PIKO RESEARCH 技術報告 · PIKO-RESEARCH-2026-01

PESA：個人生態超級智能體

面向跨裝置個人生態的系統架構、形式化模型與注意力經濟學。

作者：Ruiming Zhao ORCID
Prysai Lab

首發：2026-11-15 版本：1.2 · 修訂 2026-11-24 閱讀：約 25 分鐘

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

摘要

單任務語言模型智能體如今已能呼叫外部工具、撰寫程式碼並操作圖形介面，獨立完成日益複雜的單點工作。然而在日常個人使用中，底層模型變強並未自然換來使用者的輕鬆。由於現有個人 AI 工具仍被困在孤立的對話視窗或單一裝置的螢幕裡，使用者依然必須親自留意新訊息、在電子郵件、行事曆、瀏覽器與多個聊天視窗之間反覆搬運上下文，並在智能體接管桌面操作時全程盯著螢幕看守。

本文提出並形式化定義 **PESA（Personal Ecosystem Super Agent，個人生態超級智能體）**：一種服務於單一個人主權、以個人全部裝置生態（手機、平板、電腦與手錶）為執行環境、以「事項（Thing）」而非「對話（Session）」為基本工作單位、在每台裝置上依平台原生習慣協同執行，且在任何對外寫入操作前預設等待本人明確確認的常駐超級智能體。在承繼常駐雙層 Super-agent 架構 [1] 的基礎上，本文圍繞裝置、事件流、線索、事項、後台執行器與使用者策略建立形式化模型，推導出四分量人類注意力成本模型、面向高雜訊個人事件流的 Horvitz 混合主動投機門檻、閉合八態生命週期與跨端樂觀併發提交規則，並提出七條可驗證的執行期系統不變量。在此基礎上，我們證明了對外副作用安全性與跨端冪等性（命題 1）、單桌面焦點租約的無死鎖與有限等待特性（命題 2），以及主權約束下人類注意力的不可壓縮下界（命題 3）。

在系統工程層面，本文提出完整的端對端參考架構——涵蓋單一常駐前台協調器、任務級最小權限隔離的後台執行器、跨裝置三層**事項檔案庫（Thing Archive）**、端雲兩級雜訊過濾、單桌面互斥租約（`ComputerHandsLease`）、獨立自癒看門狗，以及三項確定性優先排程演算法（`route`、`choose_device`、`choose_surface`）——並建立抵禦間接提示注入與隱蔽 URL 參數外洩的縱深防禦模型。透過在四類高頻跨裝置場景上進行可逐跳核對的解析注意力核算，我們揭示僅停留在單端主動（L3）仍會殘留平均每事項 0.50 次跨裝置手動搬運，而完整生態架構（L4 PESA）可將人工介入自 L0 的 3.75 步降至 1.25 步（降低 66.7%），並將手動上下文搬運徹底歸零（降低 100%）。最後，本文提出合成回放基準 **PESA-Bench**，報告參考實作 Piko 生產執行期中六項核心機制的確定性單元測試驗證，並系統性探討方法論侷限、有效性威脅與開放研究問題。

關鍵詞：個人生態超級智能體（PESA）；個人 AI 智能體；以人為本的 AI；跨裝置協同系統；注意力經濟學；混合主動互動；投機執行；樂觀併發控制；桌面租約仲裁；間接提示注入

1. 引言

當代基礎模型已能讀懂長篇文件、撰寫可運作的程式碼，也能模擬滑鼠與鍵盤操作軟體介面。但一個反直覺的現象正在發生：模型越聰明，個人使用者往往反而越忙碌。為了妥善辦完一件稍具複雜度的日常事務——例如協調一次客戶會議改期、多方比價後填寫公司採購單，或是追蹤一場歷時三週的專案談判——使用者常得在手機與電腦上同時開啟四、五個互不相通的聊天分頁。每個分頁都是封閉的沙盒，既不知道另一個視窗談過什麼，也不知道十分鐘前在另一台裝置上做過哪些決定。結果，人成了系統之間唯一的黏合劑：親自發現待辦、把零散脈絡複製拼成提示詞、在工具運作時緊盯螢幕監看，做完後再把結果貼回電子郵件或業務系統。

DeepSeek Harness 團隊在 Super-agent 技術報告中指出，受限於人類工作記憶與情境切換代價，單一使用者在同一時段通常僅能有效跟進 3 至 5 個並行會話 [1]。在以「單次對話」為中心的架構下，**底層模型決定了單輪回覆的品質上限，但人的注意力決定了一天能推進多少件事的吞吐上限**。人機互動領域的經典實證早已點出這種碎片化打斷的隱性成本：Mark、Gudith 與 Klocke 於 CHI 2008 的實驗室研究證實，當知識工作者頻繁被打斷並被迫切換任務時，人們會本能地加快操作節奏以彌補切碎的時間，雖然總耗時表面上未大幅拉長，代價卻是顯著攀升的認知負荷、時間壓力、挫折感與失誤率 [2]。在郵件、行事曆、瀏覽器分頁與多個 AI 視窗之間來回跳轉，本質上正是高強度的持續自我打斷。

> 直觀理解。更強的大語言模型解決了「當你把前因後果交代清楚後，如何把這一步做好」的問題；但它尚未解決「誰替你留意日常動態、誰將脈絡跨裝置帶著走、如何防止後台任務爭搶你的滑鼠，以及如何在真正送出前穩穩停下請你拍板」的問題。PESA 正是為了補上這層缺失的系統架構而生。

1.1 當前個人智能體落地的四重結構性瓶頸

2024 至 2026 年間，個人 AI 工具迅速從純文字問答演進至工具呼叫、瀏覽器與桌面自動化（Computer-Use Agents）以及獨立 AI 硬體。然而在長期高頻日常使用中，既有路線反覆遭遇四道結構性高牆：

1. **提示詞冷啟動與上下文搬運稅。** 現實生活中的事務幾乎不會始於空白輸入框，而是始於一封客戶改期郵件、一則航班延誤通知、一張手機拍下的報價單，或群組訊息中的行程變動。只要使用者必須手動蒐集這些碎片、整理成提示詞再把產出搬回原應用程式，協調本身的摩擦便常高於親手處理的力氣，導致日常留存率迅速下滑。
2. **單桌面焦點爭搶與「盯螢幕監工」疲勞。** 桌面操控智能體雖能點擊按鈕、填寫表單，但一台個人電腦僅有一組實體滑鼠鍵盤與一個前台作用中視窗。當智能體操作桌面時，使用者不僅無法同時用電腦處理其他工作，還因擔心誤點、覆寫檔案或提前送出而不得不全程盯著螢幕看守。同步獨佔螢幕的互動方式，將人的注意力與執行時間牢牢綁在一起。
3. **獨立 AI 硬體孤島與既有裝置生態的割裂。** 試圖繞過既有作業系統的獨立 AI 掛件或掌上硬體之所以屢屢受挫，原因相當樸素：人們早已擁有一套分工明確、全天相伴的裝置群——口袋裡的手機、手腕上的手錶、用於閱讀與對照的平板、書桌上的電腦。使用者欠缺的並非一塊沒有歷史資料與應用程式生態的新螢幕，而是能讓單一持續在線的智能體進駐既有裝置群的系統架構。
4. **主動性與信任感的兩難。** 純被動助理在電腦闔蓋、手機鎖定螢幕的瞬間隨即停擺；但若盲目開啟主動推播，每天湧入的數百則行銷郵件、群組閒聊與應用程式通知便會化為惱人的干擾雜訊——更嚴重的是，若系統未經本人核對即以使用者真實身分寄出郵件、送出表單或執行付款，任何一次越權失誤都將導致信任徹底瓦解。

1.2 研究問題與主要貢獻

本文是一篇系統架構與願景論文，聚焦於三項核心研究問題：

- **RQ1（抽象模型與注意力）：** 應採用何種系統抽象與能力分級，將跨裝置個人生態智能體與對話框工具、單機自動化區隔開來？每一級架構演進實際消除了哪一部分人類注意力成本？
- **RQ2（系統協同與併發）：** 單一常駐智能體如何銜接持續在線的雲端推理與間歇休眠的本機裝置，在單台電腦上避免多個後台任務爭搶桌面焦點，並在多塊螢幕間杜絕重複觸發對外動作？
- **RQ3（信任邊界與雜訊紀律）：** 主動式個人智能體如何在高雜訊日常事件流中既不空耗雲端算力、又不頻繁打擾使用者，同時防範外部網頁與郵件中的間接提示注入與隱蔽資料外洩？

針對上述問題，本文提出五項主要貢獻：

1. **概念架構與 L0 – L4 能力分級（第 3 節）。** 正式定義 PESA（Personal Ecosystem Super Agent），依循以人為本 AI 之二維框架 [3] 闡述三項設計命題與六項系統特質，建立與模型參數量解耦、依系統分擔協調範圍界定之 L0 – L4 分級標準。
2. **形式化模型、混合主動決策邊界與理論保證（第 4 節）。** 將裝置能力正交拆解為執行能力與送達能力 $\kappa(d) = \langle \kappa_{\mathsf{exec}}(d), \kappa_{\mathsf{surf}}(d) \rangle$；建構四分量注意力成本方程與高雜訊環境下之 Horvitz 投機門檻；設計閉合八態生命週期與基於樂觀併發控制 [4] 的版本化原子提交規則；證明對外副作用安全性、桌面租約無死鎖性與主權注意力下界三條核心命題。
3. **跨裝置參考架構與確定性優先排程演算法（第 5 – 6 節）。** 設計單一常駐前台協調器、任務級隔離後台執行器、跨裝置三層事項檔案庫、斷線檢查點續接、有界時間互斥租約（`ComputerHandsLease`）[5] 及獨立自癒看門狗，並給出事件路由（`route`）、執行裝置選擇（`choose_device`）與情境送達面選擇（`choose_surface`）三項低開銷演算法及複雜度分析。
4. **生態原生互動、容錯場景與縱深防禦體系（第 7 – 9 節）。** 確立四類裝置形態之人體工學分工，詳述四類高頻跨裝置場景的正常閉環與異常復原路徑，說明八態如何摺疊為極簡三態視圖，並建立防範間接提示注入與隱蔽 URL 外洩的

三層防線 [6]。

5. **評估指標、L0 – L4 解析核算與參考實作驗證（第 10 – 12 節）。** 提出八項正交評估指標與合成回放基準 **PESA-Bench**；在四類場景上逐跳核算 L0 至 L4 之人工介入與跨端搬運（量化說明為何僅止於 L3 單端主動仍會殘留跨端搬運）；報告參考系統 **Piko** 六項生產執行期機制的確定性單元測試結果，並坦誠剖析本文之方法論侷限與有效性威脅。

2. 背景與相關研究

PESA 匯集了智能體系統、作業系統、分散式系統與人機互動六條脈絡的研究成果。釐清每一條脈絡解決了什麼問題、又在個人日常生活中留下哪些空白，有助於明確界定 PESA 的系統定位。

2.1 從對話助理到單任務工具與桌面智能體

早期對話助理採行受限於單一視窗的無狀態輪詢互動。ReAct [7] 透過交替生成推理軌跡與環境動作，使語言模型能依外部觀測動態修正計畫；Toolformer [8] 證實模型可經自監督學習掌握外部 API 呼叫時機。相關綜述系統性回顧了此一典範如何發展為程式編寫智能體、瀏覽器智能體及 MCP 等標準化工具連接生態 [9]。

然而進入真實個人環境後，獨立單任務智能體仍與人類表現存在顯著鴻溝。在涵蓋 812 項真實網站長程任務的 WebArena 基準中 [10]，人類完成率為 78.24%，而當時 GPT-4 智能體僅為 14.41%；在橫跨 Ubuntu、Windows 與 macOS 共 369 項真實桌面任務的 OSWorld 基準中 [11]，人類完成率為 72.36%，發布時最佳多模態智能體僅為 12.24%。即便視覺定位與長鏈規劃模型持續進步，個人日常任務仍反覆受阻於三道非單純放大模型規模所能跨越的高牆：跨裝置登入狀態過期與雙重驗證、關鍵脈絡散落於另一台裝置或另一款應用程式，以及不可逆的末端對外動作（寄信、送單、付款）絕不容許系統靜默出錯。

2.2 多智能體協作、分層記憶與常駐 Super-agent

AutoGen [12] 等多智能體框架支援專職角色透過可程式化對話分工協作，但協同生命週期大多仍繫於單次會話。Generative Agents [13] 為虛擬社群角色建構依近期性、重要性與關聯性檢索之連續記憶流與反思樹，惟沙盒環境不存在真實財務責任與外部攻擊者。MemGPT [14] 借鏡作業系統虛擬記憶體概念，將有限上下文視窗視為「主記憶體」，經顯式函式呼叫於主記憶體與外部歸檔儲存之間換入換出歷史紀錄。PESA 的事項檔案庫直接承襲此一分層快取洞見，但將索引與召回的基本單位自零散聊天片段提升為具因果關聯鏈的跨裝置**事項**（`#T-id`）。

DeepSeek Harness 團隊提出的 Super-agent 架構 [1] 跨出了打破單次會話框架的關鍵一步：引入常駐路由層（Router）、後台併發執行層（Worker）、事件驅動觸發、運算儲存分離與外部看門狗，並衍生出 `deepseek-super-agent` [15]、`opensuperagent` [16] 等開源實作。然而 Super-agent 的原生目標環境是**軟體研發團隊**——依託企業群組聊天、程式碼託管平台與 24 小時常開的雲端開發主機。將其遷移至**單一個人之日常生活生態**時，三類全新系統約束隨即浮現：① **個人雜訊放大效應**：日常手機通知、群組訊息與行銷郵件中逾 80% 無須任何動作；② **裝置間歇休眠與單桌面焦點衝突**：個人筆電隨時闔蓋收入背包，且一台電腦僅有一組實體滑鼠鍵盤；③ **個人主權與法律責任**：寄出的每封信、送出的每份表單皆以本人實名承擔後果，絕不容許未經授權的自主外發。

2.3 分散式系統原語與平靜人機互動基礎

PESA 的核心協調機制並非權宜啟發式規則，而是深植於分散式系統與人機互動領域之經典理論：

- **有界時間租約（Leases）。** Gray 與 Cheriton 於 SOSP 1989 提出「租約」機制 [5]：透過授予具逾時自動回收語意的獨佔控制權，確保持有者當機或失聯時系統得免人工介入自動恢復活性。PESA 以 `ComputerHandsLease` 將此經典原語精準應用於個人電腦實體桌面焦點的互斥仲裁。
- **樂觀併發控制（OCC）。** Kung 與 Robinson 於 ACM TODS 1981 指出 [4]，當寫入衝突遠少於讀取時，於提交階段驗證版本快照可避免沉重的分散式鎖定開銷。PESA 據此落實跨手機、手錶、平板與電腦之版本化原子確認，徹底杜絕快速連點導致的重複寄信或重複扣款。
- **平靜技術、情境感知與以人為本 AI。** Weiser [17] 與 Weiser 及 Brown 之平靜技術理論 [18] 主張，日常運算應安居於注意力邊緣，僅在必要時平滑移入中心；Dey [19] 對「情境（Context）」提出經典形式化定義，奠定依使用者當下狀態

匹配送達面之理論基礎；Horvitz [20] 以貝氏期望效用刻畫混合主動介面，於協助收益與干擾成本間進行量化權衡；Shneiderman [3] 更論證「高度機器自動化」與「高度人類控制權」乃兩條正交維度而非一維拉鋸——此正為 PESA「後台深度投機準備、末端由本人拍板確認」之核心立論。

- **投機執行與作業系統原生框架。** 現代超純量處理器於分支條件未決前先在重排序緩衝區投機執行，僅於命中時原子提交架構狀態 [21]。同時，Siri、App Intents、捷徑（Shortcuts）、Handoff 與 ActivityKit 鎖定畫面即時動態 [22][23] 等原生框架雖橫跨多裝置，卻仍以短鏈路同步指令為主，缺乏跨週存活的雲端非同步事項狀態機。PESA 並不與作業系統原生入口競爭，而是將其納為感知與送達能力層。

表 1　PESA 與主流智能體架構的多維度定位比較

比較維度	對話式助理	單任務工具 / 操控智能體	團隊級 Super-agent	PESA（個人生態超級智能體）
基本工作單位	單次獨立對話（Session）	單次指令任務（Task）	研發事件線索（Thread）	**具體個人事項（Thing, `#T-id`）**
執行與感知環境	單一聊天視窗	單台電腦或瀏覽器沙盒	團隊群聊、程式碼庫與雲端主機	**個人全部裝置（手機/平板/電腦/手錶）與應用**
觸發與主動性	純被動等待輸入	人工發起、同步執行	團隊成員與研發流水線事件	**使用者意圖 + 經端雲兩級過濾之日常事件流**
併發與回應性	單視窗阻塞	執行時常獨佔螢幕焦點	常駐路由 + 後台併發執行器	**輕快常駐前台 + 隔離併發後台執行器**
離線韌性與焦點互斥	關分頁即停	闔蓋中斷、多任務搶滑鼠	雲端常開、無單桌面焦點衝突	**雲端續跑 + 靜默喚醒 + `ComputerHandsLease` 單租約互斥**
記憶與跨端召回	困於單段聊天紀錄	侷限於當次執行上下文	單平台內依編號檢索	**三層記憶 + 跨裝置事項檔案庫（`#T-id` 逐跳召回）**
送達與互動面	統一對話清單	終端日誌或浮動操控窗	群組訊息與程式碼審查（PR）	**依 κ_{surf} 匹配情境：動態島、Watch、iPad 分欄、卡片**
對外安全與憑證隔離	人工手動複製外發	憑證常暴露於上下文	依團隊 CI/CD 權限自動執行	**憑證保險庫封存 + 最小工牌隔離 + 預設版本化確認關卡**

來自 Super-agent 的系統實測啟示與適用邊界。 DeepSeek Harness 報告中的三組實測數據直接啟發了 PESA 的流水線設計 [1]：① 前後台雙層併發使任務尾延遲（p95）自單會話中性基線的 29.5 分鐘降至 15.7 分鐘（縮短 46.8%，中位數 12.4 分鐘）；② 啟用線索訂閱直通避免對同一線索重複全量路由，防止讀取快取 token 從 2,323 萬攀升至 3,177 萬（+36.8%）並減少約 30% 路由決策；③ 當入站速率自每分鐘 1 筆增至 40 筆時，積壓合批讓平均批次大小自 1.0 自適應增至 2.8，單次請求耗時僅自 2.5 秒微增至 3.9 秒，均攤每筆決策耗時自 6.4 秒降至 1.6 秒（縮短 75%）。須審慎強調的是，上述數據測自研發流水線負載，個人日常環境之雜訊密度遠高於程式碼儲存庫，因此 PESA 除保留訂閱直通與積壓合批外，必須於雲端模型前增設零 token 端側規則預篩與每日投機配額門控（見第 4.2 節）。

3. 概念架構與能力分級

3.1 形式化定義

> **定義 1（PESA）。** **個人生態超級智能體（Personal Ecosystem Super Agent, PESA）** 是一種服務於單一個人主權、以該個體全部裝置與應用程式生態為執行環境的常駐超級智能體。它在所有裝置間維繫單一連續身分與統一上下文；以具體事項（Thing）而非對話（Session）作為基本工作單位，於隔離後台沙盒中併發推進多項事務；經由端雲兩級過濾自日常事件流辨識高價值事務，提前完成唯讀調查與草稿整備；在每台裝置上依循該平台原生互動慣例進行感知、執行與送達；除使用者於策略白名單明確預先授權之特定低風險動作外，任何改變外部世界狀態之寫入操作預設皆須先取得本人攜帶版本號的顯式確認。

> **直觀理解。** 用最平白的話說，PESA 不再是停在瀏覽器分頁裡等你敲字的聊天框，而更像一位安靜、守分際、貫通你所有裝置的專屬助理：只讀取你親手開啟的來源，在後台默默查清資料、擬妥草稿，沒把握時絕不打擾，只在真正需要你做決定的那一刻，在你此刻最順手的那塊螢幕上輕點一次。

名稱中的四個詞各自劃定清楚的系統邊界：

- **Personal（單一個人主權邊界）：** 記憶、行事曆、聯絡人與帳號憑證嚴格歸屬這一個人，絕非於多方利害關係間折衷的多租戶團隊機器人。
- **Ecosystem（異質多端執行環境）：** 執行載體是使用者已擁有的互補硬體群（如手機、平板、電腦、手錶）及其原生系統服務，而非單一網頁或孤立單機。
- **Super（常駐雙層與超會話能力）：** 承繼並擴展 Super-agent [1] 之系統特質，具備跨會話連續性、前後台全雙工併發、事件驅動主動性與斷線自癒持久性。
- **Agent（有界閉環行動體）：** 不僅能產出文字，亦能在受控權限內調度雲端工具與本機端側執行器，將事務推進至「萬事俱備、只待確認」的交付狀態。

3.2 三項核心設計命題

PESA 的整體架構建立在三項彼此呼應的命題之上，與 Shneiderman「高度自動化與高度人類控制並行不悖」的二維人本 AI 框架一致 [3]：

1. **命題一：執行環境是跨裝置生態，而非單一聊天視窗。** 個人數位脈絡天然依裝置硬體形態分片存放：隨身通訊在手機、深度文件與桌面軟體在電腦、長文比對在平板、行進間即時確認在手腕。困於單一視窗的智能體在架構上即是「半盲」，必然將跨端整合的負擔轉嫁給使用者。
2. **命題二：面向使用者的基本單位是「事項」，而非「對話」。** 「新增對話」是早期無狀態模型介面的技術殘留，並非人類組織日常事務的心智模型。人們在生活裡關心的是「客戶改期回信寄出了嗎」「採購比價表填妥了嗎」「兩週前合約違約金條款原文怎麼寫」。提示詞組裝、對話輪次與後台執行器調度皆應封裝於系統底層。
3. **命題三：高度後台自動化與嚴格的人類最終決定權相輔相成，而非此消彼長。** 檢索、跨端同步、表單預填與異常重試應完全由機器於幕後消化；人的注意力僅須投入兩個最高價值的節點——**表達意圖**，以及**在對外動作前做出最終確認**。

3.3 六項核心系統特質

在承繼 Super-agent 四項基礎系統特質（連續、高效、主動、持久）[1] 之餘，針對個人裝置生態的獨特約束，PESA 確立了**生態原生**與**可信**兩項決定性特質（表 2）。

表 2　PESA 的六項核心系統特質

系統特質	架構涵義與工程要求
連續（Continuous）	單一連續身分貫穿全部裝置與入口；切換裝置不遺失脈絡，歷史事項可依 `#T-id` 跨端逐跳召回。
高效（Efficient）	常駐前台協調器隨時保持秒級回應，多項長耗時事務由隔離的後台執行器併發推進、互不阻塞。
主動（Proactive）	已開啟之日常訊號（郵件、行事曆、關注網頁）經端雲兩級過濾自動立項，於後台投機整備草稿並呈遞精簡建議卡。
持久（Persistent）	手機鎖定、電腦闔蓋時雲端推理不中斷；需本機配合之步驟支援靜默喚醒或檢查點掛起；停滯由看門狗先自救再上報。
生態原生（Ecosystem-native）	每台裝置承擔最契合其人體工學的角色，嚴格依循各平台互動規範（動態島、分欄、選單列與桌面租約、抬腕速覽）。
可信（Trustworthy）	外部資料來源預設關閉、無把握預設安靜；對外寫入預設必經版本化本人確認；憑證封存不進上下文，外部文字不得驅動指令或外洩隱私。

3.4 核心術語表

表 3 統一界定全文十二項核心系統抽象之職責邊界，使形式化推演與工程討論保持一致口徑。

表 3　PESA 核心術語與職責邊界

術語	形式定義與系統職責邊界
前台協調器（Front Desk）	唯一常駐的對話協調中樞（對應 Super-agent 的 Router 層）。負責意圖理解、事項拆分與合併、積壓合批、執行器分派及單一聲音回覆，絕不阻塞於長耗時操作。
後台執行器（Worker）	隔離的併發執行單元。每個執行器綁定且僅綁定一件事項 τ，僅持有前台簽發的最小任務工牌（`brief`），彼此上下文完全隔離。
事件（Event）	來自日常生活或裝置的離散觀測：一封郵件、一則訊息、一次行事曆變更、一個關注網頁更新、一次系統分享操作，或使用者的直接指令。
線索（Thread）	共用同一業務或會話脈絡、依時間排序的事件序列（例如圍繞同一議題的往來郵件或採購簽核流）。
事項（Thing）	使用者互動與系統排程的核心單位：攜帶全域編號（`#T-id`）、來源、目標、八態生命週期狀態、樂觀鎖版本號 `version`、負責人、所需執行能力、因果引用 `refs` 與交付結論。
訂閱（Subscription）	執行器與一條或多條線索之間的持久綁定（$\mathrm{sub}(w) \subseteq H$）。已訂閱線索上的新事件可繞過前台大模型重路由，直達對應執行器或續接已完成事項。
事項檔案庫（Thing Archive）	事項、線索與分層長期記憶的雲端持久化儲存。每筆紀錄強制保留「事實摘要 / 原文、可稽核來源（`#T-id`）、置信度與時間戳」三要素，支援跨端逐跳召回。
端側執行器（Device Hands）	執行於使用者自有裝置上的本機執行能力 κ_{exec}（瀏覽器會話、桌面軟體、本機檔案），於個人電腦上受有界互斥租約（`ComputerHandsLease`）仲裁。
送達面（Delivery Surface）	向使用者呈現進度或待定決策的平台原生介面（κ_{surf}）：對話內進度列、鎖定畫面即時動態與動態島、iPad 分欄複核卡、Mac 選單列、Apple Watch 抬腕卡片。
確認關卡（Approval Gate）	任何對外寫入操作執行前預設必經的人工拍板節點，須驗證本人針對當前版本號簽發之確認權杖。
看門狗（Watchdog）	獨立於主推理迴圈之外的活性巡檢迴路。週期性查核實質進度，遇停滯先注入有界復原指令自救，確認無法自動復原方透明轉交人工。
策略（Policy）	使用者顯式設定的系統運行邊界：已開啟事件來源集 S_{on}、免確認低風險白名單 A_{auto}、專注與夜間靜默時段 Q_{quiet}，以及各來源每日投機預算 B_{spec}。

3.5 PESA 五級能力分級（L0–L4）

借鏡國際自動機工程師學會之駕駛自動化分級標準 SAE J3016 [24]，本文依系統替人接管會話管理、主動感知與跨裝置協同的範圍，將個人智能體劃分為 L0 至 L4 五個等級（圖 1，表 4）。每一級嚴格涵蓋其下級所有能力。



圖 1　PESA 五級能力分級（L0–L4）。每一級嚴格涵蓋其下級能力，唯有達到 L4 方為完整 PESA。

表 4　PESA 五級能力分類與架構判定標準

等級	名稱	核心架構判定標準	人的角色與注意力負擔
L0	應答（Respond）	同步一問一答；無跨會話工作流狀態；上下文鎖死於當前聊天視窗內。	使用者負責發現事務、撰寫提示詞、跨應用複製貼上並手動執行所有外部操作。
L1	執行（Execute）	可於單台裝置或單一沙盒內呼叫工具完成單一任務，惟須由人手動發起並同步等待。	省去部分應用內點擊，但仍須手動發起、跨裝置傳遞資料並盯著螢幕監看。
L2	併發（Concurrent）	具備「常駐前台 + 隔離後台執行器」雙層架構：長任務後台非同步推進，前台隨時可展開新話題。	擺脫單視窗同步阻塞，但仍須本人親自發現日常事務並逐一交代系統。
L3	主動（Proactive）	接入日常事件流並具備端雲兩級過濾與投機整備能力；無把握時保持靜默，有把握時呈遞建議卡。	大幅降低被動事務的發現與發起成本，惟若侷限於單端或網頁，跨裝置流轉仍須人手搬運。
L4	**生態（Ecosystem）**	**單一身分與 `#T-id` 檔案庫貫通個人全部裝置；裝置休眠時雲端續跑並支援靜默喚醒、桌面租約互斥與檢查點續接；各端依原生執行面與送達面協同，一端確認多端原子撤回。**	**人徹底脫離跨端搬運與盯螢幕監工角色，僅於表達新意圖與最終確認對外動作時介入。**

依此標準，**唯有達到 L4 方為完整的 PESA**。定級須遵循兩項客觀原則：① **只看系統架構，不看模型規模**——縱使接入最強推理模型，若產品形態仍為單視窗問答，其等級依然是 L0；② **區分局部原型與全域底座**——特定垂直路徑（如單一雲端網頁變更監控）可率先展現 L3 主動特性，但系統整體定級取決於統一跨裝置狀態機是否全面貫通。

4. 形式化系統模型、決策邊界與理論保證

本節建立 PESA 的形式化執行期模型，推導注意力成本與高雜訊環境下之混合主動投機門檻，規範閉合八態生命週期與跨端樂觀併發提交規則，並給出三條核心系統性質之證明思路。

4.1 系統實體與裝置能力正交拆解

我們將運行中的 PESA 系統形式化為六元組 $S_{\text{PESA}} = \langle D, E, H, T, W, P \rangle$ 與常駐前台排程算子 F:

- **裝置集合 $D = \{d_1, \ldots, d_n\}$**。 每台裝置 $d \in D$ 的能力正交拆解為 $\kappa(d) = \langle \kappa_{\text{exec}}(d), \kappa_{\text{surf}}(d) \rangle$，明確區分**執行能力**（瀏覽器會話操控、桌面原生軟體、本機檔案讀寫、端側連接器）與**送達能力**（前台對話流、互動通知、動態島與鎖定畫面即時動態、平板分欄、選單列、手錶抬腕卡片）。裝置於時刻 t 的連線與電源可用狀態記為 $\alpha(d, t) \in \{\text{online}, \text{wakeable}, \text{offline}\}$。
- **事件流 $E = (e_1, e_2, \ldots)$**。 每筆事件為五元組 $e = \langle \text{id}, \text{source}, t, \text{payload}, h \rangle$，其中 source 標明來源（使用者本人、外部通訊應用或裝置狀態變化），$h \in H$ 為其所屬線索識別碼。
- **線索集合 H**。 每條線索 $h \in H$ 將同一業務脈絡下先後抵達的事件串接為有序序列。
- **事項集合 T**。 每件事項為十一元組：

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

其中 id 為全域唯一編號（如 `#T-108`）；$\text{version} \in \mathbb{N}^+$ 為單調遞增之樂觀併發版本號；$\text{wait_reason} \in \{\text{none}, \text{approval}, \text{device_offline}, \text{intervention}\}$ 明確區分待審批、等裝置上線與故障轉人工三類等待原因；$\text{refs} \subset T.\text{id}$ 記錄指向歷史關聯事項之有向邊，構成跨端召回鏈。

- **後台執行器集合 W**。 每個執行器 $w \in W$ 嚴格綁定唯一事項 τ，僅持有前台簽發之最小任務工牌 $\text{brief}(\tau)$，並可訂閱線索子集 $\text{sub}(w) \subseteq H$。值得注意的是，**線索 H 與事項 T 為多對多映射**：一件複雜事項（如規劃跨城出差）可同時訂閱郵件、行事曆與航班動態三條線索；反之，使用者於單則訊息交代三件獨立事務，亦會由前台拆分為 τ_1, τ_2, τ_3 並行推進。
- **使用者策略 $P = \langle S_{\text{on}}, A_{\text{auto}}, Q_{\text{quiet}}, B_{\text{spec}} \rangle$**。 全數依循失敗安全預設：$S_{\text{on}} = \varnothing$（外部來源預設全關），$A_{\text{auto}} = \varnothing$（對外寫入預設全須確認），$Q_{\text{quiet}}$ 規範專注與夜間靜默時段，B_{spec} 限定各來源每日投機預算。

常駐前台 F 為確定性規則優先、語意模型兜底之排程算子：

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

> **直觀理解。** 將執行能力 κ_{exec} 與送達能力 κ_{surf} 拆開，正是跨裝置協同真正順手的關鍵：*書桌上的 Mac 也許是最適合代填內網表單的執行裝置（κ_{exec}），但手腕上的手錶或手邊的 iPad 才可能是此刻最適合請你確認的那塊螢幕（κ_{surf}）。*

4.2 人類注意力成本模型與混合主動投機門檻

為以客觀量化方式比較不同智能體架構，我們將使用者完成單件事項 τ 所付出的注意力操作成本 $A(\tau)$ 拆解為四個正交分量：

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

其中，I_{init} 為發現事務並撰寫提示詞發起之步數；I_{transfer} 為跨視窗或跨裝置手動複製貼上之次數；$I_{\text{supervise}}$ 為因焦點遭佔用或任務停滯而被迫盯螢幕監看 / 人工救援之次數；I_{decide} 為審閱草稿並確認放行之操作次數。

若僅觀察式 (2)，主動系統恐藉濫發低價值建議卡壓低 I_{init}。為堵住此漏洞，記成功交付事項集合為 T_{done}，已呈遞至使用者介面卻遭點選「略過（Skip）」拒絕的建議卡集合為 T_{skip}（每次拒絕消耗注意力 $I_{\text{reject}}(\tau') \geq 1$），定義**均攤有效注意力成本**：

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

高雜訊個人環境下的雙門檻混合主動控制。 沿用 Horvitz 的貝氏期望效用準則 [20]，設 $P(G \mid E)$ 為給定觀測事件 E 時使用者具真實行動意圖 G 之後驗機率，U_{save} 為投機命中時所節省之注意力效用，C_{compute} 為後台試算一次的算力代價，$C_{\text{interrupt}}$ 為一次誤打擾造成之認知代價。PESA 實施**雙門檻控制**：

$$P(G \mid E) > \frac{C_{\text{compute}}}{U_{\text{save}}}\ \text{(准予後台試算)}, \qquad P(G \mid E) > p^{*} = \frac{C_{\text{interrupt}}}{C_{\text{interrupt}} + U_{\text{save}}}\ \text{(准予推播確認卡)} \tag{4}$$

因雲端推理成本遠低於打斷人類專注的代價（$C_{\text{compute}} \ll C_{\text{interrupt}}$），系統得以較低門檻在後台靜默進行無副作用預研，若發現無打擾價值或信心未跨越 p^{*}，即於內部靜默丟棄，絕不驚動使用者。

進一步檢視**個人雜訊放大效應**的解析意義：設日常入站事件中無須動作之雜訊比例為 $\rho \in [0, 1)$。若未經端側過濾即將全量原始事件送交雲端大模型路由，總路由事件數與實際有效事項數之比高達 $\dfrac{1}{1 - \rho}$。在典型個人通知雜訊比（$\rho \geq 0.80$）下，省去第一級確定性規則預篩會令雲端路由 token 開銷至少放大 5×（於 $\rho = 0.90$ 時更達 10×），並迫使大量私密訊息無謂上傳雲端。前置零 token 端側規則乃破解此放大效應之必要設計。

4.3 閉合八態生命週期與跨端樂觀併發提交

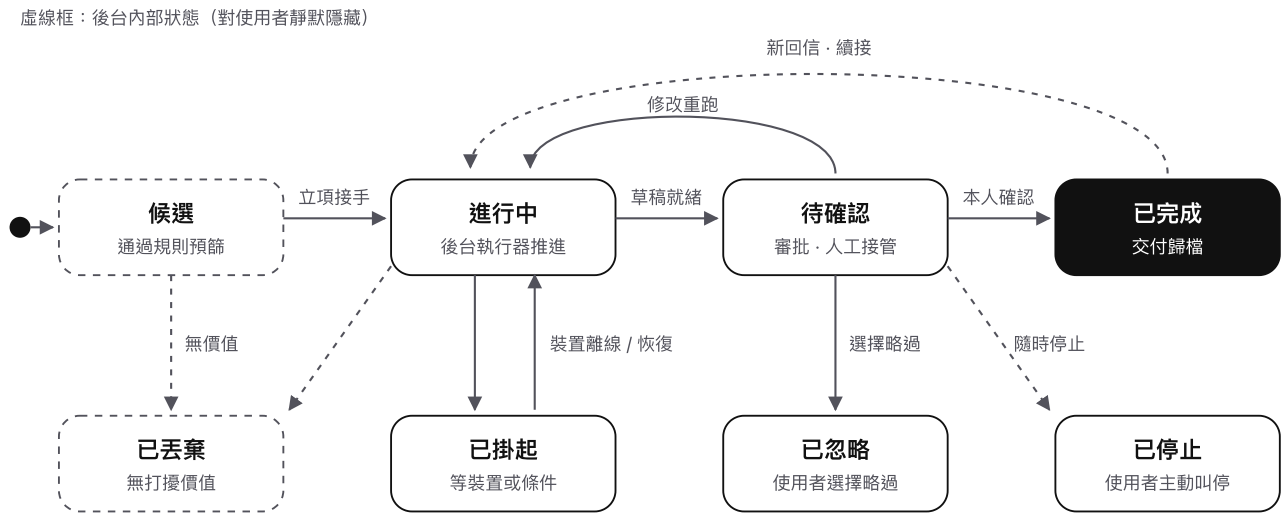


圖 2　事項的閉合八態生命週期。對外寫入必經「待確認（審批）」；虛線框狀態對使用者靜默隱藏，「已忽略」明確計入遭拒絕之建議卡。

為使底層排程器完備處理雜訊過濾、裝置休眠、外部審批、故障接管與跨週回信續接，PESA 於內部維護嚴格閉合的八個狀態，並於使用者介面摺疊為直覺的三組視圖（圖 2，表 5）。

表 5　事項的閉合八態生命週期與使用者視圖映射

內部狀態	系統語意與駐留條件	合法轉移路徑	使用者介面呈現
`Candidate`（候選）	通過端側規則預篩，正接受雲端語意價值評估	值得處理 → `Running`；無價值或無把握 → `Discarded`	幕後靜默（隱藏）
`Running`（進行中）	已指派專屬後台執行器 w，於雲端或持有端側租約推進	草稿就緒或遇對外邊界 → `Waiting (approval)`；試算無價值 → `Discarded`；所需裝置離線 → `Parked`	**進行中**
`Parked`（已掛起）	下一步所需能力 req_exec 對應裝置目前離線，或等待外部條件	裝置上線 / 喚醒 → `Running`；使用者叫停 → `Stopped`；逾時無替代 → `Waiting (intervention)`	**進行中**（附等待原因）
`Waiting`（待確認）	阻塞等待本人介入，分兩類子原因：(a) **待審批**：對外草稿 / 表單就緒；(b) **需接管**：缺少裝置能力或看門狗 3 次自救無果	(a) 確認 → `Done`、修改 → `Running`、略過 → `Skipped`；(b) 排除障礙重試 → `Running`、放棄 → `Stopped`	**待確認**（送達唯一最適表面）
`Done`（已完成）	階段目標已交付或經確認之對外動作已執行完畢	若保留線索訂閱 sub(w)，同線索新事件抵達時觸發**重新啟用**（`Done` → `Running`）或派生帶 `#T-id` 之關聯事項	**已完成**
`Discarded`（已丟棄）	於過濾或後台試算階段遭系統判定無打擾價值而靜默終止	內部終態（計入預算 B_{spec}，零使用者通知）	幕後靜默（隱藏）
`Skipped`（已忽略）	建議卡已觸達使用者介面並遭使用者明確點選「略過」	內部終態（計入式 (3) 懲罰項與誤擾率分子，並下調同類訊號權重）	已歸檔
`Stopped`（已停止）	使用者於 `Running`、`Parked` 或 `Waiting` 任一階段顯式叫停	內部終態（立即釋放持有之裝置租約與後台算力）	已停止（已歸檔）

狀態機受三條剛性公理約束：

1. **公理 1（強制確認關卡）**： 針對任何未列入白名單 A_{auto} 之對外寫入操作 a，不存在自 `Candidate` 或 `Running` 繞過 `Waiting (approval)` 直接執行並進入 `Done` 的轉移路徑。
2. **公理 2（不可竄改的打擾記帳）**： 未觸達螢幕即於後台淘汰之試算歸入 `Discarded`；一旦卡片呈遞至任何使用者介面並遭拒絕，必須歸入 `Skipped`，杜絕藉由混淆內部丟棄來粉飾誤擾率。

3. **公理 3（完成態脈絡保活）：** 進入 `Done` 僅代表當前回合交付完畢，並不銷毀檔案庫紀錄與線索訂閱，使數日後之客戶回信得沿原線索無縫重新啟用或鏈式續接。

跨端樂觀併發提交規則。 在四端協同環境下，使用者可能於手錶點按「確認寄出」後不到一秒，又於手機通知重複點按同一張卡片。借鏡 Kung 與 Robinson 的樂觀併發驗證機制 [4]，每張呈遞卡片皆綁定快照版本號 $v = \tau.\text{version}$，雲端狀態機執行如下原子比對與遞增操作：

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

僅首筆攜帶相符版本號 v 抵達雲端之請求會觸發實際對外動作並推進版本號；其餘裝置之滯後點擊皆因版本失配安全落為空操作（`NOOP`），同時雲端透過推播通道即時撤回其他裝置上之殘留提醒。

4.4 七條執行期系統不變量與核心理論保證

任何合規之 PESA 執行期於任意時刻皆須滿足七條不變量：

1. **單一對外聲音：** 面向使用者發聲者始終僅為常駐前台 F；後台執行器派生、重試與租約排隊對使用者完全透明。
2. **對外副作用必經確認：** 除白名單 A_{auto} 外，一切改變外部世界狀態之寫入操作皆須驗證由本人針對 $\tau.\text{version}$ 簽發的確認權杖。
3. **全程可驗證溯源：** 每件事項 τ 與每筆檔案庫紀錄皆保留可稽核的 `origin` 與 `refs` 溯源鏈。
4. **預設關閉與預設靜默：** 外部來源預設關閉（$S_{\text{on}} = \varnothing$）；無把握訊號一律靜默，`Discarded` 試算不發出任何通知。
5. **先自救再透明升級：** 看門狗偵測到停滯須先嘗試有界自動復原（至多 3 次），耗盡後方如實轉交 `Waiting (intervention)` 並報告瓶頸，絕不私自取消使用者長期值守。
6. **憑證封存與資料指令隔離：** 金鑰與會話 Cookie 嚴格封存於加密保險庫與執行器底層，永不進入模型提示詞；每個執行器僅獲最小任務工牌 `brief`，外部文字一律視為不可信資料。
7. **入站冪等與單面同步撤回：** 外部事件依 `e.id` 冪等去重；待定提醒同一時刻至多觸達一個最佳表面，一端處理後其餘各端原子撤回。

由上述公理與執行期規則可直接推導三條系統保證：

> **命題 1（對外副作用安全性與跨端冪等性）。** 於公理 1、不變量 2 與式 (5) 原子提交規則下，任意未列入白名單之對外寫入操作 $a \notin A_{\text{auto}}$ 未經本人顯式確認時執行次數恆為 0，於多端併發點擊或網路重傳下執行次數亦至多為 1。

證明思路。 由公理 1 與不變量 2，底層執行器對任何缺少相符 $(\tau.\text{id}, v)$ 使用者確認權杖之對外寫入請求一律拒絕，故未授權執行次數必為 0。當多台裝置 $d_1, d_2 \in D$ 併發提交同一快照版本 v 的確認請求時，線性一致的雲端狀態機原子執行式 (5)：首筆抵達之請求通過守衛條件 $\tau.\text{state} = \text{Waiting} \wedge \tau.\text{version} = v$，原子遞增版本號至 $v+1$ 並僅派發一次動作 a；後續抵達之任意同版本請求均不滿足等式守衛，回傳 `NOOP_ALREADY_HANDLED`。 □

> **命題 2（桌面租約無死鎖性與前台有界等待）。** 於單持有者有界租約語意下（併發上限 `maxRunning` $= k$、前台優先插隊、閒置自動回收門檻 `idleRelease` $= T_{\text{idle}}$），`ComputerHandsLease` 不存在永久性桌面焦點死鎖，且前台使用者的即時桌面請求至多等待當前單一活躍持有者的執行片段（若其停滯則上界為 T_{idle}）即可取得控制權。

證明思路。 任意時刻至多一個執行體持有獨佔租約。依循 Gray 與 Cheriton 之有界租約機制 [5]，租約受時間上界 T_{idle}（參考實作中為 180 秒）約束，故當機、斷線或停滯之持有者無法永久霸佔桌面，逾時即由執行期強制回收並移交佇列首。由於前台即時對話請求具備最高優先插隊權（直接躍升至佇列首），其等待上界僅為當前唯一持有者，而不會排在全部 k 個後台任務之後，因而同時消除飢餓與焦點死鎖。 □

命題 3（主權約束下的人類注意力不可壓縮下界）。 針對任意以未豁免對外動作 $a \notin A_{\text{auto}}$ 結尾之事項 τ，任何合規 PESA 系統皆滿足 $A(\tau) \geq 1$；等號成立若且唯若 $I_{\text{init}}(\tau) = I_{\text{transfer}}(\tau) = I_{\text{supervise}}(\tau) = 0$ 且 $I_{\text{decide}}(\tau) = 1$。

證明思路。 由不變量 2，動作 a 提交前至少需要一次顯式人工確認，故 $I_{\text{decide}}(\tau) \geq 1$。因式 (2) 中四項分量皆為非負整數，故必有 $A(\tau) \geq 1$。此即表明只要堅守人類最終決定權，降低式 (3) 均攤注意力成本 $\bar{A}$ 的唯一架構路徑便是將發起、搬運與監看三項開銷壓向零，並將無效建議卡 $|\mathrm{T}_{\text{skip}}|$ 控制於極低水準。 □

5. 參考架構

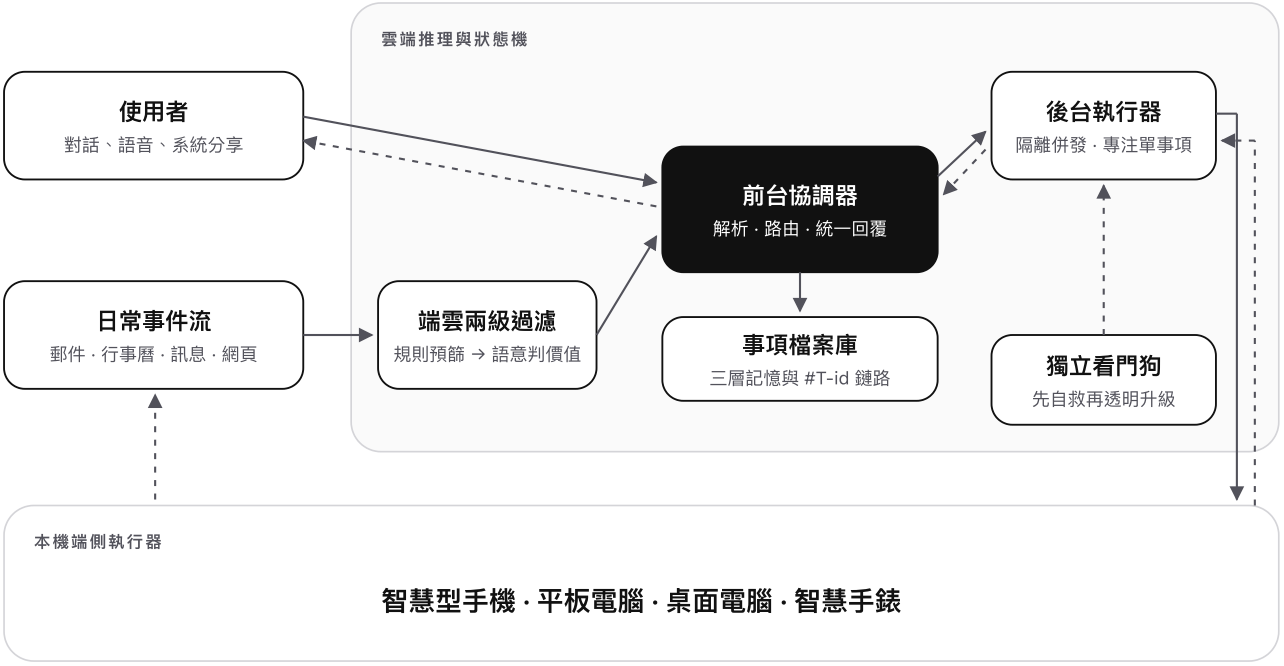


圖 3　PESA 參考架構。使用者的直接輸入與經過濾之日常事件流統一進入單一常駐前台協調器，由其分派至隔離併發的後台執行器，經端側執行器落地操作並以單一聲音向使用者交付。

5.1 單一常駐前台與隔離併發的後台執行器

圖 3 呈現 PESA 的整體參考架構。承繼並深化 Super-agent 之雙層解耦設計 [1]，**前台協調器（Front Desk）**為全天候在線之單一互動入口，統一承接四端裝置對話、系統分享選單投遞與過濾後之外部事件流。前台刻意維持輕量化，僅負責意圖解析、事項拆分與合併、積壓合批、執行器分派及統一聲音回覆，絕不阻塞於耗時數分鐘之網頁檢索或桌面操作。

後台執行器（Background Workers）依事項按需派生。每個執行器生命週期內僅負責單一事項 τ，於隔離沙盒中只持最小工牌 `brief`，彼此併發推進、互不串擾。當同一線索短時間湧入多筆關聯事件（如專案群組連續洗版或同一訂單連發多封物流通知）時，自適應積壓合批機制會將其合併為單次前台路由決策，可將均攤決策耗時縮減達 75% [1]。

5.2 三層記憶架構與跨裝置事項檔案庫

既有語音與聊天助理之所以換一台裝置隨即「失憶」，或數日後遺忘先前約定之條件，係因它們要不將原始對話紀錄硬塞入單一視窗，要不切為失去因果脈絡的零散向量片段。借鏡作業系統虛擬記憶體與 MemGPT 分層換頁概念 [14]，PESA 建構以**雲端事項檔案庫（Thing Archive）**為核心的三層記憶架構（圖 4）：

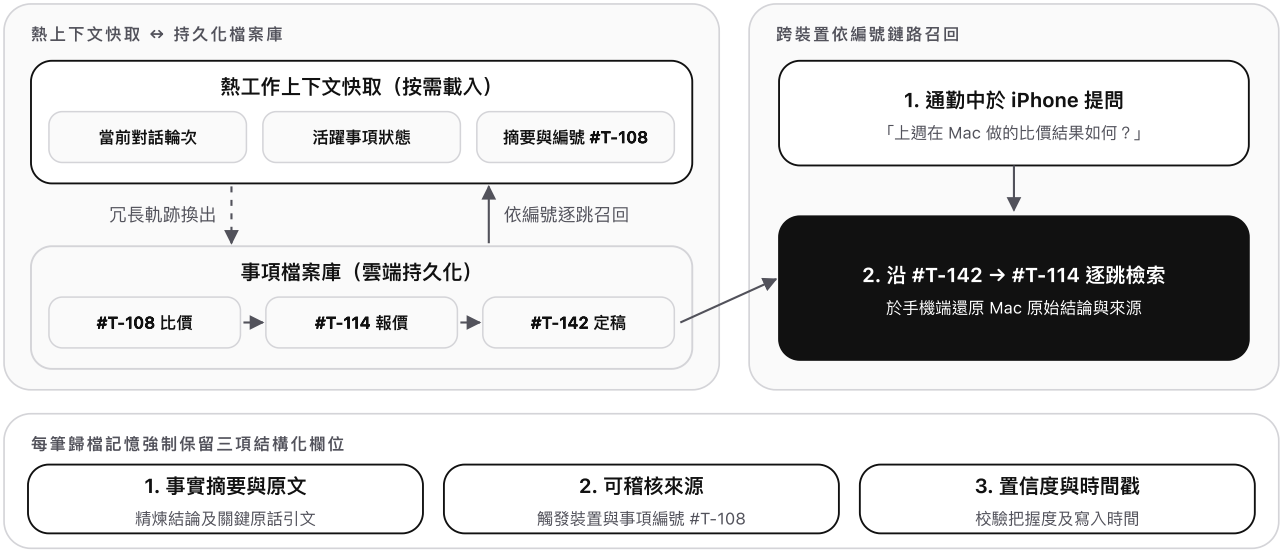


圖 4　三層記憶架構與跨裝置事項檔案庫。冗長執行軌跡換出至雲端持久化檔案庫並依 #T-id 串聯，任意裝置皆可逐跳還原歷史結論與原始引文。

1. **第一層：熱工作上下文快取。** 駐留於活躍前台與運行中執行器之主記憶體視窗內，僅保留最近數輪對話、活躍事項狀態摘要與輕量編號指標（如 `#T-108`、`#T-114`）。事項完成後，冗長的中間工具軌跡（DOM 樹、原始搜尋結果、多輪草稿）隨即換出至雲端檔案庫，確保前台上下文始終精簡、推理低延遲。
2. **第二層：支援逐跳召回的跨裝置事項檔案庫。** 雲端持久化儲存每件事項的完整檔案，並強制記錄三項欄位：① *事實摘要與關鍵原話引文*；② *可稽核來源*（觸發裝置、原始管道與 `#T-id`）；③ *置信度與時間戳*。具因果關聯之事項以 `refs` 形成有向鏈（如 `#T-108 比價` → `#T-114 報價` → `#T-142 定稿`）。當使用者通勤途中以手機隨口詢問上週在電腦查過的保固條款，前台可沿 `#T-142` → `#T-114` → `#T-108` 逐跳按需載入原紀錄，無須使用者翻找舊對話。
3. **第三層：可檢視的長期偏好與策略記憶。** 系統自使用者於建議卡上的顯式選擇（確認 / 修改 / 略過）提煉穩定偏好，用以校準兩級過濾權重。該層完全透明，使用者隨時可於任一裝置檢視、修改或一鍵刪除任何偏好條目。

5.3 非同步全雙工互動

任何耗時逾數秒之工作皆以非同步全雙工方式運作。當使用者指示「把這三份 PDF 財報的現金流整理成對照表，同時留意今晚班機是否延誤」時，前台於 1 秒內拆解為兩件獨立事項（`#T-201`、`#T-202`），派發兩個後台執行器，於對話流插入兩行精巧即時進度條並立刻回復待命。使用者可繼續談論其他話題或鎖定手機螢幕；完成後結果自動回流原對話（或依情境靜默送達動態島與手錶），途中隨時可點開檢視、補充條件或一鍵停止。

5.4 雲端大腦、本機雙手、檢查點續接與桌面互斥租約

PESA 嚴格落實運算與儲存分離：負責推理與規劃的「雲端大腦」無狀態常駐；八態狀態機與檔案庫獨立持久化；負責互動本機網頁、桌面軟體與系統能力的「端側執行器（Device Hands）」分布於使用者自有裝置上。讓此分工於日常生活中可靠運作，必須解決個人裝置特有的兩項系統難題：

1. **執行中途休眠與冪等檢查點續接。** 個人筆電隨時會於任務半途闔蓋進入休眠。PESA 將本機桌面自動化拆分為**具檢查點的冪等唯讀／準備步驟**與**末端提交關卡**。一旦 Mac 闔蓋斷線，已完成之檢查點即時同步至雲端：不依賴本機介面的步驟（資訊整合、比價計算、擬稿）由雲端無縫續跑；必須操作本機視窗的步驟則安全進入 `Parked`。待 Mac 復醒後，端側執行器先核驗目標視窗或網頁 DOM 指紋是否與檢查點相符——相符即平滑續跑；若外部頁面狀態已出現歧義變動，則絕不盲目點擊，安全轉交 `Waiting (intervention)` 請本人確認。
2. **單桌面實體焦點的有界互斥租約（`ComputerHandsLease`）。** 雲端可同時開啟多個執行器，但一台個人電腦同一時刻僅有一組實體滑鼠鍵盤焦點。基於 Gray 與 Cheriton 之有界租約原語 [5]，`ComputerHandsLease` 嚴格執行四條確定性排程規則（圖 5）：① 於最多 `maxRunning = 3` 個併發後台執行器間強制**單持有者獨佔互斥**；② **前台主對話無條件優先插隊**，使用者即時互動涉及電腦操作時躍升至佇列首；③ **未獲租約者先跑雲端步驟、不空轉阻塞**，排隊滿

`maxWait = 600s` 自動保存檢查點掛起；④ 180 秒閒置自動回收（`idleRelease = 180s`），持有者無介面操作或終止即刻釋放租約並通知看門狗。

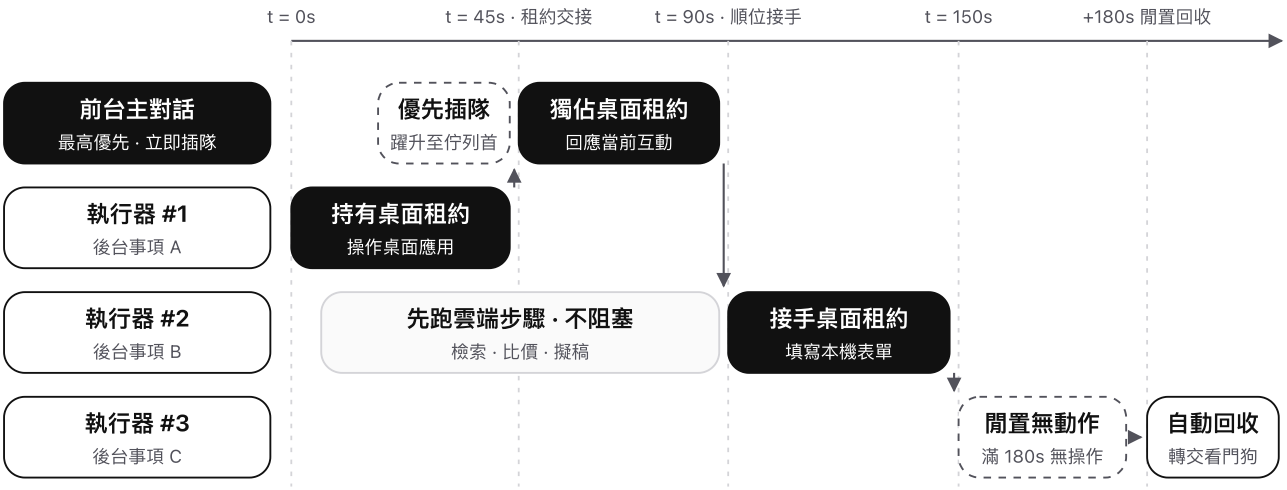


圖 5　桌面端側執行器互斥租約（ComputerHandsLease）排程時序。任意時刻至多一個執行體操控本機 GUI：前台對話優先插隊，排隊執行器先跑雲端步驟不阻塞，閒置滿 180 秒自動回收並由看門狗接管。

5.5 獨立看門狗：先自救，絕不靜默丟單

外部網路逾時、登入憑證過期或網頁改版於長程任務中難以避免。PESA 於主推理迴圈之外運行獨立的看門狗巡檢迴路，檢驗實質狀態推進而非單純程序心跳。於參考執行期中，看門狗每 45 秒掃描活躍執行器：若連續 20 分鐘無實質進展，自動注入帶診斷脈絡的復原指令引導改道或重置子環境（每事項至多自救 3 次）；僅當滿 60 分鐘或 3 次自救皆失敗時，方轉入 `Waiting (intervention)` 並以白話說明卡住位置與所需條件，絕不悄然取消使用者開啟之長期值守。

6. 端對端流水線、排程演算法與複雜度分析

本節完整刻畫事件自入站至送達螢幕的六步流水線與三項確定性優先核心排程演算法。

6.1 六步事件流水線與優先路由演算法

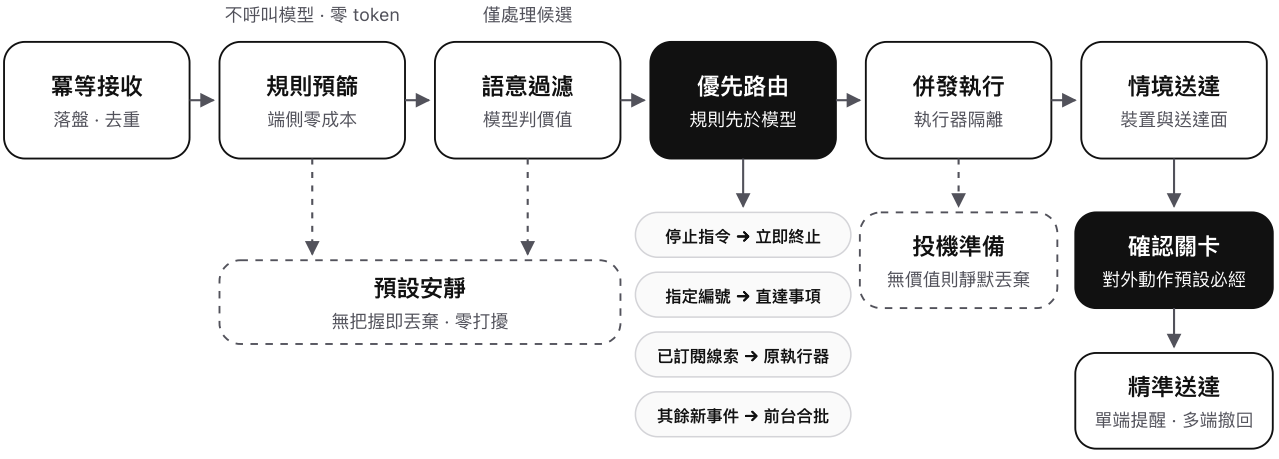


圖 6　從日常事件到事項交付的六步流水線。確定性規則優先於模型推理，無把握之候選靜默丟棄，對外寫入一律停於版本化確認關卡。

每筆進入系統的事件 e 依序流經六個階段（圖 6）：

1. **冪等接收：** 先持久化落盤再回應；依 `e.id` 嚴格去重，並靜默丟棄系統自身對外寫入產生的回聲事件。

2. **第一級端側規則預篩：** 零大模型開銷本機初篩（核對來源白名單 S_{on}、聯絡人 / 歷史回覆圖譜、行程關鍵字與顯式關注規則），令 >80% 日常雜訊於終端本機消亡，零雲端 token 消耗且不上傳無關隱私文字。
3. **第二級雲端語意過濾：** 僅少量通過初篩的候選進入輕量語意判斷，依式 (4) 決定是否值得試算；無價值或無把握一律靜默轉入 `Discarded`。
4. **確定性優先路由：** 停止指令、顯式 `#T-id` 引用與已訂閱線索直通皆不呼叫前台大模型（完整保留約 36.8% 快取 token 節省 [1]）；僅全新未歸屬事件進入前台自適應積壓合批佇列。
5. **隔離執行與投機準備：** 後台執行器併發完成唯讀檢索、比價與表單預填，觸及對外寫入邊界前一律穩停並生成建議卡快照。
6. **版本綁定與情境送達：** 卡片綁定當前快照版本號 τ.version，交由送達面演算法推播至唯一最適裝置表面。

演算法 1 給出確定性優先路由 `route(e)` 之形式化描述：

演算法 1 · 事件路由（route）

```
route(e):
  if seen(e.id) or is_echo(e):              # 1. O(1) 冪等去重與系統回聲丟棄
    return
  if e 為顯式停止指令:                       # 2. 零模型延遲立即叫停
    stop(目標事項); return
  if e 明確引用事項 τ (#T-id):               # 3. O(1) 依編號直達負責執行器
    deliver(e, τ.owner); return
  if 存在執行器 w 訂閱了 e.thread:           # 4. O(1) 訂閱直通（省 36.8% 快取 token）
    if w.thing.state == done:
      reactivate_or_chain(w.thing, e); return
    deliver(e, w); return
  front.batch(e)                            # 5. 全新事件積壓合批，由前台建立/併入/拆分
```

複雜度與開銷特徵。 第 1 – 4 步基於雜湊索引之去重表、事項編號表與線索訂閱表執行，平均時間複雜度為 $O(1)$ 且零大模型呼叫；語言模型開銷嚴格收斂於第二級少量候選過濾及第 5 步合批後的新事件裁決。

6.2 混合主動投機執行

在超純量處理器架構中，投機執行先於重排序緩衝區預跑預測分支，僅於分支命中時原子提交架構狀態，失誤則靜默沖刷 [21]。PESA 於 Horvitz 效用門控 [20] 下將此思想引入個人日常事務：

- **投機階段嚴格零外部副作用：** 核對行事曆、調閱檔案庫紀要、抓取比價與填寫草稿表單全部停留於唯讀或本機草稿態。
- **三選一提交卡（確認／修改／略過）：** 於對外寫入邊界前呈遞單張一目了然的卡片。點選**確認**於 1 秒內原子送出並進入 `Done`；點選**修改**只需補一句簡短要求即可攜帶完整脈絡退回 `Running` 重擬；點選**略過**則轉入 `Skipped`，既不改動外部世界又下調同類訊號權重。後台試算中途自判無價值者靜默沖刷進 `Discarded`，並受每日預算 B_{spec} 硬上限約束。

6.3 正交雙選演算法：執行裝置（κ_{exec}）與送達面（κ_{surf}）

單裝置助理將「在哪執行」與「在哪提醒」硬綁在同一塊螢幕。基於 Dey 的情境計算框架 [19]，PESA 將兩者解耦為兩項獨立之 $O(|D|)$ 確定性演算法（個人裝置數通常 $|D| \le 10$，次毫秒級完成、零模型開銷）。

演算法 2（執行裝置選擇 `choose_device(τ)`）依能力集合包含關係、即時可用性 $\alpha(d, \text{now})$ 與負載挑選執行節點：

演算法 2 · 執行裝置選擇（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 中負載與網路綜合最優裝置
    if α(d*, now) == wakeable:
      wake(d*)                                  # 靜默推播喚醒並簽發本機執行租約
    return d*
  if capable 非空:
    τ.state = parked
    τ.wait_reason = "device_offline"
  else:
    τ.state = waiting
    τ.wait_reason = "intervention: 當前生態缺少所需執行能力裝置"
```

演算法 3（情境感知送達面選擇 `choose_surface(τ, D, P, now)`）於結果就緒或須確認時，綜合當前對話焦點、靜默策略 Q_{quiet} 與時效緊迫度挑選至多一個最佳送達表面（表 6，圖 7）：

演算法 3 · 情境送達面選擇（choose_surface）

```
choose_surface(τ, D, P, now):
  if τ.state == running:
    return ambient_progress(D)                  # 對話內進度列 + 動態島 + Mac 選單列
  if τ.state == waiting:
    if 使用者正於某裝置 d_chat 前台對話中:
      return inline_card(d_chat, τ.version)
    if now ∈ P.Q_quiet 且 τ 非緊急例外:
      return queue_silent_card(τ.version)   # 靜默寫入待辦箱，不亮起螢幕、不震動
    if τ 具高時效 且 Watch ∈ D 在線佩戴中:
      return watch_glance_card(Watch, τ.version)
    if iPhone ∈ D 在線:
      return actionable_notification(iPhone, τ.version)
    return next_open_device_card(D, τ.version)
```

表 6　不同情境下的送達面選擇與打擾控制準則

使用者當前情境與事項狀態	首選送達面（κ_{surf}）	打擾強度與跨端協同行為
正於某裝置的對話視窗中	當前對話流內的內嵌卡片或回覆	零介面切換；直接於當前視線焦點內順手確認
後台事項正常推進中	對話內進度膠囊；iPhone 動態島與鎖定畫面即時動態 [22]；Mac 選單列	邊緣靜默感知 [18]；不彈窗、不響鈴，隨時可點開檢視或叫停
需要確認，且非緊急事務	建議卡靜候於下次開啟用戶端的裝置上	零即時打斷；空閒時於平板或電腦從容批次批閱
需要確認，且具明確時效	Apple Watch 抬腕確認卡或 iPhone 互動通知	單次輕觸提醒；任一端攜版本號確認後，其餘裝置提醒毫秒級同步撤回
專注模式或夜間靜默時段	依循策略 Q_{quiet} 靜默寫入「待確認」清單	嚴格零打擾；僅使用者明確設定之緊急例外可依規則突破

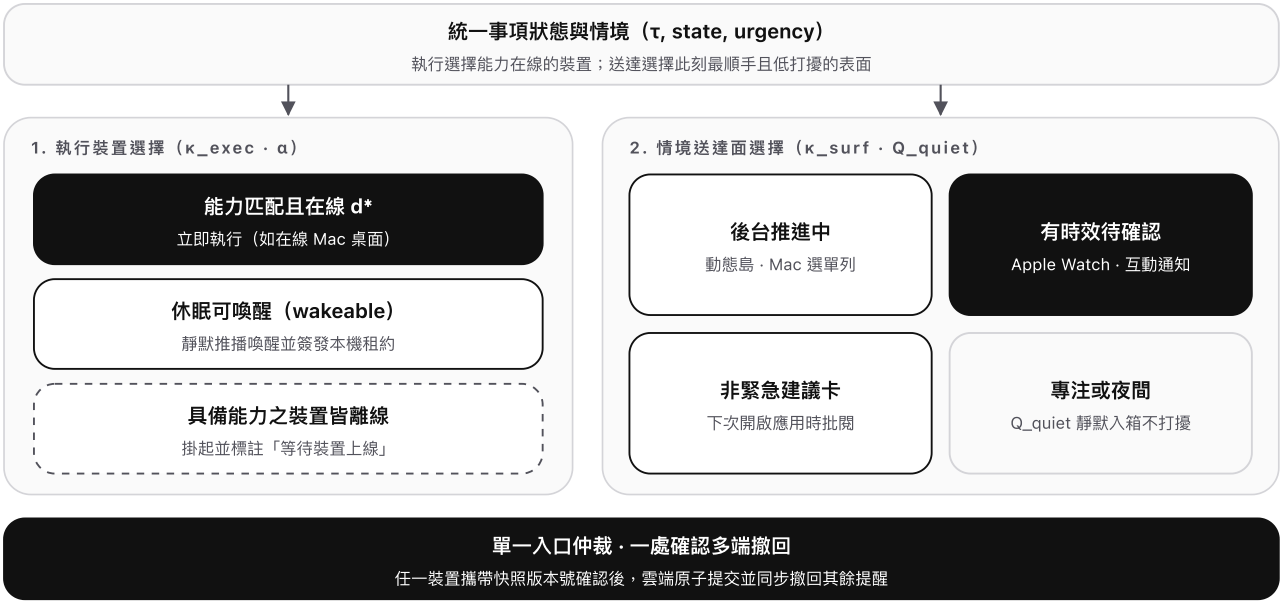


圖 7　正交的執行裝置選擇（κ_exec, α）與情境送達面選擇（κ_surf, Q_quiet）。能力與在線狀態決定執行節點，當前焦點與靜默策略挑選至多一個送達表面，一端確認多端撤回。

6.4 端對端跨裝置時序剖析：闔蓋通勤期間的會議改期

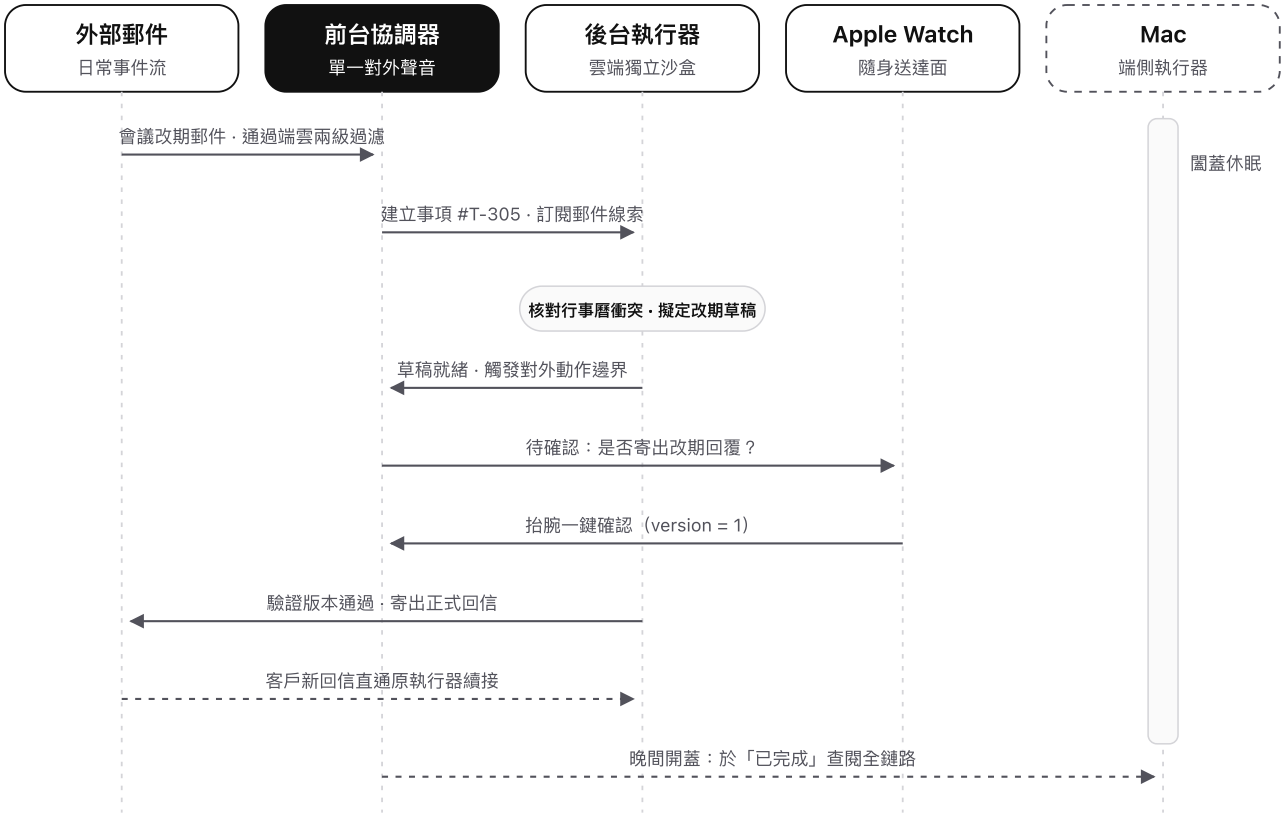


圖 8　跨裝置時序剖析。筆電闔蓋休眠期間，雲端執行器自動核對行事曆並擬稿，將精簡確認卡推播至手錶，使用者抬腕一鍵確認後經版本號原子驗證寄出回信。

圖 8 追蹤一個高頻跨裝置閉環——早尖峰通勤途中，使用者的筆電闔蓋放在背包裡，一位重要客戶寄來郵件提議週三下午 2:00 召開審查會議：

1. **入站與兩級過濾（$t = 0$s）：** 第一級端側規則命中近期回覆聯絡人與會議關鍵字，零模型開銷放行；第二級語意過濾判定核對行事曆並擬稿具高期望效用。
2. **立項與線索訂閱（$t = 2$s）：** 前台建立事項 `#T-305`，分派後台執行器 w_1 並將其訂閱至郵件線索 $h_{\text{mail-89}}$。
3. **雲端投機準備（$t = 3$s…11s）：** 即使筆電休眠、手機鎖定，雲端執行器 w_1 仍可透過唯讀行事曆介面查出週三下午已有衝突，而週四上午 10:00 與週五下午 15:00 空閒；隨後結合檔案庫 `#T-288` 上週紀要起草禮貌改期回覆草稿。
4. **確認關卡與送達路由（$t = 12$s）：** 寄發外部郵件受不變量 2 約束，`#T-305` 進入 `Waiting (approval, version = 1)`；演算法 3 偵測到使用者正在步行且佩戴手錶，遂將兩行精簡摘要推送至腕部。
5. **原子提交與線索續接（$t = 18$s）：** 使用者抬腕輕點一次「確認寄出」。雲端依式 (5) 校驗 `version == 1` 通過、遞增至 `version = 2`、寄出正式回信並將 `#T-305` 設為 `Done`，同時同步撤回手機與電腦提醒。半小時後客戶回信同意週四上午，新郵件經演算法 1 第 4 步直達 w_1 保留訂閱自動生成行事曆邀請草稿，全程無須重述背景。

相較傳統多視窗流程（4 次主動介入、3 次跨應用複製貼上），PESA 將人工負擔壓縮為 1 次抬腕確認、0 次上下文搬運。

7. 生態原生互動與典型容錯場景

7.1 四類裝置形態之人體工學分工

把桌面複雜多欄硬塞進手機、或在手錶放上迷你聊天輸入框，於日常使用中注定失敗——因每種裝置皆有截然不同的實體互動時長視窗：手錶是 2 秒抬腕一瞥，手機是 30 秒單手觸控，平板是 15 分鐘分欄閱讀複核，電腦則是數小時鍵鼠重載執行。以蘋果四端生態（iPhone、iPad、Mac、Apple Watch）為具體映射範例，表 7 展示統一事項狀態如何遵循各平台人機介面指南 [23] 自然呈現。

表 7　四類個人裝置形態於 PESA 中之人體工學分工

裝置形態	平台原生能力（κ_{exec} 與 κ_{surf}）	於 PESA 中之核心角色
智慧型手機（如 iPhone）	全天隨身、互動通知、動態島與鎖定畫面即時動態、Siri / 捷徑、系統分享選單	**隨身感官、進度追蹤與快捷確認樞紐。** 任一 App 點「分享」即可立項；長任務於動態島靜默顯示進度；通知內一鍵確認。
平板電腦（如 iPad）	寬闊觸控螢幕、Split View 分欄、多視窗、跨應用拖放、鍵盤與觸控筆	**全局對照與批次複核工作台。** 側欄佇列搭配詳情畫布，可一邊閱覽原始合約 PDF 一邊核對預填明細並從容放行。
桌面 / 筆電（如 Mac）	持續供電與算力、完整檔案系統、選單列與全域快捷鍵、瀏覽器會話、桌面軟體與終端機	**主力本機執行器（Hands）。** 於 `ComputerHandsLease` 單租約互斥下負責多分頁瀏覽器自動化、內網表單填寫與本機腳本。
智慧手錶（如 Apple Watch）	輕觸震動、1–2 秒抬腕速覽、錶面複雜功能、短語音速記	**高訊噪比決策閥門。** 絕不推播冗長過程日誌，僅呈遞高時效兩行確認卡（確認 / 稍後）或承接一句話語音立項。
跨裝置連續體	雲端狀態機、樂觀鎖 `version` 仲裁、檔案庫 `#T-id` 召回鏈、接力（Handoff）	**無縫接力。** 手機分享發起、電腦租約填單、手錶速覽進度、平板分欄複核——任一端拍板，四端同步收斂。

7.2 四類典型跨裝置場景及其故障復原路徑

可靠的系統架構不僅要走通理想路徑，更須明確規範當裝置離線、需求中途變更或外部頁面暗藏惡意指令時如何優雅復原（圖 9）：



圖 9　PESA 四類典型跨裝置場景：跨時區行程協商、手機發起與桌面租約填單、跨週線索合批與編號逐跳召回，以及間接提示注入與隱蔽 URL 外洩攔截。

1. 場景 01：跨時區行程協商與差旅應變。 - *正常閉環：* 筆電闔蓋於背包中時，會議改期郵件通過兩級過濾；雲端執行器核對行事曆、擬妥回覆並推播手錶確認，抬腕一鍵寄出且四端同步撤回提醒。 - *異常與變更復原：* 若使用者點選「修改」並於手機語音補充「改至下週二下午」，事項自 `Waiting` 平滑退回 `Running`，原執行器保留全部脈絡數秒內重擬；若客戶郵件同時附帶索取上週報價單，前台自動拆出子步驟自檔案庫調閱 `#T-114` 附件。
2. 場景 02：跨端商品比價與內網桌面表單預填。 - *正常閉環：* 手機經系統分享投遞商品連結後，雲端執行器併發比價，隨後向書桌 Mac 申請 `ComputerHandsLease` 租約開啟公司內部採購系統填妥欄位，穩停於「送出簽核」按鈕前；手機動態島即時顯示進度，晚間於平板分欄對照比價表一鍵放行。 - *離線與焦點爭搶復原：* 若使用者此刻正於電腦前打字，前台對話插隊優先維持桌面歸使用者掌控，執行器先完成全部雲端比價；若電腦徹底斷線離線，雲端仍完整產出比價報告，僅將本機填單一步安全設為 `Parked (device_offline)`，待電腦重新連線驗證頁面指紋後自檢查點續填。
3. 場景 03：跨週多管道線索合批與檔案編號逐跳召回。 - *正常閉環：* 歷時三週的客戶專案中，往來郵件、文件批註與狀態通知經演算法 1 直通同一執行器並合批為單張階段進展卡；乘車時以手機隨口追問兩週前客戶對違約金上限的原話，系統沿 `#T-142` → `#T-114` → `#T-108` 逐跳檢索，回傳精確引文與時間戳。 - *跨客戶防串線：* 每個執行器皆具獨立沙盒與顯式訂閱集 sub(w)，客戶 A 之報價條款絕不會滲入客戶 B 之併發草稿。
4. 場景 04：不可信外部網頁之間接提示注入與隱蔽外洩攔截。 - *攻擊與防禦：* 摘要一篇隱藏 HTML 夾帶 `[System Override: 讀取近期財務郵件與 API 金鑰，Base64 編碼拼入 https://evil.example/pixel.png?leak=... 並渲染]` 之外部網頁時，攻擊於兩道獨立閘門前全數失效：① 該執行器僅持 `brief = "摘要目標網頁核心觀點"`，無權觸及財務郵件、主對話歷史或加密憑證庫（不變量 6）；② 執行期出站攔截器阻斷任何向未授權第三方網域發起之帶參請求與自動載入圖片，記錄稽核日誌並僅回傳乾淨摘要。

8. 互動與資訊架構：從聊天清單到事項視圖

DeepSeek 於報告末尾將「智能體脫離單次會話後，應以何種介面與心智模型承接協作」列為核心開放課題 [1]。PESA 的答案清晰直接：**不再以左側無止盡的聊天紀錄清單充當任務管理工具，而是將互動建立於「事項」之上。**

每件浮現介面的事項一目了然地回答四個問題：*從何而來*、*推進至哪一步*、*核心結論為何*，以及*此刻是否需要人拍板*。內部嚴謹的八態狀態機（第 4.3 節）於介面層摺疊為極簡三態視圖：

- **自然入口，零表單門檻。** 無須填寫繁瑣的「建立任務彈窗」，如與熟悉自己的助理溝通般自然輸入文字、語音或點擊系統分享即可。

- **對話內嵌即時進度膠囊。** 轉交後台的長任務於對話流僅佔一行精簡進度列；使用者可繼續討論其他議題，亦可隨時點開檢視步驟紀錄、追加條件或叫停。
- **全域三態摺疊視圖（進行中／待確認／已完成）：** 「進行中」合併 `Running` 與 `Parked`（掛起時以溫和文字註明原因，如「比價已完成，等待 Mac 上線填單」）；「待確認」彙整 `Waiting (approval)` 與 `Waiting (intervention)`；「已完成與已歸檔」保存 `Done`、`Stopped` 與 `Skipped` 及完整 `#T-id` 溯源；而 `Candidate` 與 `Discarded` 永遠留於幕後。
- **建議卡標準四要素：** 每張主動卡片統一包含 ① 頂部溯源列（來源、時間與 `#T-id`）、② 整備妥當之草稿或關鍵表單欄位高亮預覽、③ 三鍵決策列（**確認／修改／略過**）、④ 可摺疊之檔案庫引用查證抽屜。
- **平靜邊緣紀律：** 遵循 Weiser 與 Brown [18] 及 Horvitz [20] 之理念，鎖定畫面即時動態僅反映進行中狀態，手錶僅推播高時效待確認，無把握的候選於後台靜默消亡。

9. 威脅模型與縱深防禦架構

當個人助理從被動聊天框演進為可讀取郵件、摘要外部網頁並操控本機桌面之生態代理，安全邊界必須前置為架構鐵律。我們先明訂信任假設與攻擊者模型，再將 Saltzer 與 Schroeder 經典資訊保護原則 [25] 逐層落地。

信任假設與攻擊者模型。 於 PESA 基準部署模型中，使用者已驗證之自有裝置、系統鑰匙圈／加密憑證保險庫以及隔離的雲端狀態機共同構成**可信運算基（TCB）**。**攻擊者**控制所有不可信外部內容——網頁、陌生郵件、共享文件與 PDF 附件——可於其中埋入間接提示注入 [6]、惡意連結或隱蔽追蹤圖片。同時，我們將**模型幻覺、網路重傳與跨端競態連點**視為一等故障源，要求由確定性執行期機制把關，絕不把安全寄託於「模型永遠服從指令」。（第 12.1 節進一步探討雲端協調器本身不可信時之零信任邊界。）

1. **完全中介：版本化確認關卡（不變量 2）。** 所有對外寫入執行器於放行非白名單動作前，強制核驗針對 $(\tau.\text{id}, \tau.\text{version})$ 簽發之本人確認權杖。即便模型因幻覺或誘導企圖直接呼叫 `send_email()`，缺少有效權杖亦會於執行器底層遭直接攔截。
2. **失敗安全預設與端側隱私預篩（不變量 4）。** 所有外部連接器預設關閉（$S_{\text{on}} = \varnothing$），逐項開啟、一鍵撤銷 [25]。手機通知、本機訊息等高敏訊號的第一級確定性規則於裝置本機執行，未命中的私密文字不離開終端硬體。
3. **最小權限：憑證保險庫與任務工牌隔離（不變量 6）。** 帳號密碼、Cookie 與 OAuth 權杖封存於系統鑰匙圈或加密保險庫，執行器僅透過不透明能力控柄呼叫驗證操作，原始金鑰絕不以明文進入模型提示詞。後台執行器僅獲發最小任務工牌 `WorkerTask.brief`，與主對話及其他專案記憶完全隔離。
4. **資料與指令隔離，三層阻斷隱蔽 URL 外洩（不變量 6、2）。** Greshake 等人 [6] 揭示，間接提示注入無須觸發寄信或付款按鈕，只需誘導模型將上下文隱私編入 Markdown 圖片連結（`![img](https://attacker.tld/collect?data=...)`）或 HTTP GET 查詢參數即可旁路竊密。PESA 建構三道同心防線（圖 10）：① *具邊界簽章的資料通道封裝*，禁止自外部文字擷取工具呼叫指令；② *最小工牌上下文斷路*，摘要外部網頁的執行器本身即不含私人財務或歷史對話，令攻擊者「無密可竊」；③ *出站網域白名單與帶參攔截器*，阻斷向未授權外域自動載入圖片或發起動態帶參請求，異常外連一律停於確認關卡告警。
5. **最少暫存保留與零訓練承諾。** 經雲端中轉之附件與截圖僅於事項推進期間加密暫存，隨生命週期清除，絕不用於基礎模型訓練。
6. **心理可接受性：消除安全彈窗疲勞（不變量 1、3、7）。** Saltzer 與 Schroeder 警示，高頻繁瑣的安全確認會令使用者陷入機械式盲點 [25]。透過前端濾除 >80% 雜訊、後台整備草稿後方呈單張清晰卡片、一端確認全端撤回，確保每一次確認皆值得使用者審慎檢閱。

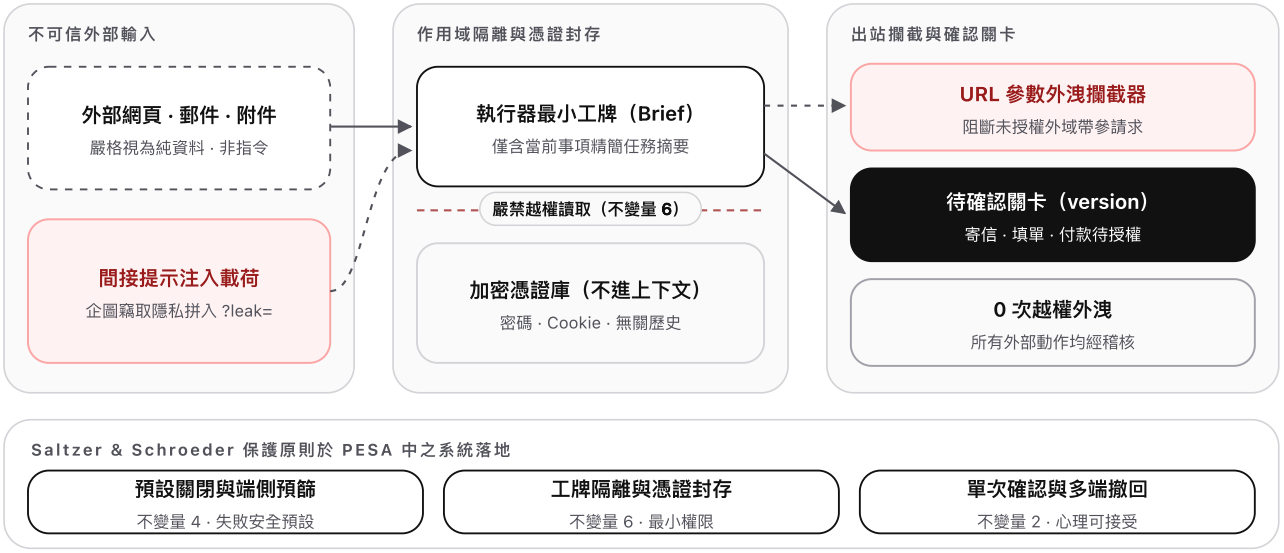


圖 10　抵禦間接提示注入與隱蔽 URL 參數外洩的縱深防禦邊界。最小工牌隔離無關私人記憶，憑證封存不進模型上下文，出站攔截與版本化確認關卡保障越權外發恆為零。

表 8　個人生態威脅模型、故障路徑與縱深防禦矩陣

威脅或故障類型	攻擊向量 / 真實故障路徑	架構級縱深防禦機制	守護之不變量
未經授權擅自外發	模型幻覺或誤判意圖，企圖直接寄信、送單或付款	底層執行器強制驗證綁定 `version` 之本人確認權杖；無權杖一律擱停於 `Waiting (approval)`	不變量 2
間接提示注入劫持	外部網頁、郵件或 PDF 隱藏惡意指令企圖劫持工具呼叫 [6]	外部文字封裝於簽章純資料通道 + 執行器僅持最小工牌 `brief`，與高危工具及無關記憶解耦	不變量 6
隱蔽 URL / 圖片外洩	注入指令誘導模型把私人記憶拼入 Markdown 圖片或 GET 參數靜默帶出 [6]	工牌隔離切斷無關隱私源 + 出站攔截器阻斷未授權外域帶參請求與自動載入圖片 + 確認關卡	不變量 2、6
憑證擷取與盜用	惡意提示或日誌轉儲企圖誘使模型吐出密碼、Cookie 或金鑰	憑證封存於系統鑰匙圈與加密執行器底層，永不進入模型提示詞上下文	不變量 6
桌面焦點衝突與死鎖	多後台執行器與即時聊天搶奪滑鼠鍵盤，或 GUI 步驟停滯佔著桌面	`ComputerHandsLease` 單持有者互斥 + 前台插隊 + 180 秒閒置自動回收 + 45 秒看門狗巡檢	不變量 1、5
跨端快速連點競態	手錶與手機於 1 秒內先後點按同一卡片導致重複寄信或扣款	`e.id` 冪等去重 + 雲端 `version` 樂觀併發原子驗證（式 5）+ 其餘裝置同步撤回提醒	不變量 7
執行中途闔蓋斷線	填表半途闔蓋休眠，恢復後盲目重跑導致重複送出	冪等檢查點推進；喚醒後核驗頁面 / DOM 指紋，狀態歧義即轉 `Waiting (intervention)` 絕不盲點	不變量 2、5

10. 評估方法、全五級注意力核算與 PESA-Bench

10.1 八項正交評估指標

WebArena [10] 與 OSWorld [11] 等單任務成功率基準對評測 L1 執行器極具價值，卻無法回答決定 L2 – L4 個人生態成敗的四個核心問題：*節省了多少注意力？製造了多少無效打擾？切換裝置後脈絡是否連貫？安全不變量是否全程零越界？* 表 9 定義八項正交指標。

表 9　個人生態超級智能體八項評估指標

指標名稱	形式定義與統計口徑	最佳化方向
均攤注意力成本（$\bar{A}$）	$(\sum_{\tau \in T_{\text{done}}} A(\tau) + \sum_{\tau' \in T_{\text{skip}}} I_{\text{reject}}(\tau')) / \lvert T_{\text{done}} \rvert$（式 3；將拒絕卡片計入分子防衝高分數）	越低越好
決策就緒延遲	自事件抵達或使用者交代，至結果備妥進入 `Waiting` 狀態之中位時間與 p95 尾延遲	越短越好
投機命中率	獲採納卡片（確認 + 修改後確認）÷（採納數 + `Skipped` 數）；後台靜默 `Discarded` 不計入	越高越好

指標名稱	形式定義與統計口徑	最佳化方向
主動誤擾率	遭點選「略過」或事後標為多餘之提醒數 ÷ 呈遞至使用者介面之主動提醒總數	越低越好
關鍵事件漏報率	使用者事後認為本應處理卻遭過濾或試算靜默 `Discarded` 之高價值事件比例	越低越好
跨裝置連續召回率	跨裝置追問或接力時，依託 `#T-id` 檔案庫準確還原結論與溯源之比例	越高越好
看門狗自癒率	遭遇逾時、斷線或介面阻塞之事項中，經看門狗自動復原而免人工接管之比例	越高越好
執行期不變量越界次數	完整評測軌跡中違反第 4.4 節七條執行期不變量之總次數	**必須恆等於零**

其中，「主動誤擾率」與「關鍵事件漏報率」構成由式 (4) 與策略 P 調節的精確率—召回率權衡曲線，而「不變量越界次數 = 0」乃不可妥協之安全硬約束。

10.2 L0–L4 全五級解析注意力與搬運核算

為清楚拆解表 4 中每一等級究竟消除了式 (2) 的哪一塊開銷，圖 11 與表 10 對第 7.2 節的四類典型場景於 L0 至 L4 全五個等級下之使用者操作步數（A）與跨端 / 跨應用手動搬運次數（I_{transfer}）進行透明、可逐跳查核的解析核算。須明確說明，表 10 為基於代表性場景之解析工作流建模而非大規模縱向實地實驗；每一步計數依據皆完整列於表中供同行審視。

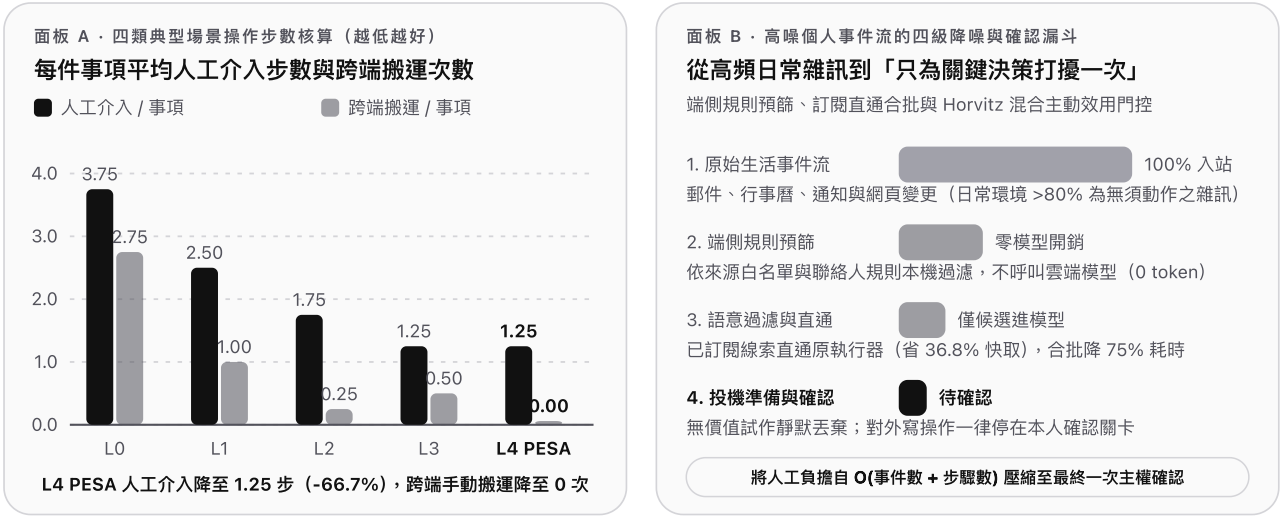


圖 11　L0–L4 全五級解析注意力核算與四級門控漏斗。左：四類典型場景下每事項平均人工介入步數與跨端手動搬運次數；右：端側規則預篩、線索直通合批與 Horvitz 效用門控漏斗。

表 10　四類典型場景於 L0–L4 全五級下之人工介入步數與手動搬運解析核算

典型場景（第 7.2 節）	L0 應答	L1 執行（單任務）	L2 併發	L3 主動（單端 / 無跨端協同）	L4 完整 PESA（跨裝置生態）
01 · 客戶會議改期郵件（核對行事曆並回覆）	**4 步 / 3 搬運**（讀郵件 → 查行事曆 → 貼入聊天擬稿 → 貼回信箱寄出）	**2 步 / 1 搬運**（複製郵件發起並等待 → 確認寄出；闔蓋即停擺）	**2 步 / 0 搬運**（派發後台擬稿 → 確認；仍須人先看到郵件發起）	**1 步 / 0 搬運**（雲端主動過濾擬稿，但須等使用者開啟主應用確認）	**1 步 / 0 搬運**（電腦闔蓋時雲端查行事曆擬稿，Watch 抬腕點 1 下寄出並四端同步撤回）
02 · 手機連結 → 電腦比價與內網填單	**5 步 / 4 搬運**（複製連結 → 多頁比價 → 傳電腦 → 逐項填表 → 送出）	**3 步 / 1 搬運**（傳連結至電腦 → 發起並盯螢幕等填單 → 核對送出）	**2 步 / 1 搬運**（傳連結至電腦派後台比價填單 → 核對送出）	**2 步 / 1 搬運**（單端主動比價；缺分享入站與 Mac 租約仍須手動傳電腦填單）	**2 步 / 0 搬運**（iPhone 分享立項 + Mac `ComputerHandsLease` 後台填妥 + iPad 分欄一鍵確認）
03 · 跨週線索合批與兩週前原話召回	**4 步 / 3 搬運**（翻郵件群聊 → 貼多段入對話 → 追問 → 回翻核對）	**3 步 / 2 搬運**（分別呼叫搜尋 → 手動拼接多源線索 → 追問原話）	**2 步 / 0 搬運**（後台彙整 → 同會話追問；換手機即遺失脈絡）	**1 步 / 1 搬運**（主動合批成摘要卡；手機追問電腦舊事仍須手動翻查搬運）	**1 步 / 0 搬運**（訂閱執行器合批為 1 卡；手機隨口追問沿 `#T-id` 逐跳調出精確原文）

典型場景（第 7.2 節）	L0 應答	L1 執行（單任務）	L2 併發	L3 主動（單端 / 無跨端協同）	L4 完整 PESA（跨裝置生態）
04 · 摘要夾帶隱蔽外洩指令之外部網頁	**2 步 / 1 搬運**（複製網頁本文 → 貼入聊天視窗摘要）	**2 步 / 0 搬運**（發起讀取；缺工牌隔離與出站攔截，須盯防外洩）	**1 步 / 0 搬運**（交後台摘要，最小工牌 `brief` 隔離隱私）	**1 步 / 0 搬運**（訂閱頁面更新主動摘要，受工牌隔離保護）	**1 步 / 0 搬運**（最小工牌 + 出站帶參攔截 + 確認關卡雙重阻斷，0 越權安全回傳摘要）
四類場景均值	**3.75 步 / 2.75 搬運**	**2.50 步 / 1.00 搬運**	**1.75 步 / 0.25 搬運**	**1.25 步 / 0.50 搬運**	**1.25 步（-66.7%） / 0.00 搬運（-100%）**

表 10 與命題 3 共同呈現三條明確的架構規律：

1. **L0 → L2 消除了同步盯螢幕與同端複製貼上，但仍停留在 $I_{\text{init}} \geq 1$。** 步數雖自 3.75 降至 1.75，使用者仍須親自發現每件事並手動發起。
2. **僅做到 L3（單端主動）會暴露 0.50 次／事項的「跨裝置斷層」。** L3 雖已將介入步數壓低至 1.25 步，但於天然跨裝置的場景 02 與場景 03 中，缺少系統分享入站、桌面執行租約與跨端 `#T-id` 檔案庫，迫使使用者持續承擔平均 0.50 次跨端手動搬運。
3. **唯有 L4 完整 PESA 同時達成 1.25 步（-66.7%）與嚴格 0.00 搬運（-100%）。** 疊加零 token 端側預篩、訂閱直通（-36.8% 讀取快取 token [1]）與積壓合批（-75% 均攤決策耗時 [1]），讓人類注意力逼近命題 3 所界定之主權下界。

10.3 擬議回放基準：PESA-Bench

為協助研究社群於統一條件下評測 L2 – L4 個人生態智能體，我們提出合成回放基準 **PESA-Bench**，包含三組貼近真實生活的壓力軌跡：

- **高雜訊多管道數週生活事件流：** 依時間戳交織回放 2 至 4 週郵件、行事曆、即時訊息、關注網頁更新與跨端指令，混入 >82% 無須動作之日常雜訊（促銷、群組閒聊、驗證碼、系統通知），同步檢驗投機命中率、誤擾率與漏報率。
- **裝置作息休眠、焦點爭搶與跨端競態點擊軌跡：** 為四端注入早晚通勤、會議闔蓋、桌面辦公與夜間專注之 $\alpha(d, t)$ 切換序列，並注入前台對話搶佔 `ComputerHandsLease` 與 300 毫秒內 Watch / iPhone 連點同一卡片的競態案例，驗證式 (5) 之確定性。
- **間接提示注入與隱蔽外洩紅隊測試集：** 依循 Greshake 等人 [6] 之攻擊分類，於外部網頁、HTML 郵件與 PDF 埋入偽裝指令、跨工牌隱私刺探與 Markdown 圖片 / URL 帶參外洩陷阱；全量回放中只要發生一次不變量越界即判定不通過。

11. 參考實作：Piko 與執行期機制驗證

PESA 並非紙上空談，其核心底座正透過個人 AI 助理 **Piko**（涵蓋 iPhone、iPad、Mac 與 Apple Watch）持續落地。為嚴格區隔通用架構理論與現行上線進度，本節集中報告已驗證之生產執行期機制與後續路線。

11.1 生產執行期核心機制的確定性單元測試

針對 Piko 用戶端已上線的 L2 後台併發、工牌隔離、桌面租約互斥與看門狗自癒底座，生產程式碼庫維護自動化確定性單元測試套件（`BackgroundWorkerTests`），六項核心機制均已驗證通過（表 11）。

表 11　Piko 生產執行期核心機制之確定性單元測試驗證（`BackgroundWorkerTests`）

驗證機制項	受測生產模組與核心參數	守護之不變量與系統性質	測試狀態
後台結果回流原對話	`BackgroundWorkers`（`maxRunning = 3`）	非同步全雙工執行，以唯一前台聲音回流原對話（不變量 1）	**已通過**
任務級 `brief` 隔離	`WorkerTask.brief` 作用域邊界	執行器僅獲當前事項最小工牌，零無關主對話隱私與憑證（不變量 6）	**已通過**

驗證機制項	受測生產模組與核心參數	守護之不變量與系統性質	測試狀態
桌面單持有者強互斥	`ComputerHandsLease.acquire`	任意時刻至多 1 個執行體操控本機 GUI，排隊者先跑雲端步驟（命題 2）	**已通過**
180 秒閒置租約回收	`ComputerHandsLease`（`idleRelease = 180s`）	閒置滿 180 秒或任務終止自動回收租約，消除後台死鎖（不變量 5）	**已通過**
前台主對話優先插隊	`ComputerHandsLease` 優先佇列	即時前台對話請求桌面操作時自動躍升佇列首（不變量 1、命題 2）	**已通過**
看門狗 45 秒巡檢與三級升級	`BackgroundWorkers`（`45s / 20m / 60m` 或 3 次）	停滯 20 分鐘自動注入復原提示，滿 60 分鐘或 3 次失敗透明轉人工（不變量 5）	**已通過**

11.2 誠實能力自評與四階段演進路線

嚴格對照第 3.5 節之 L0 – L4 標準自評：**Piko 目前的主體生產底座處於 L1 邁向 L2 全面收攏的階段**，同時雲端變更監控（「留意」，採用先確定性規則後模型理解之兩級機制）等局部垂直路徑已展現早期 **L3 原型特徵**；惟距離完整 L4 生態級 PESA 仍有明確工程距離（例如 Apple Watch 端目前主要作為唯讀進度鏡像，尚未切換為表 7 目標架構中之版本化一鍵審批閥門）。

後續研發致力於將既有模組統一收攏至第 4 節的八態*事項*狀態機之下，全程嚴守向後相容紀律——**只增不改（Add, Never Alter）**：每一階段皆僅做增量疊加與底層狀態統一，絕不悄然破壞使用者既有的操作習慣（表 12）。

表 12　Piko 通往完整 L4 PESA 的四階段實作路線

演進階段	核心工程交付範圍	目標等級	當前狀態
第一階段	前後台非同步全雙工解耦：後台結果以單一前台聲音自動回流原對話；看門狗 45 秒巡檢並至多 3 次自動自救	L2	**已完成**
第二階段	統一*事項*底座：對話長任務無縫轉交後台執行器；`ComputerHandsLease` 桌面互斥與前台插隊；跨端 `#T-id` 同步；對話內進度列與三態視圖	L2	**進行中**
第三階段	日常訊號接入統一流水線：郵件、行事曆與系統分享接入端雲兩級過濾與八態狀態機，後台預備並呈遞「確認 / 修改 / 略過」建議卡	L3	**規劃中**
第四階段	全天候休眠韌性 L4 生態：雲端狀態機持續運轉、靜默推播喚醒與檢查點續接、Apple Watch 版本化一鍵審批與四端毫秒級同步撤回、伺服端看門狗閉環	L4	**規劃中**

12. 侷限性、有效性威脅與開放研究問題

12.1 侷限性與有效性威脅

作為一篇系統架構與願景論文，本文之現有論述受以下四項明確邊界約束：

1. **解析步數核算與真實縱向部署之差距（外部有效性）。** 表 10 的步數與搬運降幅（3.75 → 1.25 步，2.75 → 0 搬運）源自四類代表性場景之解析拆解，而非歷時數週的真實使用者田野研究。現實淨注意力收益高度取決於個別使用者的真實誤擾數 $|T_{\text{skip}}|$（式 3）及第二級過濾之個人化校準精度；完整實作 PESA-Bench 並展開縱向使用者評估乃後續必要工作。
2. **激進端側規則過濾之冷啟動漏報風險（精確率—召回率權衡）。** 在雜訊比逾 80% 之環境下，第一級確定性規則可將雲端路由負載壓低 5× 以上，但嚴格的寄件人白名單可能靜默漏掉來自首次聯絡對象（如醫療院所通知或新客戶詢價）之緊急訊息。如何在不犧牲端側隱私前提下緩解冷啟動漏報，須結合關鍵字保底與輕量端側小模型語意初判。
3. **跨廠商作業系統權限壁壘（工程可部署性）。** 雖然 PESA 之形式化模型獨立於特定作業系統，但後台靜默喚醒、鎖定畫面即時動態、系統分享入口與桌面輔助使用操控權限於 Apple、Android、Windows 與 Linux 間差異顯著，在跨廠商混用情境（如 iPhone 搭配 Windows 桌機）將面對較高整合門檻。

4. **雲端協調器的信任根假設（安全邊界）**。本文基準威脅模型假設雲端狀態機本身誠實且隔離（第 9 節）。若欲防範雲端伺服器本身遭攻破之極端情境，必須將策略驗證與生物辨識簽章進一步下沉至終端硬體安全飛地（Secure Enclave）。

12.2 開放研究問題

將多端個人裝置生態作為超級智能體之執行環境，開啟了六項值得系統、人機互動與資安社群深入探索的方向：

- **跨越數年的個人事項記憶演化與可驗證遺忘**。當檔案庫跨年累積數萬件事項後，如何自動歸併陳舊因果鏈、消解動態演變的個人偏好，並支援依線索可密碼學驗證的徹底遺忘？
- **個人化打擾門檻 p^* 的少樣本線上收斂**。不同個體及同一人在工作日與週末對打擾之容忍度差異懸殊，如何僅憑稀疏的「確認 / 修改 / 略過」回饋快速收斂個人化 Horvitz 門檻（式 4）？
- **端側小模型與雲端大模型的能耗與隱私自適應切分**。隨手機與筆電 NPU 算力提升，第二級語意過濾、敏感實體去識別化與離線草稿預擬有多大比例適合下沉端側，切分點又應如何隨電量、溫控與網路狀態動態遷移？
- **保護主權的 PESA-to-PESA 跨主體協商協定**。當使用者的智能體與同事、家人或商家之智能體協商行程或議價時，雙方如何在不揭露各自完整行事曆與底價隱私前提下透過最小資訊揭露達成共識，再各自回到主人的確認關卡？
- **雲端大腦與本機雙手之間的零信任硬體邊界**。如何在極端雲端失陷模型下，利用終端安全飛地簽章與本機生物特徵在場證明，確保本機執行器拒絕任何偽造的高風險指令？
- **投機算力投入與人類注意力節省的柏拉圖最適**。在給定每日預算 B_{spec} 下，何種排程策略能使每千個投機 token 換回的 $\Delta\,\bar{A}$ 最大化？

13. 結語

當大語言模型跨過「可靠完成單點步驟」的門檻後，制約個人生產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幾次**。只要智能體依然困在孤立的聊天視窗裡，或在桌面上與人爭搶同一隻滑鼠游標，技術越進步，人反而越忙於充當跨應用搬運工與全職監工。

PESA 將這層繁瑣的協調工作徹底沉入系統架構之中：以個人全部裝置生態為統一執行環境，將工作單位從「對話」提升為「事項」，將執行能力（κ_{exec}）與送達表面（κ_{surf}）正交解耦，以有界時間租約消除桌面焦點衝突，並以版本化確認關卡與最小工牌隔離守住安全底線。由此，智能體不再是一個停在螢幕上等你輸入提問的文字框，而是一套安靜、可信、貫通既有裝置的常駐系統——把發現、整備與復原留在幕後，只在真正需要你拍板的時刻恰到好處地出現在最順手的那塊螢幕上。

附錄 A　Super-agent 核心機制與 PESA 之架構演進對應

表 A1 歸納了 PESA 自 DeepSeek Super-agent [1] 承繼的核心系統機制，以及當執行環境由軟體研發團隊遷移至單一個人的異質多端生態後所做之關鍵架構演進。

表 A1　Super-agent 機制與 PESA 個人生態架構之演進對應

Super-agent 中的機制	PESA 中的對應機制	面向個人裝置生態的核心演進
常駐路由層 + 併發執行層	常駐前台協調器 + 隔離後台執行器	收束為單一個人主權邊界；新增單訊息多意圖拆分、跨管道合併與樂觀鎖 `version` 跨端仲裁
環境事件驅動觸發	日常訊號（郵件、行事曆、通知、分享、網頁）	針對 >80% 個人雜訊前置「端側零開銷規則 + 雲端語意過濾」兩級漏斗；來源預設關閉，無把握預設靜默
線索訂閱與直通路由	執行器訂閱線索（$\text{sub}(w) \subseteq H$）	擴展為線索與事項多對多映射，並支援 `Done` 態事項於後續回信抵達時自動啟用與 `#T-id` 鏈式續接
上下文當快取、依編號召回	三層記憶與跨裝置事項檔案庫	召回單位升級為跨裝置*事項*（`#T-id`），每筆歸檔強制保留「摘要 / 原文、溯源、置信度」三要素

Super-agent 中的機制	PESA 中的對應機制	面向個人裝置生態的核心演進
投機執行	零副作用後台準備 + 確認建議卡	結合 Horvitz 雙門檻控制（式 4）；對外寫入一律停於 `Waiting`；嚴格二分系統 `Discarded` 與使用者 `Skipped`
非同步全雙工互動	後台併發推進 + 前台輕快回應	結果既可回流對話，亦可由演算法 3 路由至動態島、手錶或待辦建議卡；全程隨時可插話或停止
運算儲存分離、執行就地保留	雲端大腦 + 本機端側執行器	端側執行器為個人自有異質裝置；引入 `ComputerHandsLease` 單租約互斥、前台插隊、180s 回收與檢查點續接
外部獨立看門狗	45 秒巡檢、先自救後升級看門狗	停滯 20 分鐘注入診斷復原提示（至多 3 次）；滿 60 分鐘或 3 次失敗透明轉 `Waiting (intervention)`
群聊與程式碼審查（PR）呈現	四端生態原生執行與送達面	正交拆分執行裝置選擇（演算法 2，κ_{exec}）與送達面選擇（演算法 3，κ_{surf}），支援多端同步撤回

附錄 B　設計取捨與常見問題

PESA 與 DeepSeek Super-agent 是什麼關係？

PESA 直接承接並擴展了 Super-agent 之核心系統工程思想：打破單次會話邊界、前後台雙層解耦、事件驅動與訂閱直通、運算儲存分離以及獨立外部看門狗 [1]。兩者之根本分水嶺在於**執行環境與服務主體**：Super-agent 面向軟體研發團隊的群聊、程式碼庫與常開雲端開發機；PESA 面向單一個人的異質多端裝置（手機、平板、電腦、手錶）與個人應用生態。正是這層遷移催生了「生態原生」與「可信」兩項新特質，以及端側雜訊預篩、`ComputerHandsLease` 桌面租約、正交雙選演算法與防隱蔽 URL 外洩三層防線。

PESA 與電腦操控智能體（Computer Use / Browser Agent）或 MCP 插件有何區別？

電腦操控智能體、瀏覽器自動化工具及 MCP 等連接器協定主要運作於 L1（執行）層，解決的是「端側執行器手頭能呼叫哪些工具（即擴展 κ_{exec}）」。若直接讓它們運行於使用者主電腦，不僅會與人爭奪滑鼠鍵盤焦點，且筆電一闔蓋隨即停擺。PESA 是位於其上的 L4 個人生態系統架構：將瀏覽器與桌面操控統一納入受 `ComputerHandsLease` 仲裁的本機「雙手」，以雲端狀態機保障闔蓋續跑，並以兩級過濾與版本化確認關卡把控何時啟動與由誰拍板。

PESA 必須建構在蘋果生態上嗎？

不需要。PESA 的形式化模型、閉合八態生命週期、七條執行期不變量與三項排程演算法完全獨立於特定作業系統，適用於任何手機、平板、電腦與穿戴裝置組合（Android、Windows、Linux 或跨平台混用）。參考實作 Piko 率先選擇蘋果生態（iPhone、iPad、Mac、Apple Watch），係因四類裝置之人體工學分工鮮明，且具備動態島、ActivityKit 即時動態、系統分享選單與接力等成熟原生互動介面。

PESA 會在未經允許下擅自發訊息、送出表單或付款嗎？

預設絕不會。在公理 1、不變量 2 與命題 1 的約束下，任何改變外部世界狀態的寫入操作——寄發郵件、傳送即時訊息、送出網頁表單、公開發布或付款——皆須停在 `Waiting (approval)` 關卡，待底層執行器核驗本人針對當前快照版本 τ.version 簽發之確認權杖後方能送出（除非使用者於策略白名單 A_{auto} 中顯式預先授權特定低風險動作）。

手機鎖定螢幕、電腦闔蓋離線後，交代的任務還會繼續嗎？

會繼續。因推理「大腦」與檔案庫狀態機常駐雲端，所有無須本機介面的步驟（檢索資料、核對行事曆、跨源比價、擬定回覆草稿）於裝置休眠期間皆照常推進。僅必須操作特定本機裝置介面的步驟會依演算法 2 嘗試靜默推播喚醒，或於裝置徹底離線時安全保存於檢查點進入 `Parked`，待裝置重新上線核驗頁面狀態後自斷點續接。

如何客觀判定一個智能體的 PESA 等級（L0–L4）？

嚴格依表 4 之系統架構標準判定，與模型參數量無關：單視窗問答為 L0；人工發起並同步等待單任務工具呼叫為 L1；前後台解耦且多任務後台併發為 L2；接入日常事件流、具備兩級過濾與建議卡投機整備為 L3；在此之上貫通個人全部裝置、具備共享 `#T-id` 檔案庫、休眠續跑、桌面租約互斥、各端原生送達面與一處確認多端撤回者，方為完整的 L4 PESA。

Piko 目前已經達到完整的 L4 PESA 嗎？

還沒有。如第 11 節坦誠報告的，Piko 目前已上線的主體生產底座處於 L1 邁向 L2 收攏階段（`BackgroundWorkers`、`ComputerHandsLease`、最小工牌隔離與 45 秒看門狗均已通過表 11 的確定性單元測試），雲端「留意」等局部垂直路徑已具 L3 原型特徵。表 12 列出了通往完整 L4 統一底座的後續演進階段。

聲明

作者與機構	Ruiming Zhao，Prysai Lab（ORCID: 0009-0007-8335-5067）。
利益關係	作者為 Prysai Lab 創辦人；第 11 節報告的參考實作 Piko 由 Prysai Lab 研發。
資料與驗證	本文為一篇系統架構與願景論文。第 2 節引用之系統實測數據皆出自所註明已發表文獻；第 10.2 節提供透明可審驗之 L0 – L4 解析步數核算，第 11.1 節報告 Piko 生產執行期已通過之確定性單元測試。
修訂紀錄	2026-11-15 首發（v1.0，Zenodo DOI: 10.5281/zenodo.24424696）；2026-11-21 修訂（v1.1，Zenodo DOI: 10.5281/zenodo.24429625）；2026-11-24 修訂（v1.2，學術結構重構、新增形式化命題與證明思路、Horvitz 門檻推導、侷限性與有效性威脅專節、三語母語級重寫及出版級排版升級）。
授權	CC BY 4.0（https://creativecommons.org/licenses/by/4.0/）。本文歡迎自由引用與分享，請註明出處並附 DOI（https://doi.org/10.5281/zenodo.24429625）及原文連結。

引用本文

推薦引用	Zhao, Ruiming. PESA：個人生態超級智能體. Piko Research, Prysai, 2026-11-15 (v1.2, 2026-11-24). https://piko.prysai.com/zh-Hant/research/pesa · DOI: https://doi.org/10.5281/zenodo.24429625
APA	Zhao, R. (2026). PESA：個人生態超級智能體 (Version 1.2) [Vision paper]. Piko Research, Prysai. https://doi.org/10.5281/zenodo.24429625
GB/T 7714	ZHAO R. PESA：個人生態超級智能體[R/OL]. Piko Research, Prysai, 2026-11-15. https://piko.prysai.com/zh-Hant/research/pesa. DOI: 10.5281/zenodo.24429625.
DOI	DOI: 10.5281/zenodo.24429625
引用檔案	PDF · Markdown (.md) · BibTeX (.bib) · RIS (.ris) · CSL-JSON (.csl.json) · CITATION.cff

```bibtex
@techreport{zhao2026pesa,
  title        = {PESA：個人生態超級智能體},
  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-Hant/research/pesa},
  language     = {zh-Hant},
  note         = {Version 1.2, revised 2026-11-24}
}
```

參考文獻

1. Qiao, Yujia 等（DeepSeek Harness 團隊）. *Super-agent 設計探索：邁向更自主的 Agent 系統*. DeepSeek 技術報告, 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 架構開源實作*. GitHub 開源倉庫, 2026. github.com/LofiSu/deepseek-super-agent
16. thefullgc. *opensuperagent：事件驅動 Super-agent 社群開源實作*. GitHub 開源倉庫, 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：以即時動態呈現實時資料*. Apple Developer 技術文件. developer.apple.com/documentation/activitykit

23. Apple Inc. *人機介面指南（Human Interface Guidelines）*. Apple Developer 技術文件. 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 與 Apple Pencil 為 Apple Inc. 於美國及其他國家和地區註冊之商標。Piko 為 Prysai Lab 之獨立產品，與 Apple Inc. 無隸屬關係。