# 系統詳細設計圖集 (System Detailed Design Atlas)

**專案名稱**：Bombus 企業管理系統

**版本**：V6.0 (PRD Strict Alignment Ver.)

**文件日期**：2025-11-21

**用途**：補充 PRD/FDD/TAD，提供開發團隊具體的物件結構與邏輯流程圖。

**撰寫人**：Gemini (資深系統設計規劃師)

## L0 系統核心與儀表層 (System Core & Dashboard)

### L0.0 企業管理儀表板

圖表類型：序列圖 (Sequence Diagram)

描述儀表板如何聚合跨模組數據並透過 AI 生成摘要。

sequenceDiagram  
 participant User as 使用者 (CEO/HR)  
 participant Dashboard\_UI as 儀表板前端  
 participant Aggregator as 資料聚合服務  
 participant Cache as Redis 快取  
 participant AI\_Svc as AI 摘要引擎  
 participant DB as 核心資料庫  
  
 User->>Dashboard\_UI: 登入並存取首頁  
 Dashboard\_UI->>Aggregator: 請求儀表板數據 (UserRole)  
   
 par 平行撈取數據  
 Aggregator->>Cache: 查詢熱點數據 (OKR/KPI)  
 Aggregator->>DB: 查詢即時異動 (L1/L4/L5)  
 end  
   
 Cache-->>Aggregator: 回傳快取數據  
 DB-->>Aggregator: 回傳最新數據  
   
 alt 快取過期或無摘要  
 Aggregator->>AI\_Svc: 發送關鍵指標數據 (Prompt)  
 AI\_Svc-->>Aggregator: 回傳 "一句話營運摘要"  
 Aggregator->>Cache: 更新快取 (TTL: 1hr)  
 end  
   
 Aggregator-->>Dashboard\_UI: 回傳聚合 JSON  
 Dashboard\_UI-->>User: 渲染戰情室視圖

**圖表細節說明**：

1. **Aggregator (聚合器)**: 這是 L0 的核心組件，負責向後端多個微服務（L1-L6）發起請求，並將結果標準化。
2. **Redis Cache**: 為了確保首頁載入 < 2 秒（非功能需求），高頻存取的數據（如全公司 OKR 進度）必須被快取。
3. **AI 摘要**: 透過 LLM 對冷冰冰的數據進行自然語言處理，產出具備洞察力的文字摘要（如「本週研發進度落後，建議關注專案 Alpha」）。

設計整合說明：

L0.0 不僅是一個顯示層，更是整個系統的「神經中樞」。本序列圖展示了「效能」與「智能」的平衡策略。考量到高階主管的時間寶貴，系統採用了「快取優先 (Cache-First)」與「平行處理」架構來極大化回應速度。同時，引入 AI 服務作為數據的翻譯者，將複雜的報表數據即時轉化為可閱讀的決策建議。這種設計確保了決策者能在登入的第一時間，就掌握組織的健康狀況，而非迷失在數據叢林中。

### L0.1 系統管理與權限控制

圖表類型：類別圖 (Class Diagram)

定義 RBAC 模型與列級權限結構。

classDiagram  
 class User {  
 +String userId  
 +String deptId  
 +List~Role~ roles  
 }  
 class Role {  
 +String roleId  
 +int level "1-99"  
 +List~Permission~ permissions  
 }  
 class Permission {  
 +String resource "L1\_Salary, L6\_EAP"  
 +String action "Read, Write, Approve"  
 +String scope "Self, Dept, All"  
 }  
 class AuditLog {  
 +String logId  
 +String userId  
 +String action  
 +Date timestamp  
 +String ipAddress  
 }  
  
 User "1" --> "\*" Role : assigned  
 Role "1" --> "\*" Permission : contains  
 User "1" -- "\*" AuditLog : generates

**圖表細節說明**：

1. **Scope (範圍)**: 權限物件中的關鍵欄位，決定了使用者能看到多廣的資料（僅自己、本部門、全公司）。這是實作「列級權限 (Row-Level Security)」的基礎。
2. **AuditLog (稽核日誌)**: 記錄所有關鍵操作，特別是涉及 L1 薪資與 L6 EAP 的存取，以滿足資安合規要求。

設計整合說明：

L0.1 的權限模型是系統安全的基石。本設計採用了標準的 RBAC（基於角色的存取控制）模型，但強化了 Data Scope 的定義。這意味著系統不僅控制「誰能做什麼功能（如編輯薪資）」，還能精細控制「誰能看到哪幾筆資料（如只能看本部門員工）」。配合完整的 Audit Log 設計，我們構建了一個既靈活又具備可追溯性的安全架構，這對於處理 L5 獎金與 L6 隱私資料至關重要。

## L1 員工管理模組 (Employee Management)

### L1.1 招募與候選人管理

圖表類型：類別圖 (Class Diagram)

定義招募流程核心實體與 AI 分析結果的儲存。

classDiagram  
 class JobOpening {  
 +String jobId  
 +String title  
 +List~String~ keywords  
 +String status  
 }  
 class Candidate {  
 +String candidateId  
 +String name  
 +String resumeUrl  
 +JSON aiAnalysisResult  
 +float totalScore  
 }  
 class InterviewRecord {  
 +String recordId  
 +String transcriptText  
 +Map~String, Float~ keywordScores  
 +float sentimentScore  
 }  
   
 JobOpening "1" -- "\*" Candidate : attracts  
 Candidate "1" -- "\*" InterviewRecord : has

**圖表細節說明**：

1. **aiAnalysisResult**: 这是一个 JSON 欄位，儲存 AI 對候選人履歷與面試表現的綜合分析結構資料。
2. **keywordScores**: 儲存面試過程中命中 JobOpening 關鍵字的次數與權重，是量化評分的基礎。

設計整合說明：

此類別圖強調了從「職缺定義」到「候選人評估」的數據一致性。JobOpening 中定義的 Keywords 直接驅動了 InterviewRecord 中的評分邏輯。系統將面試過程中的非結構化數據（語音、文字）轉化為結構化的分數（keywordScores, sentimentScore），並最終匯總於 Candidate 實體。這為 HR 提供了客觀的數據支持，減少了人為偏見，並建立了可被 L1.2 繼承的人才數據資產。

### L1.2 員工檔案與歷程管理

*(參照 L1.1 類別圖，Employee 實體繼承自 Candidate，確保數據延續性)*

### L1.3 人才庫與再接觸管理

*(邏輯依賴 L1.1 的 Candidate 資料庫與關鍵字匹配演算法)*

### L1.4 職涯晉升與接班規劃

圖表類型：流程圖 (Flowchart)

描述週休三日試行機制的自動化監控流程。

graph TD  
 Start((每月1號)) --> FetchUsers[撈取試行員工]  
 FetchUsers --> Check[檢核指標: 任務率 & 協作分]  
   
 Check --> |未達標| Fail[累計次數 +1]  
 Check --> |達標| Pass[重置次數]  
   
 Fail --> Threshold{連續 >= 3次?}  
 Threshold -- Yes --> Terminate[觸發退場: 恢復週休二日]  
 Threshold -- No --> Warning[發送預警信]  
   
 Terminate --> Notify[通知 HR 與主管]

**圖表細節說明**：

1. **自動排程**: 每月 1 號觸發，無需人工介入。
2. **三振機制**: 採用連續 3 次未達標才退場的邏輯，提供員工改善緩衝期。

設計整合說明：

此流程圖具體化了 PRD 中對於「週休三日管理」的政策邏輯。透過自動化的監控與預警，系統在保障員工福利（彈性工時）與維護組織績效之間取得了平衡。這種「先輔導、後處置」的設計，減少了管理者的心理負擔，並確保了制度執行的公平性與透明度。

### L1.5 會議管理

圖表類型：序列圖 (Sequence Diagram)

描述內部會議與外部 Google Calendar 的同步邏輯。

sequenceDiagram  
 participant User  
 participant L1\_Meeting as 會議系統  
 participant G\_API as Google Calendar API  
 participant DB as 資料庫  
  
 User->>L1\_Meeting: 建立會議 (時間/與會者)  
 L1\_Meeting->>DB: 寫入會議紀錄 (Status: Pending)  
 L1\_Meeting->>G\_API: 呼叫 Create Event API  
   
 alt Google 回傳成功  
 G\_API-->>L1\_Meeting: 回傳 Event ID  
 L1\_Meeting->>DB: 更新 Event ID & Status: Synced  
 L1\_Meeting-->>User: 預約成功  
 else 衝突或失敗  
 G\_API-->>L1\_Meeting: Error (Time Conflict)  
 L1\_Meeting-->>User: 預約失敗，建議其他時段  
 end

**圖表細節說明**：

1. **Event ID**: 儲存 Google 回傳的 ID，作為後續更新或刪除會議的 Key。
2. **衝突檢測**: 利用 Google API 的回應來判斷會議室或人員是否忙碌。

設計整合說明：

會議管理模組雖小，卻是員工體驗的關鍵。本設計採用「雙向同步」策略，以 Bombus 系統為發起端，Google Calendar 為執行端。這確保了員工在企業系統內的操作能無縫反映在個人行事曆上，避免了資訊不同步造成的會議衝突，提升了協作效率。

## L2 職能管理模組 (Competency Management)

### L2.1 職等職級管理

*(參照 L2.3 類別圖中的 JobDescription 與 Competency 關聯)*

### L2.2 職務說明書管理 (JD)

*(參照 L2.3 類別圖，JD 為職能模型的輸入端)*

### L2.3 職能框架開發

圖表類型：類別圖 (Class Diagram)

定義職能模型結構。

classDiagram  
 class JobDescription {  
 +String jdId  
 +String title  
 +List~CompetencyReq~ requirements  
 }  
 class Competency {  
 +String compId  
 +String name (KSA)  
 +List~BehaviorIndicator~ indicators  
 }  
 class CompetencyReq {  
 +String compId  
 +Int requiredLevel  
 +Int weight  
 }  
   
 JobDescription "1" -- "\*" CompetencyReq : defines  
 CompetencyReq --> Competency : references

**圖表細節說明**：

1. **CompetencyReq**: 中間表，允許不同職位對同一職能有不同的等級要求。
2. **KSA**: Competency 實體封裝了知識、技能與態度，是 L2 的核心原子單位。

設計整合說明：

此模型實現了職位與能力的解耦。透過中間表設計，HR 可以靈活地組合不同的職能來定義新職位，而無需重複建立職能標準。這為 L2.6 的 AI 生成引擎提供了標準化的數據結構，使得非結構化的 JD 文本能被精確地映射到系統的職能庫中。

### L2.4 職能評估系統

*(流程邏輯參照 L2.6 序列圖)*

### L2.5 職能落差分析

*(計算邏輯包含於 L2.6 序列圖中)*

### L2.6 AI 職能生成引擎

圖表類型：序列圖 (Sequence Diagram)

描述從 JD 生成到落差分析的完整流程。

sequenceDiagram  
 participant HR  
 participant AI as AI 引擎  
 participant L2 as 職能模組  
 participant DB as 資料庫  
  
 HR->>L2: 上傳 JD 職位描述  
 L2->>AI: 解析 KSA 要素  
 AI-->>L2: 回傳結構化職能列表  
 L2->>DB: 儲存職能標準 (Standard)  
  
 Note over L2: 評估週期  
 L2->>DB: 讀取員工實測分數 (Actual)  
 L2->>L2: 計算 Gap = Standard - Actual  
 L2->>DB: 儲存 Gap Analysis 結果

**圖表細節說明**：

1. **KSA 解析**: AI 將自然語言轉換為系統可識別的 Competency ID。
2. **Gap 計算**: 系統自動比對標準與實測，為 L3 推薦提供數學依據。

設計整合說明：

此序列圖展示了 Bombus V6.0 如何利用 AI 自動化解決「職能標準建立難」的痛點。系統自動從 JD 中萃取標準，並在評估後即時計算落差。這不僅節省了 HR 的前置作業時間，更確保了職能標準與實際工作內容（JD）的高度一致性。

## L3 教育訓練管理模組 (Training & Development)

### L3.1 培訓計畫管理

*(參照 L3.2 類別圖，TrainingPlan 為頂層容器)*

### L3.2 課程與報名管理

圖表類型：類別圖 (Class Diagram)

定義課程與人才地圖的聚合關係。

classDiagram  
 class Course {  
 +String courseId  
 +String type  
 +List~String~ targetCompetencies  
 }  
 class TrainingRecord {  
 +String recordId  
 +float postTestScore  
 +float conversionRate  
 }  
 class TalentMapMetrics {  
 +String deptId  
 +Map~String, Float~ avgCompetency  
 +List~String~ starEmployees  
 }  
  
 Course "1" -- "\*" TrainingRecord : generates  
 TrainingRecord --|> TalentMapMetrics : aggregates\_to

**圖表細節說明**：

1. **targetCompetencies**: 課程與職能的關聯鍵，用於 L2 的自動推薦。
2. **TalentMapMetrics**: 聚合表，用於快速渲染 L0 與 L3 的熱力圖。

設計整合說明：

此類別圖展示了「培訓數據」如何轉化為「戰略視圖」。系統透過追蹤個別員工的 TrainingRecord，定期聚合運算出部門層級的 TalentMapMetrics。這讓管理層能跳脫單一課程的細節，直接看到培訓對組織能力（人才地圖）的具體提升，實現數據驅動的人才發展決策。

### L3.3 線上測驗系統

圖表類型：狀態機圖 (State Diagram)

描述測驗過程中的狀態流轉與防作弊機制。

stateDiagram-v2  
 [\*] --> Ready  
 Ready --> InProgress: 開始測驗  
   
 state InProgress {  
 [\*] --> Answering  
 Answering --> BlurDetected: 切換視窗  
 BlurDetected --> Answering: 警告並記錄  
 Answering --> Submit: 交卷  
 }  
   
 Submit --> Grading: 自動評分  
 Grading --> Completed: 產生證書  
 Grading --> Failed: 未通過

**圖表細節說明**：

1. **BlurDetected**: 偵測瀏覽器 blur 事件，作為防作弊的觸發點。
2. **Grading**: 系統後端自動比對答案，無需人工介入。

設計整合說明：

L3.3 專注於驗證學習成效的真實性。狀態機清晰定義了測驗的生命週期，特別是加入了防作弊（切換視窗偵測）的狀態流轉。這確保了線上測驗不僅是形式，而是能真實反映學員知識掌握度的評估工具，為 L3.4 的成效追蹤提供可信的基線數據。

### L3.4 培訓成效追蹤與回饋

圖表類型：流程圖 (Flowchart)

描述三個月後的行為轉化評估流程。

graph TD  
 End((完訓)) --> Wait[等待 90 天]  
 Wait --> Trigger[觸發反饋會議]  
 Trigger --> Star[收集 STAR 案例]  
 Star --> AI[AI 分析轉化率]  
 AI --> Score{轉化率分數}  
   
 Score --> |High| UpdateL2[升級 L2 職能]  
 Score --> |Low| ReviewCourse[標記課程需改善]

**圖表細節說明**：

1. **90天延遲**: 確保評估的是長期行為改變。
2. **雙向回饋**: 高分回饋員工（升級），低分回饋課程（優化）。

設計整合說明：

此流程落實了 Kirkpatrick 模型的 L3 層級評估。系統自動化的追蹤機制解決了人工追蹤困難的問題，並透過 AI 分析學員的實踐案例，將定性的行為改變轉化為定量的數據。這使得培訓成效不再是模糊的感覺，而是可被量化、可被優化的具體指標。

## L4 專案管理模組 (Project Management)

### L4.1 專案與任務管理

圖表類型：類別圖 (Class Diagram)

定義專案財務結構。

classDiagram  
 class Project {  
 +String projId  
 +float budgetBAC  
 }  
 class ProjectFinance {  
 +float actualCost\_AC  
 +float earnedValue\_EV  
 +float aiRiskFactor  
 +float predictedMargin  
 }  
 class WorkLog {  
 +float hours  
 +float cost  
 }  
  
 Project "1" -- "1" ProjectFinance : tracks  
 Project "1" -- "\*" WorkLog : accumulates

**圖表細節說明**：

1. **ProjectFinance**: 獨立實體，專門儲存財務預測數據。
2. **aiRiskFactor**: AI 根據非結構化數據計算出的風險係數。

設計整合說明：

L4 的設計重心在於將「執行數據」轉化為「財務洞察」。透過 WorkLog 精確捕捉隱性成本，並匯總至 ProjectFinance，系統為每個專案建立了一本即時的財務帳本。這為 L4.3 的 AI 預測提供了堅實的數據基礎。

### L4.2 專案協作與進度追蹤

*(功能主要在前端 IDP 呈現，後端依賴 L4.1 實體)*

### L4.3 專案績效與毛利

圖表類型：序列圖 (Sequence Diagram)

描述 AI 未來損益預測邏輯。

sequenceDiagram  
 participant Scheduler  
 participant L4\_Svc  
 participant AI  
 participant DB  
  
 Scheduler->>L4\_Svc: 每日觸發預測  
 L4\_Svc->>DB: 撈取 EVM 數據 (AC/EV)  
 L4\_Svc->>AI: 傳送留言文本 (Sentiment)  
 AI-->>L4\_Svc: 回傳 Risk Factor  
   
 L4\_Svc->>L4\_Svc: 修正 EAC = BAC/CPI \* Risk  
 L4\_Svc->>L4\_Svc: 預測毛利 = 預算 - 修正 EAC  
   
 opt 毛利 < 0  
 L4\_Svc->>DB: 寫入警示  
 end

**圖表細節說明**：

1. **混合模型**: 結合數學 (EVM) 與 AI (Sentiment) 進行雙重驗證。
2. **主動預警**: 預測虧損時立即寫入警示表。

設計整合說明：

此序列圖展示了 Bombus V6.0 的核心競爭力——「預知能力」。不同於傳統系統只能檢討過去，本系統利用 AI 偵測專案中的情緒與風險信號，對財務預測進行修正。這讓管理者能在財務赤字發生前獲得預警，從而採取行動挽救專案利潤。

### L4.4 專案報表與分析

*(基於 L4.1 數據的查詢與匯出功能)*

### L4.5 專案自動化引擎

*(後端排程服務，依賴 L4.1 狀態變更觸發)*

## L5 績效管理模組 (Performance Management)

### L5.1 過程管理系統與毛利計算

圖表類型：序列圖 (Sequence Diagram)

描述獎金池計算的跨系統流程。

sequenceDiagram  
 participant L5\_Svc  
 participant ERP  
 participant L4\_Svc  
 participant DB  
  
 L5\_Svc->>ERP: 獲取營收 (Revenue)  
 L5\_Svc->>L4\_Svc: 獲取直接成本 (Direct Cost)  
 L5\_Svc->>DB: 讀取分攤率 (Overhead Rate)  
   
 L5\_Svc->>L5\_Svc: 毛利 = 營收 - 直接 - 間接  
 L5\_Svc->>DB: 查表取得提撥率  
 L5\_Svc->>L5\_Svc: 獎金池 = 毛利 \* 提撥率  
   
 L5\_Svc->>DB: 儲存計算結果

**圖表細節說明**：

1. **資料聚合**: L5 是數據的終點站，匯集了 L4 的成本與 ERP 的營收。
2. **參數化**: 間接成本分攤率可由管理者設定，確保計算彈性。

設計整合說明：

L5.1 是將「努力」變現為「獎勵」的關鍵環節。本設計透過自動化的數據聚合與參數化的計算引擎，確保了獎金分配的精確性與公正性。它解決了傳統人工計算耗時且易錯的問題，讓企業的利潤分享機制能高效運轉。

### L5.2 考核週期管理

*(依賴 L5.1 計算結果啟動考核流程)*

### L5.3 360 度回饋系統

*(作為 L5.5 績效分析的輸入數據)*

### L5.4 績效紀錄與日誌系統

*(作為 L5.5 評分時的佐證資料)*

### L5.5 績效分析與改善計畫

*(整合 L5.1-L5.4 數據產出最終報告)*

## L6 文化管理模組 (Culture Management)

### L6.1 企業文化手冊管理

圖表類型：序列圖 (Sequence Diagram)

描述 EAP 匿名預約流程。

sequenceDiagram  
 participant Employee  
 participant L6\_Sys  
 participant Provider  
  
 Employee->>L6\_Sys: 預約諮商  
 L6\_Sys->>L6\_Sys: 生成 Hash ID & Token  
 L6\_Sys->>Provider: 傳送匿名預約單 (Token)  
 L6\_Sys-->>Employee: 回傳核銷碼  
   
 Note over Employee, Provider: 線下諮商  
   
 Provider->>L6\_Sys: 核銷 Token  
 L6\_Sys->>L6\_Sys: 標記完成 (無個資連結)