以使用案例管理需求的可追蹤性策略

Ian Spence 和 Leslee Probasco

Rational Software 白皮書

TP 166, 1998

Rational®
the software development company

目錄

摘要

在使用案例建模技術的許多商業應用程式中，使用案例模型必須與更傳統的需求擷取技術相結合，以提供專案有關的所有關係人可接受的需求管理流程。本白皮書探索可追蹤性策略，採用使用案例建模技術的組織可使用這些策略作爲其需求管理策略的一部分。

簡介與背景

可追蹤性項目

討論「需求管理」時，尤其在使用像 RequisitePro 之類的工具時，「需求」一詞的超載常常令人感到困惑。除了一般定義爲「需求」的項目之外，我們需要擷取和追蹤許多其他種類的項目的屬性和彼此之間的可追蹤性。這些其他的可追蹤性項目包括問題、假設、要求、名詞解釋、參與者、測試等等。

擷取和追蹤這些其他種類的可追蹤性項目，可幫助我們有效管理專案的需求。

定義：可追蹤性項目

任何需要從另一個文字或模型項目明確追蹤的文字或模型項目，以追蹤它們之間的相依關係。

關於 RequisitePro 方面，此定義可改寫爲：

任何在 RequisitePro 內以 RequisitePro 需求類型的實例代表的文字或模型項目。

RequisitePro 本身提供出色的工具，來定義、擷取和追蹤與軟體開發有關的許多種可追蹤性項目的值、屬性和彼此之間的可追蹤性鏈結。

隱含的和明確的可追蹤性

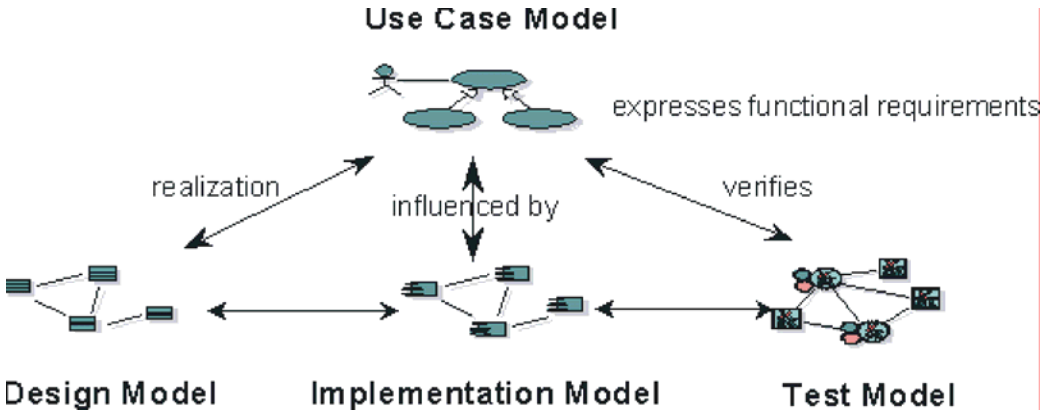
在任何開發流程中都隱含某種程度的可追蹤性。這通常是由流程中各構件之間的正式關係來提供。

這種隱含的可追蹤性的範例如下：

- 命名慣例
  亦即，設計模型中名稱爲 Fred 的類別由實作模型中名稱爲 Fred 的類別來實作
- 模型之間的對映建構
  亦即，Rose 中的元件視圖可讓 Rose 中的邏輯視圖中的套件和類別，明確地對映到實作模型中的套件。此對映可包含模型與不同套裝策略的應用程式之間的重新命名
- 模型項目本身之間的關係
  亦即，在 Rational Unified Process 中，設計模型中的使用案例實現化會追蹤到它們實現的使用案例。
- 不同視景的建立，說明一個模型的元素如何滿足另一個模型的元素所隱含的需求
  亦即，在設計模型中的使用案例實現化示範設計模型的模型元素如何分工合作來實現一個使用案例。這些在設計模型中提供一個使用案例視景，以驗證及補充設計模型中的類別和套件的靜態套裝。

所有這些範例都提供可追蹤性的層次，並允許使用開發模型所保留的資訊進行影響分析。

如下圖所示，使用案例驅動的開發涉及一系列相互關聯的模型。



此圖顯示模型和這些模型之間隱含的關係。模型之間的關係提供在開發流程中隱含的可追蹤性層次。

在進行使用案例驅動的開發時，需要一些支援的構件來支援使用案例模型，及啓用完整軟體需求規格的定義。在 Rational Unified Process 中，這些是指增補規格和名詞解釋。企業案例和願景文件也很重要，它們包含專案的需求、目標和特性的定義。

模型之間的關係不涉及這些支援的構件，因此內建於開發流程的隱含可追蹤性並不涵蓋它們。

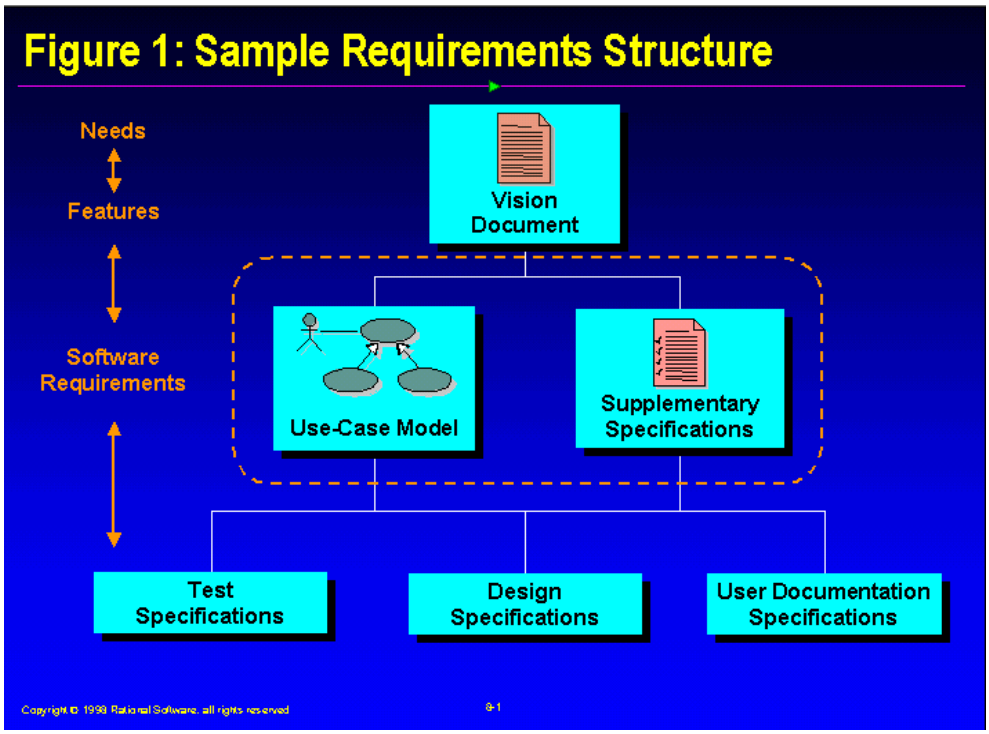
這些隱含的關係是開發流程的基礎，且因爲內建爲開發人員原有工作的一部分而從中受惠。這些關係對建模流程很重要，當模型成熟時，會建構及維護這些關係。

隱含的可追蹤性限於建模表示法中可用的關係。

因此，我們爲何還要其他明確的可追蹤性？

如果我們要採用需求可追蹤性的原則，就必須將「需求」、模型項目和其他可追蹤項目整合到一個可追蹤性階層中。我們也想要在開發流程中新增其他可追蹤性關係。

下圖是一個可追蹤性階層範例，它顯示願景文件所定義的「特性」與使用案例模型所定義的「軟體需求」和增補規格之間的關係。它也顯示如何在測試需求、設計和使用者文件中追蹤軟體需求。



如果我們查看增補需求（定義在「增補規格」文件中）和設計及實作模型之間的關係，就會知道模型之間隱含的可追蹤性並未涵蓋它。

這是專案通常需要的另一層明確可追蹤性的好例子。由於與使用案例模型之間隱含的關係，許多需求追蹤整個模型系列。沿著增補規格中的使用案例模型而擷取的增補需求，沒有直接與設計模型中需要考量它們的任何套件相關，或與實作模型中需要實現它們的任何元件相關。

其他範例包括下列各項之間的關係：

- 系統和使用案例模型的特性
- 使用案例模型和使用者文件
- 使用案例模型和測試需求。

在設定需求可追蹤性流程時，我們需要做的其中一個重大決定，就是我們需要的可追蹤性層次以及需要多大的明確可追蹤性才能達到此目標。我們希望需求、可追蹤性和管理的方法能夠促進開發流程，而不是使它更複雜和受到限制。

我們瞭解在開發構件中新增明確的可追蹤性會大幅增加專案的成本。當我們考慮到移入和維護此額外資訊的長期成本時，其成本更多。我們需要做的是確保建立專案可追蹤性的適當層次，並確保我們決定要維護的其他明確可追蹤性能得到投資報酬。我們希望開發人員花費時間在開發上而不是在追蹤上。爲了達到此目的，在專案中新增明確可追蹤性的成本之前，我們需要建立及預估可追蹤性策略。

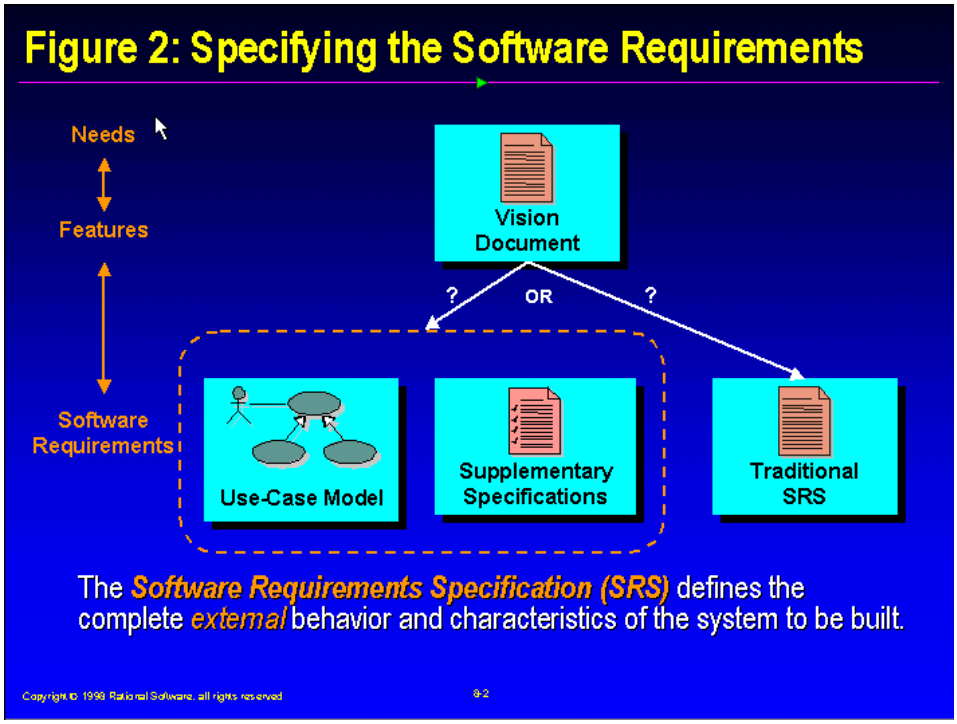
可追蹤性策略將定義我們想要新增至軟體開發流程的明確可追蹤性的層次。

管理支援的構件

上面的「圖 1」顯示 RUP 中的需求規格有關的構件。

值得注意的是，使用案例模型和增補規格形成完整的軟體需求規格 (SRS)。這表示我們不需要有傳統需求管理技術所需要的正式「軟體需求規格」文件。

下面的「圖 2」顯示傳統 SRS 文件通常與 RUP 構件是如何相關的。傳統 SRS 只是記錄「軟體需求」的一個替代方式。請務必瞭解，這兩個方法都可以提供「軟體需求規格」來定義要建置的系統的完整外部行爲。



此關係通常被誤解爲暗示「需求管理」的兩個模型無法共存。人們通常認爲，他們必須在傳統需求管理技術（使用正式「軟體需求規格」文件）與以利用使用案例模型和增補規格的需求管理技術爲基礎的使用案例模型之間做選擇。事實上，在某些狀況下，這兩種形式的「軟體需求規格」必須共存於相同專案內。

可能的可追蹤性策略

是否有多個可接受的可追蹤性策略？

有許多可追蹤性策略可用來促進需求管理流程；即使運作於 Rational Unified Process 的架構內，仍有許多可能的方法。作者看過已在使用中的所有最常見方法的其中四個如下：

- 僅使用案例模型。

  在此情況下，使用案例模型是系統需求的唯一陳述。選擇這種方法的專案，其特色爲客戶和開發人員之間密切關聯與信任。

  使用案例模型、名詞解釋和增補規格共同形成系統需求的整個陳述 –沒有其他「需求」、「產品特性」或軟體需求的定義存在。

- 特性驅動使用案例模型

  這是 Rational Unified Process 建議的預設策略。使用案例模型和增補規格形成完整的軟體需求規格。特性記載在「願景文件」中，並追蹤到使用案例。如果它們沒有反映在使用案例模型中，則會追蹤到「增補規格」中的增補需求。

  在此情況下，使用案例模型成爲功能需求的主要陳述。除了名詞解釋和增補需求之外，「使用案例模型」和「增補需求」將以需求和產品特性來補充。

- 使用案例模型是「軟體需求規格」的解譯

  在此情況下，使用案例模型是正式傳統「軟體需求規格」的解譯。當正式傳統「軟體需求規格」因爲規定或內部通訊協定而成爲必要時，或需要有使用案例模型才能啓用使用案例驅動開發的作法時，這是最常用的解譯。

  「特性」追蹤到正式「軟體需求規格」文件（如傳統需求管理），但軟體需求則明確地追蹤到「使用案例模型」。

- 使用案例模型使許多組傳統軟體需求一致。

  「使用案例模型」是來自多個來源的一組正式「軟體需求規格」的解譯，並提供單一一般系統的規格。

  在此情況下，關係人有他們自己的一組「產品特性」和「軟體需求」，並在他們擁有的「願景和軟體需求規格」文件中加以詳述。這些多重觀點和可能的衝突需求，將在指定系統執行動作的單一「使用案例模型」內取得一致。在處理大量獨立關係人時，此策略非常有效。

除了選項 1 之外，在所有案例中，我們結合自己的使用案例模型和傳統需求可追蹤性流程的元素。

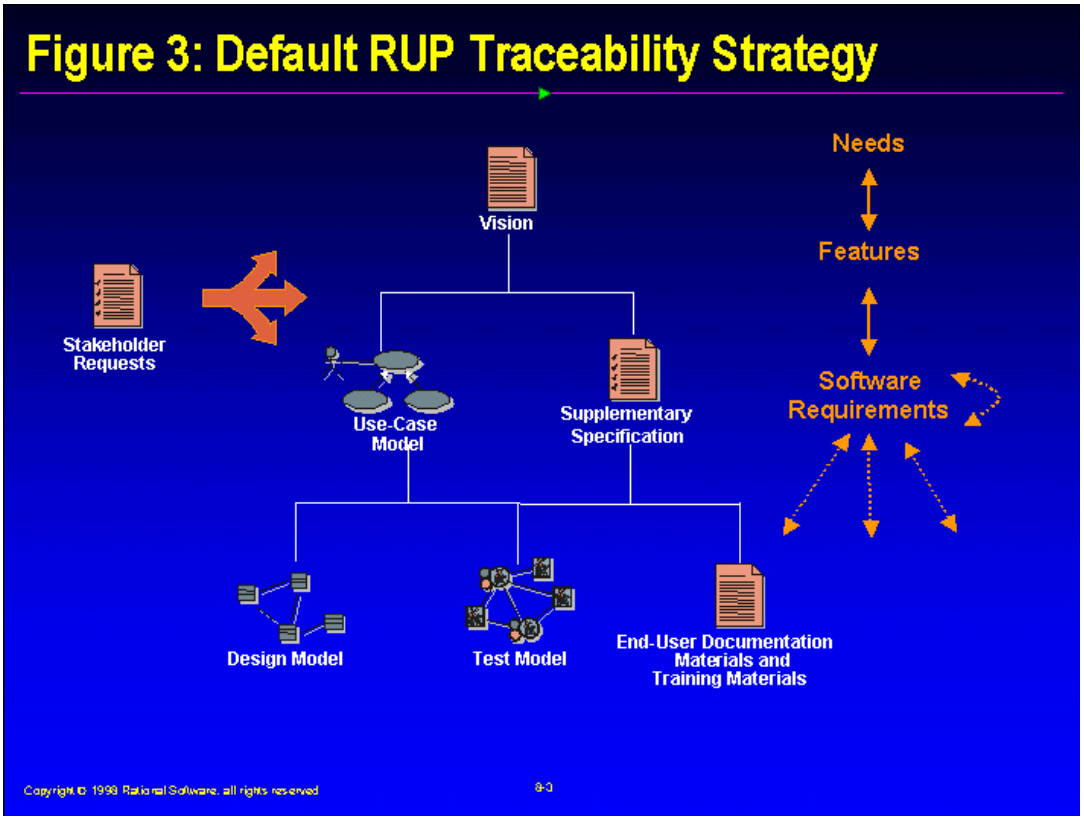
當然，還有許多其他可能的選項，其中之一就是完全不要有使用案例模型。我們會把這個選項稱爲「無使用案例模型」。

這些方法和其他方法將在以下「可追蹤性策略目錄」中詳細討論。

爲何我們要採用其中一個混合式方法？

正如我們在上面看到的兩個極端解決方案（「僅使用案例模型」和「無使用案例模型」），它們採取非常純粹的視圖，只容許一種需求擷取形式。兩者都假設一體適用方法適用所有專案和所有關係人社群。兩個方法的好壞在伯仲之間，但仍有所爭議，因爲它們缺乏靈活性而無法處理所有複雜情況，還有「真實」專案所引發的關係人關係。

Rational Unified Process 建議下列可追蹤性階層。



這是上述的「特性驅動使用案例模型」選項，可能是最有效的可追蹤性策略，但值得注意的是，即使已採用 Rational Unified Process，此方法也不一定是最有效的。

證明利用使用案例建模作爲功能軟體需求規格之唯一機制會造成問題的範例包括：

- 有許多對立需求來源（亦即，需要追蹤和管理許多對立需求）。
- 在主張依循現有的傳統需求擷取流程的組織內進行專案。
- 無法讓關係人參與建模研討會來產生單一共識需求模型。
- 在現有的需求擷取流程的限制下以使用案例建模來啓用物件導向軟體開發。
- 關係人社群無法或不願意直接在使用案例模型內表達其所有功能軟體需求層次的意願。
- 客戶已定義要以一組傳統軟體需求的形式交付的產品。當開發案正式提出時這是極爲常見的狀況 – 傳統需求陳述即變成交付合約的一部分。

我們認爲，要採取何種方法的決策必須在每一個專案和開發組織的環境定義內提出。此問題並無一體適用的解決方案，試圖強制所有專案實施單一方法之需求管理是不明智的作法。

請記住，Rational Unified Process 是一種可設定的流程，除了「無使用案例模型」方法之外，可對付本文件提到的所有可追蹤性策略（Rational Unified Process 的使用案例驅動本質杜絕本選項的採用）。關於採用哪一個方法的決策是在 Rational Unified Process 開發案例正式作業期間完成的決策之一。

關於可追蹤性策略目錄

若要能夠定義可追蹤性策略，我們需要一種機制來分類及定義可追蹤性項目：

定義：可追蹤性類型

可追蹤性時間的分類（例如需求、產品特性、使用案例、軟體需求、測試需求、參與者、名詞解釋等等），以一般性質和屬性爲基礎。

附註：在 RequisitePro 中，可追蹤性類型將以「需求類型」表示。

因此，設定可追蹤性策略涉及三個密切相關的活動：

- 識別可追蹤性類型的設定，這是要定義可追蹤性項目所需要的。
- 識別這些可追蹤性類型之間的有效可追蹤性關係。
- 識別可追蹤性項目所需的屬性，以啓用專案之有效需求管理。

「可追蹤性策略目錄」記載已知可追蹤性項目集合及其可追蹤性關係，以幫助這些步驟的前兩個步驟。它不涵蓋第三個活動，因爲可追蹤性類型的適當屬性的定義目前是在本白皮書的範圍之外。

目錄中所描述的可追蹤性策略全都使用同一組基本可追蹤性類型的子集。

- 需求
- 產品特性
- 軟體需求（功能和非功能）
- 名詞解釋
- 使用案例
- 使用案例區段
- 參與者

附註：在使用案例建模時，唯一軟體需求通常是指增補規格所定義的增補需求。

在所有傳統可追蹤性類型和使用案例模型的元件部分之間追蹤的可能性，開放了專案可用的可追蹤性策略數。

可追蹤性項目之間有兩個可追蹤性層次：

- 基本可追蹤性

  任選一個可追蹤性策略時有適用的基本可追蹤性。此基本可追蹤性隱含在可追蹤性類型的本質中。這包括像使用案例與使用案例區段之間或使用案例和參與者之間的關係。

  在閱讀下面的「可追蹤性策略區段」的總覽圖時，不針對每一個策略重複此基本可追蹤性，依預設，會是將它併入適用的可追蹤性策略中。

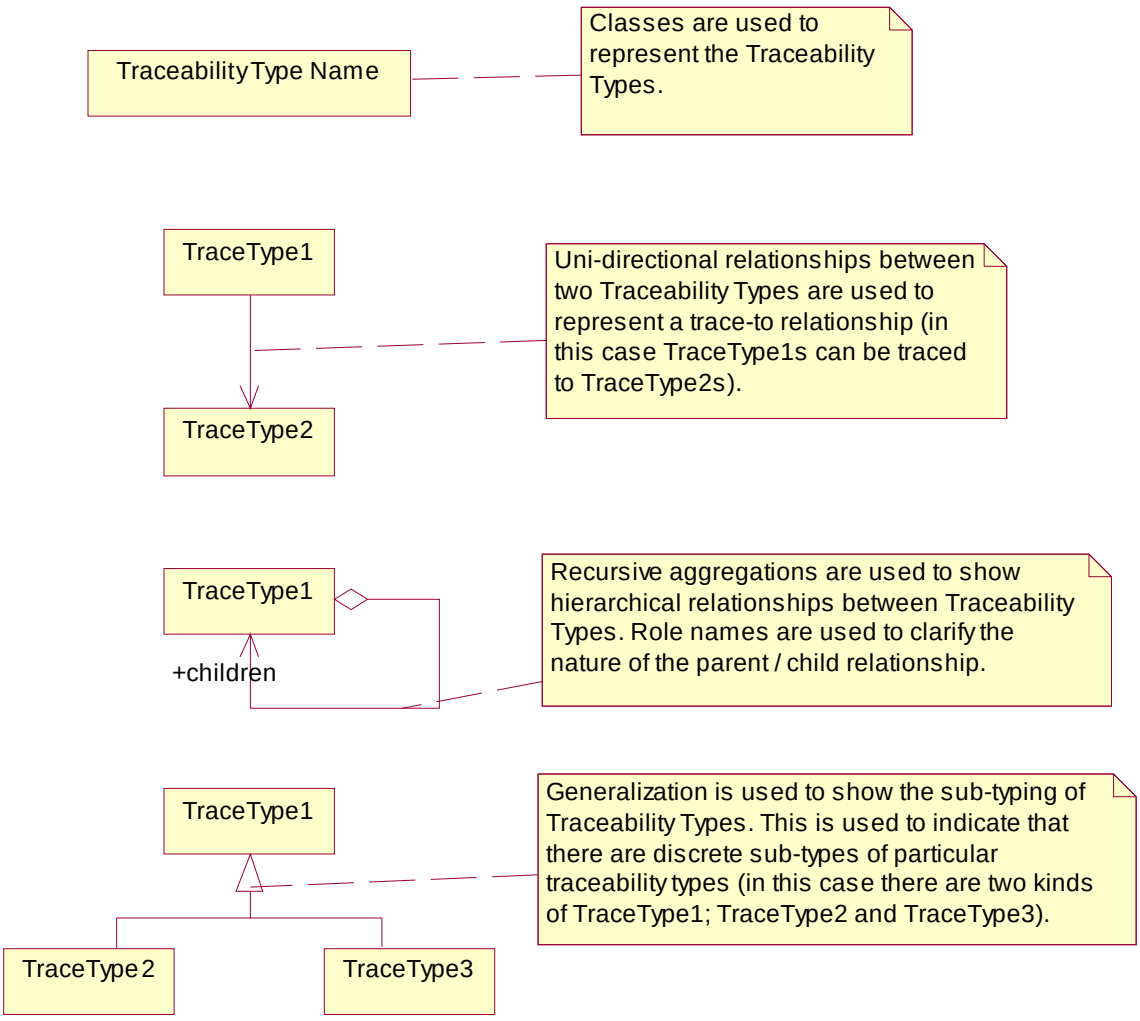
- 延伸的可追蹤性

  這是爲了支援其中一個特定可追蹤性策略而引進的可追蹤性。這種可追蹤性比較主觀，在不同的可追蹤性策略之間並不相同。

可追蹤性策略目錄

圖解

「可追蹤性類型」及其可追蹤性關係以「統一建模語言（Unified Modeling Language, UML)」圖表顯示。下圖顯示如何解譯 UML 在此環境定義的用法。


Traceability Type Name
Classes are used to represent the Traceability Types.
TraceType1
TraceType2
Uni-directional relationships between two Traceability Types are used to represent a trace-to relationship (in this case TraceType1s can be traced to TraceType2s).
TraceType1
+children
Recursive aggregations are used to show hierarchical relationships between Traceability Types. Role names are used to clarify the nature of the parent / child relationship.
TraceType1
TraceType2
TraceType3
Generalization is used to show the sub-typing of Traceability Types. This is used to indicate that there are discrete sub-types of particular traceability types (in this case there are two kinds of TraceType1; TraceType2 and TraceType3).

若要完全瞭解圖表，最好知道在 RequisitePro 中實作定義時使用的實作對映表。下表解釋圖解要如何對映到 RequisitePro 專案。

圖解	RequisitePro 對映表
類別 / 可追蹤性類型	需求類型
關係	RequisitePro "trace-to" 關係
聚集	階層式需求
一般化	新增另一個「子分類」屬性的超需求類型的分類。

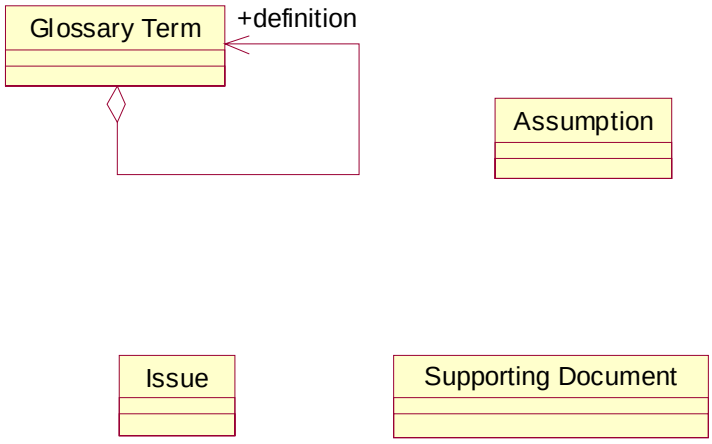
附註：RequisitePro 允許以任何可追蹤性項目追蹤到其他項目。可追蹤性策略定義的是有意義的可追蹤性鏈結，這些將成爲專案需求管理策略的核心。

支援的可追蹤性類型

說明

在此章節中，我們定義一組支援的可追蹤性類型，可用來支援任何已選取的可追蹤性策略。

概觀



可追蹤性類型

可追蹤性類型	說明
名詞解釋	這種可追蹤性類型定義代表名詞解釋及其定義的可追蹤性項目。 這已併入支援的可追蹤性類型集合中，因爲不論您選擇採用何種可追蹤性策略，都需要名詞解釋。
問題	這種可追蹤性類型可讓您新增可追蹤性項目來代表您要在 RequisitePro 內追蹤的問題。於是，這些問題便可與任何受它們影響的可追蹤性項目相關聯。 追蹤與「名詞解釋項目」相關聯的問題，就是使用「問題」可追蹤性類型的一個範例。如果定義不確定或有爭議，就會引發問題並包括在 RequisitePro 中。這可確保問題不被遺忘，並允許建置視圖來報告有爭議問題的所有名詞解釋項目。這種可追蹤性類型的另一個好處，是追蹤在檢閱使用案例及其他開發構件時引發的問題。
假設	這種可追蹤性類型可讓您追蹤您所提出的假設。於是，這些假設便可與任何受它們影響的可追蹤性項目相關聯。
支援的文件	這種可追蹤性類型可讓您在可追蹤性階層中新增您要的任何文件。這對於併入預存的範例或文件來釐清另一個可追蹤性項目的意義或用途特別有用。 RequisitePro 的彈性可追蹤性機制可讓您使支援的文件與任何類型的任何可追蹤性項目相關聯。 使用「支援的文件」類型的一個範例：併入詳細 EDI 訊息規格作爲名詞解釋的支援資訊，或作爲使用訊息的使用案例的附錄。

基本可追蹤性

可追蹤性鏈結	說明
名詞解釋到名詞解釋	此關係可讓我們使用單一可追蹤性類型來擷取名詞解釋的名稱及其定義。
支援的可追蹤性類型對任何其他可追蹤性類型	這些支援的可追蹤性類型可追蹤到所選擇的可追蹤性策略中的任何其他可追蹤性類型。

無使用案例模型

說明

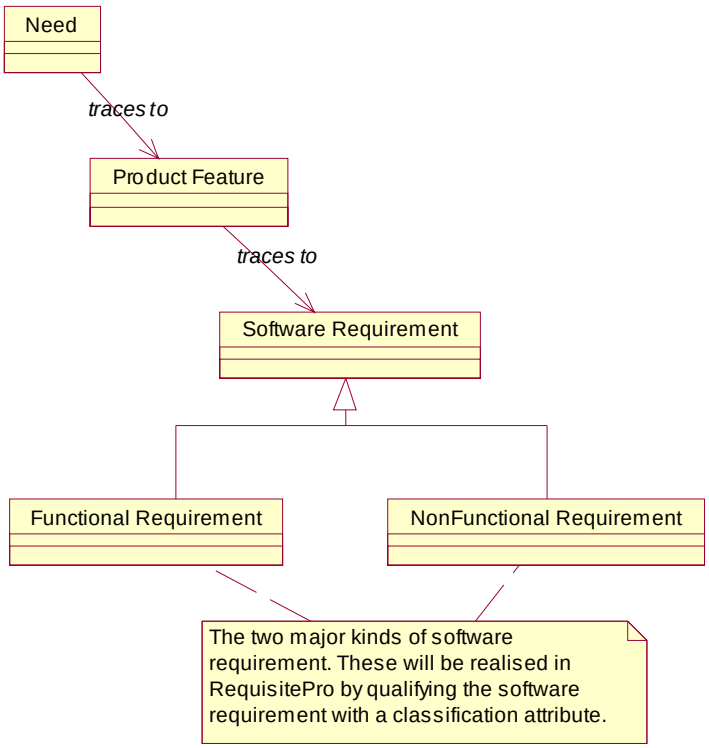
在此情況下，沒有使用案例模型。「需求」引起「產品特性」，「產品特性」再引起「軟體需求規格」所記載的「軟體需求」。

專案經理最常說的一句話：「我不需要令人討厭的使用案例模型！」

性質

性質	值	註解
明確可追蹤性	高	實作需求管理技術而不利用使用案例的專案，通常在可追蹤性類型之間維護高層次的明確可追蹤性。
信任	低	
責任	高	
形式	高	
完整性	低	要評定一組傳統軟體需求的完整性非常困難。
文件集	大	文件集通常是以英尺而不是英寸來衡量。
焦點	合約	需求擷取流程的焦點是要在客戶和開發人員之間建立可合法實施的合約，而不是對於要解決的問題和提出的解決方案建立共識。
可懂性	低	使用者群組和開發人員通常無法存取需求文件。它通常由許多個別行項目組成，按類型或功能範圍分組，但對檢閱人員提供的環境定義很少。
流程	典型瀑布式	傳統需求擷取技術通常以瀑布式開發流程的一部分來實作。任何需求子集的環境定義之缺乏以及評定完整性的困難度，都無法促進反覆及漸進式開發流程的採用。
開發樣式	功能分解	當需求轉換成解決方案時，按類型或功能範圍將需求分組，將傾向於導致持續功能分解。

可追蹤性概觀



可追蹤性類型

需求類型	說明
需求	必須達到才能證明採購或使用的商業問題或作業問題（機會）。亦稱爲目標或目的。
產品特性	直接滿足需求的系統的功能或性質。通常被認爲是系統的「廣告利益」。
軟體需求	所建置的軟體必須符合的條件或功能。

可追蹤性*摘要*

可追蹤性鏈結	說明
需求追蹤產品特性	每一個需求由一組特性實現。此關係允許追蹤每一個特性的商業利益。
產品特性追蹤軟體需求。	每一個特性由一組軟體需求實現。 此關係允許追蹤每一個「軟體需求」的商業利益，並在「產品特性」層次啓用「軟體需求」的範圍管理。

優點和缺點

優點：

- 非常瞭解
- 被認爲適合合法合約（請參閱許多與上市軟體能否滿足指定需求的進行中法院案例）。
- 許多標準流程建議。
- 啓用詳細、低層次、正式可追蹤性。
- 採用「該死的新奇」點子而不推翻現狀

缺點：

- 很難完成需求擷取 - 在需求階段很容易陷入同一種模式。
- 很難瞭解以此形式表達的需求。
- 很難評定需求變更的影響分析。
- 個別需求沒有環境定義。
- 高維護成本。缺乏隱含的可追蹤性，會爲專案留下維護大量明確可追蹤性關係的成本。
- 缺乏環境定義很難識別有意義的需求子集。這使得產品的範圍管理和漸進式交付更加雪上加霜。

範例

在許多商業領域的許多專案中，普遍使用需求可追蹤性的「無使用案例模型」方法。許多組織需要正式傳統「軟體需求規格」作爲正式合法談判的基礎。這使它們認爲傳統需求管理方法是唯一適合其專案的方法。

僅使用案例模型

說明

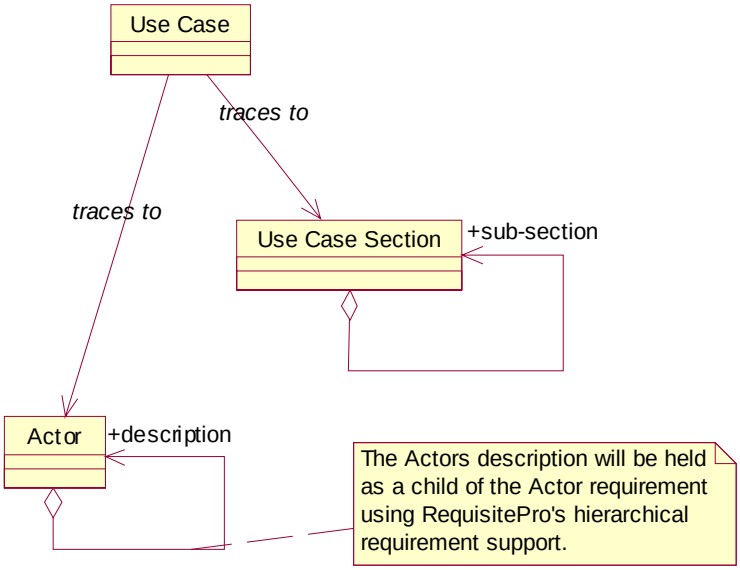
「使用案例模型就是我的需求」。客戶和開發人員之間的密切關聯和高層次信任通常是採用此方法的專案的特色。它通常用於內部低責任性專案，其開發人員與客戶一起（或經客戶核准）開發使用案例模型，希望示範或取得對需求的明確瞭解。

在此情況下，使用案例模型是系統需求的唯一陳述。使用案例模型、名詞解釋和增補規格共同形成系統需求的整個陳述。

性質

性質	值	註解
明確可追蹤性	低	不需要明確可追蹤性。採用使用案例驅動方法而提供的隱含可追蹤性已經足夠。可能不維護明確可追蹤性，因此不使用需求管理工具。
信任	高	缺乏任何需求或特性層次分析，表示共同開發人員給予使用案例模型的開發人員高層次的信任來交付適當的系統。
責任	低	
形式	低	
完整性	低	雖然使用案例模型本身可幫助建立軟體需求規格的完整性，但缺乏對關係人的可追蹤性通常會導致產生不完整或太過複雜的系統。
文件集	小	這種方法涉及最小文件集。
焦點	使用者	使用案例模型有使用者視景。
可懂性	高	使用案例模型容易讓專案的所有關係人瞭解。
流程	通常爲反覆和漸進式	使用案例將軟體需求放入環境定義中，促進反覆和漸進式開發（使用案例提供理想的交付單元）。使用案例也可以與瀑布式開發流程一起使用。
開發樣式	通常爲物件導向	雖然使用案例可適用於任何開發樣式，但通常以使用案例來驅動物件導向軟體開發。如果不採用物件導向樣式，則需要高層次的明確可追蹤性。

可追蹤性概觀



可追蹤性類型

可追蹤性類型	說明
使用案例	這種可追蹤性類型定義代表使用案例的可追蹤性項目。
使用案例區段	「使用案例區段」可讓您將「使用案例」的區段併入可追蹤性階層中。 這可讓我們追蹤個別流程和構成使用案例的其他內容。 與子區段之間階層式關係的存在可讓我們擷取每一個區段的個別片段。例如，這可讓我們識別構成先決條件區段的個別先決條件。 附註：在某些情況下，適合在使用案例的事件流程中識別個別軟體需求（在此情況下爲很小的區段），但要等到使用案例本身穩定時才適合這麼做。
參與者	這種可追蹤性類型定義代表參與者的可追蹤性項目。

可追蹤性摘要

可追蹤性鏈結	說明
使用案例對使用案例區段	每一個使用案例是由一組使用案例區段所組成。此一關係可讓我們追蹤哪些使用案例區段構成哪些使用案例。
使用案例區段對使用案例區段	有些較複雜的使用案例區段是由許多子區段組成。例如，事件流程可能由許多子流程組成，先決條件區段可能由許多先決條件組成。
使用案例對參與者	此一關係可讓我們瞭解哪些參與者涉及哪些使用案例。
參與者對參與者	此關係可讓我們使用單一可追蹤性類型來擷取參與者的名稱及其簡要說明。

優點和缺點

優點：

- 最小文件集
- 涉及需求管理的最少人力
- 對範圍管理、影響分析和漸進式開發的良好支援。
- 使用案例容易瞭解。

缺點：

- 與關係人需求沒有關係。在開始進行解決方案的定義之前，不實際嘗試分析問題。
- 有些人認爲很難接受一份只以使用案例模型爲依據的合約。
- 若沒有採取任何需求分析，就很難知道使用案例模型本身何時描述適合的解決方案。撰寫使用案例時，可天馬行空。
- 如果執行一般版本，除了使用案例本身以外沒有較高層次的任何資訊，則很難進行產品管理及符合關係人期望。

範例

這種方法通常用於小規模的非正式內部專案，開發人員和使用者合作非常密切。

使用案例模型定義產品特性

說明

在此情況下，使用案例建模將作爲主要需求誘導方法，而使用案例模型將成爲由系統提供的「產品特性」的定義以及軟體需求的陳述。

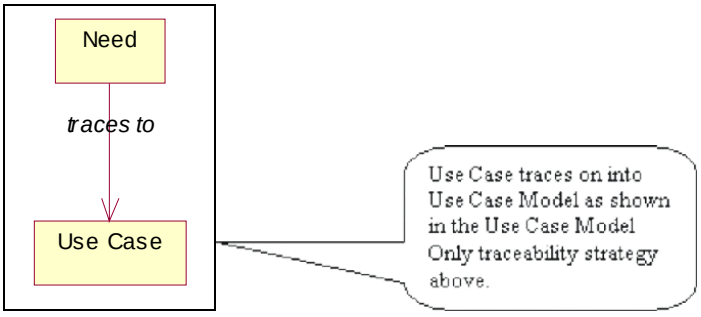
這個選項只適用於生命週期很短的小規模開發，因爲它不做調整。即使每一個使用案例代表一個系統特性，特性數量也會比使用案例多，實際上，許多特性可能影響許多使用案例。隨著系統發展，每一版的新特性越來越不可能採取新使用案例的形式。

性質

這是先前的「僅使用案例模型」方法的一種變體。在討論此方法時，我們只注意到一些差異。

性質	值	註解
明確可追蹤性	低	僅使用案例模型。只有一點點需求，而這少量的額外可追蹤性仍會導致少量必要的明確可追蹤性。
信任	中 / 高	將「需求」新增到「使用案例模型」會變成信任度小於「僅使用案例模型」的策略。
責任	低	
形式	低	
完整性	中	使用案例模型本身可幫助建立軟體需求規格的完整性，對共同開發人員的進一步可追蹤性則需要幫助才能驗證使用案例模型的適用性。
文件集	小	這種方法涉及最小文件集。使用案例模型加上包含「需求」的願景文件。
焦點	使用者	使用案例模型和需求兩者都有使用者視景。
可懂性	高	需求和使用案例模型兩者都容易讓專案的所有關係人瞭解。
流程	通常爲反覆和漸進式	僅作爲使用案例模型
開發樣式	通常爲物件導向	僅作爲使用案例模型

可追蹤性概觀



可追蹤性類型

可追蹤性類型	說明

需求	定義給「無使用案例模型」
使用案例	定義給「僅使用案例模型」

可追蹤性摘要

可追蹤性鏈結	說明
需求到使用案例	在此情況下，「需求」直接追蹤到使用案例。假設在產品和範圍的管理上，使用案例可扮演產品特性的角色。

優點和缺點

這種方法十分類似「僅使用案例模型」策略。其優點和缺點相同，但與下列新增項目和警示。

優點：

- 在此情況下，使用案例模型與關係人需求相關，後者有助於評定使用案例模型的適合性。

缺點：

- 使用案例雖然在專案初期階段定義系統特性，但這兩個概念會隨著專案的成長而背離。
- 使用案例不是特性—看起來省時省力的策略很快變成一團糟。

範例

雖然在小型內部專案上有嘗試使用可追蹤性策略，但因爲可調整性和長期產品評估問題，我們不建議使用這種方法。如果使用案例模型將增補關係人需求的可追蹤性，則建議採用「特性驅動使用案例模型」策略。

特性驅動使用案例模型

說明

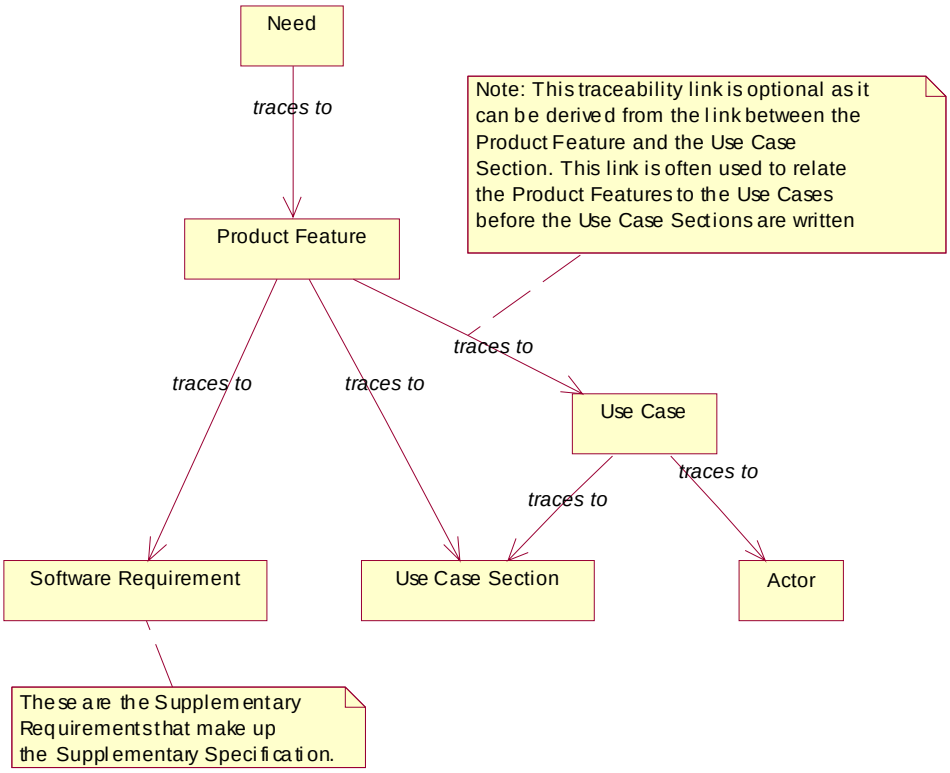
使用案例模型和增補規格形成我的 SRS。這是 RUP 概述及建議的策略。需求和產品特性記載在「願景文件」中，並追蹤到使用案例。如果它們沒有反映在使用案例模型中，則會追蹤到「增補規格」中的增補需求。

在此情況下，使用案例模型成爲軟體需求的主要陳述。這由增補規格加以補充，增補規格包含的軟體需求無法在使用案例本身輕易表達清楚。

種類

性質	值	註解
明確可追蹤性	中	除了使用案例模型的隱含可追蹤性之外，在此情況下，我們還必須明確維護需求、特性和使用案例模型之間的可追蹤性。
信任	中	
責任	高	
形式	中	新增「需求」和「產品特性」到「使用案例模型」，與只維護「使用案例模型」相比較，將產生更正式的需求管理流程。
完整性	高	在「軟體需求」上若具有「特性」和「使用案例」視景，將可在「軟體需求」的擷取和優先順序上達到高層次的完整性。
文件集	中	現在我們有一個「願景」文件，包含了需求和特性、使用案例模型和增補規格。
焦點	使用者、關係人和專案經理	需求和特性新增到使用案例模型之後，可拓寬需求活動的焦點，使它更主動圍繞產品經理和所有其他關係人以及使用者。特性是一個非常強大的工具，用來管理關係人期望，並對軟體需求的使用案例視景提供良好的補充。
可懂性	高	和使用案例模型並排靠攏的需求和特性的定義以及增補規格，提供一個容易讓專案所有關係人瞭解的需求模型。
流程	通常爲反覆和漸進式	僅作爲使用案例模型。
開發樣式	通常爲物件導向	僅作爲使用案例模型。

可追蹤性概觀



可追蹤性類型

可追蹤性類型	說明
需求	定義給「無使用案例模型」
產品特性	定義給「無使用案例模型」
使用案例	定義給「僅使用案例模型」
軟體需求	套用到整個系統或不太適合使用案例的任何軟體需求。軟體需求是要建置之軟體必須符合的條件或功能。
使用案例區段	定義給「僅使用案例模型」
參與者	定義給「僅使用案例模型」

可追蹤性摘要

可追蹤性鏈結	說明
需求到產品特性	定義給「無使用案例模型」
產品特性到使用案例	選擇性的可追蹤性鏈結。產品特性可直接追蹤到使用案例。這可在撰寫使用案例區段之前，讓產品特性指派到使用案例，並可讓您在產品特性層次執行使用案例模型的影響分析，反之亦然。
產品特性到使用案例區段*	產品特性追蹤到使用案例區段。這可讓使用案例模型以特性基礎管理範圍，並在比使用案例本身更適用的層次上促進特性集與使用案例模型之間的影響分析。 追蹤到使用案例的所有產品特性也必須追蹤到其中一個使用案例區段。
產品特性到軟體需求*	產品特性也追蹤到增補規格中的軟體需求。未追蹤到使用案例模型的所有產品特性必須追蹤到增補規格中的至少一個軟體需求。
使用案例對參與者	定義給「僅使用案例模型」。
使用案例對使用案例區段	定義給「僅使用案例模型」。

*每一個產品特性必須追蹤到至少一個使用案例區段或增補軟體需求。如果沒有，則軟體中不會包含它。

優點和缺點

這種方法將使用案例和傳統需求管理方法兩者所提供的優點最大化，同時將缺點最小化。

優點：

- 非常瞭解
- Rational Unified Process 建議。
- 啓用詳細、低層次、正式可追蹤性。
- 在軟體需求上同時具有產品特性和使用案例視景，有助於完成需求擷取 - 這使得需求誘導和擷取活動陷入同一種模式的機會降至最小。
- 軟體需求以更容易瞭解的形式表達。
- 這個可追蹤性策略促進需求變更的影響分析 - 可清楚瞭解不實作特性或使用案例區段的影響。
- 個別需求有使用案例和/或產品特性提供的環境定義。這樣可以輕易識別有意義的需求子集。這也使得產品的範圍管理和漸進式交付更加容易。
- 最小完整文件集。
- 使「需求管理」的相關工作減至最少。
- 此解決方案可靈活調整。如果執行一般版本，則同時在特性和使用案例層次上進行範圍管理的能力，可讓所有關係人在他們認爲適合的詳細層次上追蹤專案進度。

- 在此情況下，透過幫助所有關係人評定使用案例模型適合性的產品特性，使用案例模型是與關係人需求相關的。

缺點：

- 並非所有組織都接受。
- 雖然有許多組織已成功達到目的，但有些人認爲要撰寫一份主要以使用案例模型表達的軟體需求爲依據的合約，並不容易。

範例

這種方法適合接受使用案例作爲表達大部分軟體需求的適當格式的所有專案。

使用案例模型是「軟體需求規格」的解譯

說明

使用案例模型是正式 SRS 的解譯。若因爲規定或內部通訊協定而需要正式 SRS 時，最常使用這種解譯。

在外包或承擔固定價格開發案時，傳統 SRS 通常被視爲達成合約協定的重要一環。這導致兩個典型狀況：

客戶提供傳統 SRS 給開發組織，作爲其系統開發的起跑點。

在專案生命週期初期，SRS 文件是強制的或規定的交付項。每一個專案必須有傳統正式 SRS 文件，以和所有其他專案一樣的方式來表達系統需求。

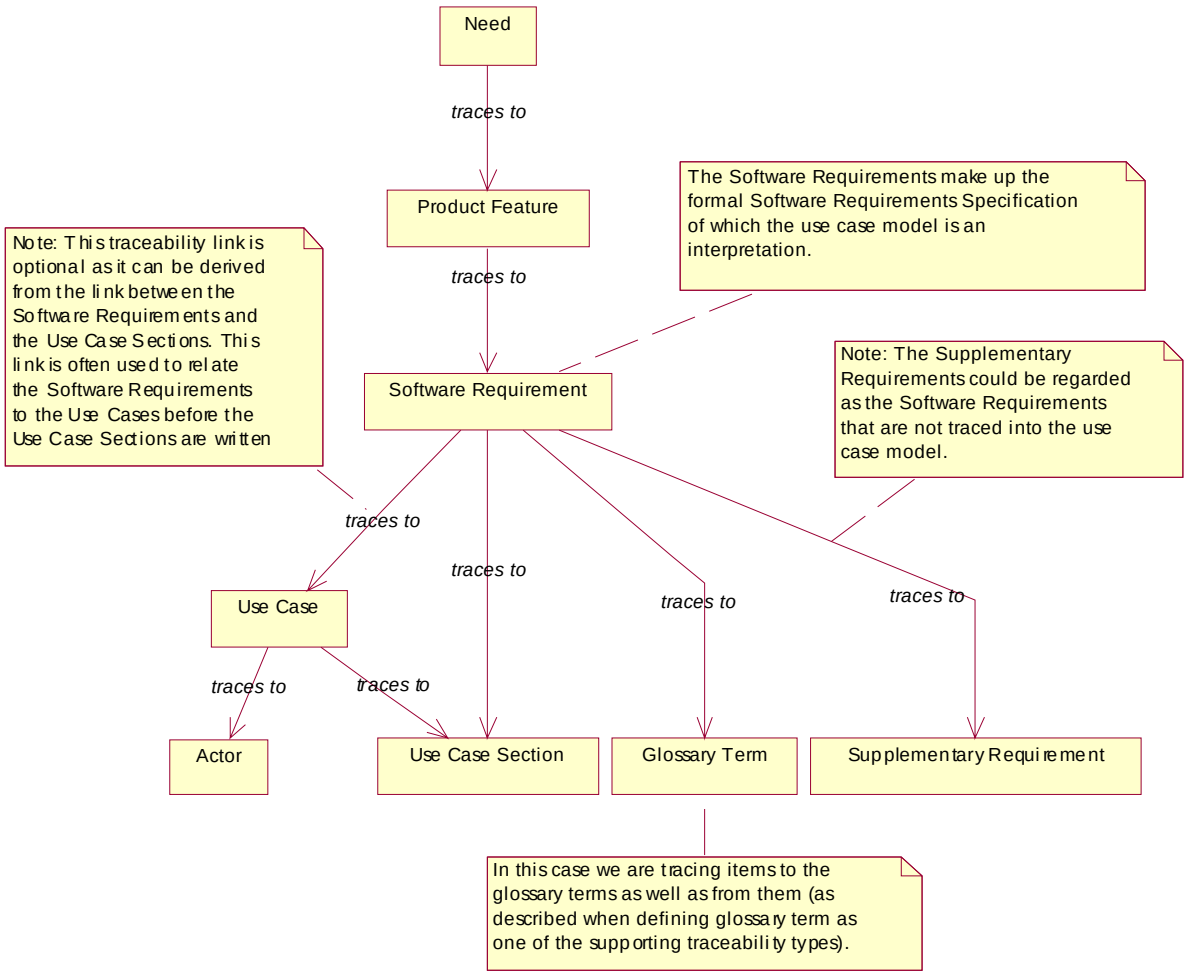
在這些情況下，會利用使用案例模型來建模，並在專案範圍內重新解譯所有軟體需求。採用這種方法時，通常以 SRS 爲優先 - 還有其他技術可用來呈現使用案例模型所保留的資訊，其格式類似正式傳統 SRS（尤其是採用「特性驅動使用案例模型」方法時），而且不必建立第二個「軟體需求」定義。

附註：採用這種方法時，傳統軟體需求的集合不一定是必要功能的完整陳述—使用案例模型將提供或確保功能規格的完整性。傳統軟體需求可以只用來擷取直接識別的或關係人引發的軟體需求。

性質

性質	值	註解
明確可追蹤性	非常高	「無使用案例模型」方法需要的所有明確可追蹤性是維護正式 SRS 所必需的，而傳統軟體需求若追蹤到使用案例模型會有額外負荷。
信任	非常低	這是一種非常「提心吊膽」的方法，暗示信任度非常低。
責任	高	
形式	非常高	同樣地，這是非常正式的方法，同時並列套用兩個需求管理方法。
完整性	非常高	以使用案例模型補充傳統軟體需求規格而成爲非常高完整性的方法。附註：在此情況下，已假設這是使用案例模型，可確保系統功能的完整規格。軟體需求規格的集合不一定要在相同完整性層次。
文件集	非常大	在此狀況下，我們基本上指定系統兩次。
焦點	合約	會採用這種方法，其驅動力是因爲需要履行現有合約，合約以傳統 SRS 表達，或與現有的開發方法一致，需要以 SRS 作爲開發人員和客戶之間的合約。
可懂性	中	製作兩個軟體需求定義一開始或許令人混淆，但以使用案例模型作爲主要軟體需求定義，應可產生一目了然的系統需求陳述。
流程	開啓	雖然有足夠資源可支援幾乎任何開發流程，不過爲了使用反覆和漸進式技術，使用案例模型常常會新增到傳統 SRS。
開發樣式	開啓	雖然有足夠資源可支援幾乎任何開發樣式，不過爲了使用物件導向技術，使用案例模型常常會新增到傳統 SRS。

可追蹤性概觀



可追蹤性類型

可追蹤性類型	說明
需求	定義給「無使用案例模型」
產品特性	定義給「無使用案例模型」
軟體需求	定義給「無使用案例模型」
使用案例	定義給「僅使用案例模型」
參與者	定義給「僅使用案例模型」
使用案例區段	定義給「僅使用案例模型」
名詞解釋	定義給「僅使用案例模型」
增補需求	套用到整個系統或不太適合使用案例的任何軟體需求。這些可能不需要從原始軟體需求集合重新陳述，但有可能屬於未追蹤到使

	用案例模型的範圍軟體需求。

可追蹤性摘要

可追蹤性鏈結	說明
需求到產品特性	定義給「無使用案例模型」
產品特性到軟體需求	定義給「無使用案例模型」
軟體需求到使用案例	「功能軟體需求」追蹤到使用案例。 非功能軟體需求的子集也追蹤到使用案例。 此關係可在使用案例的需求和商業利益方面進行使用案例模型的範圍限定和評定。 附註：範圍功能需求的全部**必須**追蹤到使用案例或名詞解釋。如果沒有以此方式反映它們，則不會實作它們。
軟體需求到使用案例區段	追蹤到使用案例的軟體需求也必須追蹤到其中一個使用案例的使用案例區段。 關於使用案例的需求方面，此關係可對使用案例的使用案例區段進行驗證。追蹤到使用案例的所有軟體需求必須以使用案例內的其中一個區段來滿足。雙重可追蹤性可讓我們對此進行驗證，並在考慮使用案例區段是否爲必要之前，先配置軟體需求到使用案例本身。 有些區段可能沒有相符的軟體需求。 附註：追蹤到使用案例的所有功能需求也**必須**追蹤到其中一個使用案例區段。如果沒有以此方式反映它們，則不會實作它們。
軟體需求到名詞解釋	功能和非功能需求可追蹤到名詞解釋中的項目。這對於識別系統所涉及之實體的屬性和關係的「靜態需求」，尤其如此。如果名詞解釋是從軟體需求追蹤，則名詞解釋必須使用於其中一個使用案例，否則它可能無法運用到設計模型中。
使用案例對參與者	定義給「僅使用案例模型」
使用案例對使用案例區段	定義給「僅使用案例模型」
軟體需求到增補需求	增補需求可重新陳述套用到整個系統或不太適合使用案例模型的軟體需求。替代方法是將範圍軟體需求中未追蹤到使用案例模型的一切都視爲增補需求—這樣可避免重新陳述它們。

優點和缺點

老實說，這種方法過份隆重了一點。實際上，這只適合已有傳統 SRS 但想要以使用案例建模來瞭解所提供的需求及協助使用案例驅動方法的專案。

優點：

- 啓用非常詳細、低層次、正式可追蹤性。
- 軟體需求以更容易瞭解的形式表達。
- 這個可追蹤性策略促進需求變更的影響分析 - 可清楚瞭解不實作特性、軟體需求或使用案例區段的影響。
- 個別需求有使用案例和/或產品特性提供的環境定義。使用案例模型的呈現可輕易識別有意義的需求子集。這也使得產品的範圍管理和漸進式交付更加容易。
- 在此情況下，透過軟體需求和幫助所有關係人評定使用案例模型適合性的產品特性，使用案例模型實際上是與關係人需求相關的。
- 大部分組織可接受（有警示）- 這種方法滿足所有人的願望。這種方法通常以並列需求流程的形式使用於初始使用案例專案（亦即，專案同時以新舊兩種方式執行），或可採取此方法來隱藏開發人員利用使用案例的事實。
- 第一次採用或實驗使用案例時，這是使組織分裂減至最輕的理想方式。外界看到的還是傳統 SRS，所以可以使用標準程序和合約。

缺點：

- 不容易瞭解 - 同時有傳統需求陳述和使用案例模型，會令人混淆。
- 同時有傳統軟體需求規格和使用案例模型，讓您變成雙邊從事需求活動。這樣很容易搞不清楚哪一個才是完整的軟體需求規格
- 您必須維護一個很大的文件集。
- 有許多重複發生，使需求管理流程複雜化。由於使用案例是直接隨需求變更而更新，所以傳統「軟體需求」可能廢而不用。
- 這是一個非常高成本高維護性方法。

範例

這種方法對於利用使用案例驅動開發技術的開發公司很有幫助，這些公司是以傳統軟體需求規格作爲其合約的一部分。使用案例的引進讓開發公司能夠證明他們對需求的瞭解，並以反覆及漸進式方式交付軟體。

將使用案例技術引進到使用傳統需求擷取技術並抵制變更爲使用案例驅動方法的公司時，它也是一個非常有用的策略。在此情況下，其意圖是要讓使用案例證明它們對於開發組織的價值，並隨著對使用案例越來越有信心，而讓傳統軟體需求規格逐步淘汰。這是邁向「特性驅動使用案例模型」方法的第一步。

使用案例模型使許多組傳統軟體需求一致

說明

「使用案例模型是來自多個來源的正式 SRS 的解譯，並提供單一一般系統的規格。」

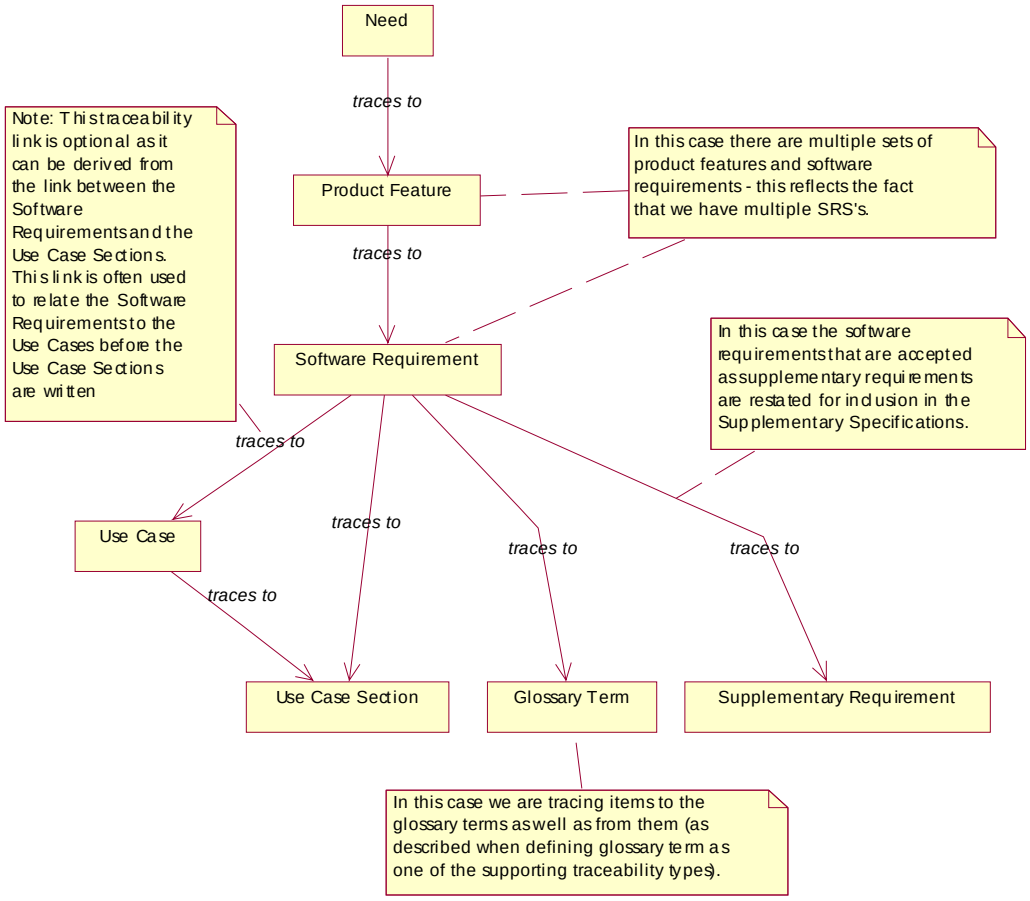
這是「使用案例模型是軟體需求規格的解譯」的一個變體，只不過，在此情況下，有多個獨立的關係人集合提供多個傳統 SRS。開發單一應用程式來滿足許多不同獨立未連線客戶的需求的軟體公司，通常會發生此狀況。在此情況下，使用案例模型是開發人員對系統需求的合併視圖，而個別 SRS 是個別關係人對其自己的需求的視圖（不整合或反映其他關係人的需求）。在許多個別需求集合和使用案例模型之間的追蹤，可讓開發人員評定它們在評定不同關係人需求上的表現。

性質

此策略是先前的「使用案例模型是軟體需求規格的解譯」方法的一個變體。在討論此方法時，我們只注意到一些差異。

性質	值	註解
明確可追蹤性	非常高	使用案例模型是「軟體需求規格」的解譯
信任	非常低	使用案例模型是「軟體需求規格」的解譯
責任	非常高	在此情況下，我們在其自己的 SRS 中保留所有個別客戶視景。
形式	非常高	使用案例模型是「軟體需求規格」的解譯
完整性	非常高	使用案例模型是「軟體需求規格」的解譯
文件集	非常大	在此情況下，我們有多個規格的系統，由單一使用案例模型使其一致。
焦點	管理多個獨立的客戶。	這種方法的焦點是放在多個獨立的、可能對立的需求來源的管理上，這些來源在政治上、地理上或組織上無法密切合作。
可懂性	中	使用案例模型是「軟體需求規格」的解譯
流程	通常爲反覆和漸進式。	在此情況下，開發人員利用使用案例模型作爲其 SRS。請參閱「僅使用案例模型」
開發樣式	通常爲物件導向	在此情況下，開發人員利用使用案例模型作爲其 SRS 來驅動軟體的開發。請參閱「僅使用案例模型」

可追蹤性概觀



需求類型

需求類型	說明
需求	定義給「無使用案例模型」
產品特性	定義給「無使用案例模型」
軟體需求	定義給「無使用案例模型」
使用案例	定義給「僅使用案例模型」
使用案例區段	定義給「僅使用案例模型」
名詞解釋	定義給「僅使用案例模型」
增補需求	套用到整個系統或不太適合使用案例的任何軟體需求。

可追蹤性摘要

可追蹤性鏈結	說明
需求到產品特性	定義給「無使用案例模型」
產品特性到軟體需求	定義給「無使用案例模型」
軟體需求到使用案例	使用案例模型是 SRS 的解譯
軟體需求到使用案例區段	使用案例模型是 SRS 的解譯
軟體需求到名詞解釋	使用案例模型是 SRS 的解譯
軟體需求到增補規格	在此情況下，必須在支援使用案例模型的增補規格文件中重新陳述軟體需求，以便開發人員從多個個別關係人的 SRS 中汲取開放資訊，來製作單一、一致、有競爭力的 SRS。

優點和缺點

此策略是先前的「使用案例模型是軟體需求規格的解譯」方法的一個變體，具有差不多的優點和缺點。

與其他策略不同，這種方法的好處是它以其自己的正式個別 SRS 形式來處理及保留獨立關係人觀點的能力。

它還有一個缺點，即產生更大的文件集，使維護和追蹤更難。

範例

英國的軟體公司要開發一個支援保險經紀人的系統，讓保險公司能夠以電子方式分送新產品。

此專案是在有 22 個關係人的區域中，其中大約有三分之二是經紀業務公司，三分之一是保險公司。在這些關係人之間，需求大大不同；在許多案例中，保險公司與經紀人之間有完全對立的需求。

在此情況下，已決定要產生軟體需求規格，讓每一個關係人的公司能夠詳述其特定需求及輕易維護其個別視景。使用案例模型是用來呈現系統的合併願景給所有關係人。從原始 SRS 到使用案例模型的可追蹤性讓關係人能夠真正看到系統符合哪一個需求，及驗證系統是否適合其需求。它也可以讓軟體公司針對每一個關係人履行所有關係人需求的 80% 的目標追蹤其進度。

Rational
the software development company

兩個總公司：

Rational Software
18880 Homestead Road
Cupertino, CA 95014
電話：(408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
電話：(781) 676-2400

免付費專線：(800) 728-1212
電子郵件：info@rational.com
網址：www.rational.com
國際辦事處：www.rational.com/worldwide