
# Web3 x AI 系列 | 為什麼人工智慧代理需要經濟基礎設施

人工智慧是目前最熱門的話題，持續吸引資本、人才和關注，但Web3仍然被上一輪的失敗所束縛。人們對加密貨幣感到厭倦是有一定道理的，尤其是在多年來基礎設施建設幾乎從未實際應用之後。但這種比較過於侷限於當前的討論，忽略了人工智慧一旦超越聊天介面就會出現的問題。

聊天機器人回答產品內部的提示，而智慧代理則可以要求運算資源、呼叫API、購買資料集、將任務分配給其他模型、檢查結果、交付成果，並在使用者離開後繼續運行。

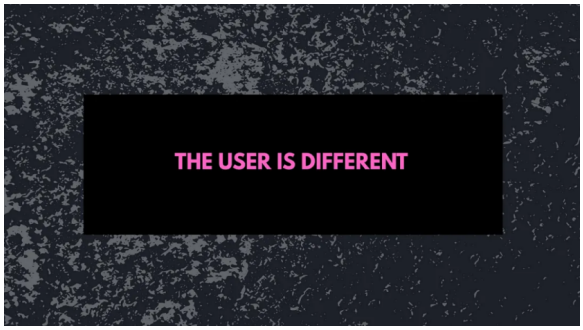
平台可以暫時將這些活動隱藏在一個帳戶背後，內部計量使用量，並在月底將所有費用計入用戶的信用卡。當智慧代理在受控環境中運行，平台承擔所有後續工作時，這種方式就足夠了。

但一旦智慧代理跨服務運行，這種機制就會失效，因為經濟活動會帶來模型品質本身無法解答的問題。哪個帳戶在執行操作？誰授予了它權限？它可以消費什麼？任務執行過程中發生了什麼事？輸出結果被使用後誰獲得報酬？其他系統可以依賴哪一個記錄？

人工智慧賦予軟體更強的行動能力，但只有當行動能夠跨系統進行授權、支付、記錄、核查和結算時，才能真正發揮商業價值。 Web3 的優點在於它能夠比平台帳戶、訂閱或私人資料庫更好地處理這些環節。但這並不意味著區塊鏈應該應用於所有人工智慧工作流程，而是意味著自主軟體開始暴露圍繞人類用戶、平台帳戶和計費系統構建的基礎設施的局限性——這些系統假定仍然有人能夠批准下一步操作。

大多數代理商不需要自己的新代幣，許多代理商仍將留在封閉的產品中，許多人工智慧工作流程使用普通資料庫、普通帳戶和普通計費系統將會更有效率。真正的挑戰在於代理需要支出資金、接收資金、購買資源、僱用其他代理，以及向受控環境以外的交易對手證明工作成果。

到那時，代理人需要經濟基礎設施，軟體可以直接運行，規則可以被其他系統讀取，而不是信任隱藏在一家公司的後端。




大多數網路基礎設施仍然假定使用者就在附近，需要使用者註冊郵箱地址、新增付款方式、接受條款、點擊介面、等待確認，並在產品需要時建立新帳戶。這種模式雖然混亂，但人們已經習慣了。他們可以閱讀警告訊息、重設密碼、批准付款、聯絡客服，並理解一個半成品的結帳流程。

代理商則對整個系統的不同部分提出了更高的要求，因為他們不需要為了介面而介面，也不關心按鈕是否友善。他們需要的是可以閱讀的規則、可以執行的權限、可以查詢的餘額、可以呼叫的服務，以及無需用戶坐在螢幕前也能正常運作的支付系統。代理商並不要求網路操作簡單，他們需要的是網路以軟體可以執行的方式呈現規則、餘額、限額和承諾。

私鑰、簽章、gas費、智慧合約呼叫、多重簽章邏輯和可程式帳戶對一般使用者來說極為繁瑣，主要是因為它們暴露了太多底層機制。而代理程式則更接近這些機制運行，它們可以簽署交易、監控餘額、檢查合約狀態、重試失敗的調用，並執行條件邏輯，而無需將每一步轉化為用戶可感知的用戶體驗。

對於只想使用應用程式的用戶來說，錢包功能過於複雜；但對於需要在限額內支出的軟體而言，策略控制的帳戶可以成為一個有效的操作邊界。

對於只想一鍵完成操作的用戶來說，gas費令人煩惱；但贊助gas費或帳戶抽象可以將交易成本轉化為後台策略。

智能合約對消費者來說介面笨拙，但對於只關心規則是否執行的代理程式來說，它可以作為一個服務端點。


人工智慧領域最容易犯的錯誤是將智慧本身視為問題的全部。一旦模型能夠編寫程式碼、分析文件、生成設計或規劃工作流程，人們就會覺得最困難的部分已經解決。然而，模型一旦離開沙盒環境，那些常見的經濟問題又會重新出現。

即使智能體找到了合適的資料集，仍然需要存取條款和付款方式。如果它選擇計算服務提供者，就需要製定並執行預算。如果它將工作交給專業的評估人員，就需要資金管理、支付款項並保留支票記錄。如果輸出結果之後重複使用，系統還需要歸屬機制和收益分配機制，否則工作成果將與其產生的輸入、支付和支票脫節。

平台可以透過以下方式在自身內部解決部分問題：為智能體提供內部帳戶、在資料庫中計量使用情況、按月向客戶收費，並透過支援服務解決糾紛。當代理僅限於單一產品時，這種方法行之有效；但當代理程式跨越信任邊界時，其效力就會減弱，因為每個外部服務都有其自身的帳戶系統、計費方式、權限模型以及私密的事件記錄。

API 金鑰是為軟體存取而設計的，但它們並非旨在將軟體轉化為可問責的經濟主體。支付系統可以向個人或公司收費，但無法賦予小型代理人對單次任務付款的完全自主權。 SaaS 計費模式適用於訂閱服務，但對於那些可能只想為評估、資料提取、推理呼叫或從其他代理處獲得的特定工作支付幾美分的代理來說，這種模式並不適用。

錢包改變了問題的性質，因為它為代理商提供了一個可尋址的帳戶，該帳戶可以承載價值並與跨服務的合約互動。

Uptick 的 AA Wallet 就是一個很好的例子，它將帳戶推向了更接近策略層的位置，帳戶價值與軟體的支出、可調用的服務以及何時需要其他審批層介入掛鉤。該錢包並沒有讓模型更智能，也沒有消除產品設計的必要性，但它為代理商提供了一種在單一公司資料庫之外承擔責任的方式。


即使人工在聊天視窗中閱讀，人工智慧的輸出結果也很難驗證，一旦代理商開始在不同系統間傳遞工作，驗證的壓力就會更大。結果可能看起來合情合理，但下一個代理需要的不僅僅是答案，它還需要知道使用了哪些數據、調用了哪些工具、哪個評估人員檢查了輸出結果、是否已支付款項，以及工作流程的任何部分是否失敗、更改或事後獲得批准。

中心化平台可以提供自身產品的日誌，但這些日誌通常止步於平台邊緣。一旦工作在代理、工具、資料提供者、計算提供者和人工審核人員之間流轉，記錄就會分割成私有的部分。每位參與者都可以證明自己的部分，但除非某個平台擁有相關的帳戶、付款、權限和歷史記錄，否則整個工作流程將難以重構。

Web3 的作用比為代理程式的所有操作提供完整的稽核追蹤要窄。某些執行過程應該保持私密，某些資料應該鏈下存儲，某些檢查仍然依賴外部系統。更強大的應用場景是共享選定操作、付款、權限和證明的記錄，不同的系統可以讀取這些記錄，而無需依賴單一公司來保存完整的歷史記錄。

一旦輸出結果包含付費資料、租用運算資源、外部評估和手動審核等因素，所有權的認定就變得更加複雜。如果沒有可移植的記錄，每個下游使用者只能依賴平台內部對事件的描述。有了更強大的溯源層，歸屬、許可、收入分配和爭議解決將更容易評估，因為作品本身承載了更多歷史記錄。


一款優秀的 Web3 x AI 產品在移除區塊鏈後效能應該會有所下降。如果區塊鏈層主要增加成本、拖慢工作流程，或創造了類似代幣的融資管道，那麼該產品只是把 Web3 當作噱頭。區塊鏈必須能夠處理一些系統難以取代的功能，例如其他服務可以識別的帳戶、代理商無法繞過的支出規則、雙方都能驗證的支付條件，或者工作流程離開公司後端後仍然有效的記錄。

讀取錢包資料的聊天機器人或許很方便，但這並不能說明代理商的基礎設施狀況。與 AI 產品綁定的代幣或許能夠創造市場，但這並不能證明該代幣能夠協調任何事務。在社群媒體上發文的機器人或許有助於分發，但發文和回覆並不會改變工作的支付、審核或記錄方式。

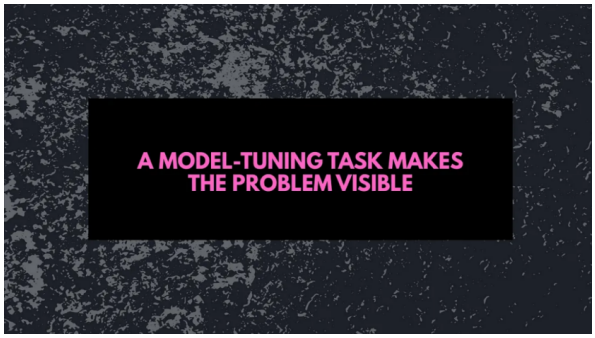
真正的考驗始於工作流程內部。

當代理商購買服務時，錢包會為其分配一個其他系統可以存取的帳戶。當從該帳戶支出時，智慧帳戶可以限制金額、已批准的服務以及需要額外審批層的操作。當任務依賴另一方時，支付條件可以凍結資金，直到工作完成。當輸出流向下一個系統時，記錄會提供足夠的歷史訊息，以便下一個系統在信任之前判斷發生了什麼。

當使用者只想要一個感覺正常的應用程式時，開放式結算、可程式帳戶、可移植歷史記錄和可組合合約就很難被接受。代理商並不要求普通的消費者流程，他們需要的是機器可讀的承諾和執行路徑，而區塊鏈只有在提供這些路徑的同時，又不強制所有參與者使用同一平台時，才能發揮作用。

內部客戶支援人員不需要鏈上結算，財務團隊內部的報告人員最好還是留在公司現有的技術堆疊中。當工作流程跨越多個組織、資金按操作流動、任務從一個代理商傳遞到另一個代理

商，或者需要在原始平台不再參與後仍然保留記錄時，Web3 才有資格被實施。




一個簡單的模型調優任務就足以暴露協調問題。

使用者請求代理針對特定用例提升模型效能，代理需要識別效能差距，請求或購買數據，租用計算資源，運行訓練或微調，將結果發送給評估人員，針對特殊情況引入人工審核，交付更新後的模型，並將款項支付給所有參與其中的人員。

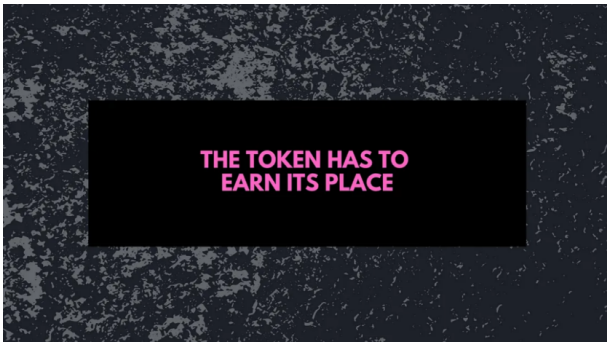
這些操作並非天方夜譚，但困難在於協調，因為在傳統的平台架構中，每個步驟都依賴不同的帳戶。計算資源提供者使用一套計費系統，資料提供者使用另一套，評估人員使用另一套，而人工審核人員的報酬則完全來自其他管道。代理程式可以呼叫 API，但經濟往來卻分散在私有的儀表板、發票、使用日誌和支援郵件中。

代理原生架構將從智慧帳戶和既定策略開始。代理商可以設定消費限額，使用已核准的供應商，超過閾值時需要申請批准，並透過雙方都能讀取的支付通道進行支付。支付方式需要轉變，因為當代理商的工作分解成涉及多個服務的小操作時，他們並不適合訂閱、發票和信用卡表單等付款方式。他們可能需要按請求、按推理、按評估、按數據提取或按下游使用付費。

如果服務流程圍繞著 Uptick 微支付服務構建，那麼重複的小額請求就不必變成單獨的結帳環節。一次授權即可涵蓋在批准限額內的多筆小額支付，驗證可以在服務使用過程中進行，結算可以稍後批量處理，而不是強制每個操作都進入單獨的鏈上交易。工作交付後，支付記錄、輸入、檢查和發布條件可以隨輸出一起遷移，而不是被困在每個提供者的私有系統中。

將所有細節放在公共帳本上是錯誤的，因為大多數代理商的工作流程都需要私密執行、選擇性揭露、資料加密、鏈下儲存以及防止敏感資訊被公開的策略。其有用之處在於共享的經濟框架，參與者可以就誰執行了操作、支付了什麼、資金釋放的條件以及哪個帳戶擁有支出權限達成一致。

帳戶抽象化使該框架更加安全，因為代理程式的操作權限不必等同於帳戶擁有者的全部權限。智慧型帳戶可以設定限額、批量操作、支付 gas 費、對異常行為要求額外批准、輪換密鑰以及在故障後恢復存取權限。對於使用者而言，這些功能使錢包的使用者體驗更加友善；而對於代理商而言，這些功能定義了其實際可以執行的操作策略。



Web3 x AI 的弱化版本在系統尚未具備任何協調能力之前就急於使用代幣。一旦代理人開始出現，這種本能反應就更容易被合理化，因為自主軟體聽起來似乎應該擁有自己的資金、市場和治理體系。但更難的問題是，代幣在工作流程中究竟扮演什麼角色。

當代幣支援共享基礎設施的存取、結算、驗證、收益分配或治理時，它就具有意義。計算提供者可能需要基於使用量的報酬，資料提供者可能需要在代理商使用其工作成果時獲得報酬，評估人員可能需要激勵措施來誠實地檢查輸出結果，工具開發者可能需要一種方式，讓代理人在不同產品中調用他們的服務時獲得收益。

測試方法很簡單。

如果代幣消失後，工作流程仍然運作得更好、更便宜、更順暢，那麼它可能只是一個噱頭。如果移除代幣會破壞存取、結算、驗證或共享所有權，那麼它實際上可能是基礎設施的一部分，而不僅僅是附加在它上面的市場概念。


第一波人工智慧價值的湧現集中在技術堆疊中最容易被感知的部分。模型需要電力、資料中心、晶片和工程人才，因此最接近這些資源限制的公司抓住了市場需求。這一層仍然很重要，但隨著模型存取變得更加便捷，價值不再局限於此。

隨著技術棧的成熟，瓶頸向上轉移。更棘手的問題不再只是誰能訓練模型或提供GPU，而是代理商如何存取服務、支付資源費用、證明工作成果、分配價值、儲存記錄以及與平台以外的系統進行協調。

正是在這個時候，真正有用的基礎設施開始更靠近代理商本身。

代理需要一個圍繞模型運行的環境，其中包含工具、記憶體、帳戶、權限、支付、策略、資料存取、評估和結算等功能。其中一些功能仍將保持集中式，因為集中式基礎設施通常更便宜、更易於控制。當代理商需要與不共享相同資料庫的參與者協調時，開放的支付通道（Open Rails）就變得至關重要。因為如果工作流程需要在多個服務之間流動，帳戶、付款和記錄層就無法完全位於同一個後端。

隨著代理功能日益強大，平台控制的價值也隨之提升。如果一家公司擁有代理商介面、帳戶系統、支付通道、任務歷史記錄和分發管道，它就可以決定代理商可以使用哪些服務、哪些貢獻者可以獲得報酬、哪些記錄可見以及哪些外部系統可以參與其中。這或許效率很高，但也重新建構了Web3一直聲稱要消除的平台依賴性，只不過現在這種依賴性存在於可以代表使用者執行操作的軟體之下。

Web3不必成為人工智慧的大腦，它必須在代理需要錢包、可編程支出規則、可驗證的任務記錄、託管、收入路由和跨服務結算等功能時發

揮作用。這本身就足以完成大量工作，也讓討論聚焦於真正的瓶頸，而不是泛泛地談論網路的未來。

下一代實用的 Web3 x AI 產品乍看之下或許平平無奇，但它們將使代理更容易管理帳戶餘額、支付 API 呼叫費用、證明任務已透過審核、在貢獻者之間分配收益，或在不同服務之間共享信譽記錄。 Uptick 的 AA Wallet、AI Agent Wallet 和相容 x402 的微支付服務就屬於此操作層。在這一層，代理在執行操作前需要設定帳戶限額，支付需要隨請求轉移，並且記錄在工作流程離開後端後仍可讀取。

只能透過單一平台帳戶進行操作的代理商在該平台內功能強大，但在其他平台則顯得笨拙。一旦需要購買服務、證明工作、路由支付或跨系統共享歷史記錄，後端無法獨自處理所有權限、爭議和記錄。

這正是市場比較所忽略的部分。一旦代理跨服務工作，帳戶、付款和事件記錄就需要隨工作轉移，否則代理的效用仍然僅限於其所依賴的平台帳戶。

Uptick Network

hello@uptickproject.com

@Uptickproject

@Uptickproject

Uptick Network

Uptick Network