前言

25 年前，一群美國同事來英國協助我們開發一套方式與課程，訓練安全性架構師。當時，我們維護的系統多數都是執行各種版本 Windows 或 Unix 的中階或大型主機，網路多屬於內部或私有。時至今日，網際網路、雲端運算、容器化應用、敏捷式流程、自動化與人工智慧等技術的崛起，已然徹底改變資訊系統的樣貌。

為簡化，一律稱混合雲

全書為簡化而一律統稱混合雲（hybrid cloud）。出現「混合雲」一詞時，可以理解為多重雲端服務或「多重雲端」（multicloud）。

雖然歷經許多變革，但我們 25 年前建立的主要安全架構概念仍然維持不變，最初採用的是以下標準的概念：

- 以保障機密性、完整性及可用性為指導原則或安全性設計目標，奠定安全控制措施的基礎。
- 使用名為子系統的安全性領域，分組安全控制。
- 威脅模型分析（Threat modeling）檢視傳輸中與靜止時的資料，識別威脅並找出保護資料的因應措施。不過當時不是叫這個名稱，而是用 Common Criteria（*https://oreil.ly/zRlfE*）的構想描述這項技術。
- 利用區域與防火牆，為資料中心進行網路分段。

- 依據 Common Criteria 的保護輪廓建立控制框架，為安全控制提供需求目錄。
- 文件紀錄與測試提供落實安全控制的保證。

本書含括上述概念，但很多部分有程度不一的變化。對架構性思維產生重大影響的主因，在於移往本地部署與雲端同時使用的混合與多重雲端，這已經成為眾多企業的標準平台架構，相應而來的各種技術平台與套用的安全性政策數量也大幅成長。

威脅日益擴大、技術平台漸趨複雜，連帶使得保護機制的設計也更精良。企業組織如今使用不同技術平台與服務供應商，強化使用各種技術時需遵循的各類不同安全性政策；在這種複雜環境下，架構有效安全性當然出現極大挑戰。

何謂人為產物？

本書「人為產物」（artifact）一詞指的是在架構性思維過程中產生的各種圖示或表格。將這些人為產物整合至文件或簡報中，能夠清晰地呈現解決方案的架構；值得注意的是，部分人為產物的出現奠基於其他已存在之人為產物。為了理解整體架構人為產物及彼此相依性，人為產物相依性的圖表是一種相當實用的工具。

我們改良 25 年前的方式，納入處理混合雲這種複雜狀況的技術，並花 7 年時間與超過 1000 名 IBM 的員工與碩士生共同測試。如今的處理方式，更能夠反應業界變化與架構性思維的改良，包括：

- 人為產物相依圖，展現架構性思維領域與人為產物之間的相依性。
- 處理方式全面導入零信任原則。
- 整合其他架構性思維與專案管理技巧，包括設計思維、敏捷性實作與 DevSecOps。
- 架構圖表在描述功能性元件與部署架構的一致性。
- 導入新人為產物，敘述混合雲複雜的共同責任與新雲端架構圖。
- 採用新的控制框架作為需求目錄，包括美國國家標準與技術研究院（NIST）的網路安全框架、NIST SP 800-53 資訊系統與組織的安全性與隱私控制，以及雲端安全聯盟（CSA）雲端控制矩陣（CCM）。
- 其他針對功能性需求定義的新技術，包括設計思維與使用者背景。
- 因應產業已發展逐漸成型新實作方式，導入威脅模型分析的革新技術。

- 網路分段轉型成微區段，因為能更輕鬆設定虛擬私有雲端網路的網路組態。
- 使用架構模式與可部署架構，加速架構性思維的流程。
- 擴展處理方式，納入安全營運流程與程序定義，包括威脅偵測與回應。
- 為了支援證明合規性，以及威脅的偵測與應變，納入需求、架構分解與威脅的可追溯性。

資訊系統的設計與交付中，安全性架構的每個階段都有重要意義，本書將重點闡述那些可適用於現有系統與軟體開發方法的安全性設計人為產物與技術。透過採用處理流程與人為產物，明確地溝通實作階段所需之安全控制措施，將有助於提升整體開發流程的嚴謹性。希望這種思維模式能廣泛應用在開發流程的制定，與現正使用的實作上。

請將本書視為你的「工具包」，在需要的時候選擇使用其中的工具與技術即可，不必完全依照本書傳授的所有內容。重點在於，本書逐步灌輸正確的系統化架構性思維，將安全性嵌入到系統當中，才能夠有效地保障機敏性資產的處理方式。當然，我們更期待讀者能夠創造自己的技術，補充本書提及的內容。

就安全性架構而言，雖然終極目標是透過保障寶貴資產進而緩解風險，但解決方案必要的其他品質也不容忽略，例如效能、可用性、彈性或成本。雖然你的職責是保障企業營運安全，但也必須理解企業組織其他的驅動因子。

目標讀者

正在架構安全控制措施，並計畫將它們整合到資訊系統裡的人，就是本書目標讀者。安全的架構性思維不僅針對安全架構師，也是設計資訊系統所有架構師的責任。因此，本書適合所有參與架構系統的人。

我們目前在英國兩所大學教授這些概念與技術，是英國 NCSC 認證碩士學位之網路安全學位單元的一部分。本書循序漸進建立這些概念與技術，使之適於教學；這都要感謝過去每位要求擁有這樣一本教科書的學生，是他們催生出這本書。

本書假設讀者對資訊系統與雲端運算已有基本認知，並具備有助於制定良好架構決策的實作經驗。架構師的學習方式並非只是繪製圖表，而是透過實際交付應用程式和基礎架構，建立自身的專業能力，全球許多以此為業的架構師都是以這種方式學習。

第五章

系統關係

「資料就是一切！」是否耳熟能詳？就安全性而言，重點在於保護資料，避免失去可用性、完整性與機密性。但多數時候重點往往從保護應用程式正在處理的資料，轉為保護儲存與處理資料的基礎結構。這種輕忽的態度，證明了採用「資料為先」處理方式，來設計安全控制的必要性，架構師應該專注在保護資料的方式上。

本章將探索資訊系統的安全設計流程，並強調在資料傳輸中、靜止時或使用時都必須保護資料的重要性，進而解釋安全保障的核心在於保障重要的資訊資產，而非只有 IT 系統。接著，檢視以資料類別歸類資料資產的方式，並且點出資料處理過程中，會產生詮釋資料等等的新資料。最後以資料對機密性、完整性與可用性缺失的敏感度為基礎，為資料分級。

建立系統關係圖，再找出掌控資料進出系統的交易，是資訊資產清單詳列系統資料流的起點。之後會依敏感度與適用的法律與法規限制，歸類資料，這樣做可設計出依資料敏感度調整的安全控制。

階段性人為產物

本章主要目的在為一般性的架構開發奠定基礎。

本書各章都會在人為產物相依圖裡以黑底白字方框強調人為產物，如圖 5-1 所示，說明將安全性架構到系統裡正在經歷的旅程。系統關係圖與資訊資產清單，是審查過企業背

景環境下各種人為產物後，建立的兩大關鍵人為產物。務必記錄影響架構的關鍵需求與背景資訊，將重要假設記錄在 RAID 日誌裡，如第 10 章所述。

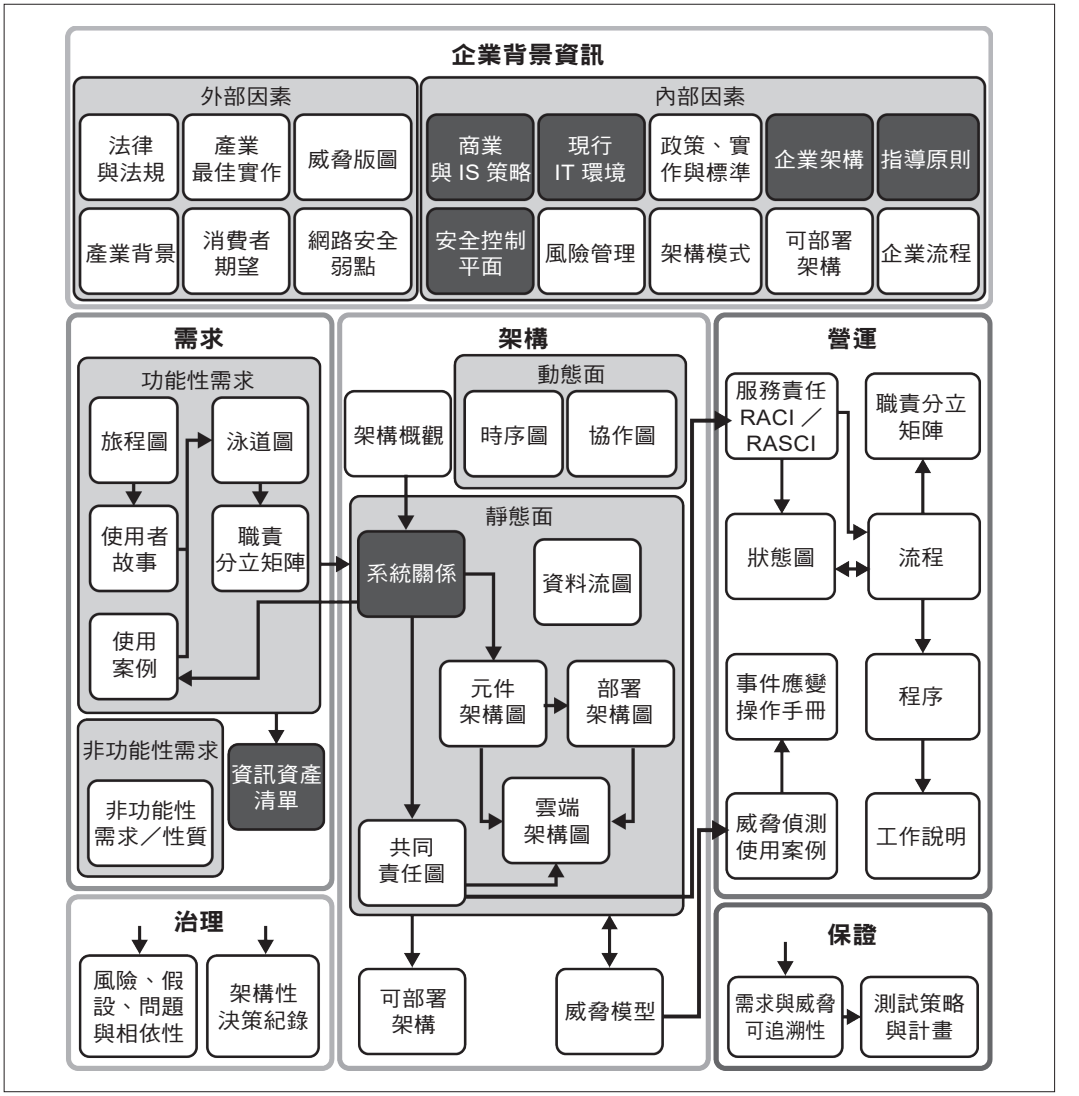

圖 5-1　系統關係的階段性人為產物

案例研究背景

再次提醒，為強化學習效果，附錄 A 的案例研究備有本章企業背景與現有 IT 環境的相關資訊。

本章將延續資料保護相關資訊，包括資料價值的探討、需要考量的資料生命週期，以及與零信任架構間的關係。

資料保護

資訊安全的重點不是實施各種安全功能，而是保護企業組織與其客戶資料，藉此保障企業組織商譽與企業遵循適用法律與法規等寶貴價值。

識別通過系統邊界的資料流，再依機敏性歸類資料，並依據政策定義必要的安全控制，就能為企業真正重要的保護措施排列優先順序，資訊安全繼而成為策略優勢而非義務。

接著將依下列小節討論資料保護的概念：

- 資料價值
- 資料安全生命週期
- 詮釋資料
- 零信任與資料流

這些內容有助於理解本書的技術背景。

資料價值

資料對企業組織的價值，是實施安全控制決策時的關鍵要素。先找出對企業組織而言重要的資料，再利用類別或資產類型，協助識別必須保護的資料，例如下列資產類型：

核心資產（*Crown jewels*）

代表企業組織最寶貴資訊資產的資料，若被揭露、遭到修改或無法使用，可能導致企業倒閉，這裡指的不只是原始資料，還有資料處理後產生的新資料。

個人資訊（*Personal information，PI*）

此類資料涉及個人或可辨識的個體，敏感度不算高，公開揭露不至於對當事人造成無法彌補的損害，例如姓名或住址。

機敏個人資訊（*Sensitive personal information，SPI*）

涉及個人或可辨識個體的資料，相當敏感，以致於揭露可能對當事人造成無法挽回的傷害，例如種族或族群、性生活、政治主張、犯罪紀錄、宗教信仰、工會活動與身心健康等。

財務資訊（*Financial information*）

個人或可辨識個體的財務紀錄資料，包括銀行紀錄、稅務紀錄與會計紀錄等。

若企業組織沒有這類資產類型清單，應該為正在架構的解決方案建立一份自有清單，以便在專案間清楚傳達資料價值。

NIST 出版品「NIST SP 800-160 Vol. 1 Engineering Trustworthy Secure Systems」（*https://oreil.ly/AhLi9*）敘述的一般資產類型，可作為補充用的參考，雖然它的重點在於系統治理，但或許能為企業組織提供開發自有資產類型的想法。

資料安全生命週期

系統在處理資料時，從資料的建立到處理與最後的銷毀，會經歷一段生命週期。了解系統的資料生命週期，就能了解資料流動路徑與系統處理階段，積極找出處理流程各個階段適合的安全控制。

資料安全生命週期的範例很多，雲端安全聯盟（CSA）制定的 *Security Guidance for Critical Areas of Focus In Cloud Computing*（*https://oreil.ly/7m8nU*）生命週期即為一例。不過，生命週期的敘述至今仍沒有一致的產業標準，本書所使用的資料生命週期模型如圖 5-2，用八大階段強調重要環節。

建立（*Creation*）

初始的資料建立，例如使用者將資訊輸入表單，或由感應器產生的資料。

儲存（*Storage*）

資料留存，若非靜止狀態就是正在處理，例如磁碟儲存裝置或資料庫保存資料。

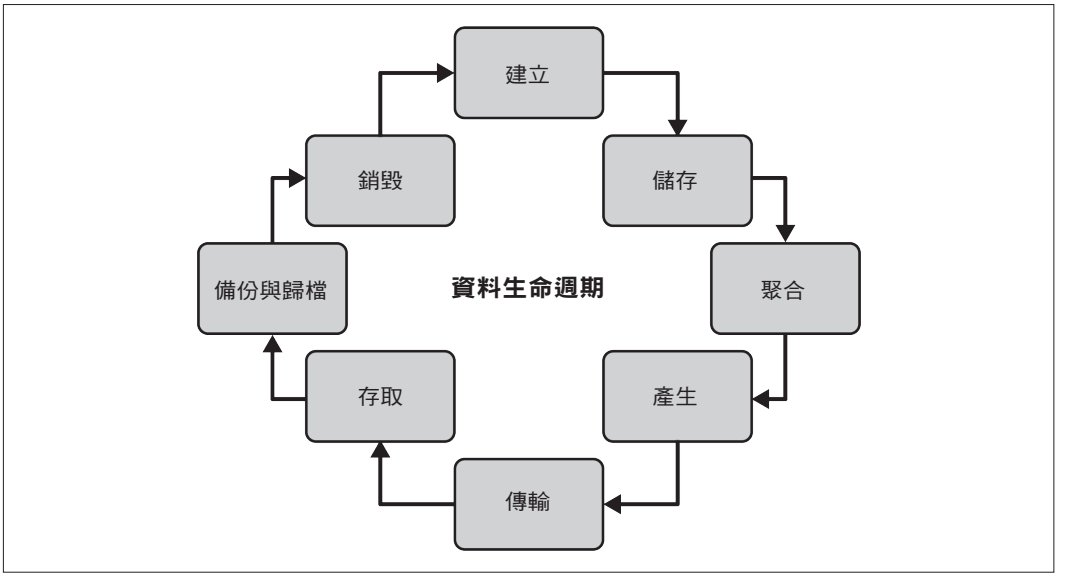

圖 5-2　資料安全生命週期

聚合（*Aggregation*）

多個來源的資料整合成單一資料集。這個環節相當重要，因為聚合資料可能提高最終成果資訊的機敏性。

產生（*Generation*）

現有的資料經過處理或分析會產生新資料。如先前所述，這種衍生的資料同樣可能提高最終成果資訊的機敏性。

傳輸（*Transmission*）

資料在不同位置移動，例如在公開或私有網路間傳輸，要留意的是，傳輸過程中的某些節點也可能暫時儲存資料。

存取（*Access*）

檢索與使用資料，例如使用者審核報告，或系統執行運算。

備份與歸檔（*Backup and archive*）

為了復原、合規性與歷史紀錄等目的，中 / 長期保留資料。

銷毀（*Destruction*）

將資料從系統移除，刪除或安全抹除儲存裝置裡的資料即為一例。

可依據資料生命週期模型，考量資料在傳輸、儲存與處理時的需求；同時，資料也需要安全控制，才能避免資料生命週期各個階段未經授權存取、修改或揭露資料；最後，依據資料的機敏性與指定的類別，決定設計與實施控制措施的方式。

詮釋資料

資料處理後會產生記錄原始資料洞見的新資料，稱為詮釋資料，內容為資料相關的資訊，包括：

敘述性詮釋資料（*Descriptive metadata*）

附加上主題、原創、發布者這類標籤，主要在敘述資料的內容。

結構性詮釋資料（*Structural metadata*）

敘述書籍的章節或音樂專輯的曲目這類組織性結構。

技術性詮釋資料（*Technical metadata*）

描述書本尺寸或影片紀錄格式這類資料的技術特性。

管理性詮釋資料（*Administrative metadata*）

支援資料管理的活動，例如最後變更紀錄的日期、資料安全分級，或數位版權管理系統的資料。

請謹記，詮釋資料是提供洞見、提升搜尋能力與協助資料管理的寶貴資源，對提供搜尋功能與人工智慧的企業組織而言，這是商業營運的根基。

若不能有效保護組織與客戶的詮釋資料，而導致機敏資訊遺失，企業組織就會有生存風險，因為這些資料可能屬於「核心資產」的資料，比原始資料更敏感，所以需要比原始資料更慎重的安全控制。

零信任與資料流

實施零信任原則的解決方案必須檢驗資料流，因為零信任以嚴格控制措施保護資料為基礎，並透過持續安全監控，針對機敏資料偵測是否有未經授權存取。

為了降低或除去盲目的信任，每個試圖存取資源的裝置、服務與使用者，都要先驗證，才能授權存取零信任解決方案的系統。「假定入侵」原則帶來的啟示就是：安全控制應盡可能部署在接近資料的位置，因為必須持續假定其他系統可能已遭入侵。

架構師檢驗資料流，可以更了解使用者與裝置在網路上的行為模式，找出可能象徵安全威脅的非常規活動。因為安全團隊有能力監控資料流、偵測威脅、找出潛在弱點，所以在弱點遭不當利用前，可以主動修復，降低攻擊者造成的危害。

與企業組織的合作過程中，我們發現橫跨多個商業流程的整體考量，可以減少零信任解決方案的數量；而先前探討的資料分類，則有助於引導系統套用安全控制的等級。這兩種策略都能有效降低實施的成本、提升交付的速度，並減少持續的技術支援問題。

整體而言，檢驗資料流是零信任架構的重點，架構師需要藉此找出有效的安全控制，還要能確保持續監控資料流偵測威脅。

接著要了解資料進出系統時，在系統邊界的互動。並由這個邊界定義系統範疇，以找出資料需要的保護措施。

系統關係圖

繼了解資料機敏性的重要程度後，系統交易流程將依序傳輸、處理及儲存這些資料。

這裡從識別系統邊界，與交易流程跨越邊界的方法，來開始架構系統。交易流程可以是人類行動者觸發，也可以是系統行動者。

圖 1-7 為架構分解圖。第一層為敘述系統邊界的系統關係圖，處於架構分解的頂層，如圖 5-3 所示。這層考量人類及系統行動者與之互動的系統，同時隱藏內部機制。

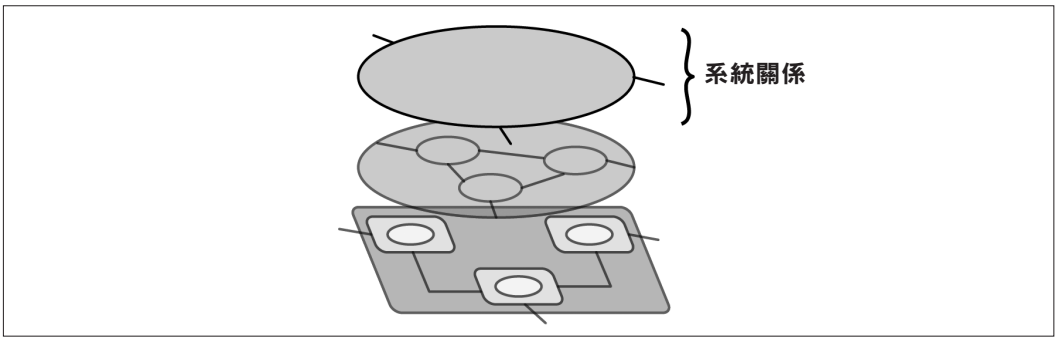

圖 5-3　架構分解 - 頂層

1960 年代的**系統關係**（*system context*，*https://oreil.ly/PI6JO*）圖，是系統工程普遍使用的技術，收錄於 International Council on Systems Engineering（INCOSE）的 Systems Engineering Body of Knowledge（SEBok，*https://www.sebokwiki.org*），公認為架構性思維流程的開端。Tilak Mitra 的著作《Practical Software Architecture》（*https://oreil.ly/0oV7b*），與 Simon Brown 所寫的 C4 Model（*https://c4model.com*）皆有相關說明。

第 6 章與第 8 章將檢視架構的另外兩層。

如同第 4 章所述，使用案例或使用者故事描述行動者執行的活動，這些活動會觸發系統內外的資料流。

深入細節前，先來了解系統關係圖的負責人。

系統與安全架構師角色

應用程式與基礎結構的架構師運用系統關係圖，視覺化呈現解決方案與周邊環境的互動。他們用這張圖指出產生互動的系統邊界、參與的行動者與使用案例，再檢視由使用案例驅動的系統交易流程。

為了安全性，本書進一步改良這個構想，找出這些交易流程的資料流動，這樣做將有助於架構師著手製作系統的資訊資產完整清單。為找到零信任技術的解決方案，請識別已知介面型態，例如 API、HTTPS 或 MQTT，再依資料分類，找出適切安全控制。後續章節會檢視系統內部的資料處理流程。

安全架構師與應用程式架構師協同合作設計應用程式時，安全架構師可另外補充安全相關資訊，強化應用架構師的人為產物。然而，若安全架構師獨立負責解決方案的架構，就必須自行完成所有分析；例如，安全架構師負責建立身分識別與存取管理系統這類安全服務，就必須完全負責解決方案的架構。

現在，繼續探討概念。

系統關係概念

建立解決方案架構時系統關係圖非常重要，因為它提供系統的高階檢視，包括與之互動的外部行動者與系統，同時也能定義系統邊界，明確架構涵蓋的範圍。

架構師設計解決方案時，若了解系統外部背景，可找出相關使用案例與資料流。同時也能識別所有對現行 IT 環境的約束或限制，在這樣的基礎上，能進一步識別外部行動者可能採取的相關潛在安全威脅與其他風險。

綜上所述，系統關係圖在打造解決方案架構方面扮演決定性的角色，既能確保架構符合商業營運需求，又能遵循企業組織的整體 IT 策略。

這裡將從簡單的系統關係圖範例開始，說明繪製的概念。圖 5-4 呈現的線上商店簡圖遺漏了一些行動者，隨著後續的探討，你會發現需要多次更新。

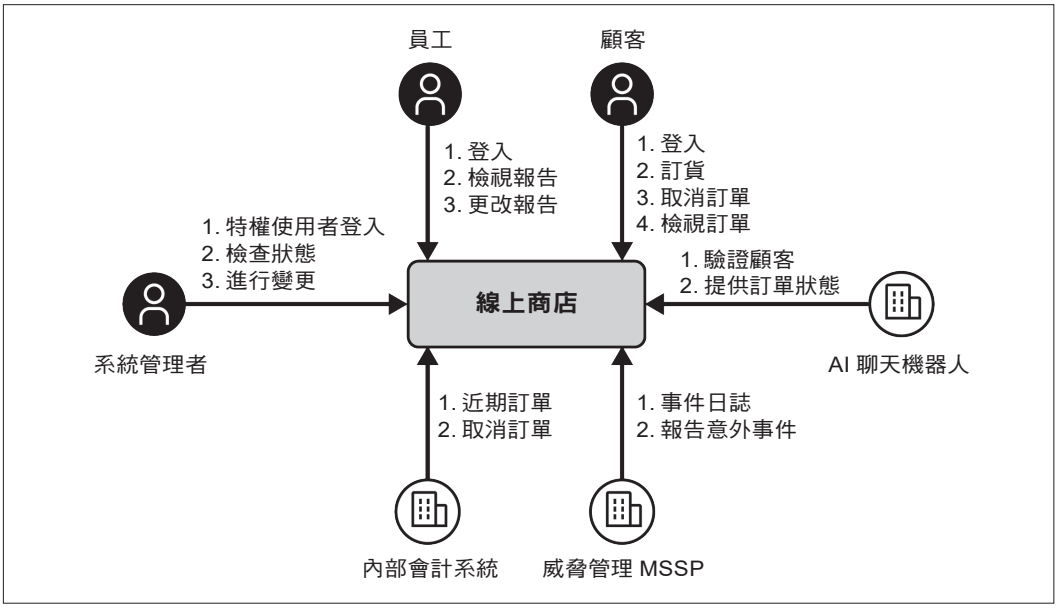

圖 5-4　系統關係範例

圖 5-4 內含屬性如下：

人類行動者（*Human actor*）

系統關係圖的人類行動者，是指資料進出系統的一組流程或使用案例中，與系統互動的個人或一群人，其中含括擁有該系統的企業組織內外部人員，內部人員包括會計、開發人員、IT 支援人員、網路安全人員、客戶、內部合規性等與系統互動的任何個體或群體。重點在於，應以角色，而非以個人識別與系統互動的行動者。

這張圖表納入「員工」角色，但不夠明確，用「業務管理」更能識別也更合適。「系統管理者」角色有比較明確，但管理者角色的型態很多，可能需要再分解。

系統行動者（*System actor*）

系統關係圖裡的系統行動者，指的是系統外部的資訊科技（IT）、營運科技（OT）與物聯網系統，它們透過資料進出系統的一組交易或使用案例與系統整合；這些系統行動者，可能有支付供應商、外部代理安全服務或後勤系統等。在圖 5-4 範例中，我們將威脅管理代管的安全服務供應商（MSSP）整合到系統裡。

這裡說的「外部」，指的是系統之外而非企業組織外部，因此，其他部門及並非專案系統設計的皆屬外部，這種狀況下，外部系統會包含內部會計系統。

一般而言，受制於不同於主系統安全政策的系統行動者，不應該出現在系統裡。若系統有本身的政策，可能是 SaaS 服務，會透過與組織的合約關係執行安全治理。這種屬於與第三方的合約，例如 SWIFT（*https://www.swift.com*）支付閘道，遵從的政策可能與內部系統行動者不同，因為 SWIFT 會要求為了連結，必須實施特定控制。

盡量避免過於瑣碎也很重要，有時圖表會顯示全部的人類與系統行動者，而顯得太複雜，因此一個專案可以建立兩個系統關係圖：一個人類行動者，一個系統行動者。

使用案例（*Use case*）

每位行動者都有一組使用案例，列出它觸發傳輸與處理資料的交易流程，而交易流程可以識別進出系統的資料類型。第 4 章已探討過使用案例或使用者故事的定義。

本章將舉例探討系統關係的建構，若想進一步了解建立系統關係圖的相關資訊，請參考《Practical Software Architecture》的〈Chapter 4: The System Context〉（*https://oreil.ly/TanKz*）。

技術設計圖符號表示法

架構師的主要工作是溝通複雜的主題，不僅使用文字，也會用到繪圖與圖表。如同所有溝通，在語言上達成一致很重要，如此各方才能以簡明易懂的方式交流，這不僅適用於文字，設計圖表也是如此。通用的設計語言才能讓檢視者輕鬆理解看到的內容。

我們遇到的狀況是：各家 CSP（雲端服務廠商）自有繪製雲端架構圖的符號表示法，那是一種品牌烙印、特定的實施規格還加上色彩，在在使得這些圖表難以處理。因此本書以獨立於 CSP 的方式繪製圖表，使之能夠支援混合雲架構。

這裡選定的技術設計符號表示法，是獨立於 CSP 同時支援 IBM 架構師的 IBM 統一方法框架。IBM 設計語言站台（*https://oreil.ly/EZXcj*）提供技術圖表的專屬章節（*https://oreil.ly/Ts3s4*），可供參考。

圖表可以有不同的抽象層級。**邏輯**（*logical*）**檢視**呈現系統的高層概念，與實際運作方式不同。過去，架構師會參考系統的實際檢視，但隨著虛擬化與容器的發展，現在使用的是稱為**規範檢視**（*prescribed view*）的圖表。

本書使用如圖 5-5 所述技術設計符號表示法的子集。[1] 符號已針對印刷書籍頁面大小與出版的一致性作過調整。原始圖表請參考本書配套網站（*https://securityarchitecture.cloud*）。

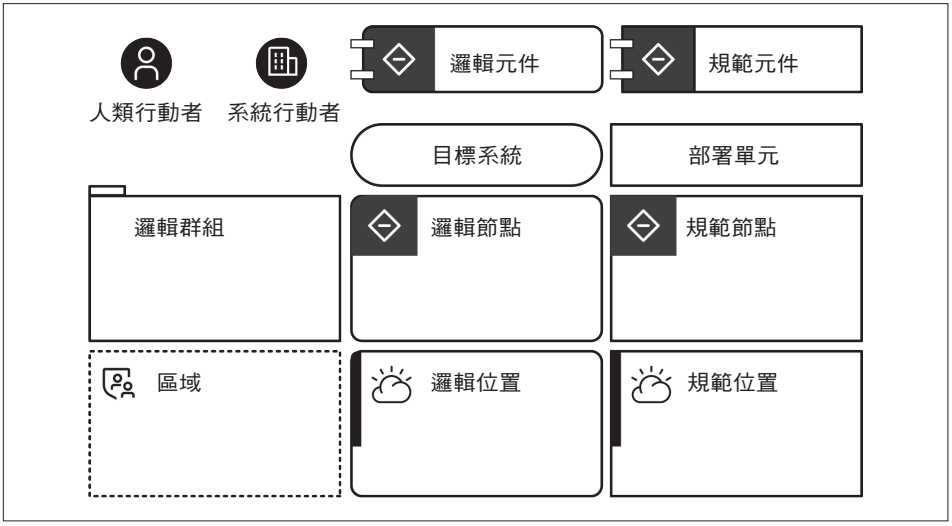

圖 5-5　技術設計圖符號表示法；請參閱原圖（*https://oreil.ly/SAHC*）。

1　若所處企業組織使用不同的符號表示法也無妨，重要的是所有圖表一致，才能有效率溝通。

本圖表使用下列基本元素：

行動者（*Actor*）

以圓形代表行動者，並明確區分與解決方案互動的人類行動者與系統行動者。第 5 章會介紹行動者概念。

目標系統（*Target system*）

目標系統即為正在設計的系統，位於系統關係圖正中央。

部署單元（*Deployment unit*）

方角矩形代表部署單元。指的是節點上已部署的功能性實體，例如應用程式或資料庫。有可能一個節點擁有許多部署單元，所以為防止圖表單一節點布滿太多矩型，也可以直接使用文字清單，在節點上列出部署單元，無須各單元單獨表示。

位置（*Location*）

左上角暗黑標記的矩形代表位置，圓角為邏輯位置，方角則是規範位置。位置可以是地理區域或是組織性位置，可見第 8 章的介紹。

元件（*Component*）

帶有兩個連接埠且左上角有個方框包圍的符號，代表元件。圓角代表邏輯元件，方角則為規範元件。元件代表解決方案功能，可見第 6 章的介紹。

邏輯群組（*Logical group*）

左上方以方角標記的矩型代表邏輯群組。這些元件是在邏輯上分組，簡單來說，分組的這些元件，共享類似的非功能性需求或其他共同特性。

節點（*Nodes*）

為了功能的執行或運作，將功能性元件部署（或置放）在節點上。例如，可以將功能性元件當成可執行程式（部署單元），部署（或置放）到虛擬伺服器（節點）上。

左上角框住符號的矩型表示節點。圓角代表邏輯節點，方角代表規範節點。元件會透過部署單元部署在節點上，依選定技術與規模指定節點，所以節點也可以是虛擬伺服器。

為了騰出更多空間給紙本書頁面，本書簡化圖表選擇只繪製規範節點，所以沒有雲端服務或資源實體的方塊。

區域（*Zone*）

記錄使用通用安全政策與威脅的位置，我們使用網路區域表示，以方角虛線框呈現，元件與節點皆置於其中，此外，一個區域可橫跨多個位置。

本書選擇使用 draw.io GitHub 站台 issue 3914（*https://oreil.ly/qUgp1*）釋出的 IBM Cloud 圖示，對於黑白出版品而言相當清晰。出版前夕，這個架構圖示重新發表（*https://oreil.ly/sbgwf*），使用彩色方框，但圖示太小無法用於出版品，讀者可自行考量設計圖表使用的圖示標準。

建立系統關係圖需要一些背景資訊，因此在套用到案例研究前，這裡先釐清概念。

商業與 IT 背景環境

開發系統關係圖時，必須了解系統的商業背景與現有 IT 環境，後者可能會對解決方案造成限制。商業目標將決定目標解決方案架構的商業流程，繼而成為系統關係上，識別人類與系統行動者的依據，同時，現有 IT 環境會產生需要與系統整合的現有行動者，可能會限制解決方案的開發。

本書附錄 A 提供的案例研究，包含商業背景與現有 IT 環境的資訊，說明套用這些資訊開發人為產物的方式。開發的專案若處於早期階段，可能沒有這方面資訊，建議與關鍵利害關係人協同合作，開發系統關係圖，以利釐清商業背景。

接著繼續套用這個概念，使用附錄 A 的案例研究，開發系統關係圖。

案例研究：系統關係圖

找出系統邊界是定義系統關係的第一步，藉此可界定系統內部的系統功能性元件，與系統外部的人類與系統行動者。

人類與系統行動者的定義先前已探討過，現在要從案例研究找出行動者。案例研究中，有架構概況圖的協助，若無，則必須透過文字或面談以識別。案例研究中，以**實線方框**標記強調的潛在系統行動者。建議如下圖所示，自行標記案例研究。

標記案例研究以識別系統行動者

- 債務催收機構**Clean Air Debt**收到未在 48 小時期限內支付費用的車主資訊後，以信件方式通知遭到罰款的車主，並在一段期間後催收罰款。
- 車主可以使用自己的**Google 或 Microsoft IDs**登入入口網站，使用程式註冊他們的車輛，以便進入 Guildford 時自動付款。
- 這個方案使用客服中心的**Guildford Service SaaS**應用程式回應電話查詢並接受付款。
- 使用公有雲廠商的雲端服務提供**AI chatbot**，可以更快處理駕駛人查詢，無須等待客服中心的電話支援。
- 威脅管理公司**Clean Threats**在其安全營運中心（SOC）管理所有安全服務。

圖 5-6 系統關係的繪製，將行動者安排在居中方框的系統四周；方框內沒有已識別元件，是因為這裡的重點是系統邊界。方框內部元件將於第 6 章說明。

人類行動者有兩種人員圖示，暗影圖示呈現與營運流程有關的行動者，淺色圖示顯示與系統操作相關的行動者；建物圖示代表系統行動者，第 6 章會探討它們的差異。

行動者圖示

不同型態的使用者使用不同的圖示說明，並非這個方法的規定，企業組織可以自訂樣式。

這裡針對部分行動者，以範例使用案例說明事件觸發通過系統的交易流程方式，試想缺少的使用案例，記錄其他行動者。

接著，我們要識別人類行動者。

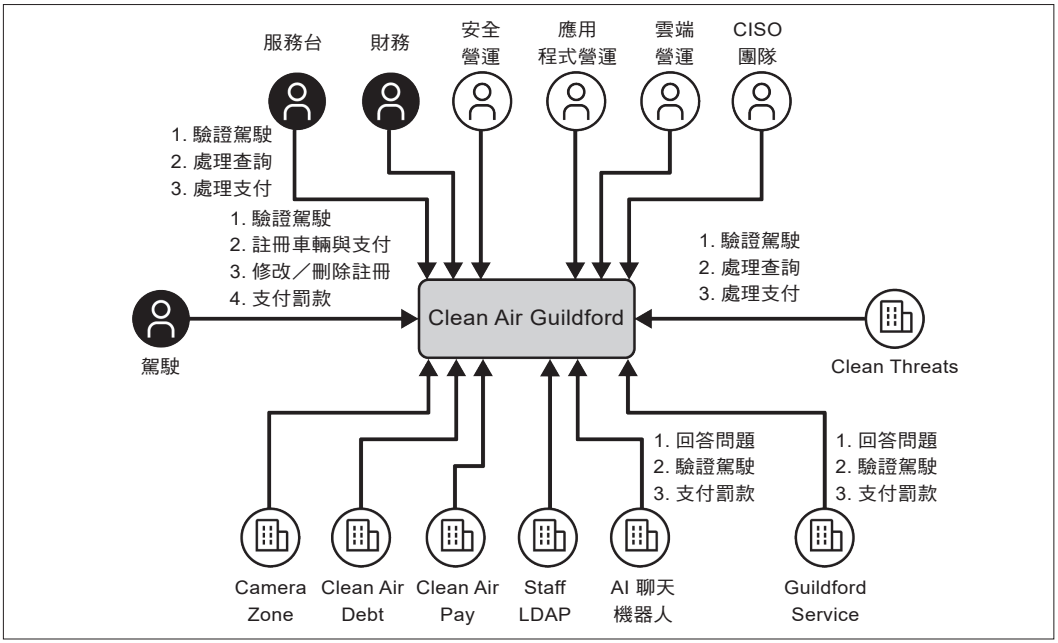

圖 5-6　案例研究的系統關係；請參閱原圖（*https://oreil.ly/SAHC*）

識別人類行動者

為了繪製附錄 A 案例研究的系統關係圖，必須找出所有提及的人類行動者。案例研究僅提及一位人類行動者，但與系統互動的其實很多，請研讀案例研究，識別可能需要存取系統的角色。

案例研究提及設有服務台且有員工負責運作。這個案例中，服務台員工為 Clean Air Guildford 職員，所以是人類行動者；若是委外的 SaaS 應用程式提供服務台應用，請繪製名為「Guildford Service」的系統行動者。

營運系統一定有顯然不同的角色，若案例研究沒有確切提及，架構師就要負責找出這些不同的角色。例如，考量「財務」時，需要存取與支付相關的財務功能，請加入名為「Finance」的人類行動者。

這個案例中，「雲端營運」（Cloud Operations）團隊負責基礎結構、「應用程式營運」（Application Operations）團隊負責應用程式，而「安全營運」（Security Operations）團隊負責安全性。當委外團隊透過系統存取而不是互動式登入，就可能從人類行動者轉

變成系統行動者，舉例來說，系統管理活動可以在委外團隊資料中心代管的服務管理入口網站自動化處理。若這些團隊還透過跳版主機或堡壘主機存取，就要以人類行動者圖示表示；總之，有兩種存取系統的途徑，就要用兩種圖示。

檢驗資料生命週期

想像整個系統的資料生命週期。例如，存放在資料庫的資料可能需要備份、歸檔與網路復原的能力，這種時候處理網路復原需要人類行動者也需要系統行動者，因此，提供這些能力的系統元件必須放在系統內部還是外部？這些可以先放在待辦清單，或者如第 10 章所述放進 RAID 日誌的議題（issue）。重點在於，不能忽略這個必要動作。

檢視這份系統關係，是否發現缺少負責批准相關流程的人類行動者？例如，新增系統使用者時，可能需要部門主管的核准。因此，建議增加「Line Manager」（部門主管）這個新角色，以確保系統支援這樣的批准機制。

討論過識別各種行動者後，最終找出的行動者可能不影響架構。舉例來說，SaaS 應用程式運作的電子郵件系統會有許多人類行動者。雖然管理者與一般職員在電子郵件系統裡可能有不同功能，但不影響架構，不需要全部識別，用「Email User」（Email 使用者）這種一般角色也可以。

接著，繼續探討識別系統行動者。

識別系統行動者

架構 IT 系統時，應該把重點放在功能性元件，而不是從底層實施的技術開始。找出這些功能性元件後，將它們拆分為二：**屬於系統**（*to the system*）內部與外部的元件。

找出功能性元件後，一定要確認所處企業組織是否擁有或負責營運這些元件。若企業組織無法掌控應用程式的設計，而由服務供應商控制雲端帳號；或者執行的安全政策與自身企業組織政策不一致，這個元件就屬於設計中系統的外部行動者，而針對外部系統，必須有一份合約，能與供應商在實施安全控制上取得共識，或者明定供應商要求你設計的系統必須具備的控制措施。

企業組織內部其他系統可能屬於外部系統行動者，而非內部元件，特別是處於企業組織不同部門的系統，這種狀況下，就可能有必要與其他內部系統的供應商協商。例如，案例研究可能沒有識別出會計系統，但若它是不屬於專案範疇的企業內部系統，在系統關

係圖裡就應該是外部系統行動者。這些可能會需要備忘錄（DoU）這類內部協議，雙方一致同意必須採取的安全控制。

介面修改

外部系統介面可能與系統行動者不相容，因此要選擇必須修改的系統。不論必須修改的介面是外部行動者還是新系統，都要由企業設計授權委員會決定。

還有遺漏系統行動者嗎？回顧附錄 A 案例研究的架構概況圖，想想駕駛如何使用應用程式進行身分識別與驗證處理支付？你可能已經發現，缺少 Google 與 Microsoft 的身分識別供應商的整合。

你是否認為系統行動者其實屬於案例研究設計系統裡的元件？在這個案例裡，有個行動者是公有雲平台託管應用程式的 PaaS 服務，指仍未完整客製化的服務。此外，AI 聊天機器人應該歸屬於系統內部元件，因為它需要當成專案的一部分訓練。

整體而言，要得到正確圖表，必須分析案例研究提及的所有人類與系統行動者，還要考量功能性元件的擁有者與負責營運的人，以及連結系統的內外部行動者。

現在，將上述資訊融入現正記錄的系統關係裡。

記錄系統關係

雖然這份圖表對某些專案已足夠，但其他專案有可能會需更全面的說明。表 5-1 列出行動者與各自的敘述與介面，剛開始可能無法理解這些介面，表單稍後會另外詳述。

表 5-1　行動者與介面表

行動者	敘述	介面
駕駛	車主進出空氣淨化區	駕駛透過 Clean Air Guildford 網路應用程式的 TLSa 1.2 對話連線。 第二個介面是電話，撥打至 Clean Air Guildford 服務台。
服務台	Clean Air Guildford 員工使用應用程式為車主提供支援	Clean Air Guildford 網頁應用程式的 TLS 1.2 對話連線。 第二個介面電話，接收駕駛來電。

行動者	敘述	介面
Camera Zone	企業組織管理提供自動車牌辨識（ANPR）的攝影機	提供 REST API，檢索專屬私有網路連結 TLS 1.3 對話中的攝影機資料。
…	…	…

[a] 傳輸層安全性（TLS）的對話層加密版本至少要 1.2，但 1.3 更好，因為先前實作存在已知漏洞。

注意，駕駛與服務台各有網頁與電話兩種介面，後續分析請考量所有可能的介面，例如電子郵件與 SMS 訊息。

以下檢查清單，可協助審核系統關係圖開發的品質與後續可能的活動。

系統關係 QA 檢查表

- 考量與系統互動的所有人類行動者，包括公司員工與透過合約提供服務的第三方。這裡的範例是 CISO 團隊，但可能也會有專責的合規性或稽核團隊。
- 將人類行動者分解到相關使用案例或使用者故事可識別角色的程度。這裡的範例是財務角色，在實施職責分立時，可能需要多重角色。
- 可以考量依地理位置分解人類行動者，例如考量到不同交易流程，辦公室工作者一個角色，遠端工作者另一個角色。
- 識別與系統互動的系統行動者。這個範例裡沒有電子郵件閘道器或文字訊息服務，但如果是運輸實體貨物的系統，可能會有某種後勤物流或送貨服務。
- 識別你無法掌控且需要訂定協議才能與系統建立介面的系統行動者。
- 考量可能定期或批次觸發交易流程的事件。這些流程可能會處理不同的資料，路徑也可能與僅考量外部事件觸發的流程不同。

系統介面實作角度

這裡敘述的系統關係是概念性的，但可以利用同類型圖表，考量系統實作方式，協助釐清如何整合流程。

使用系統關係圖的變體，除了可以描述特定網路連結的資料流，還能定義網路安全控制。這裡直接跳到實施的細節，對於開發軟體設備的軟體工程師而言，這種系統關係圖可能正是你會採用的形式。

圖 5-7 範例中，單一方框呈現系統的進出連結，描述特定網路通訊埠的資料流。該圖表分別顯示軟體系統公開與私有網路介面流動的連線。圖表頂端為公開網路介面使用的網路連結，下方為私有網路，側面的 ICMP 連結，必須同時具備私有與公開網路的連結能力。

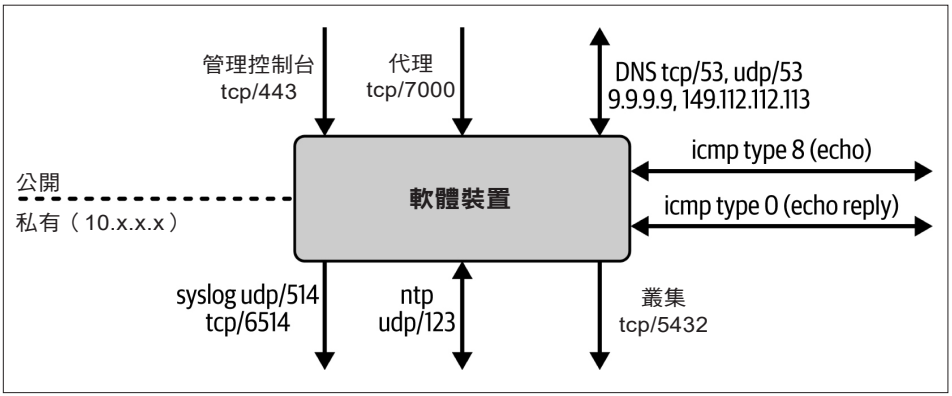

圖 5-7　系統介面實作圖

完成系統關係圖後，接著記錄已識別資料，再依據資產清單裡的敏感度，歸類資料，依據這樣的資料分類決定所需的安全控制。

資訊資產清單

為了識別與歸類系統傳輸、處理與儲存的資料，建立資訊資產清單相當重要，透過識別與分級資料，就能找出資料需要的安全措施。

除此之外，資訊資產清單也可以確保資料的處理符合法律與法規需求。例如，處理的資料含括個人可識別資訊，後續整理出來的資訊資產清單，就能確保遵循資料隱私法律與法規處理這些資料。

整體而言，資訊資產清單可供全面了解系統資訊相關資產，讓企業能更有效率並更精準地保障與管理這些資產。

探討記錄資訊資產清單前，先來了解找出不同敏感度資料所需控制措施的資料分級方式。

資料分級

涉及資料安全時，企業組織依資料敏感度歸類資料是必要手段，可以導出保護資料所需的控制措施。一般來說會依據 CIA 鐵三角：機密性（confidentiality）、完整性（integrity）與可用性（availability）為資料分級。

資料揭露造成的潛在影響決定機密性分級。傳統的分級系統通常從公開的資料開始，再進展到遺失會對企業組織造成重大影響的內部極機密資料。

企業組織機密性損失的分級系統範例，如表 5-2 所示。

表 5-2　機密性的分級系統

種類	敘述	控制指南
公開	大眾可以取得的資訊，遺失時不會對企業組織或客戶造成影響。 網際網路公開網站發布的所有內容即為一例。	不需要安全控制。
內部	供所有員工使用的資訊，但未於公開網站發布。 例如餐廳菜單或申請新筆記型電腦流程相關資訊。 揭露不致造成重大影響，但沒有公開的必要。	資料必須託管於公司內部網路，且無須為了問責執行身分識別或驗證。[a]
機密	可能危及企業組織，但未觸犯法律或法規的資訊。 例如正式釋出前的產品資訊草稿，或內部產品發展的藍圖。	針對個人問責並依特定角色實施控制措施，需要個人身分識別與驗證才能存取資料，且所有傳送中與靜止時的資料都必須加密。

種類	敘述	控制指南
高度機密	此類資訊如遭揭露，可能嚴重危害企業組織與客戶，還可能觸犯法律與法規。 例如洩漏客戶的機敏性個人資訊，可能違反資料外洩保護法，導致企業遭到罰款，也嚴重危害公司商譽。	除了前述控制外，資料庫個人欄位資料也需要封裝或加密，確保就連特權管理者也無法檢視資料。

[a] 本例假設企業組織設有控制邊界，且內部網路未採取零信任原則。

注意，表格最後欄位是依資料分級，套用不同控制措施的想法，最後可能會產生一份備有各種安全控制的詳盡清單。

雖然針對完整性缺失的資料分級較少，但不代表不重要。舉例來說，在參與者不知情的情況下變更金融交易，或者是 AI 模型遭到竄改，就會導致做出不正確的決策，這就是喪失完整性所造成的嚴重後果。

針對企業組織完整性損失的分級系統範例，如表 5-3 所示。

表 5-3　完整性分級系統

種類	敘述	控制指南
低風險	資料的完整性遭到破壞對企業組織影響極小。 公開資訊或次要歷史資料皆屬於此類。 例如，餐廳昨天的菜單或免費瑜伽課程的海報。	
中度風險	資料完整性遭到破壞，對企業組織會造成中等程度的影響。 包括客戶電話號碼或明天的餐廳菜單。	這種資料可以套用簡單的控制措施，例如檢查電話號碼是否為有效格式，或是菜單品項是否為標準項目。
高風險	完整性遭破壞，將對企業組織造成重大影響的資料。 包括客戶的機敏性個人資訊、法律文件或商業機密。	必須套用雜湊值處理，以便偵測資料是否遭到修改。
重大風險	資料完整性遭破壞，將對企業組織造成全面且不可逆的影響。 包括企業組織營運的必要資料，例如金融交易、醫療紀錄或個人身分證號碼。	資料必須加密簽章，確保就算特權管理者修改也能偵測到。

當今混合雲環境，保障商業流程的可用性是網路安全的關鍵考量，通常是由復原力團隊負責處理這類風險，請注意，DDoS 保護這類控制措施，屬於安全團隊的職責範圍。

資訊資產 QA 檢查表

- 識別涉及所有人類與系統行動者的每個使用案例資料。
- 加入定期事件觸發或交易流程所衍生的資料。
- 根據企業組織的分級系統歸類資料後，找出需要遵循其他法律與法規控制的資料。

最後，分享一些運用這些技術的心得與想法，來結束本章。

總結

有些人低估系統關係的複雜度，認為不用花費太多時間開發，但我們的經驗顯示，沒有經過討論、文件紀錄與傳達這份簡單圖表的意義，很可能對專案有極大誤解，對專案範疇有不同想法若不盡早解決，會導致後續許多問題。

圖表很可能遺漏外部的整合點或人類行動者、可能忽略系統管理元件的整合方式，而導致需要另外的 IT 或非 IT 行動者與系統互動。請與各個團隊成員共同審核這張圖表，以利找出其他行動者。

開發安全服務解決方案的安全架構師，有責任開發系統關係；若是協助架構應用程式的安全架構師，雖然不是系統關係圖的擁有者，還是有責任補充相關資訊。並且，經過重要的利害關係人審查後，可能還需要另外的使用案例或使用者故事。

安全架構師可能自有一份資訊資產清單，若應用程式架構師沒有，安全架構師就必須自備。再次強調，一如常被輕忽的系統關係圖，資訊資產清單對整個團隊的溝通與保持一致而言，也相當重要，例如，系統處理支付資料的紀錄必須符合 PCI DSS，以便專案追蹤需求，若忽略這個環節，到了專案後期可能必須變更專案而造成延遲交付。

納入新需求反覆調整

至此，應該對系統關係有所了解，也對資料處理的型態有些想法了。架構性思維流程第一階段產生的人為產物，會為了反應識別的新需求而反覆更新，並制定新架構性決策。後續章節還會回來記錄新的行動者、資料與使用案例。