財務報帳成為第一個實驗場

115 資工專題 • 2026/07/17 Meeting

通用 Agent 架構開始碰到一個真正的 domain problem

當時的系統構想仍然很「上層」

專題還沒有正式定案成經費核銷預檢，當時主軸仍是：

META layer

- 收集需求
- 維護規格書
- Architect 審核規格

下層開發自動化

- AI coding
- QA / 驗收
- 讓規格可以往實際系統落地

財務報帳只是第一個實驗場：希望 Agent 能把過去踩過的坑留下來，讓新人不用重踩。

協作機制：把軟體開發流程變成可追蹤紀錄

當時開始把 GitHub 的協作概念搬進架構：

- **Issue**：需求、問題與待辦有明確來源。
- **Pull Request**：規格或實作變更先被審核，再進正式版本。
- **不同模型分工**：昂貴模型只放在真的需要判斷力的步驟，其他流程用較低成本方案。

> 目標不是「AI 幫我寫 code」，而是讓每次需求到上線之間都有可驗收的流程。

老師第一次直接追問：和現成工具差在哪？

如果只說：

> 「使用者問規章，系統可以正確回答。」

那跟 Google 搜尋、校內既有 AI 客服或其他專題組的想法太接近。

老師要求的是更明確的差異：

- **哪一種問題是現有工具真的處理不好？**
- **你的系統為什麼不是另一個規章 QA？**
- **答案要怎麼被驗證，而不是只相信模型？**

同場另一個構想，改變了後面的重點

另一組用停修案例談到 neural-symbolic 驗證：

- AI 可以理解自然語言與找候選資訊。
- 但引用的法規要真的存在。
- 推論不能只靠模型自己說得合理。

這個想法後來成為 Campus Agent 很重要的一條線：

> **差別不只在「答對」，而在「能不能查證答案依據」。**

這場之後真正要解的問題

原本的問題是：

怎麼做一個能持續演進的 Agentic software workflow？

這場之後開始變成：

能不能找到一個真實、規則複雜、現成工具又不夠好的場景，讓可追蹤、可驗證的架構有必要存在？

> **下一週的關鍵不是再擴大架構，而是把 domain 與研究問題正式定案。**