Domain Driven Design Workshop – Feedback

NM6121030 余振揚

這次由陳勉修先生主講的 Domain-Driven Design (DDD) 講座，深入探討了如何以業務需求為核心，進行軟體系統設計。講師從問題空間（Problem Space）與解決空間（Solution Space）的區分開始，強調了以業務邏輯為導向的重要性，避免傳統資料導向設計所導致的耦合與靈活性不足。他也介紹了戰略設計與戰術設計的核心概念，特別是限界上下文（Bounded Context）的應用與價值流的映射，幫助開發者更有效地處理複雜業務邏輯。

最讓我印象深刻的是「領域故事化（Domain Storytelling）」的技術，透過具體的業務場景來幫助團隊建立共同語言與清晰的領域模型。此外，講師還討論了如何運用聚合、實體與值對象等戰術建模工具，結合事件驅動架構，實現模組化設計。

作為研究 AI 領域的學生，我認為 DDD 的概念在 AI 模型的設計與整合上有相當大的潛力。當我們開發 AI 系統時，常常需要整合不同子領域的模型，比如自然語言處理或圖像辨識。透過限界上下文的概念，我們可以清楚劃分每個子領域的範圍，避免模型之間的功能重疊或混淆，也能讓系統設計更有條理，方便管理和維護。

另外，DDD 裡的事件驅動設計特別適合用在 AI 系統的即時處理上。舉個例子，當系統偵測到異常情況時，可以馬上觸發對應的分析模組來處理，這樣不僅節省時間，還能讓系統更快回應。

在規劃 AI 解決方案時，我覺得戰略設計這套方法也很有用。它讓我們能專注在解決具體的業務問題，而不只是追求技術的尖端。特別是搭配價值流分析工具，可以幫助我們更清楚知道每個階段的 AI 系統對業務的貢獻。

最後，AI 團隊和業務單位之間常會因為專業術語不同而溝通不良。DDD 提倡的「通用語言」正好可以解決這個問題，幫助大家站在同一陣線上，一起確保 AI 系統真正解決了業務的核心需求。

總結來說，DDD 不僅是一種軟體設計方法，更是一種設計思維模式。在 AI 領域，透過引入 DDD 的框架，我們可以打造出更有彈性、更高效益的智能系統。