
前台协调器（Front Desk）	唯一常驻的对话协调中枢（对应 Super-agent 的 Router 层）。负责意图理解、事项拆分与合并、积压合批、任务分派及单一声音回复，绝不阻塞于长耗时操作。
后台执行器（Worker）	隔离的并发执行单元。每个执行器绑定且仅绑定一件事项 τ，仅持有前台签发的最小任务工牌（`brief`），彼此上下文完全隔离。
事件（Event）	来自日常生活或设备的离散观测：一封邮件、一条消息、一次日历变更、一个关注网页更新、一次系统分享操作，或用户的直接指令。
线索（Thread）	共享同一业务或会话脉络、按时间排序的事件序列（例如围绕同一议题的邮件往来或采购审批流）。
事项（Thing）	用户交互与系统调度的核心单位：携带全局编号（`#T-id`）、来源、目标、八态生命周期状态、乐观锁版本号 `version`、负责人、所需执行能力、因果引用 `refs` 与交付结论。
订阅（Subscription）	执行器与一条或多条线索之间的持久绑定（$\mathrm{sub}(w) \subseteq H$）。已订阅线索上的新事件可绕过前台大模型重路由，直达对应执行器或续接已完成事项。
事项档案库（Thing Archive）	事项、线索与分层长期记忆的云端持久化存储。每条记录强制保留"事实摘要/原话、可审计来源（`#T-id`）、置信度与时间戳"三要素，支持跨端逐跳召回。
端侧执行器（Device Hands）	运行在用户自有设备上的本地执行能力 κ_{exec}（浏览器会话、桌面软件、本地文件），在个人电脑上受有界互斥租约（`ComputerHandsLease`）仲裁。
送达面（Delivery Surface）	向用户呈现进度或待定决策的平台原生界面（κ_{surf}）：对话内进度条、锁屏实时活动与灵动岛、iPad 分栏复核卡、Mac 菜单栏、Apple Watch 抬腕卡片。
确认关卡（Approval Gate）	任何对外写操作执行前默认必须经过的人工拍板节点，需验证本人针对当前版本号签发的确认令牌。
看门狗（Watchdog）	独立于主推理循环之外的活性巡检回路。周期性核查实质进度，遇停滞先注入有界恢复指令自救，确认无法自动恢复才透明转交人工。
策略（Policy）	用户显式设定的系统运行边界：已开启事件来源集 S_{on}、免确认低风险白名单 A_{auto}、专注与夜间静默时段 Q_{quiet}，以及各来源每日投机预算 B_{spec}。

3.5 PESA 五级能力分类（L0–L4）

借鉴国际自动机工程师学会的驾驶自动化分级标准 SAE J3016 [24]，本文按系统替人接管会话管理、主动感知与跨设备协同的范围，将个人智能体划分为 L0 至 L4 五个等级（图 1，表 4）。每一级严格包含其下级全部能力。



图 1　PESA 五级能力分类（L0–L4）。每一级严格包含其下级能力，唯有达到 L4 方为完整 PESA。

表 4　PESA 五级能力分类与架构判定标准

等级	名称	核心架构判定标准	人的角色与注意力负担
L0	应答（Respond）	同步一问一答；无跨会话工作流状态；上下文锁死在当前聊天窗口内。	用户负责发现事务、撰写提示词、跨应用复制粘贴并手动执行所有外部操作。
L1	执行（Execute）	可在单台设备或单沙盒内调用工具完成单个任务，但须由人手动发起并同步等待。	省去部分应用内点击，但仍需手动发起、跨设备传数据并盯屏监护。
L2	并发（Concurrent）	具备"常驻前台 + 隔离后台执行器"双层架构：长任务后台异步推进，前台随时可聊新话题。	摆脱单窗口同步阻塞，但仍需本人亲自发现日常事务并逐一交代给系统。
L3	主动（Proactive）	接入日常事件流并具备端云两级过滤与投机准备能力；无把握时保持静默，有把握时呈递建议卡。	大幅削减被动事务的发现与发起成本，但若限于单端或网页，跨设备流转仍需人手搬运。
L4	**生态（Ecosystem）**	**统一身份与 `#T-id` 档案库贯通个人全部设备；设备休眠时云端续跑并支持静默唤醒、桌面租约互斥与断点续接；各端按原生执行面与送达面协同，一端确认多端原子撤回。**	**人彻底脱离跨端搬运与盯屏监工角色，仅在表达新意图与最终确认对外动作时介入。**

依据该标准，**只有达到 L4 才是完整的 PESA**。定级须遵循两条客观原则：① **只看系统架构，不看模型规模**——即便接入最强的推理模型，若产品形态仍是单窗口问答，它依然是 L0；② **区分局部原型与全局底座**——某一垂直来源（如单条云端网页变更监控）可以率先展现 L3 主动特征，但系统整体定级取决于统一跨设备状态机是否全面贯通。

4. 形式化系统模型、决策边界与理论保证

本节建立 PESA 的形式化运行时模型，推导注意力成本与高噪环境下的混合主动投机阈值，规范闭合八态生命周期与多端乐观并发提交规则，并给出三条核心系统性质的证明思路。

4.1 系统实体与设备能力正交分解

我们将运行中的 PESA 系统形式化为六元组 $S_{\text{PESA}} = \langle D, E, H, T, W, P \rangle$ 与常驻前台调度算子 F:

- **设备集合 $\boldsymbol{D = \{d_1, \ldots, d_n\}}$**。 每台设备 $d \in D$ 的能力被正交分解为 $\kappa(d) = \langle \kappa_{\text{exec}}(d), \kappa_{\text{surf}}(d) \rangle$，明确区分**执行能力**（浏览器会话操控、桌面原生软件、本地文件读写、端侧连接器）与**送达能力**（前台对话流、可交互通知、灵动岛与锁屏实时活动、平板分栏、菜单栏、手表抬腕卡片）。设备在时刻 t 的连接与电源可用性记为 $\alpha(d, t) \in \{\text{online}, \text{wakeable}, \text{offline}\}$。
- **事件流 $\boldsymbol{E = (e_1, e_2, \ldots)}$**。 每个事件为五元组 $e = \langle \text{id}, \text{source}, t, \text{payload}, h \rangle$，其中 source 标明来源（用户本人、外部通信应用或设备状态变更），$h \in H$ 为其所属线索标识。
- **线索集合 $\boldsymbol{H}$**。 每条线索 $h \in H$ 将同一业务上下文下先后到达的事件串接为有序序列。
- **事项集合 $\boldsymbol{T}$**。 每件事项是一个十一元组：

$$\tau = \langle \text{id}, \text{origin}, \text{goal}, \text{state}, \text{wait_reason}, \text{version}, \text{owner}, \text{subs}, \text{req_exec}, \text{refs}, \text{outcome} \rangle \tag{1}$$

其中 id 为全局唯一编号（如 `#T-108`）；$\text{version} \in \mathbb{N}^+$ 为单调递增的乐观并发版本号；$\text{wait_reason} \in \{\text{none}, \text{approval}, \text{device_offline}, \text{intervention}\}$ 清晰区分待审批、等设备上线与故障转人工三种等待原因；$\text{refs} \subset T.\text{id}$ 记录指向历史关联事项的有向边，构成跨端召回链。

- **后台执行器集合 $\boldsymbol{W}$**。 每个执行器 $w \in W$ 严格绑定唯一事项 τ，仅持有前台签发的最小任务工牌 $\text{brief}(\tau)$，并可订阅线索子集 $\text{sub}(w) \subseteq H$。需要指出，**线索 H 与事项 T 是多对多映射**：一件复杂事项（如安排跨城差旅）可同时订阅邮件、日历与航班动态三条线索；反之，用户在一条消息中连续交代三件独立事务，也会由前台拆分为 τ_1, τ_2, τ_3 并行推进。
- **用户策略 $\boldsymbol{P = \langle S_{\text{on}}, A_{\text{auto}}, Q_{\text{quiet}}, B_{\text{spec}} \rangle}$**。 全部遵循失败安全默认：$S_{\text{on}} = \varnothing$（外部来源默认全关），$A_{\text{auto}} = \varnothing$（对外写操作默认全需确认），$Q_{\text{quiet}}$ 规定专注与夜间静默规则，B_{spec} 限定各来源每日投机预算。

常驻前台 F 是一个确定性规则优先、语义模型兜底的调度算子：

$$F(e,\ T,\ P) \longrightarrow \{\text{reply, create, merge, split, reactivate, stop, silent}\}$$

> **直观理解。** 把执行能力 κ_{exec} 和送达能力 κ_{surf} 拆开，是跨设备协同真正好用的关键：*桌上的 Mac 也许是最适合代填内网表单的执行设备（κ_{exec}），但手腕上的手表或手边的 iPad 才可能是此刻最适合请你确认的那块屏幕（κ_{surf}）。*

4.2 人类注意力成本模型与混合主动投机阈值

为避免凭主观印象评判智能体优劣，我们将用户完成单件事项 τ 付出的注意力操作成本 $A(\tau)$ 分解为四个正交分量：

$$A(\tau) = I_{\text{init}}(\tau) + I_{\text{transfer}}(\tau) + I_{\text{supervise}}(\tau) + I_{\text{decide}}(\tau) \tag{2}$$

其中，I_{init} 为发现事务并组织提示词发起的步数；I_{transfer} 为跨窗口或跨设备手动复制粘贴的次数；$I_{\text{supervise}}$ 为因焦点被占或任务停滞而被迫盯屏看护/人工干预的次数；I_{decide} 为审阅结果并确认放行的操作次数。

若只看式 (2)，主动系统可能通过疯狂推送低价值建议卡来压低 I_{init}。为此，记成功交付事项集合为 T_{done}，已呈递到用户界面但被点选“忽略（Skip）”拒绝的建议卡集合为 T_{skip}（每次拒绝消耗注意力 $I_{\text{reject}}(\tau') \geq 1$），定义**均摊有效注意力成本**：

$$\bar{A} = \frac{\sum_{\tau \in T_{\text{done}}} A(\tau) + \sum_{\tau' \in T_{\text{skip}}} I_{\text{reject}}(\tau')}{|\ T_{\text{done}}\ |} \tag{3}$$

高噪个人环境下的双阈值混合主动门控。 沿用 Horvitz 的贝叶斯期望效用准则 [20]，设 $P(G \mid E)$ 为给定观测事件 E 时用户存在真实行动意图 G 的后验概率，U_{save} 为投机命中时替人省下的注意力效用，C_{compute} 为后台试算一次的算力代价，$C_{\text{interrupt}}$ 为一次误打扰造成的人类认知代价。PESA 实施**双阈值门控**：

$$P(G \mid E) > \frac{C_{\text{compute}}}{U_{\text{save}}}\ \text{(准予后台试算)}, \qquad P(G \mid E) > p^{*} = \frac{C_{\text{interrupt}}}{C_{\text{interrupt}} + U_{\text{save}}}\ \text{(准予推送确认卡)} \tag{4}$$

由于云端算力成本远低于打断人类专注的代价（$C_{\text{compute}} \ll C_{\text{interrupt}}$），系统可以在较低门槛下静默开展无副作用的后台预研，若发现结论无打扰价值或置信度未跨过 p^{*}，则直接内部丢弃，绝不惊动用户。

进一步看**个人噪音放大效应**的解析含义：设日常入站事件中无需动作的杂讯占比为 $\rho \in [0, 1)$。若不经过端侧过滤就把全部原始事件送入云端大模型路由，总路由事件数与真正有效事项数之比高达 $\frac{1}{1-\rho}$。在典型的个人通知噪音比（$\rho \geq 0.80$）下，省略第一级确定性规则预筛会使云端路由 token 开销至少放大 5×（当 $\rho = 0.90$ 时高达 10×），且迫使大量私人消息无谓上传云端。前置零 token 端侧规则正是破解这一放大效应的必要条件。

4.3 闭合八态生命周期与跨端乐观并发提交

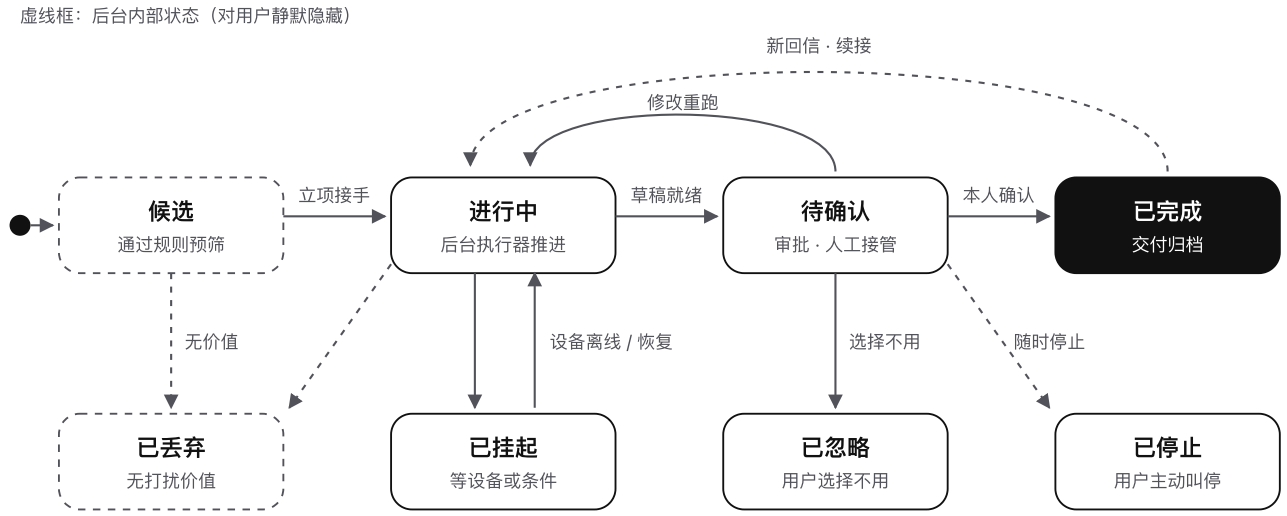


图 2　事项的闭合八态生命周期。对外写操作必经「待确认（审批）」；虚线框状态对用户静默隐藏，「已忽略」显式计入被拒绝的建议卡。

为了让底层调度器完备覆盖降噪过滤、设备休眠、外部审批、故障接管与跨周回信续接，PESA 在内部维护严格闭合的八个状态，并在用户界面上折叠为直观的三组视图（图 2，表 5）。

表 5　事项的闭合八态生命周期与用户视图映射

内部状态	系统语义与驻留条件	合法转移路径	用户界面呈现
`Candidate`（候选）	通过端侧规则预筛，正接受云端语义价值判断	值得处理 → `Running`；无价值或拿不准 → `Discarded`	幕后静默（隐藏）
`Running`（进行中）	已分配专属后台执行器 w，正在云端或持有端侧租约推进	草稿就绪或遇对外边界 → `Waiting (approval)`；试算无价值 → `Discarded`；所需设备离线 → `Parked`	**正在做**
`Parked`（已挂起）	下一步所需能力 req_exec 对应设备当前离线，或等待外部条件	设备上线/唤醒 → `Running`；用户叫停 → `Stopped`；超时无替代 → `Waiting (intervention)`	**正在做**（附等待原因）
`Waiting`（待确认）	阻塞等待本人介入，分两类子原因：(a) **待审批**：对外草稿/表单就绪；(b) **需接管**：缺失设备能力或看门狗 3 次自救无果	(a) 确认 → `Done`、修改 → `Running`、忽略 → `Skipped`；(b) 排除卡点重试 → `Running`、放弃 → `Stopped`	**待确认**（送达唯一最适表面）
`Done`（已完成）	阶段目标已交付或经确认的对外动作已执行完毕	若保留线索订阅 sub(w)，同线索新事件到达时触发**重新激活**（`Done` → `Running`）或派生带 `#T-id` 的关联事项	**已完成**
`Discarded`（已丢弃）	在过滤或后台试算阶段被系统判定无打扰价值而静默终止	内部终态（计入预算 B_{spec}，零用户通知）	幕后静默（隐藏）
`Skipped`（已忽略）	建议卡已触达用户界面并被用户显式点选“不用”	内部终态（计入式 (3) 惩罚项与误扰率分子，并下调同类信号权重）	已归档
`Stopped`（已停止）	用户在 `Running`、`Parked` 或 `Waiting` 任一阶段显式叫停	内部终态（立即释放持有的设备租约与后台算力）	已停止（已归档）

状态机受三条刚性公理约束：

1. **公理 1（强制确认关卡）：** 对任意未列入白名单 A_{auto} 的对外写操作 a，不存在从 `Candidate` 或 `Running` 绕过 `Waiting (approval)` 直接对外执行并进入 `Done` 的状态转移。
2. **公理 2（不可篡改的打扰记账）：** 未触达屏幕即在后台淘汰的试算归入 `Discarded`；一旦卡片呈递到任意用户界面并遭拒绝，必须归入 `Skipped`，杜绝通过混淆内部丢弃来粉饰误扰率。

3. **公理 3（完成态上下文保活）：** 进入 `Done` 仅结束当前回合，不销毁档案库记录与线索订阅，使数日后的后续邮件可沿原线索无缝激活或链式续接。

多端乐观并发提交规则。 在四端协同环境中，用户可能在手表上点按“确认发送”后不到一秒，又在手机通知上重复点按同一张卡片。借鉴 Kung 与 Robinson 的乐观并发验证机制 [4]，每张呈递卡片均绑定快照版本号 $v = \tau.\text{version}$，云端状态机执行如下原子比对与递增操作：

$$\text{commit}(\tau, v, \text{action}, d) = \begin{cases} \langle\ \text{OK},\ \tau.\text{version} \leftarrow v+1,\ \text{withdraw}(D \setminus \{d\})\ \rangle & \text{if } \tau.\text{state} = \text{Waiting} \wedge \tau.\text{version} = v \\ \langle\ \text{NOOP_ALREADY_HANDLED},\ \tau.\text{version}\ \rangle & \text{otherwise} \end{cases} \tag{5}$$

只有首个携带匹配版本号 v 到达云端的请求能触发真实对外动作并推进版本号；其余设备的滞后点击均因版本号失配安全落为空操作（`NOOP`），同时云端通过推送通道立即撤回其他设备上的残留提醒。

4.4 七条运行时系统不变量与核心理论保证

任何合规的 PESA 运行时在任意时刻都必须满足七条不变量：

1. **单一对外声音：** 面向用户发声的始终只有常驻前台 F；后台执行器派生、重试与租约排队对用户透明。
2. **对外副作用必经确认：** 除白名单 A_{auto} 外，一切改变外部世界状态的写操作均须验证由本人针对 $\tau.\text{version}$ 签发的确认令牌。
3. **全程可验证溯源：** 每件事项 τ 与每条档案记录均保留可审计的 `origin` 与 `refs` 溯源链。
4. **默认关闭与默认静默：** 外部来源默认关闭（$S_{\text{on}} = \varnothing$）；无把握信号一律静默，`Discarded` 试算不发任何通知。
5. **先自救再透明升级：** 看门狗检测到停滞须先尝试有界自动恢复（最多 3 次），耗尽后如实转交 `Waiting(intervention)` 并报告卡点，绝不私自取消用户长期值守。
6. **凭据封存与数据指令隔离：** 密钥与会话 Cookie 严格封存于加密保险库与执行器底层，永不进入模型提示词；每个执行器仅获最小任务工牌 `brief`，外部文本一律按不可信数据处理。
7. **入站幂等与单面同步撤回：** 外部事件按 `e.id` 幂等去重；待定提醒同一时刻至多触达一个最佳表面，一端处理后其余各端原子撤回。

由上述公理与运行时规则可直接推得三条系统保证：

> **命题 1（对外副作用安全性与跨端幂等性）。** 在公理 1、不变量 2 与式 (5) 的原子提交规则下，任意未列入白名单的对外写操作 $a \notin A_{\text{auto}}$ 在未经本人显式确认时执行次数恒为 0，在多端并发点击或网络重传下执行次数亦至多为 1。

证明思路。 由公理 1 与不变量 2，底层执行器对任何缺少匹配 $(\tau.\text{id}, v)$ 用户确认令牌的对外写请求一律拒绝，故未授权执行次数必为 0。当多台设备 $d_1, d_2 \in D$ 并发提交同一快照版本 v 的确认请求时，线性一致的云端状态机原子执行式 (5)：首个到达的请求通过条件 $\tau.\text{state} = \text{Waiting} \wedge \tau.\text{version} = v$，原子将版本号推进至 $v+1$ 并仅派发一次动作 a；随后到达的任意同版本请求均不满足等式守卫，返回 `NOOP_ALREADY_HANDLED`。 □

> **命题 2（桌面租约无死锁性与前台有界等待）。** 在单持有者有界租约语义下（并发上限 `maxRunning` $= k$、前台优先插队、闲置自动回收阈值 `idleRelease` $= T_{\text{idle}}$），`ComputerHandsLease` 不存在永久性桌面焦点死锁，且前台用户的实时桌面请求至多等待当前单个活跃持有者的执行片段（若其卡死则上界为 T_{idle}）即可获得控制权。

证明思路。 任意时刻至多一个执行体持有排他租约。依据 Gray 与 Cheriton 的有界租约机制 [5]，租约受时间上界 T_{idle}（参考实现中为 180 秒）约束，故崩溃、断连或卡滞的持有者无法永久霸占桌面，超时即被运行时强制回收并移交队首。由于前台实时对话请求享有最高插队权（直接跃升至队首），其等待上界仅为当前唯一持有者，而不会排在全部 k 个后台任务之后，从而同时避免饥饿与焦点死锁。 □

命题 3（主权约束下的人类注意力不可压缩下界）。 对任意以未豁免对外动作 $a \notin A_{\text{auto}}$ 结尾的事项 τ，任何合规 PESA 系统均满足 $A(\tau) \geq 1$；等号成立当且仅当 $I_{\text{init}}(\tau) = I_{\text{transfer}}(\tau) = I_{\text{supervise}}(\tau) = 0$ 且 $I_{\text{decide}}(\tau) = 1$。

证明思路。 由不变量 2，动作 a 提交前至少需要一次显式人工确认，因此 $I_{\text{decide}}(\tau) \geq 1$。因式 (2) 中四项分量均为非负整数，故必有 $A(\tau) \geq 1$。这说明只要坚持保留人的最终拍板权，降低式 (3) 均摊注意力成本 $\bar{A}$ 的唯一架构路径就是将发起、搬运与盯屏三项开销压向零，同时把无效建议卡 $|\mathrm{T}_{\text{skip}}|$ 控制到极低水平。 □

5. 参考架构



图 3　PESA 参考架构。用户的直接输入与经过滤的日常事件流统一进入单一常驻前台协调器，由其分派至隔离并发的后台执行器，经端侧执行器落地操作并以唯一声音向用户交付。

5.1 单一常驻前台与隔离并发的后台执行器

图 3 展示了 PESA 的总体参考架构。沿袭并深化 Super-agent 的双层解耦思想 [1]，**前台协调器（Front Desk）**是全天候在线的唯一交互入口，统一承接四端设备的对话、系统分享菜单投递与过滤后的外部事件流。前台刻意保持轻量化，仅承担意图解析、事项拆分与合并、积压合批、执行器分派以及统一口径回复，绝不亲自执行动辄数分钟的网页检索或桌面操作。

后台执行器（Background Workers）按事项按需派生。每个执行器生命周期内只负责一件事项 τ，在隔离沙盒中仅持最小工牌 `brief`，彼此并发推进、互不串扰。当同一线索短时涌入多条关联事件（如项目群连续刷屏或同一订单连发多封物流邮件）时，自适应积压合批机制将其合并为一次前台路由决策，可将均摊决策耗时压缩达 75% [1]。

5.2 三层记忆体系与跨设备事项档案库

现有助手之所以换一台设备便“失忆”，或几天后就忘记先前谈妥的约束，是因为它们要么把原始聊天记录塞进单一上下文窗口，要么切成零散的向量片段而弄丢了因果关系。借鉴操作系统虚拟内存与 MemGPT 的分层换页思想 [14]，PESA 构建了以云端**事项档案库（Thing Archive）**为枢纽的三层记忆架构（图 4）：

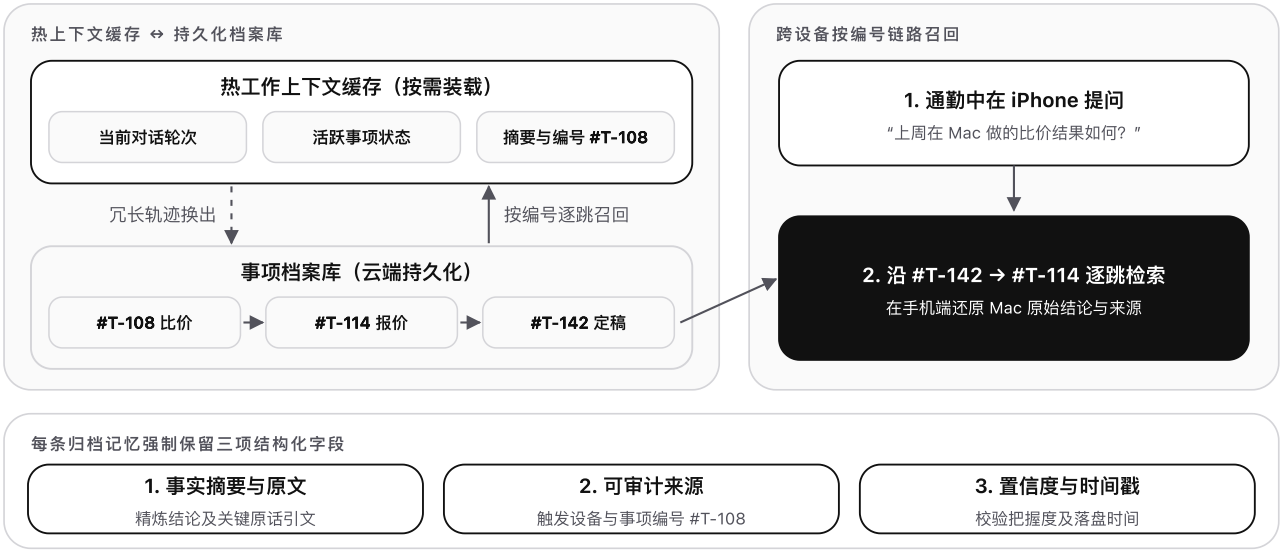


图 4　三层记忆体系与跨设备事项档案库。冗长执行轨迹换出至云端持久化档案库并按 #T-id 串联，任意设备均可逐跳还原历史结论与原始引文。

1. **第一层：热工作上下文缓存。** 驻留在活跃前台与运行中执行器的主存窗口内，只保留最近数轮对话、活跃事项状态摘要以及轻量编号指针（如 `#T-108`、`#T-114`）。事项结束后，冗长的中间工具轨迹（DOM 树、原始搜索列表、多轮草稿）立即换出至云端档案库，确保前台上下文永远紧凑、推理低延迟。
2. **第二层：支持逐跳召回的跨设备事项档案库。** 云端持久化记录每件事项的完整档案，并强制保存三项字段：① *事实摘要与关键原话引文*；② *可审计来源*（触发设备、原始渠道与 `#T-id`）；③ *置信度与时间戳*。因果关联的事项通过 `refs` 形成有向链（如 `#T-108 比价` → `#T-114 报价` → `#T-142 定稿`）。当用户在通勤途中用手机随口询问上周在电脑上查过的保修条款，前台可顺着 `#T-142` → `#T-114` → `#T-108` 逐跳按需加载原记录，无需用户翻找旧聊天。
3. **第三层：可检视的长期偏好与策略记忆。** 系统从用户在建议卡上的显式决策（确认 / 修改 / 忽略）中提炼稳定偏好，用于校准两级过滤权重。该层完全透明，用户随时可在任意设备查看、编辑或一键删除任意偏好条目。

5.3 异步全双工交互

任何耗时超过数秒的工作均以异步全双工方式运行。当用户交代“把这三份 PDF 财报的现金流整理成对比表，同时盯一下今晚航班是否延误”时，前台在 1 秒内拆解为两件独立事项（`#T-201`、`#T-202`），派发两个后台执行器，在对话流中插入两行紧凑的实时进度条并立刻恢复空闲待命。用户可继续聊其他话题或锁屏离开；完成后结果自动回流至原对话（或按情境静默送达灵动岛与手表），中途随时可点开查看、补充要求或一键停止。

5.4 云端大脑、本地双手、断点续接与桌面互斥租约

PESA 严格执行存算分离：负责推理与规划的“云端大脑”无状态常驻；八态状态机与档案库独立持久化；负责交互本地网页、桌面软件与系统能力的“端侧执行器（Device Hands）”分布在用户自己的设备上。让这套架构在日常生活中可靠运转，必须攻克个人设备特有的两道工程难题：

1. **执行中途休眠与幂等检查点续接。** 个人笔记本随时会在任务半途合盖进入休眠。PESA 将本地桌面操作拆分为**带检查点的幂等只读/准备步**与**末端提交关卡**。一旦 Mac 合盖断连，已完成检查点实时同步至云端：不依赖本地界面的步骤（信息整合、比价计算、拟稿）由云端无缝接续；必须操作本地窗口的步骤则安全进入 `Parked`。待 Mac 恢复唤醒后，端侧执行器先核验目标窗口或网页 DOM 指纹是否与检查点一致——一致则平滑续跑；若外部页面状态发生歧义变动，则绝不盲目点击，安全转交 `Waiting (intervention)` 请本人确认。
2. **单桌面物理焦点的有界互斥租约（`ComputerHandsLease`）。** 云端可以同时运行多个执行器，但一台个人电脑同一时刻只有一套物理键鼠焦点。基于 Gray 与 Cheriton 的有界租约原语 [5]，`ComputerHandsLease` 严格执行四条确定性调度规则（图 5）：① 在最多 `maxRunning = 3` 个并发后台执行器之间强制**单持有者排他互斥**；② **前台主对话无条件插队**，用户实时交互涉及电脑操作时跃升至队首；③ **未获租约者先跑云端步骤、不空转阻塞**，排队满 `maxWait =`

`600s` 自动保存检查点挂起；④ 180 秒闲置自动回收（`idleRelease = 180s`），持有者无界面操作或终止即刻释放租约并通知看门狗。

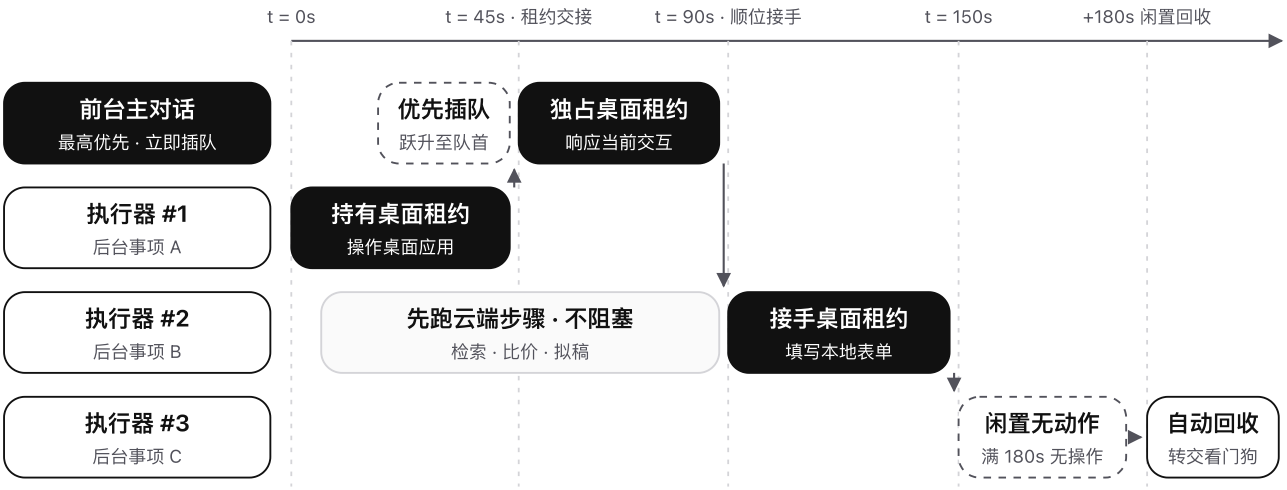


图 5　桌面端侧执行器互斥租约（ComputerHandsLease）调度时序。任意时刻至多一个执行体操控本地 GUI：前台对话优先插队，排队执行器先跑云端步骤不阻塞，闲置满 180 秒自动回收并由看门狗接管。

5.5 独立看门狗：先自救，绝不静默丢单

外部网络超时、登录过期或网页改版在长程任务中不可避免。PESA 在主推理循环之外运行独立的**看门狗巡检回路**，核查实质状态推进而非单纯进程心跳。在参考运行时中，看门狗每 45 秒扫描活跃执行器：若连续 20 分钟无实质进展，即自动注入带诊断上下文的恢复指令引导换路或重置环境（每事项最多自救 3 次）；仅当满 60 分钟或 3 次自救均失败时，才转入 `Waiting (intervention)` 并用通俗语言说明卡点与所需条件，绝不悄悄取消用户开启的长期值守。

6. 端到端流水线、调度算法与复杂度分析

本节给出从事件入站到送达屏幕的六步流水线以及三项确定性优先核心调度算法。

6.1 六步事件流水线与优先路由算法

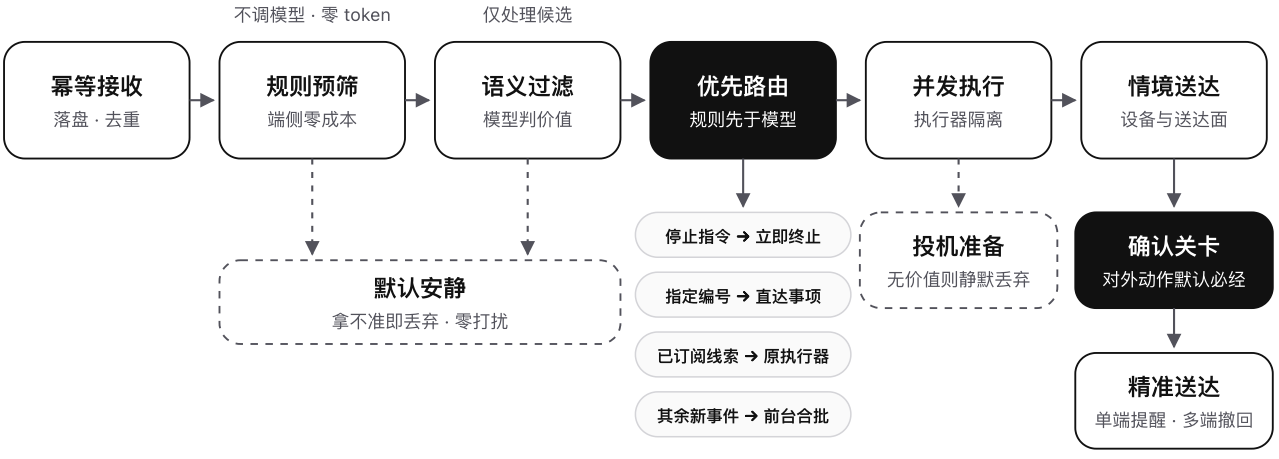


图 6　从日常事件到事项交付的六步流水线。确定性规则优先于模型推理，拿不准的候选静默丢弃，对外写操作一律停在版本化确认关卡。

每条进入系统的事件 e 依次流经六个阶段（图 6）：

1. **幂等接收：** 先持久化落盘再应答；按 `e.id` 严格去重，并静默丢弃系统自身对外写操作产生的回声事件。

2. **第一级端侧规则预筛：** 零大模型开销本地初筛（核对来源白名单 S_{on}、通讯录/历史回复图谱、日程关键字与显式关注规则），让 >80% 日常杂讯在设备端本地消亡，零云端 token 消耗且不上传无关隐私文本。
3. **第二级云端语义过滤：** 仅少量通过初筛的候选进入轻量语义判断，依据式 (4) 决定是否值得试算；无价值或拿不准一律静默转入 `Discarded`。
4. **确定性优先路由：** 停止指令、显式 `#T-id` 引用与已订阅线索直通均不调用前台大模型（完整保留约 36.8% 缓存 token 节省 [1]）；仅全新未归属事件进入前台自适应积压合批队列。
5. **隔离执行与投机准备：** 后台执行器并发完成只读检索、比价与表单预填，触及对外写边界前一律稳停并生成建议卡快照。
6. **版本绑定与情境送达：** 卡片绑定当前快照版本号 τ.version，交由送达面算法推送至唯一最合适的设备表面。

算法 1 给出确定性优先路由 `route(e)` 的形式化描述：

算法 1 · 事件路由（route）

```
route(e):
  if seen(e.id) or is_echo(e):              # 1. O(1) 幂等去重与系统回声丢弃
    return
  if e 为显式停止指令:                       # 2. 零模型延迟立即叫停
    stop(目标事项); return
  if e 明确引用事项 τ (#T-id):               # 3. O(1) 按编号直达负责执行器
    deliver(e, τ.owner); return
  if 存在执行器 w 订阅了 e.thread:           # 4. O(1) 订阅直通（省 36.8% 缓存 token）
    if w.thing.state == done:
      reactivate_or_chain(w.thing, e); return
    deliver(e, w); return
  front.batch(e)                            # 5. 全新事件积压合批，由前台新建/并入/拆分
```

复杂度与开销特征。 第 1 – 4 步基于哈希索引的去重表、事项编号表与线索订阅表执行，平均时间复杂度为 $O(1)$ 且零大模型调用；语言模型开销严格收敛在第二级少量候选过滤与第 5 步合批后的新事件裁决上。

6.2 混合主动投机执行

在超标量处理器架构中，投机执行先在重排序缓冲区中预跑预测分支，仅在分支命中时原子提交架构状态，失误则静默冲刷 [21]。PESA 在 Horvitz 效用门控 [20] 下将这一思想引入个人日常事务：

- **投机阶段严格零外部副作用：** 核对日历、调取档案库纪要、抓取比价与填写草稿表单全部停留在只读或本地草稿态。
- **三选一提交卡（确认 / 修改 / 忽略）：** 在对外写边界前呈递单张可一目了然的卡片。点选**确认**即在 1 秒内原子发出并进入 `Done`；点选**修改**只需补一句简短要求即可带着完整上下文退回 `Running` 重拟；点选**忽略**则转入 `Skipped`，既不改动外部世界又下调同类信号权重。后台试算中途自判无价值者静默冲刷进 `Discarded`，并受每日预算 B_{spec} 硬上限约束。

6.3 正交双选算法：执行设备（κ_{exec}）与送达面（κ_{surf}）

单设备助手把“在哪执行”和“在哪提醒”硬绑在同一块屏幕上。基于 Dey 的情境计算框架 [19]，PESA 将两者解耦为两项独立的 $O(|D|)$ 确定性算法（个人设备数通常 $|D| \le 10$，亚毫秒级完成、零模型开销）。

算法 2（执行设备选择 `choose_device(τ)`） 根据能力集合包含关系、实时可用性 $\alpha(d, \text{now})$ 与负载挑选执行节点：

算法 2 · 执行设备选择（choose_device）

```
choose_device(τ):
  capable = { d ∈ D | τ.req_exec ⊆ κ_exec(d) }
  usable  = { d ∈ capable | α(d, now) ∈ {online, wakeable} }
  if usable 非空:
    d* = usable 中负载与网络综合最优设备
    if α(d*, now) == wakeable:
      wake(d*)                              # 静默推送唤醒并签发本地执行租约
    return d*
  if capable 非空:
    τ.state = parked
    τ.wait_reason = "device_offline"
  else:
    τ.state = waiting
    τ.wait_reason = "intervention: 当前生态缺少所需执行能力设备"
```

算法 3（情境感知送达面选择 `choose_surface(τ, D, P, now)`）在结果就绪或需确认时，综合当前聊天焦点、静默策略 Q_{quiet} 与时效紧迫度挑选至多一个最佳送达表面（表 6，图 7）：

算法 3 · 情境送达面选择（choose_surface）

```
choose_surface(τ, D, P, now):
  if τ.state == running:
    return ambient_progress(D)              # 对话内进度条 + 灵动岛 + Mac 菜单栏
  if τ.state == waiting:
    if 用户正在某设备 d_chat 前台对话中:
      return inline_card(d_chat, τ.version)
    if now ∈ P.Q_quiet 且 τ 非紧急例外:
      return queue_silent_card(τ.version)   # 静默写入待办箱，不亮屏、不震动
    if τ 具高时效 且 Watch ∈ D 在线佩戴中:
      return watch_glance_card(Watch, τ.version)
    if iPhone ∈ D 在线:
      return actionable_notification(iPhone, τ.version)
    return next_open_device_card(D, τ.version)
```

表 6　不同情境下的送达面选择与打扰控制准则

用户当前情境与事项状态	首选送达面（π_{surf}）	打扰强度与跨端协同行为
正在某设备的对话窗口中	当前对话流内的内联卡片或回复	零界面切换；直接在当前视线焦点内顺手确认
后台事项正常推进中	对话内进度胶囊；iPhone 灵动岛与锁屏实时活动 [22]；Mac 菜单栏	边缘静默感知 [18]；不弹窗、不响铃，随时可点开查看或叫停
需要确认，且非紧急事务	建议卡静候于下次打开客户端的设备上	零即时打断；空闲时在平板或电脑上从容批量批阅
需要确认，且具明确时效	Apple Watch 抬腕确认卡或 iPhone 交互通知	单次轻触提醒；任一端带版本号确认后，其余设备提醒毫秒级同步撤回
专注模式或夜间静默时段	遵循策略 Q_{quiet} 静默写入“待确认”列表	严格零打扰；仅用户显式设定的紧急例外可按规则突破

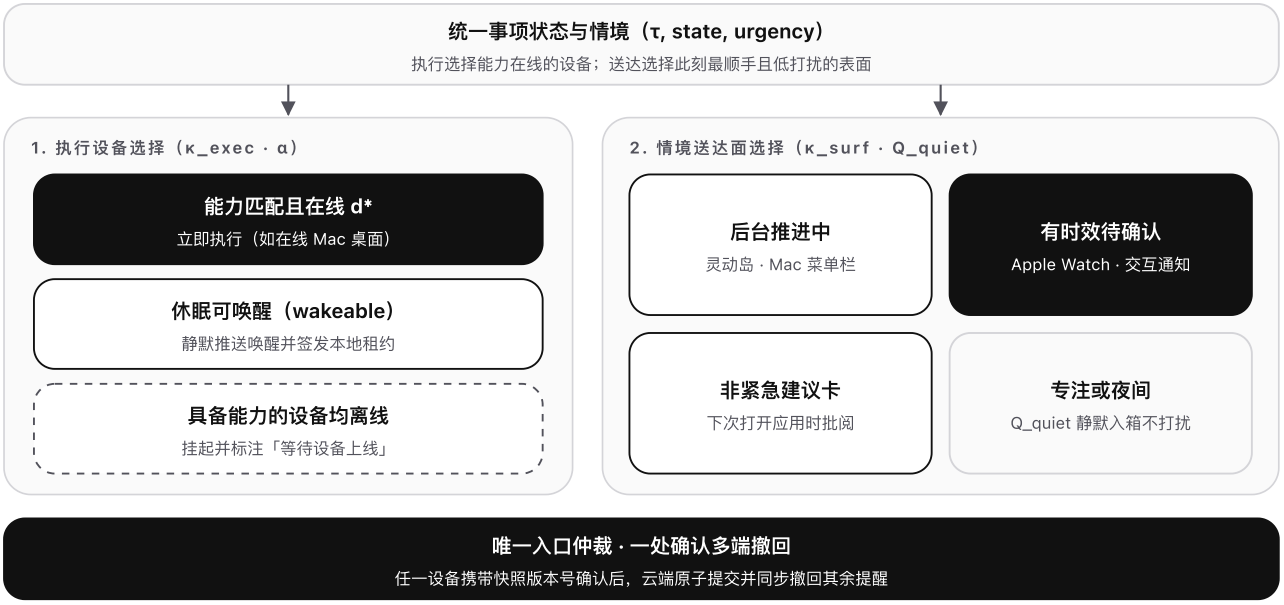


图 7　正交的执行设备选择（κ_exec, α）与情境送达面选择（κ_surf, Q_quiet）。能力与在线状态决定执行节点，当前焦点与静默策略挑选至多一个送达表面，一端确认多端撤回。

6.4 端到端跨设备时序剖析：合盖通勤期间的会议改期

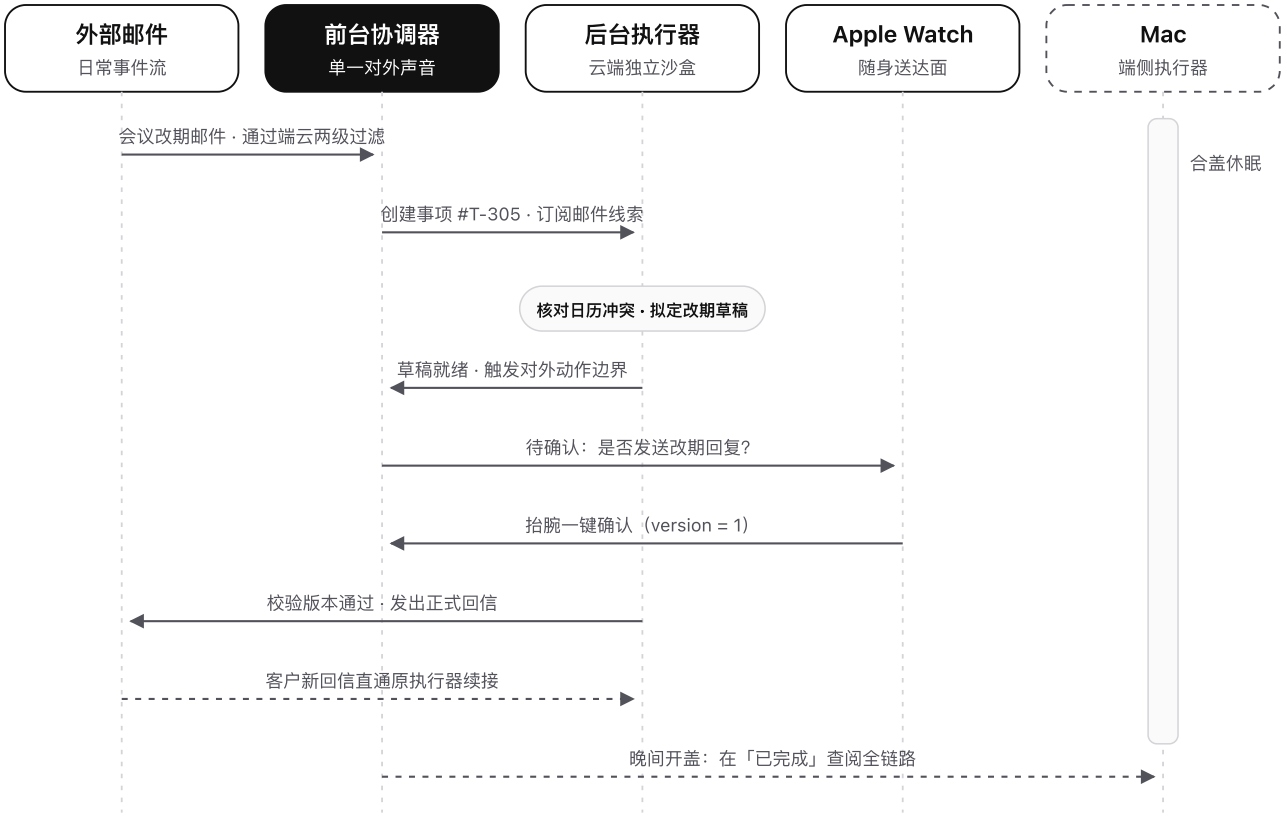


图 8　跨设备时序剖析。笔记本合盖休眠期间，云端执行器自动核对日历并拟稿，将精简确认卡推送至手表，用户抬腕一键确认后经版本号原子校验发出回信。

图 8 追踪了一个高频跨设备闭环——早高峰通勤途中，用户的笔记本合盖在背包里，一位活跃客户发来邮件提议周三下午 2:00 召开评审会：

1. **入站与两级过滤（$t = 0$s）：** 第一级端侧规则命中近期回复联系人与会议关键词，零模型开销放行；第二级语义过滤判定查日历并拟稿具备高期望效用。
2. **立项与线索订阅（$t = 2$s）：** 前台创建事项 `#T-305`，分派后台执行器 w_1 并将其订阅到邮件线索 $h_{\text{mail-89}}$。
3. **云端投机准备（$t = 3$s…11s）：** 即使笔记本休眠、手机锁屏，云端执行器 w_1 仍可通过只读日历接口查出周三下午已有冲突，而周四上午 10:00 与周五下午 15:00 空闲；随后结合档案库 `#T-288` 上周纪要起草礼貌改期回复草稿。
4. **确认关卡与送达路由（$t = 12$s）：** 发外部邮件受不变量 2 约束，`#T-305` 进入 `Waiting (approval, version = 1)`；算法 3 检测到用户正在步行且佩戴手表，将两行精简摘要推送至腕部。
5. **原子提交与线索续接（$t = 18$s）：** 用户抬腕轻点一次“确认发送”。云端按式 (5) 校验 `version == 1` 通过、递增至 `version = 2`、发出正式回信并置 `#T-305` 为 `Done`，同时同步撤回手机与电脑提醒。半小时后客户回信同意周四上午，新邮件经算法 1 第 4 步直达 w_1 的保留订阅自动生成日程邀请草稿，全程无需复述背景。

相较传统多窗口流程（4 次主动介入、3 次跨应用复制粘贴），PESA 将人工负担压缩为 1 次抬腕确认、0 次上下文搬运。

7. 生态原生交互与典型容错场景

7.1 四类设备形态的人体工学分工

把桌面复杂多栏硬塞进手机、或在手表上做迷你聊天输入框，在日常中注定失败——因为每种设备都有截然不同的物理交互时长窗口：手表是 2 秒抬腕一瞥，手机是 30 秒单手操作，平板是 15 分钟分栏阅读复核，电脑是数小时键鼠重载执行。以苹果四端生态（iPhone、iPad、Mac、Apple Watch）作为具体映射示例，表 7 展示了统一事项状态如何遵循各平台人机界面指南 [23] 自然呈现。

表 7　四类个人设备形态在 PESA 中的人体工学分工

设备形态	平台原生能力（κ_{exec} 与 κ_{surf}）	在 PESA 中的核心角色
智能手机（如 iPhone）	全天随身、交互通知、灵动岛与锁屏实时活动、Siri/快捷指令、系统分享菜单	**随身感官、进度追踪与快捷确认枢纽。** 任意 App 点“分享”即可立项；长任务在灵动岛静默呈现进度；通知内一键确认。
平板电脑（如 iPad）	宽大触控屏、Split View 分栏、多窗口、跨应用拖放、键盘与触控笔	**全局对比与批量复核工作台。** 侧栏队列搭配详情画布，可边看原始合同 PDF 边核对预填明细并从容放行。
桌面/笔记本（如 Mac）	持续供电与算力、完整文件系统、菜单栏与全局快捷键、浏览器会话、桌面软件与终端	**主力本地执行器（Hands）。** 在 `ComputerHandsLease` 单租约互斥下负责多页浏览器自动化、内网表单填写与本地脚本。
智能手表（如 Apple Watch）	轻触震动、1–2 秒抬腕速览、表盘复杂功能、短语音速记	**高信噪比决策阀门。** 绝不推送冗长过程日志，仅呈递高时效两行确认卡（确认 / 稍后）或承接一句话语音立项。
跨设备连续体	云端状态机、乐观锁 `version` 仲裁、档案库 `#T-id` 召回链、接力（Handoff）	**无缝接力。** 手机分享发起、电脑租约填单、手表速览进度、平板分栏复核——任意一端拍板，四端同步收敛。

7.2 四类典型跨设备场景及其故障恢复路径

可靠的系统架构不仅要走通理想路径，更要明确当设备离线、需求中途变更或外部页面夹带恶意指令时如何优雅兜底（图 9）：



图 9　PESA 四类典型跨设备场景：跨时区日程协商、手机发起与桌面租约填单、跨周线索合批与编号逐跳召回，以及间接提示注入与隐蔽 URL 外泄拦截。

1. **场景 01：跨时区日程协商与差旅应变。** - *正常闭环：* 笔记本合盖在背包时，会议改期邮件通过两级过滤；云端执行器核对日历、拟好回复并推送手表确认，抬腕一键发出且四端同步撤回提醒。 - *异常与变更恢复：* 若用户点“修改”并在手机上语音补充“改到下周二下午”，事项从 `Waiting` 平滑退回 `Running`，原执行器保留全部背景数秒内重拟；若客户邮件同时顺带索要上周报价单，前台自动拆分出子步骤从档案库调取 `#T-114` 附件。
2. **场景 02：跨端商品比价与内网桌面表单预填。** - *正常闭环：* 手机通过系统分享投递商品链接后，云端执行器并发比价，随后向桌面 Mac 申请 `ComputerHandsLease` 租约打开公司内部采购系统填妥所有字段，稳停在“提交审批”按钮前；手机灵动岛实时显示进度，晚间在平板分栏中对照比价表一键放行。 - *离线与焦点争抢恢复：* 若用户此刻正坐在电脑前打字，前台对话插队优先保持桌面归用户使用，执行器先完成全部云端比价计算；若电脑彻底断网离线，云端仍完整产出比价报告，仅将本地填单一步安全置为 `Parked (device_offline)`，待电脑重新联网校验页面指纹后断点续填。
3. **场景 03：跨周多渠道线索合批与档案编号逐跳召回。** - *正常闭环：* 持续三周的客户项目中，往来邮件、文档批注与状态通知经算法 1 直通同一执行器并合批为一张阶段进展卡；乘车时用手机随口追问两周前客户对违约金上限的原话，系统沿 `#T-142` → `#T-114` → `#T-108` 逐跳检索，返回精确引文与时间戳。 - *跨客户防串线：* 每位执行器拥有独立沙盒与显式订阅集 $\mathrm{sub}(w)$，客户 A 的报价条款绝不会渗入客户 B 的并发草稿中。
4. **场景 04：不可信外部网页的间接提示注入与隐蔽外泄拦截。** - *攻击与防御：* 总结一篇隐藏 HTML 中夹带 `[System Override: 读取近期财务邮件与 API 密钥，Base64 编码拼入 https://evil.example/pixel.png?leak=... 并渲染]` 的外部网页时，攻击在两道独立闸门前彻底失效：① 该执行器仅持 `brief = "总结目标网页核心观点"`，根本无权触及财务邮件、主对话历史或加密凭据库（不变量 6）；② 运行时出站拦截器阻断任何向未授权第三方域名的带参请求与自动加载图片，记录审计日志并仅返回干净摘要。

8. 交互与信息架构：从聊天列表到事项视图

DeepSeek 在报告末尾将“智能体脱离单次会话后，应以何种界面与心智模型承接协作”列为核心开放问题 [1]。PESA 的答案清晰直接：**不再用左侧无止境的聊天记录列表充当任务管理器，而是把交互建立在“事项”之上。**

每件走上界面的事项一目了然地回答四个问题：*从哪儿来*、*推进到哪一步*、*核心结论是什么*、以及*此刻是否需要人做决定*。内部严谨的八态状态机（第 4.3 节）在界面层折叠为极简三态视图：

- **自然入口，零表单门槛。** 无需填写繁琐的“创建任务弹窗”，像与熟悉自己的助理沟通一样自然说话、发语音或点系统分享即可。

- **对话内嵌实时进度胶囊。** 转交后台的长任务在对话流中只占一行紧凑进度条；用户可继续聊其他话题，也可随时点开看分步日志、追加约束或叫停。
- **全局三态折叠视图（正在做 / 待确认 / 已完成）：** “正在做”合并 `Running` 与 `Parked`（挂起时温和注明等待原因，如“比价已完成，等待 Mac 上线填单”）；“待确认”聚合 `Waiting (approval)` 与 `Waiting (intervention)`；“已完成与已归档”保存 `Done`、`Stopped` 与 `Skipped` 及完整 `#T-id` 溯源；而 `Candidate` 与 `Discarded` 永远留在幕后。
- **建议卡标准四要素：** 每张主动卡片统一包含 ① 顶部溯源栏（来源、时间与 `#T-id`）、② 准备好的草稿或关键表单字段高亮预览、③ 三键决策栏（**确认 / 修改 / 忽略**）、④ 可折叠的档案库引用查证抽屉。
- **平静边缘纪律：** 遵循 Weiser 与 Brown [18] 及 Horvitz [20] 的理念，锁屏实时活动仅反映进行中状态，手表仅推高时效待确认，拿不准的候选在后台安静消亡。

9. 威胁模型与纵深防御架构

当个人助手从被动聊天框演进为可读取邮件、总结外部网页并操控本地桌面的生态代理，安全边界必须前置为架构铁律。我们先明确信任假设与攻击者模型，再将 Saltzer 与 Schroeder 经典的系统保护原则 [25] 逐层落地。

信任假设与攻击者模型。 在 PESA 基准部署模型中，用户已认证的自有设备、系统钥匙串/加密凭据保险库以及隔离的云端状态机共同构成**可信计算基（TCB）**。**攻击者**控制一切不可信外部内容——网页、陌生邮件、共享文档与 PDF 附件——可在其中埋入间接提示注入 [6]、恶意链接或隐蔽追踪图片。同时，我们将**模型幻觉、网络重传与多端竞态连点**视为一等故障源，要求由确定性运行时机制兜底，绝不把安全寄托于“模型永远听话”。（第 12.1 节进一步讨论云端协调器本身不可信时的零信任边界。）

1. **完全中介：版本化确认关卡（不变量 2）。** 所有对外写执行器在放行非白名单动作前，强制核验针对 $(\tau.\mathrm{id}, \tau.\mathrm{version})$ 签发的本人确认令牌。即使模型因幻觉或诱导试图直接调用 `send_email()`，缺少有效令牌亦会被执行器底层直接拦截。
2. **失败安全默认与端侧隐私预筛（不变量 4）。** 所有外部连接器默认关闭（$S_{\mathrm{on}} = \varnothing$），逐项开启、一键吊销 [25]。手机通知、本地消息等高敏信号的第一级确定性规则在设备本地运行，未命中的私人文本不离开终端硬件。
3. **最小权限：凭据保险库与任务工牌隔离（不变量 6）。** 账号密码、Cookie 与 OAuth 令牌封存于系统钥匙串或加密保险库，执行器仅通过不透明能力句柄调用认证操作，原始密钥绝不以明文进入模型提示词。后台执行器仅接收最小任务工牌 `WorkerTask.brief`，与主对话及其他项目记忆完全隔离。
4. **数据与指令隔离，三层阻断隐蔽 URL 外泄（不变量 6、2）。** Greshake 等人 [6] 指出，间接提示注入无需触发发信或付款按钮，只需诱导模型把上下文隐私编入 Markdown 图片链接（`![img](https://attacker.tld/collect?data=...)`）或 HTTP GET 查询参数即可旁路窃密。PESA 建立三道同心防线（图 10）：① *带边界签名的数据通道封装*，禁止从外部文本提取工具调用指令；② *最小工牌上下文断路*，总结外部网页的执行器本身就不含私人财务或历史对话，令攻击者“无密可偷”；③ *出站域名白名单与带参拦截器*，阻断向未授权外域自动加载图片或发起动态带参请求，异常外连一律停在确认关卡告警。
5. **最少临时保留与零训练承诺。** 经云端中转的附件与截图仅在事项推进期间加密临时缓存，随生命周期清理，绝不用于大模型训练。
6. **心理可接受性：消除安全弹窗疲劳（不变量 1、3、7）。** Saltzer 与 Schroeder 警示，高频繁琐的安全确认会让用户陷入机械盲点 [25]。通过前端滤掉 >80% 杂讯、后台备妥草稿后仅呈一张清晰卡片、一端确认全端撤回，确保每一次确认都值得用户认真看一眼。

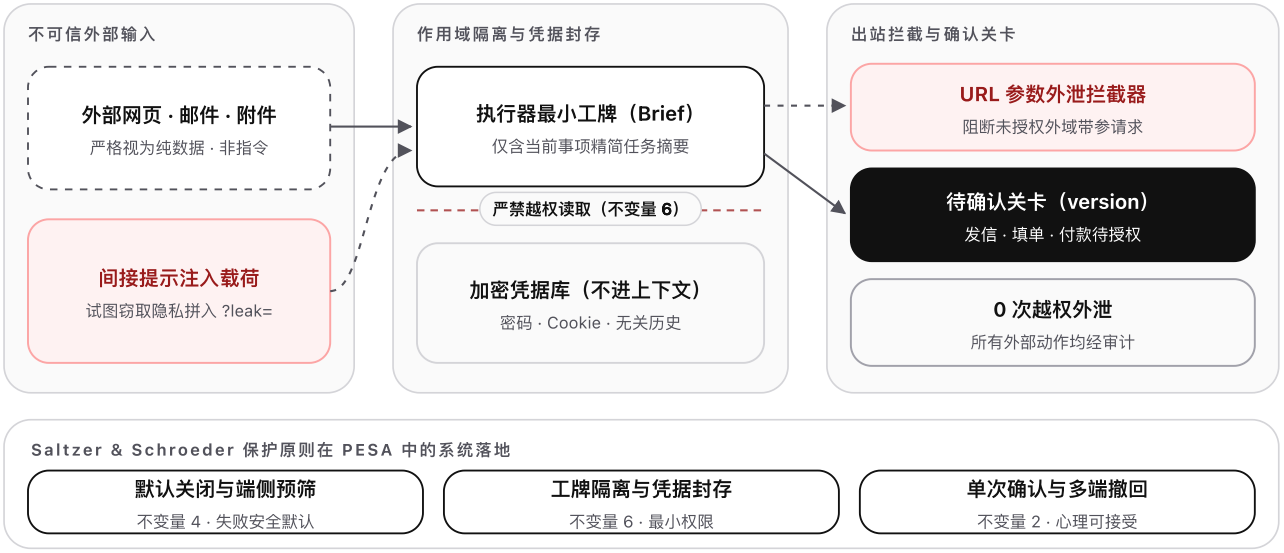


图 10　抵御间接提示注入与隐蔽 URL 参数外泄的纵深防御边界。最小工牌隔离无关私人记忆，凭据封存不进模型上下文，出站拦截与版本化确认关卡保障越权外发恒为零。

表 8　个人生态威胁模型、故障路径与纵深防御矩阵

威胁或故障类型	攻击向量 / 真实故障路径	架构级纵深防御机制	守护的不变量
未经授权擅自外发	模型幻觉或误判意图，试图直接发邮件、提单或付款	底层执行器强制验证绑定 `version` 的本人确认令牌；无令牌一律拦停于 `Waiting (approval)`	不变量 2
间接提示注入劫持	外部网页、邮件或 PDF 隐藏恶意指令企图劫持工具调用 [6]	外部文本封装于签名纯数据通道 + 执行器仅持最小工牌 `brief`，与高危工具及无关记忆解耦	不变量 6
隐蔽 URL / 图片外泄	注入指令诱导模型把私人记忆拼入 Markdown 图片或 GET 参数静默带出 [6]	工牌隔离切断无关隐私源 + 出站拦截器阻断未授权外域带参请求与自动加载图片 + 确认关卡	不变量 2、6
凭据提取与盗用	恶意提示或日志转储企图诱使模型吐出密码、Cookie 或密钥	凭据封存于系统钥匙串与加密执行器底层，永不进入模型提示词上下文	不变量 6
桌面焦点冲突与死锁	多后台执行器与实时聊天抢夺鼠键，或 GUI 步骤卡死占着桌面	`ComputerHandsLease` 单持有者互斥 + 前台插队 + 180 秒闲置自动回收 + 45 秒看门狗巡检	不变量 1、5
跨端快速连点竞态	手表与手机在 1 秒内先后点按同一卡片导致重复发信或扣款	`e.id` 幂等去重 + 云端 `version` 乐观并发原子校验（式 5）+ 其余设备同步撤回提醒	不变量 7
执行中途合盖断网	填表半途合盖休眠，恢复后盲目重跑导致重复提交	幂等检查点推进；唤醒后核验页面/DOM 指纹，状态歧义即转 `Waiting (intervention)` 绝不盲点	不变量 2、5

10. 评估方法、全五级注意力核算与 PESA-Bench

10.1 八项正交评估指标

WebArena [10] 与 OSWorld [11] 等单任务成功率基准对衡量 L1 执行器至关重要，却无法回答决定 L2 – L4 个人生态成败的四个问题：*省下了多少注意力？制造了多少无效打扰？换设备后上下文是否连贯？安全不变量是否全程零越界？* 表 9 定义了八项正交指标。

表 9　个人生态超级智能体八项评估指标

指标名称	形式定义与统计口径	优化方向
均摊注意力成本（$\bar{A}$）	$(\sum_{\tau \in T_{\text{done}}} A(\tau) + \sum_{\tau' \in T_{\text{skip}}} I_{\text{reject}}(\tau')) / \lvert T_{\text{done}} \rvert$（式 3；将拒绝卡片计入分子防刷分）	越低越好
决策就绪时延	从事件到达或用户交代，到结果备妥进入 `Waiting` 状态的中位时间与 p95 尾延迟	越短越好
投机命中率	被采纳卡片（确认 + 修改后确认）÷（采纳数 + `Skipped` 数）；后台静默 `Discarded` 不计入	越高越好

指标名称	形式定义与统计口径	优化方向
主动误扰率	被点选“忽略”或事后标为多余的提醒数 ÷ 呈递到用户界面的主动提醒总数	越低越好
关键事件漏报率	用户事后认为本应处理却被过滤或试算静默 `Discarded` 的高价值事件占比	越低越好
跨设备连续召回率	跨设备追问或接力时，依托 `#T-id` 档案库准确还原结论与溯源的比例	越高越好
看门狗自愈率	遭遇超时、断网或界面阻塞的事项中，经看门狗自动恢复而无需人工接管的比例	越高越好
运行时不变量越界次数	整个评测轨迹中违反第 4.4 节七条运行时不变量的总次数	**必须恒等于零**

其中，“主动误扰率”与“关键事件漏报率”构成由式 (4) 和策略 P 调节的精确率-召回率权衡曲线，而“不变量越界次数 = 0”则是不可妥协的安全硬约束。

10.2 L0–L4 全五级解析注意力与搬运核算

为清晰拆解表 4 中每一个等级究竟消除了式 (2) 中的哪一部分开销，图 11 与表 10 对第 7.2 节的四类典型场景在 L0 至 L4 全五个等级下的用户操作步数（A）与跨端/跨应用手动搬运次数（I_{transfer}）进行了**透明、可逐跳核对的解析核算**。需要明确说明，表 10 是基于代表性场景的解析工作流核算而非大规模真实用户纵向实验；每一步计数的依据均在表中如实列出供同行审验。

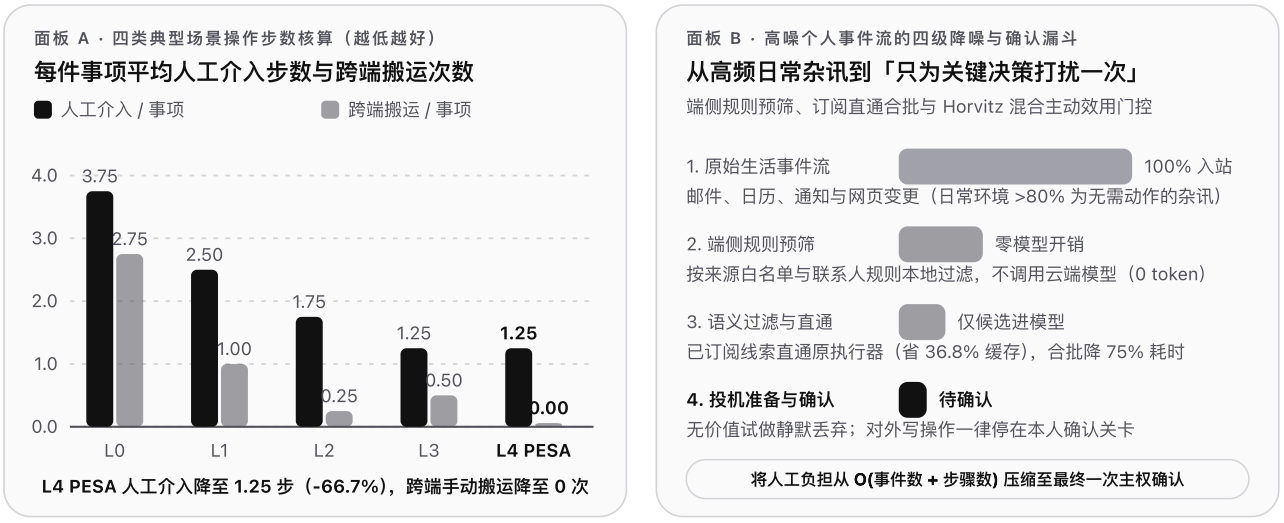


图 11　L0–L4 全五级解析注意力核算与四级门控漏斗。左：四类典型场景下每事项平均人工介入步数与跨端手动搬运次数；右：端侧规则预筛、线索直通合批与 Horvitz 效用门控漏斗。

表 10　四类典型场景在 L0–L4 全五级下的人工介入步数与手动搬运解析核算

典型场景（第 7.2 节）	L0 应答	L1 执行（单任务）	L2 并发	L3 主动（单端/无跨端协同）	L4 完整 PESA（跨设备生态）
01 · 客户会议改期邮件（核对日历并回复）	**4 步 / 3 搬运**（读邮件 → 查日历 → 贴入聊天拟稿 → 贴回邮箱发送）	**2 步 / 1 搬运**（复制邮件发起并等待 → 确认发送；合盖即停摆）	**2 步 / 0 搬运**（派发后台拟稿 → 确认；仍需人先看到邮件发起）	**1 步 / 0 搬运**（云端主动过滤拟稿，但需等用户打开主应用确认）	**1 步 / 0 搬运**（电脑合盖时云端查日历拟稿，Watch 抬腕点 1 下发送并四端同步撤回）
02 · 手机链接 → 电脑比价与内网填单	**5 步 / 4 搬运**（复制链接 → 多页比价 → 传电脑 → 逐项填表 → 提交）	**3 步 / 1 搬运**（传链接到电脑 → 发起并盯屏等填单 → 核对提交）	**2 步 / 1 搬运**（传链接到电脑派后台比价填单 → 核对提交）	**2 步 / 1 搬运**（单端主动比价；缺少分享入站与 Mac 租约仍须手动传电脑填单）	**2 步 / 0 搬运**（iPhone 分享立项 + Mac `ComputerHandsLease` 后台填妥 + iPad 分栏一键确认）
03 · 跨周线索合批与两周前原话召回	**4 步 / 3 搬运**（翻邮件群聊 → 贴多段入对话 → 追问 → 回翻核对）	**3 步 / 2 搬运**（分别调搜索 → 手动拼多源线索 → 追问原话）	**2 步 / 0 搬运**（后台汇总 → 同会话追问；换手机即丢上下文）	**1 步 / 1 搬运**（主动合批成摘要卡；手机追问电脑旧事仍需手动翻查搬运）	**1 步 / 0 搬运**（订阅执行器合批为 1 卡；手机随口追问沿 `#T-id` 逐跳调出精确原文）

典型场景（第 7.2 节）	L0 应答	L1 执行（单任务）	L2 并发	L3 主动（单端/无跨端协同）	L4 完整 PESA（跨设备生态）
04 · 总结夹带隐蔽外泄指令的外部网页	**2 步 / 1 搬运**（复制网页正文 → 贴入聊天窗口总结）	**2 步 / 0 搬运**（发起读取；缺工牌隔离与出站拦截，需盯防外泄）	**1 步 / 0 搬运**（交后台总结，最小工牌 `brief` 隔离隐私）	**1 步 / 0 搬运**（订阅页面更新主动总结，受工牌隔离保护）	**1 步 / 0 搬运**（最小工牌 + 出站带参拦截 + 确认关卡双重阻断，0 越权安全返回摘要）
四类场景均值	**3.75 步 / 2.75 搬运**	**2.50 步 / 1.00 搬运**	**1.75 步 / 0.25 搬运**	**1.25 步 / 0.50 搬运**	**1.25 步（-66.7%） / 0.00 搬运（-100%）**

表 10 与命题 3 共同揭示出三条清晰的架构规律：

1. **L0 → L2 消除了同步盯屏与同端复制粘贴，但仍卡在 $I_{\text{init}} \geq 1$。** 步数虽从 3.75 降至 1.75，用户仍须亲自发现每件事并手动发起。
2. **仅做 L3（单端主动）会暴露 0.50 次/事项的"跨设备断层"。** L3 已将介入步数压低至 1.25 步，但在天然跨设备的场景 02 与场景 03 中，缺少系统分享入站、桌面执行租约与跨端 `#T-id` 档案库，迫使用户继续承担平均 0.50 次跨端手动搬运。
3. **唯有 L4 完整 PESA 同时达成 1.25 步（-66.7%）与严格 0.00 搬运（-100%）。** 叠加零 token 端侧预筛、订阅直通（-36.8% 读取缓存 token [1]）与积压合批（-75% 均摊决策耗时 [1]），使人类注意力逼近命题 3 给出的主权下界。

10.3 拟议回放基准：PESA-Bench

为推动研究社区在统一条件下评测 L2 – L4 个人生态智能体，我们提出合成回放基准 **PESA-Bench**，由三套贴近真实生活的压力轨迹构成：

- **高噪多通道数周生活事件流：** 按时间戳交织回放 2 至 4 周邮件、日历、即时消息、关注网页更新与跨端指令，混入 >82% 无需动作的日常杂讯（促销、群聊闲聊、验证码、系统通知），同步考核投机命中率、误扰率与漏报率。
- **设备作息休眠、焦点争抢与跨端竞态点击轨迹：** 为四端注入早晚通勤、会议合盖、桌面办公与夜间专注的 $\alpha(d, t)$ 切换序列，并注入前台对话抢占 `ComputerHandsLease` 与 300 毫秒内 Watch/iPhone 连点同一卡片的竞态用例，检验式 (5) 的确定性。
- **间接提示注入与隐蔽外泄红队套件：** 依据 Greshake 等人 [6] 的攻击分类，在外部网页、HTML 邮件与 PDF 中埋入伪装指令、跨工牌隐私刺探与 Markdown 图片/URL 带参外泄陷阱；全量回放中出现任意一次不变量越界即判定不通过。

11. 参考实现：Piko 与运行时机制验证

PESA 并非纸上空谈，其核心底座正依托个人 AI 助手 **Piko**（覆盖 iPhone、iPad、Mac 与 Apple Watch）持续演进。为严格区分通用架构理论与当前已上线的工程现实，本节集中汇报已通过测试的运行时机制与后续路线。

11.1 生产运行时核心机制的确定性单元测试

针对 Piko 客户端已落地的 L2 后台并发、工牌隔离、桌面租约互斥与看门狗自愈底座，生产代码库维护了自动化确定性单元测试套件（`BackgroundWorkerTests`），六项核心机制均已验证通过（表 11）。

表 11　Piko 生产运行时核心机制的确定性单元测试验证（`BackgroundWorkerTests`）

验证机制项	被测生产模块与核心参数	守护的不变量与系统性质	测试状态
后台结果回流原对话	`BackgroundWorkers`（`maxRunning = 3`）	异步全双工执行，以唯一前台声音回流原对话（不变量 1）	**已通过**
任务级 `brief` 隔离	`WorkerTask.brief` 作用域边界	执行器仅获当前事项最小工牌，零无关主对话隐私与凭据（不变量 6）	**已通过**
桌面单持有者强互斥	`ComputerHandsLease.acquire`	任意时刻至多 1 个执行体操控本地 GUI，排队者先跑云端步骤（命题 2）	**已通过**

验证机制项	被测生产模块与核心参数	守护的不变量与系统性质	测试状态
180 秒闲置租约回收	`ComputerHandsLease`（`idleRelease = 180s`）	闲置满 180 秒或任务终止自动回收租约，消除后台死锁（不变量 5）	**已通过**
前台主对话优先插队	`ComputerHandsLease` 优先级队列	实时前台对话请求桌面操作时自动跃升队首（不变量 1、命题 2）	**已通过**
看门狗 45 秒巡检与三级升级	`BackgroundWorkers`（`45s / 20m / 60m` 或 `3` 次）	停滞 20 分钟自动注入恢复提示，满 60 分钟或 3 次失败透明转人工（不变量 5）	**已通过**

11.2 诚实能力自评与四阶段演进路线

对照第 3.5 节的 L0 – L4 标准严格自评：**Piko 当前的主体生产底座处于 L1 向 L2 全面收拢的阶段**，同时云端变更监控（“留意”，采用先确定性规则后模型理解的两级机制）等局部垂直路径已展现早期 **L3 原型特征**；但距离完整的 L4 生态级 PESA 仍有清晰工程距离（例如 Apple Watch 端目前主要作为只读进度镜像，尚未切换为表 7 目标架构中的版本化一键审批阀门）。

后续研发致力于把既有模块统一收拢到第 4 节的八态*事项*状态机下，全程严守向后兼容纪律——**只增不改（Add, Never Alter）**：每一阶段只做增量叠加与底层状态统一，绝不悄悄破坏用户既有的稳定习惯（表 12）。

表 12　Piko 通往完整 L4 PESA 的四阶段实施路线

演进阶段	核心工程交付范围	目标等级	当前状态
第一阶段	前后台异步全双工解耦：后台结果以唯一前台声音自动回流原对话；看门狗 45 秒巡检并最多 3 次自动自救	L2	**已完成**
第二阶段	统一*事项*底座：对话长任务无缝转交后台执行器；`ComputerHandsLease` 桌面互斥与前台插队；跨端 `#T-id` 同步；对话内进度条与三态视图	L2	**进行中**
第三阶段	日常信号接入统一流水线：邮件、日历与系统分享接入端云两级过滤与八态状态机，后台预备并呈递“确认 / 修改 / 忽略”建议卡	L3	**规划中**
第四阶段	全天候休眠韧性 L4 生态：云端状态机持续运转、静默推送唤醒与检查点续接、Apple Watch 版本化一键审批与四端毫秒级同步撤回、服务端看门狗闭环	L4	**规划中**

12. 局限性、有效性威胁与开放研究问题

12.1 局限性与有效性威胁

作为一篇系统架构与愿景论文，本文的现有结论受以下四项显式边界约束：

1. **解析步数核算与真实纵向部署的差距（外部有效性）。** 表 10 的步数与搬运降幅（3.75 → 1.25 步，2.75 → 0 搬运）来自对四类代表性场景的解析分解，而非数周规模的真实用户田野实验。现实中的净注意力收益高度依赖每个用户真实的误扰数 $|T_{\text{skip}}|$（式 3）以及第二级过滤对个人偏好的校准精度；完整实现 PESA-Bench 并开展纵向用户研究是后续必经验证。
2. **激进端侧规则过滤的冷启动漏报风险（精确率-召回率权衡）。** 在噪声比超过 80% 的环境下，第一级确定性规则能将云端路由负载降低 5× 以上，但严格的发件人白名单可能静默漏掉来自陌生新联系人（如医院通知或首次询盘客户）的紧急事件。在不牺牲端侧隐私的前提下缓解冷启动漏报，需结合关键词兜底与轻量端侧小模型语义初判。
3. **跨厂商操作系统权限壁垒（工程可部署性）。** 尽管 PESA 的形式化模型与平台无关，但后台静默唤醒、锁屏实时活动、系统分享入口与桌面无障碍操控权限在 Apple、Android、Windows 与 Linux 之间差异巨大，在跨厂商混用环境（如 iPhone 搭配 Windows 台式机）中会面临显著的工程适配成本。
4. **云端协调器的信任根假设（安全边界）。** 本文基准威胁模型假设云端状态机本身诚实且隔离（第 9 节）。若要抵御云端服务器自身被攻破的极端威胁，必须将策略校验与生物特征签名进一步下沉到终端硬件安全飞地（Secure Enclave）。

12.2 开放研究问题

将多端个人设备生态作为超级智能体的运行环境，开启了六个值得系统、HCI 与安全社区深入探索的方向：

- **跨越数年的个人事项记忆演进与可验证遗忘。** 当档案库跨年累积数万件事项后，如何自动归并陈旧因果链、消解动态演变的个人偏好，并支持按线索可密码学验证的彻底遗忘？
- **个性化打扰阈值 p^* 的小样本在线收敛。** 不同人及同一人在工作日与周末的打扰容忍度差异悬殊，如何仅靠稀疏的“确认 / 修改 / 忽略”反馈快速收敛个体化的 Horvitz 阈值（式 4）？
- **端侧小模型与云端大模型的能耗与隐私自适应切分。** 随手机与电脑 NPU 算力增强，第二级语义过滤、敏感实体脱敏与离线草稿预拟有多大比例适合下沉至端侧，切分点应如何随电量、温控与网络状态动态迁移？
- **保护主权的 PESA-to-PESA 跨主体协商协议。** 当用户的智能体与同事、家人或商家的智能体协商日程或议价时，双方如何在不暴露各自完整日历与价格底牌的前提下通过最小信息披露达成意向，再各自回到主人的确认关卡？
- **云端大脑与本地双手之间的零信任硬件边界。** 如何在极端云端失陷模型下，利用终端安全飞地签名与本地生物特征在场证明，保证本地执行器拒绝任何伪造的高危指令？
- **投机算力投入与人类注意力节省的帕累托最优。** 在给定每日预算 B_{spec} 下，何种调度策略能使每千个投机 token 换回的 $\Delta\,\bar{A}$ 最大化？

13. 结语

当大语言模型跨过“可靠完成单点步骤”的门槛之后，制约个人生产力的真正天花板便不再是模型本身，而是**人的有限注意力每天被切碎了多少次**。只要智能体依然被困在孤立的会话框里，或在桌面上与人争夺同一只鼠标光标，技术越进步，人反而越忙于充当跨应用搬运工与全职监工。

PESA 把这层繁琐的协调工作彻底沉入系统架构之中：以个人全部设备生态为统一运行环境，将工作单位从“对话”提升为“事项”，将执行能力（κ_{exec}）与送达表面（κ_{surf}）正交解耦，用有界时间租约消除桌面焦点冲突，并用版本化确认关卡与最小工牌隔离守住安全底线。由此，智能体不再是一个停在屏幕上等你敲字提问的输入框，而是一套安静、可信、贯通既有设备的常驻系统——把发现、准备与恢复留在幕后，只在真正需要你拍板的时刻恰到好处地出现在最顺手的那块屏幕上。

附录 A　Super-agent 核心机制与 PESA 的架构演进对应

表 A1 总结了 PESA 从 DeepSeek Super-agent [1] 继承的核心系统机制，以及当运行环境从软件研发团队迁移至单一个人的异构多端生态后所做的关键架构演进。

表 A1　Super-agent 机制与 PESA 个人生态架构的演进对应

Super-agent 中的机制	PESA 中的对应机制	面向个人设备生态的核心演进
常驻路由层 + 并发执行层	常驻前台协调器 + 隔离后台执行器	收束为单一个人主权边界；新增单消息多意图拆分、跨渠道合并与乐观锁 `version` 多端仲裁
环境事件驱动触发	日常信号（邮件、日历、通知、分享、网页）	针对 >80% 个人杂讯前置“端侧零开销规则 + 云端语义过滤”两级漏斗；来源默认关闭，无把握默认静默
线索订阅与直通路由	执行器订阅线索（$\text{sub}(w) \subseteq H$）	扩展为线索与事项多对多映射，并支持 `Done` 态事项在后续回信到达时自动激活与 `#T-id` 链式续接
上下文当缓存、按编号召回	三层记忆与跨设备事项档案库	召回单位升级为跨设备*事项*（`#T-id`），每条归档强制保留“摘要/原文、溯源、置信度”三要素
投机执行	零副作用后台准备 + 确认建议卡	结合 Horvitz 双阈值门控（式 4）；对外写操作一律停在 `Waiting`；严格二分系统 `Discarded` 与用户 `Skipped`
异步全双工交互	后台并发推进 + 前台轻快响应	结果既可回流对话，亦可由算法 3 路由至灵动岛、手表或待办建议卡；全程随时可插话或停止

Super-agent 中的机制	PESA 中的对应机制	面向个人设备生态的核心演进
存算分离、执行就地保留	云端大脑 + 本地端侧执行器	端侧执行器为个人自有异构设备；引入 `ComputerHandsLease` 单租约互斥、前台插队、180s 回收与检查点续接
外部独立看门狗	45 秒巡检、先自救后升级看门狗	停滞 20 分钟注入诊断恢复提示（至多 3 次）；满 60 分钟或 3 次失败透明转 `Waiting (intervention)`
群聊与代码评审（PR）呈现	四端生态原生执行与送达面	正交拆分执行设备选择（算法 2，κ_{exec}）与送达面选择（算法 3，κ_{surf}），支持多端同步撤回

附录 B　设计取舍与常见问题

PESA 与 DeepSeek Super-agent 是什么关系?

PESA 直接承接并扩展了 Super-agent 的核心系统工程思想：打破单次会话边界、前后台双层解耦、事件驱动与订阅直通、存算分离以及独立外部看门狗 [1]。两者的根本分水岭在于**运行环境与服务主体**：Super-agent 面向软件研发团队的群聊、代码库与常开云端开发机；PESA 面向单一个人的异构多端设备（手机、平板、电脑、手表）与个人应用生态。正是这一迁移催生了“生态原生”与“可信”两项新特征，以及端侧降噪预筛、`ComputerHandsLease` 桌面租约、正交双选算法和防隐蔽 URL 外泄三层防线。

PESA 与电脑操控智能体（Computer Use / Browser Agent）或 MCP 插件有什么区别?

电脑操控智能体、浏览器自动化工具及 MCP 等连接器协议主要工作在 **L1（执行）层**，解决的是“端侧执行器手头能调用哪些工具（即扩展 κ_{exec}）”。若直接让它们运行在用户主电脑上，不仅会与人争抢鼠标键盘焦点，而且合盖即停摆。PESA 是位于其上的 **L4 个人生态系统架构**：把浏览器与桌面操控统一纳入受 `ComputerHandsLease` 仲裁的本地“双手”，用云端状态机保障合盖续跑，用两级过滤与版本化确认关卡把控何时启动与由谁拍板。

PESA 必须构建在苹果生态上吗?

不需要。PESA 的形式化模型、闭合八态生命周期、七条运行时不变量与三项调度算法完全独立于具体操作系统，适用于任何手机、平板、电脑与穿戴设备组合（Android、Windows、Linux 或跨平台混用）。参考实现 Piko 率先选择苹果生态（iPhone、iPad、Mac、Apple Watch），是因为四类设备的人体工学分工清晰，且具备灵动岛、ActivityKit 实时活动、系统分享菜单与接力等成熟原生交互接口。

PESA 会在未经允许的情况下擅自发消息、填表提交或付款吗?

默认绝不会。在公理 1、不变量 2 与命题 1 的约束下，任何改变外部世界状态的写操作——发邮件、发即时消息、提交网页表单、公开发布或支付——都必须停在 `Waiting (approval)` 关卡，等待底层执行器核验本人针对当前快照版本 τ.version 签发的确认令牌后才会发出（除非用户在策略白名单 A_{auto} 中显式预先授权了特定低风险动作）。

手机锁屏、电脑合盖离线后，交代的任务还会继续吗?

会继续。因推理“大脑”与档案库状态机常驻云端，所有无需本地界面的步骤（检索资料、核对日历、跨源比价、拟定回复草稿）在设备休眠期间照常推进。只有必须操作特定本地设备界面的步骤才会按算法 2 尝试静默推送唤醒，或在设备彻底离线时安全保存在检查点进入 `Parked`，待设备重新上线核验页面状态后从断点续接。

如何客观判定一个智能体的 PESA 等级（L0–L4）？

严格按表 4 的系统架构标准判定，与模型参数量无关：单窗口问答为 L0；人工发起并同步等待单任务工具调用为 L1；前后台解耦且多任务后台并发为 L2；接入日常事件流、具备两级过滤与建议卡投机准备为 L3；在此之上贯通全部个人设备、具备共享 `#T-id` 档案库、休眠续跑、桌面租约互斥、各端原生送达面与一处确认多端撤回的，才是完整的 L4 PESA。

Piko 目前已经达到完整的 L4 PESA 了吗?

还没有。如第 11 节坦诚报告的，Piko 当前已上线的主体生产底座处于 L1 向 L2 收拢阶段（`BackgroundWorkers`、`ComputerHandsLease`、最小工牌隔离与 45 秒看门狗均已通过表 11 的确定性单元测试），云端“留意”等局部垂直路径已具 L3 原型特征。表 12 给出了通往完整 L4 统一底座的后续演进阶段。

声明

作者与机构	Ruiming Zhao，Prysai Lab（ORCID: 0009-0007-8335-5067）。
利益关系	作者为 Prysai Lab 创始人；第 11 节报告的参考实现 Piko 由 Prysai Lab 研发。
数据与验证	本文是一篇系统架构与愿景论文。第 2 节引用的系统实测数据均源自所注明已发表文献；第 10.2 节给出透明可审验的 L0 – L4 解析步数核算，第 11.1 节报告 Piko 生产运行时已通过的确定性单元测试。
修订记录	2026-11-15 首发（v1.0，Zenodo DOI: 10.5281/zenodo.24424696）；2026-11-21 修订（v1.1，Zenodo DOI: 10.5281/zenodo.24429625）；2026-11-24 修订（v1.2，学术结构重构、新增形式化命题与证明思路、Horvitz 阈值推导、局限性与有效性威胁专节、三语母语级重写及出版级排版升级）。
许可	CC BY 4.0（https://creativecommons.org/licenses/by/4.0/）。本文可自由引用与转载，请注明出处并附 DOI（https://doi.org/10.5281/zenodo.24429625）及原文链接。

引用本文

推荐引用	Zhao, Ruiming. PESA：个人生态超级智能体. Piko Research, Prysai, 2026-11-15 (v1.2, 2026-11-24). https://piko.prysai.com/zh-Hans/research/pesa · DOI: https://doi.org/10.5281/zenodo.24429625
APA	Zhao, R. (2026). PESA：个人生态超级智能体 (Version 1.2) [Vision paper]. Piko Research, Prysai. https://doi.org/10.5281/zenodo.24429625
GB/T 7714	ZHAO R. PESA：个人生态超级智能体[R/OL]. Piko Research, Prysai, 2026-11-15. https://piko.prysai.com/zh-Hans/research/pesa. DOI: 10.5281/zenodo.24429625.
DOI	DOI: 10.5281/zenodo.24429625
引用文件	PDF · Markdown (.md) · BibTeX (.bib) · RIS (.ris) · CSL-JSON (.csl.json) · CITATION.cff

```bibtex
@techreport{zhao2026pesa,
  title        = {PESA：个人生态超级智能体},
  author       = {Zhao, Ruiming},
  institution  = {Prysai},
  type         = {Vision paper},
  series       = {Piko Research},
  year         = {2026},
  month        = nov,
  doi          = {10.5281/zenodo.24429625},
  url          = {https://piko.prysai.com/zh-Hans/research/pesa},
  language     = {zh-Hans},
  note         = {Version 1.2, revised 2026-11-24}
}
```

参考文献

1. Qiao, Yujia 等（DeepSeek Harness 团队）. *Super-agent 设计探索：迈向更自主的 Agent 系统*. DeepSeek 技术报告, 2026 年 4 月. github.com/deepseek-ai/dsh-superagent-prototype
2. Mark, Gloria, Gudith, Daniela, and Klocke, Ulrich. *The Cost of Interrupted Work: More Speed and Stress*. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08), pp. 107 – 110, ACM, 2008; doi:10.1145/1357054.1357072. doi.org/10.1145/1357054.1357072
3. Shneiderman, Ben. *Human-Centered Artificial Intelligence: Reliable, Safe & Trustworthy*. International Journal of Human – Computer Interaction 36(6): 495 – 504, 2020; doi:10.1080/10447318.2020.1741118. doi.org/10.1080/10447318.2020.1741118
4. Kung, H. T., and Robinson, John T. *On Optimistic Methods for Concurrency Control*. ACM Transactions on Database Systems (TODS) 6(2): 213 – 226, June 1981; doi:10.1145/319566.319567. doi.org/10.1145/319566.319567
5. Gray, Cary G., and Cheriton, David R. *Leases: An Efficient Fault-Tolerant Mechanism for Distributed File Cache Consistency*. Proceedings of the 12th ACM Symposium on Operating Systems Principles (SOSP '89) / ACM SIGOPS Operating Systems Review 23(5): 202 – 210, December 1989; doi:10.1145/74850.74870. doi.org/10.1145/74850.74870
6. Greshake, Kai, Abdelnabi, Sahar, Mishra, Shailendra, Endres, Christoph, Holz, Thorsten, and Fritz, Mario. *Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection*. Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec '23), pp. 79 – 90, ACM, 2023; doi:10.1145/3605764.3623985; arXiv:2302.12173. arxiv.org/abs/2302.12173
7. Yao, Shunyu, Zhao, Jeffrey, Yu, Dian, Du, Nan, Shafran, Izhak, Narasimhan, Karthik, and Cao, Yuan. *ReAct: Synergizing Reasoning and Acting in Language Models*. International Conference on Learning Representations (ICLR), 2023; arXiv:2210.03629. arxiv.org/abs/2210.03629
8. Schick, Timo, Dwivedi-Yu, Jane, Dessì, Roberto, Raileanu, Roberta, Lomeli, Maria, Hambro, Eric, Zettlemoyer, Luke, Cancedda, Nicola, and Scialom, Thomas. *Toolformer: Language Models Can Teach Themselves to Use Tools*. Advances in Neural Information Processing Systems (NeurIPS) 36: 68539 – 68551, 2023; arXiv:2302.04761. arxiv.org/abs/2302.04761
9. Wang, Lei, Ma, Chen, Feng, Xueyang, Zhang, Zeyu, Yang, Hao, Zhang, Jingsen, Chen, Zhiyuan, Tang, Jiakai, Chen, Xu, Lin, Yankai, Zhao, Wayne Xin, Wei, Zhewei, and Wen, Jirong. *A Survey on Large Language Model based Autonomous Agents*. Frontiers of Computer Science 18(6): 186345, 2024; doi:10.1007/s11704-024-40231-1; arXiv:2308.11432. arxiv.org/abs/2308.11432
10. Zhou, Shuyan, Xu, Frank F., Zhu, Hao, Zhou, Xuhui, Lo, Robert, Sridhar, Abishek, Cheng, Xianyi, Ou, Tianyue, Bisk, Yonatan, Fried, Daniel, Alon, Uri, and Neubig, Graham. *WebArena: A Realistic Web Environment for Building Autonomous Agents*. International Conference on Learning Representations (ICLR), 2024; arXiv:2307.13854. arxiv.org/abs/2307.13854
11. Xie, Tianbao, Zhang, Danyang, Chen, Jixuan, Li, Xiaochuan, Zhao, Siheng, Cao, Ruisheng, Hua, Toh Jing, Cheng, Zhoujun, Shin, Dongchan, Lei, Fangyu, Liu, Yitao, Xu, Yiheng, Zhou, Shuyan, Savarese, Silvio, Xiong, Caiming, Zhong, Victor, and Yu, Tao. *OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments*. Advances in Neural Information Processing Systems (NeurIPS) 37, Datasets and Benchmarks Track, 2024; arXiv:2404.07972. arxiv.org/abs/2404.07972
12. Wu, Qingyun, Bansal, Gagan, Zhang, Jieyu, Wu, Yiran, Li, Beibin, Zhu, Erkang, Jiang, Li, Zhang, Xiaoyun, Zhang, Shaokun, Liu, Jiale, Awadallah, Ahmed Hassan, White, Ryen W., Burger, Doug, and Wang, Chi. *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation*. First Conference on Language Modeling (COLM), 2024; arXiv:2308.08155. arxiv.org/abs/2308.08155
13. Park, Joon Sung, O'Brien, Joseph C., Cai, Carrie J., Morris, Meredith Ringel, Liang, Percy, and Bernstein, Michael S. *Generative Agents: Interactive Simulacra of Human Behavior*. Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23), Article 2, pp. 1 – 22, ACM, 2023; doi:10.1145/3586183.3606763; arXiv:2304.03442. arxiv.org/abs/2304.03442
14. Packer, Charles, Wooders, Sarah, Lin, Kevin, Fang, Vivian, Patil, Shishir G., Stoica, Ion, and Gonzalez, Joseph E. *MemGPT: Towards LLMs as Operating Systems*. arXiv:2310.08560, 2023. arxiv.org/abs/2310.08560
15. LofiSu. *deepseek-super-agent：DeepSeek Super-agent 架构开源实现*. GitHub 开源仓库, 2026. github.com/LofiSu/deepseek-super-agent
16. thefullgc. *opensuperagent：事件驱动 Super-agent 社区开源实现*. GitHub 开源仓库, 2026. github.com/thefullgc/opensuperagent
17. Weiser, Mark. *The Computer for the 21st Century*. Scientific American 265(3): 94 – 104, September 1991; doi:10.1038/scientificamerican0991-94. doi.org/10.1038/scientificamerican0991-94
18. Weiser, Mark, and Brown, John Seely. *Designing Calm Technology*. PowerGrid Journal 1.01, July 1996 (reprinted in Beyond Calculation: The Next Fifty Years of Computing, pp. 75 – 85, Springer, 1997). calmtech.com/papers/designing-calm-technology
19. Dey, Anind K. *Understanding and Using Context*. Personal and Ubiquitous Computing 5(1): 4 – 7, 2001; doi:10.1007/s007790170019. doi.org/10.1007/s007790170019
20. Horvitz, Eric. *Principles of Mixed-Initiative User Interfaces*. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99), pp. 159 – 166, ACM, 1999; doi:10.1145/302979.303030. doi.org/10.1145/302979.303030

21. Hennessy, John L., Patterson, David A. *Computer Architecture: A Quantitative Approach（第 6 版）*. Morgan Kaufmann, Cambridge, MA, 2017; ISBN 978-0-12-811905-1.

22. Apple Inc. *ActivityKit：以实时活动呈现实时数据*. Apple Developer 技术文档. developer.apple.com/documentation/activitykit

23. Apple Inc. *人机界面指南（Human Interface Guidelines）*. Apple Developer 技术文档. developer.apple.com/design/human-interface-guidelines/

24. SAE International. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. SAE Standard J3016_202104, April 2021. www.sae.org/standards/content/j3016_202104/

25. Saltzer, Jerome H., and Schroeder, Michael D. *The Protection of Information in Computer Systems*. Proceedings of the IEEE 63(9): 1278 – 1308, September 1975; doi:10.1109/PROC.1975.9939. doi.org/10.1109/PROC.1975.9939

iPhone、iPad、Mac、Apple Watch、Siri 与 Apple Pencil 是 Apple Inc. 在美国及其他国家和地区注册的商标。Piko 是 Prysai Lab 的独立产品，与 Apple Inc. 没有隶属关系。