重新定義系統架構：從萬能定義到通觀層面與萬能模組

引言：邁向通觀層面的演進

本報告旨在正式介紹一項關鍵的架構重新定義：從現有的「萬能定義」與「三世界」範式，轉變為一個整合性的新框架，稱之為「通觀層面」。這項轉變不僅是語義上的，更代表著複雜系統在概念化、設計與管理方式上的根本性哲學與實踐轉向。

過去的「萬能定義」可能暗示著一種單一、包羅萬象的定義，潛在地難以應對現代系統固有的複雜性與動態特性；而「三世界」則可能代表著相互獨立、甚至可能孤立的領域。轉向「通觀層面」正是為了解決這些限制，強調系統內部的互聯性與其湧現特性。

本文件將全面分析「通觀層面」，詳細闡述其基本原則、構成層面——「概念面」、「執行面」和「數據面」——以及「萬能屬性」和「萬能分析」在定義「萬能模組」中的轉化作用。本報告旨在為系統架構師和開發人員提供一份基礎指南，確保對此新框架的共同理解和一致應用。

這項重新定義標誌著一項策略性舉措，旨在實現更整合、適應性更強、更易於維護的系統設計，這對於現代複雜系統，特別是那些利用先進人工智慧能力的系統至關重要。它承認系統不僅僅是其各部分的總和，其整體行為源於錯綜複雜的相互作用。

此一轉變從「孤立的螢幕或功能」轉向「所有事物如何連結」，這不僅是一種設計偏好，更是處理複雜系統的根本範式轉變。傳統設計通常將問題孤立化，一次處理一個元素。相較之下，通觀設計則考慮體驗的所有部分如何跨時間、跨通道以及使用者情感協同運作，並審視整個生態系統：產品、服務、品牌和環境 。這與使用者將「萬能定義」（可能暗示單一、整體定義或孤立組件）和「三世界」（三個離散領域）轉變為「通觀層面」的需求完全一致。這意味著從孤立或碎片化的視角轉向整合的、相互連接的生態系統視角。這種轉變表明，為了管理現代技術系統日益增長的複雜性和相互依賴性，特別是那些整合了人工智慧的系統，從以組件為中心或以領域為中心的視角轉向以生態系統為中心的視角，是至關重要的策略性舉措。

表1：框架演進：新舊術語對照

舊術語	新術語	舊術語簡述（暗示意義/限制）	新術語簡述（核心概念/優勢）
萬能定義	通觀層面	可能暗示單一、僵化的定義，難以適應複雜系統的動態性。	強調系統的整體性、互聯性與湧現特性，提供全面視角。
三世界	概念面	可能代表獨立、孤立的領域或功能筒倉。	系統的智慧與推理核心，定義系統的理解、決策與知識運用。
	執行面		將概念層的智慧轉化為具體行動，管理系統的操作流程。
	數據面		系統的感測與記憶系統，負責所有數據的獲取、準備、儲存與管理。
核心功能	萬能屬性	僅關注功能本身，缺乏普適性與跨域適用性。	定義核心功能的普適性、基本特性，使其具備

舊術語	新術語	舊術語簡述(暗示意義/限制)	新術語簡述(核心概念/優勢)
			廣泛適用性與適應性。
	萬能分析		貫穿系統生命週期的全面、數據驅動分析能力，驅動持續優化。
	萬能模組		結合萬能屬性與萬能分析，實現可重用、適應性強、高效能的核心功能單元。

通觀層面:一個基礎框架

在系統架構中,「通觀層面」是一種總體方法，它考慮到整個系統、其組件、它們之間錯綜複雜的相互作用，以及系統運作的廣泛背景 。它超越了孤立的元素，以理解所有事物如何連結以實現總體目標和使用者需求 。這種視角承認，系統的真實行為和性能往往是其各部分複雜相互作用所產生的湧現特性，而非僅僅是各個組件功能的總和 。

通觀方法的關鍵原則

- 系統思維：將產品或系統視為更大生態系統的組成部分。從使用者介面到支援、內容和品牌，每個元素都必須協同運作 。這包括深入了解系統的架構，包括其組件、介面和相互作用。
- 使用者旅程導向：為跨所有接觸點的整個使用者體驗進行設計，而不僅僅是單個螢幕或任務。這包括繪製使用者如何通過不同階段和上下文移動的路徑，識別摩擦點、愉悅時刻以及實現一致性的機會 。
- 相互依賴性分析：深入分析所有系統組件之間的相互作用和相互依賴性，以理解一個領域的變化如何影響其他領域 。
- 湧現特性理解：認識到系統的整體行為和能力可能源於其各部分複雜的相互作用，其方式往往無法僅從單個組件預測 。
- 多學科和多視角整合：結合多樣化的觀點，包括技術、社會、經濟和環境影響，以確保對系統的背景和影響有全面的理解 。

採用通觀層面對於複雜系統開發的益處

- 提升效率與最佳化：透過考慮整個系統，工程師可以識別潛在問題和改進機會，從而最佳化系統性能並減少浪費 。
- 改進決策與問題解決：通觀視角能夠做出更明智的決策，並通過從多個角度理解系統行為來解決複雜問題 。
- 增強永續性與可靠性：考慮相互作用和相互依賴性有助於識別和減輕風險，確保系統在其整個生命週期中保持穩健和永續 。
- 一致的使用者體驗：確保所有接觸點的一致性、可訪問性和情感共鳴，從而提供無縫且有意義的使用者體驗 。
- 適應性與未來驗證：以整個生態系統為念進行設計，使得系統更容易適應不斷變化的需求並整合新技術。

通觀設計建議了一個清晰、協作的流程(理解上下文、繪製旅程、共同創作、原型測試、一致性實施、持續演進)。這個流程為「通觀層面」的實際應用提供了一個實踐框架，而不僅僅是一個理論概念。它強調了持續演進和反饋循環的重要性。例如，理解上下文和繪製旅程直接關聯到概念理解

和數據收集，而共同創作解決方案和原型測試則涉及概念的細化和執行層面的實踐。持續實施和不斷演進則描述了執行層面的持續操作，以及數據層面所促進的數據驅動的迭代改進。這種以流程為導向的視角將「通觀層面」從一個抽象概念轉變為一個可操作的框架，這對於指導複雜系統的開發生命週期至關重要。
此外，通觀思維不僅僅是設計，更是一種解決問題和系統設計的「方法論」，它強調「複雜關係」和「湧現特性」。這將「通觀層面」從單純的概念轉變為一種嚴謹的實踐方法，用於開發和管理複雜系統。對「湧現特性」和「複雜關係」的強調表明，新框架專門設計用於處理大型系統中固有的非線性行為、不可預見的結果以及錯綜複雜的動態，特別是那些涉及人工智慧的系統。這為重新定義的必要性和益處提供了有力的理由，突顯了其解決傳統還原論方法可能忽略的挑戰的能力。
通觀思維的另一個核心原則是「整合多學科和多視角」，這明確包括了「社會、經濟和環境影響」以及技術性能。這表明「通觀層面」不僅關注技術組件，還關注系統的更廣泛影響和上下文。這為「概念面」和整體系統理解增加了一個關鍵維度。這意味著「通觀層面」旨在引導創建更負責、更道德和更永續的系統，這在現代系統設計中是一項重要的策略優勢，超越了純粹的功能需求，涵蓋了更廣泛的利害關係人考量和長期可行性。

通觀層面的層次結構

「通觀層面」通過三個相互連接的層次結構實現：概念面、執行面和數據面。這些層面協同運作，以確保全面的系統理解、有效的運作和持續的改進。

概念面 (概念面)

- 作用：此層主要關注系統的智慧、推理能力和知識表示。它定義了系統「理解什麼」、「如何決策」以及「利用什麼知識」。它是系統的大腦，提供高層次的理解和戰略方向。
- 組成與功能：
    - 知識表示：利用知識圖譜、本體論和語義網絡等機制來組織和儲存結構化知識，使系統能夠理解關係和上下文 。
    - 推理引擎：處理知識和數據以推斷新資訊、進行邏輯推導並支持決策。這包括檢索增強生成（RAG）和向量搜索等能力，使系統能夠獲取和合成相關資訊 。
    - 學習模型：包含人工智慧模型的訓練和完善（例如，深度學習、轉換器、強化學習），使系統能夠識別模式、進行預測並隨時間適應 。這是系統從原始數據中建立理解的地方。
    - 目標設定與策略適應：對於代理系統，此層根據環境回饋和內部知識定義目標、規劃策略並實時適應 。
    - 使用者研究與上下文映射：作為通觀設計流程的一部分，此層始於深入的使用者研究，以發現需求、行為和痛點，並映射更廣泛的生態系統（技術、業務、情感），以指導系統的概念模型 。

執行面 (執行面)

- 作用：此層將概念面的智慧和決策轉化為具體行動，並管理系統的運作流程。它是系統的馬達，使其能夠從「思考轉為行動」。
- 關鍵組成與功能：
    - 行動執行：直接執行命令、啟動工作流程並與外部環境互動。此層確保系統完成驅動結果的任務，使人工智慧成為自主運營者 。
    - 執行引擎：作為工作流程的控制中心，根據實時數據、使用者輸入和預定義邏輯協調行動。它們將高層次決策轉化為精確的自動化步驟，管理任務執行、錯誤處理和資源

分配 。範例包括 LangGraph、AutoGPT、CrewAI 和 MetaGPT 等框架，它們管理複雜的多步驟工作流程、分支邏輯和任務優先級 。
    - **API 與介面**：作為系統內部邏輯與外部軟體系統、第三方工具或公共網路服務之間的關鍵介面。它們使代理能夠提取實時數據並觸發行動 。這與「數據連結層」相符，後者描述了使用 API 和框架將人工智慧模型連接到實際應用 。
    - 狀態管理：管理代理的短期和長期記憶，隨著任務完成或變化更新內部知識，這對於目標追蹤、重新規劃和提高準確性至關重要 。
    - 實時計算：核心人工智慧邏輯在此實時執行，實現即時決策和響應 。

數據面 (數據面)

- 作用：這個基礎層負責所有流經和支持系統的數據的獲取、準備、儲存和管理。它是系統的感測和記憶系統，為智慧和行動提供原始材料。
- 數據流與管理機制：
    - 實體基礎設施：底層硬體組件，包括 GPU、TPU、雲端基礎設施或邊緣設備，為模型訓練和推理提供動力 。
    - 數據獲取與整合：涉及通過 API 從各種來源（內部平台、公共網路服務）提取實時數據 。這也包括「數據連結層」，它使用 API 和框架將人工智慧模型連接到實際應用 。
    - 數據準備與表示：將原始數據轉換為人工智慧模型可消費的格式，例如令牌、向量或嵌入 。這對於確保數據品質和可用性至關重要。
    - 數據飛輪與反饋循環：利用反饋循環使人工智慧代理能夠隨時間學習和改進，根據新數據增強其功能和有效性 。這確保了持續的完善和適應。
    - 數據收集研究方法：採用定性（使用者訪談、實地研究、民族誌研究）和定量（調查、分析）方法來收集有關使用者行為、趨勢和性能的全面數據 。旅程映射和服務藍圖也跨接觸點可視化數據 。

「數據飛輪」概念 強調了人工智慧代理如何通過反饋循環「隨時間學習和改進」，這與通觀設計的「持續演進」原則 直接相關。這表明「數據面」和「執行面」並非靜態的，而是動態、自我改進系統的一部分，這對於「萬能模組」至關重要。這意味著數據的獲取、處理和系統執行之間存在著動態且迭代的關係。系統不僅是被動地處理數據，更是積極地利用數據來完善自身的行為、性能和決策能力。這種持續的數據驅動優化，使得系統更具韌性、智慧和自我改進能力。

「執行引擎」在管理「代理如何規劃、優先排序和執行複雜工作流程中的行動」以及處理「短期和長期記憶」方面扮演關鍵角色 。這超越了簡單的命令執行，達到了複雜的協調和狀態管理，這對於「萬能模組」維持上下文並執行多步驟流程至關重要。執行引擎作為「代理工作流程的控制中心」，管理「任務執行、錯誤處理和資源分配」。它們根據邏輯、上下文和狀態決定任務順序，並管理記憶，這表明執行層遠比運行命令複雜；它涉及複雜的協調、智能決策和穩健的狀態管理。這意味著「萬能模組」將需要設計有穩健的內部狀態和工作流程能力，使其真正自主並能夠處理需要跨各種操作維持上下文的複雜多步驟任務，從而增強其「萬能」實用性。

研究資料中提出的「計算層」直接對應於「執行面」，「知識層」對應於「概念面」，而「數據連結層」、「表示層」和「實體層」共同構成了「數據面」。這提供了一個具體、業界認可的架構分解，驗證了使用者所選擇的層次結構。這種對應關係證明了使用者框架與既定的人工智慧系統設計原則相符，為所提出的「通觀層面」賦予了顯著的可信度和實際適用性。

此外，關於瓶頸通常出現在基礎設施或整合層（數據連結/實體層），而非僅僅模型層（計算/學習層）的提示 ，突顯了這些層之間關鍵的相互依賴性。這強化了對整個堆疊進行通觀最佳化的必要性，而不僅僅是單個組件，以實現整體系統性能。這意味著「執行面」和「概念面」的有效運作和性能，嚴重依賴於底層「數據面」所提供的穩健性、效率和無縫整合。這強調了優化「萬能模組」需要通觀地考慮整個系統堆疊，而不僅僅是其內部邏輯或所採用的特定人工智慧模型，從而強化了「通觀層面」作為一個整合框架的核心前提。

表2:通觀層面層次結構:組成與功能

層次結構	主要作用/功能	關鍵組成/技術	相關資料來源
概念面	系統的智慧、推理與知識表示，提供高層次理解與戰略方向。	知識圖譜、推理引擎、學習模型、目標設定、使用者研究與上下文映射。	
執行面	將概念層的智慧轉化為具體行動，管理系統的操作流程。	行動執行、執行引擎(協調、狀態管理)、API、實時計算。	
數據面	所有數據的獲取、準備、儲存與管理，為智慧與行動提供原始材料。	實體基礎設施、數據獲取與整合、數據準備與表示、數據飛輪、研究方法。	

從核心功能到萬能模組

從核心功能到「萬能模組」的轉變，代表著模組化設計原則的策略性應用，由「萬能屬性」驅動，並通過「萬能分析」進行完善。這種方法確保核心系統能力被封裝為可重用、適應性強且高效的單元。

定義萬能屬性 (萬能屬性)

「萬能屬性」是定義和區分核心系統功能的普遍、基本特徵或屬性，使其本質上具有適應性和跨多樣化上下文的適用性。這些屬性超越了特定的實現，捕捉了能力的本質。它們作為設計「萬能模組」的指導原則。例如，一個屬性可能是「實時處理能力」、「上下文感知」或「互操作性」。以這些屬性為念設計的模組，本質上就是為了廣泛適用性和整合性而構建的。例如，在一個用於數據處理的「萬能模組」中，「可擴展性」和「數據完整性」將是萬能屬性，確保它能夠處理不同的數據量並在不同應用中保持數據品質。

運用萬能分析 (萬能分析)

「萬能分析」指的是在整個系統生命週期中應用無處不在且深入的分析能力。它涉及利用大數據演算法、自動化參數訓練和複雜的數據處理，以理解系統行為、識別模式並為策略決策提供依據。這種分析超越了被動觀察，轉向主動訓練，其中手動操作被轉化為系統學習，並且各種演算法被合併以實現全面的洞察。這確保了模組不僅基於初步理解進行設計，而且通過真實世界的數據持續進行最佳化。

「萬能分析」對於識別系統中潛在問題和改進機會至關重要。它使系統能夠根據數據驅動的洞察和過去的互動做出明智決策，從而實現最佳化的性能、減少浪費並提高整體效率。例如，通過將人工智慧與大數據整合來預測中國房地產企業的財務困境，或改善醫療保險理賠效率。

萬能模組設計原則

「萬能模組」是根據既定的模組化設計原則構建的，確保它們堅固、靈活且真正「萬能」地實用。

- 目的性：每個模組必須具有單一、明確定義的目的，其內部元素應緊密相關並專注於該特定職責。這降低了複雜性，增強了理解、維護和重用性。
- 封裝(資訊隱藏)：模組應封裝特定功能，隱藏其內部運作，僅暴露清晰、明確定義的介面。這對於降低後續設計變更的成本並使模組更易於重用至關重要。這意味著使用者無需了解其內部工作原理，只需了解其功能以及如何與之互動。

- 低耦合：模組應具有最小的相互依賴性，這意味著對一個模組的更改不太可能影響其他模組。這促進了可修改性、靈活性，並有助於並行開發 。
- 可修改性：模組的設計應易於修改或擴展，以滿足不斷變化的需求，而不會影響整個系統 。
- 可重用性：模組應設計為可在不同上下文或項目中重用，從而減少開發時間和精力，促進項目一致性，並鼓勵組織或社群內部共享程式碼和最佳實踐 。這是「萬能」功能的核心原則。
- 清晰的介面設計：提供簡單、直觀且有良好文檔的介面，定義系統其他部分如何使用和與模組互動 。這種抽象意味著無需考慮其實現即可理解其功能 。
- 可測試性：模組應設計為易於作為獨立單元進行測試，從而簡化維護和調試 。
- 版本控制與兼容性：應有機制確保對模組的更改與其他組件保持兼容，從而實現無縫升級和新版本的採用 。
- 維護與升級能力：封裝功能使得更容易識別、隔離和修改特定功能，而不會影響系統的其餘部分 。
- 文檔：提供全面的文檔和指南，以確保團隊在使用和整合模組時的清晰度和連續性 。

模組化設計原則，特別是可重用性、可修改性和低耦合，是「萬能模組」概念的基石 。這些原則不僅僅是關於程式碼組織，更是關於實現可擴展性、降低開發成本和確保長期適應性。對於「萬能模組」而言，這意味著它們本質上被設計為可以無縫整合到各種上下文中（「萬能」），而無需進行大量重新設計，使其成為複雜系統中極具價值的資產。這直接解釋了為什麼選擇模組化來構建「萬能模組」，因為它為其「萬能」特性提供了架構基礎。

「資訊隱藏」作為一項策略性設計選擇，是模組化設計的關鍵要素 。它明確指出「每個模組都應該封裝不對程式其餘部分開放的資訊。這種資訊隱藏降低了後續設計變更的成本。」它與僅僅將「邏輯相關的功能」分組形成對比。這對於「萬能模組」來說是一個關鍵的、微妙的區別，它暗示了其內部複雜性被刻意抽象化，使其更易於使用且不易產生破壞性變更。這項策略性設計選擇顯著減少了模組內部變更對系統其餘部分的漣漪效應，促進了穩健的整合並降低了維護開銷。這深化了對「萬能模組」設計哲學的理解，解釋了它們如何在內部強大且在外部易於使用，這是其「萬能」實用性的關鍵推動因素。

「萬能」一詞在提供的資料中呈現出兩種截然不同的涵義。一方面，「萬能人工智慧」被描述為一種強大、整合、數據驅動的系統，用於執行複雜任務，這與使用者對「萬能分析」和「萬能屬性」的意圖相符 。另一方面，「無處不在的連字符」則用來闡釋人工智慧在理解細微差別和「為什麼」方面的局限性 。這種並置產生了一個重要的張力：雖然目標是實現「萬能分析」（積極意義），但系統設計必須旨在減輕人工智慧「無處不在」的局限性（消極意義），特別是在需要深入上下文理解或細緻決策的領域。這意味著即使進行「萬能分析」，也需要穩健的驗證和人為監督。這為「萬能」概念增加了實踐上的謹慎、倫理考量和設計穩健性，確保系統不僅強大，而且可靠和值得信賴。

「萬能分析」的運作機制具體體現在「演算法合併」、「使用大數據的自動化參數訓練」以及「將手動操作轉化為系統主動訓練」等技術中 。這將「萬能分析」從一個高層次概念轉變為一系列具體的、可操作的方法。這意味著「萬能分析」不僅僅是數據的收集和處理，更是動態整合和完善分析模型，並利用人類專業知識來主動訓練和改進系統。這種主動、自適應和整合的性質是複雜系統中真正「萬能」且有效分析的關鍵特徵，能夠實現持續最佳化和響應能力。

表3：萬能模組設計原則

原則	萬能模組的應用	萬能模組的益處	相關資料來源
目的性	每個模組專注於單一、明確定義的職責。	降低複雜性，易於理解、維護與重用。	
封裝	隱藏內部運作，僅暴露清晰介面。	降低設計變更成本，提升模組獨立性。	

原則	萬能模組的應用	萬能模組的益處	相關資料來源
低耦合	模組間相互依賴性最小化。	促進可修改性與靈活性，便於並行開發。	
可修改性	易於修改或擴展，不影響系統其他部分。	提升系統適應性，降低維護風險。	
可重用性	可在不同上下文或項目中重複使用。	減少開發時間與精力，促進程式碼共享與一致性。	
清晰的介面設計	提供簡單、直觀且有良好文檔的介面。	易於整合與使用，降低學習曲線。	
可測試性	設計為易於作為獨立單元進行測試。	簡化維護與調試，提升模組品質。	
版本控制與兼容性	確保模組變更與其他組件的兼容性。	允許無縫升級，避免系統中斷。	
維護與升級能力	封裝功能，易於識別、隔離與修改。	簡化長期維護，提升系統演進能力。	
文檔	提供全面的文檔與指南。	確保團隊協作清晰度與連續性。	

表4: 萬能屬性與分析在模組定義中的相互作用

概念	在模組定義中的作用	對萬能模組的影響	相關資料來源
萬能屬性	定義核心功能的普適性、基本特性。	確保模組具備廣泛適用性與內在兼容性。	
萬能分析	提供貫穿系統生命週期的全面、數據驅動的分析能力。	實現模組的持續最佳化、主動學習與智能決策。	

新框架的協同作用與策略優勢

本節綜合闡述了「通觀層面」、其分層架構以及指導「萬能屬性」、「萬能分析」和「萬能模組」的原則如何整合形成一個連貫而強大的系統。

層次與模組的整合

「概念面」提供總體智慧和決策框架，其基礎是深入的使用者和系統上下文理解 。它定義了系統的知識、推理和學習能力，為整個系統提供高層次的戰略指導。
「執行面」作為操作骨幹，通過複雜的協調和狀態管理，將概念決策轉化為實際行動 。它負責實際的任務執行、工作流程管理以及與外部環境的互動，確保系統能夠自主地將思考轉化為行動。
「數據面」作為基礎的真相來源，持續為整個系統提供洞察並促成學習 。它負責數據的獲取、準備、表示和管理，是系統智慧和行動的基礎。
「萬能模組」作為封裝的、可重用的構建塊，在這些層次內部和之間無縫運作。其設計由「萬能屬性」指導，確保內在兼容性和廣泛適用性，而「萬能分析」則通過數據驅動的反饋循環不斷完善其性能和行為。這些模組不僅是功能單元，更是設計用於長期演進、成本效益和整合到多樣化應用中的組件。它們的「萬能」特性源於其穩定的、簡單的介面，而其內部複雜性則被刻意抽象化，使其更易於使用且不易產生破壞性變更。

通觀反饋與持續演進

該框架促進了持續的反饋循環，其中來自「數據面」的數據為「萬能分析」提供資訊，進而完善「概念面」的理解和「執行面」的行動，最終導致「萬能模組」的演進 。這種動態過程確保系統保持適應性並持續最佳化。數據飛輪機制使人工智慧代理能夠通過利用反饋循環隨時間學習和改進，增強其功能和有效性。這確保了系統的持續完善和適應，使其成為一個動態且自我改進的實體。

整合框架的整體益處

- 提升效率與性能：通過最佳化整個系統而非孤立的部分，該框架帶來卓越的性能、減少浪費並提高運營效率 。
- 改進決策：在所有層次利用「萬能分析」提供更深入、數據驅動的洞察，從而實現更明智和主動的決策 。
- 增強永續性與可靠性：對相互依賴性和風險的通觀考量，結合模組化以簡化維護和升級，確保系統的長期永續性和可靠性 。
- 更大的可重用性與可擴展性：「萬能模組」以低耦合和清晰介面原則設計，顯著減少開發時間和精力，促進一致性並實現跨多樣化應用的快速擴展 。
- 適應複雜性：該框架本質上設計用於管理現代系統的複雜性，包括湧現特性和多學科整合，從而產生更穩健和面向未來的解決方案 。
- 一致的使用者體驗：通觀設計方法確保系統的每個方面都有助於提供無縫且有意義的使用者體驗，從而提升滿意度和忠誠度 。

結論

從「萬能定義」和「三世界」到「通觀層面」的轉變，標誌著系統架構的關鍵演進。這個新框架，圍繞著概念面、執行面和數據面構建，為設計複雜、智能系統提供了一個穩健且整合的方法。
將「萬能屬性」和「萬能分析」作為定義「萬能模組」的基礎元素，確保核心功能被開發為高度可重用、適應性強且持續最佳化的組件。這項策略性轉變不僅僅是術語上的改變，它代表著一種更深層次的承諾，即以整體、互聯的方式來理解和構建系統，從而應對現代技術環境中日益增長的複雜性和動態性。
這種通觀和模組化的範式使組織能夠構建不僅強大高效，而且本質上具有適應性、永續性並能夠持續自我改進的系統。它鼓勵系統思維模式，超越孤立問題，解決定義真正系統智慧和韌性的錯綜複雜的相互依賴關係。
該框架為開發複雜的人工智慧驅動解決方案提供了清晰的路線圖，強調了數據驅動的洞察、智能協調以及對技術和上下文因素的深入理解的重要性。它為未來在自主系統、智能自動化和高度整合的數位生態系統方面的創新奠定了基礎。

引用的著作

1. What is Holistic Design? — updated 2025 | IxDF, https://www.interaction-design.org/literature/topics/holistic-design 2. Holistic Thinking in Systems Engineering - Number Analytics, https://www.numberanalytics.com/blog/holistic-thinking-systems-engineering 3. 7 Layers That Make AI Work (and Why They Matter) | by Generative AI | Medium, https://medium.com/@genai.works/7-layers-that-make-ai-work-and-why-they-matter-d0ebe3f32b56 4. Agentic AI Action Layer: Powering True Autonomy with Tools, APIs, and Execution Engines,

https://www.aziro.com/blog/agentic-ai-action-layer-tools-apis-execution-engines-for-true-autonomy/ 5. Full article: Big data and Omnipresent AI - Taylor and Francis, https://www.tandfonline.com/doi/full/10.1080/17517575.2025.2490920 6. Modular Design: Crafting Reusable and Interchangeable Components - Verpex, https://verpex.com/blog/website-tips/modular-design-crafting-reusable-and-interchangeable-components 7. 4.1 Modular Design Review, https://www.mcs.anl.gov/~itf/dbpp/text/node40.html 8. The Story of the Omnipresent Hyphen: Why AI Can't Get Punctuation Right - Medium, https://medium.com/@devapratimm/the-story-of-the-omnipresent-hyphen-why-ai-cant-get-punctuation-right-b26e88426bad