

謝禎錕、陳仲儼(2018)，『運用限制理論為基礎的敏捷發展轉型歷程模式』，
中華民國資訊管理學報，第二十五卷，第一期，頁 23-53。

運用限制理論為基礎的敏捷發展轉型歷程模式

謝禎錕

精誠軟體服務股份有限公司

陳仲儼*

國立中央大學資訊管理學系

摘 要

近年來由於商務環境的資訊化與快速變遷，使得敏捷軟體開發逐漸受到重視，許多軟體組織也開始從原有的開發方式轉換成敏捷模式，這樣的轉換過程稱為『敏捷轉型過程 (agile transition process; ATP)』。對於敏捷轉型過程而言，由於敏捷開發與傳統方式差異甚大，加上目前尚未有標準或模式可依循，使得組織在轉型過程中面臨許多障礙與挑戰。有鑒於限制理論 (Theory of Constraints) 強調突破限制以改善組織績效的特性，本研究嘗試運用其來協助軟體企業於進行 ATP 時能有效思考與檢視流程限制與障礙，並建立一個轉換歷程模式—本研究稱之為 Constraint-based Agile Transition (CBAT) Model。CBAT 模式為一個實務性之研究，故本文以個案研究方式來探索其內容，並透過一實際企業之軟體開發團隊來探討其在轉換敏捷開發過程中所發生的不良效應、以及如何運用該模式來協助其順利轉型，並建立一個完整的轉型歷程框架供其他組織進行敏捷轉型時參考。研究結果指出，個案企業透過此轉換模式能有效地改善其轉型中面臨之問題，並順利轉變為具備自我形成決策與快速應變能力的敏捷開發組織。

關鍵字：軟體開發、敏捷開發、敏捷轉型、限制理論、轉型歷程框架

*本文通訊作者。電子郵件信箱：cychen@mgt.ncu.edu.tw。
2016/10/05 投稿；2017/06/06 修訂；2017/07/13 接受

Using Theory of Constraints to Aid Software Organizations in Conducting Agile Transition Process

Chen Kun Hsieh

Sincere Software Services Co., Ltd.

Chung Yang Chen*

National Central University Department of Information Management

Abstract

Purpose— To introduce a transition model named Constraint-based Agile Transition (CBAT) and report its empirical usage to show how it effectively aids software organizations in conducting the agile transition process (ATP)

Design/methodology/approach— CBAT is methodically based on the Theory of Constraints and the Five-Thinking Processes that are designed to handle constraints, limitations and obstacles encountered during ATP and to provide ongoing improvement guidance that in turns lead an effective way to agile transition. This paper employs the case research methodology to demonstrate and validate CBAT under the real software development environment settings.

Findings—The results of the case showed that CBAT has significantly mitigated issues encountered in the case studied, and helped the case organization establish self-continual software process improvement.

Research limitations/implications— Current research is scoped to software organizations in Taiwan and those organizations who intend to adopt agile software development.

Practical implications—An agile transition process framework is provided for the future reference of other software organizations in conducting their ATP.

* Corresponding author. Email: cychen@mgt.ncu.edu.tw °
2016/10/05 received; 2017/06/06 revised; 2017/07/13 accepted

Originality/value—Software organizations who adopt agile development often face obstacles and challenges, yet currently there are no standards, models or frameworks that help them conduct ATP. This research innovatively develops a transition model (CBAT) to address such an empirical need.

Keyword: Software development, agile development, agile transition process (ATP), theory of constraints, agile transition process framework

壹、緒論

在以前資訊系統尚未普及時，人們與作業環境對於系統的需求較為單純，因此傳統的軟體開發可依據既定之步驟或所謂瀑布式過程、並透過文件以完整確立需求與系統規劃後，始按部就班地進行開發(Leau et al. 2012; Nikiforova et al. 2009; Wysocki 2013)。然而，隨著資訊化時代的來臨與資訊環境快速變遷，軟體交付速度成為現今軟體開發的重要目標(Denning 2012; Thangasamy 2012)。這是由於市場與環境的快速變化使得使用者或客戶經常提出新需求、並且期望負責開發的團隊能夠盡快完成建置。因此，在允許需求變化的情況下，軟體開發組織除了預先規劃之外，還要能夠快速開發及具備因應變化與修正能力。

為了提升軟體組織與團隊的快速實作與應變能力，敏捷軟體開發(Agile Software Development)因運而生。敏捷開發方法強調快速發展與隨時修正以配合現今商業環境的需求，因此許多軟體組織開始嘗試從傳統的方式轉換為敏捷開發(Carvalho & Azevedo 2013)，而轉換的過程即謂之敏捷轉型過程(agile transition process; ATP)(Gandomani & Nafchi 2016; Gandomani et al. 2015b; Olsson et al. 2012; Armour & Kaisler 2001)。敏捷轉型過程影響了軟體企業組織環境的許多構面，其可視為軟體開發思維的一場革命(Abrahamsson et al. 2009; Dybå & Dingsøyr 2008; Savolainen et al. 2010; Tolfo et al. 2011)。這是由於敏捷方式與傳統開發的高度差異，使得原本習於詳盡規劃與按部就班的開發團隊在進行 ATP 中面臨許多障礙和瓶頸(Pikkarainen et al. 2012; Ganesh & Thangasamy 2012; Schova 2010)。此外許多研究指出，這也是由於其並沒有一個標準流程或模式(Abdelnour-Nocera & Sharp 2012; Srinivasan & Lundqvist 2010)，來協助與導引軟體組織突破所遭遇之障礙與瓶頸、順利完成敏捷轉型。

在組織變革與管理領域中，Goldratt 所提出的限制理論(theory of constraints; TOC)指出，一個組織有如一一個系統般，其是由許多的環節所組成。這如同一條鏈子，整個組織的最終績效是由最弱的環節(Critical Chain)所決定；而這最弱的環節成為了阻礙組織成長的限制(Constraint)與瓶頸(Bottleneck)。因此，為了能夠達成更高的績效，組織必須打破這些限制、並做到持續改進(Noreen et al. 1995; Rahman 1998)。本研究認為，敏捷轉型過程所遭遇的障礙與挑戰，或許可以透過限制理論來探討組織在進行敏捷轉型過程中所發生的瓶頸問題、判斷造成問題的根本原因、瞭解組織的現況及目標、進而提出可行的破除障礙與改善措施。

因此，本研究嘗試應用限制理論來協助軟體組織進行敏捷轉型。具體而言，本研究以限制理論中的『聚焦改善五步驟(Five Focusing Steps for On-going Improvement Process)(Goldratt 1999)』與『思考流程(Thinking Process)(Goldratt 1990; Kim et al. 2008; Watson et al. 2007)』為基礎，建立一個敏捷轉換歷程模式 constraint-based agile transition model (CBAT)來導引順利完成轉型。由於 CBAT 為一個實務性之研究，我們以個案研究方式來探索其內容，並透過一個實際企業來深入檢視其在進行 ATP 中如何運用 CBAT 來界定、分析並解決轉型過程中的障礙與不良效應。對於模式成效的檢討，本研究分別從個案組織實施敏捷轉型的量化效益，以及透過訪談來分析模式所帶給組織的質性轉變。研究結果顯示，個案企業透過此轉換模式能有效地改善其轉型中面臨之問題，並順利轉變為具備自我形成決策與快速應變能力的敏捷式開發組織。

貳、 文獻探討

一、 敏捷開發與傳統方式之比較

組織在導入敏捷軟體開發之前，必須先了解其與傳統方式之不同。本研究彙整了相關文獻並從流程觀點來看，主要是在於目標設定與執行方式、價值觀、團隊規模、溝通、以及需求變更處理方式等五大方面進一步說明如下：

(一) 目標設定與執行方式。傳統軟體開發方式多以序列型如瀑布式 (Waterfall Model) 為基礎，強調分層施工、一次交付 (Wysocki 2013; Saxena & Kaushik 2013)。其主張軟體開發須先針對專案整體目標妥善規劃後，再依既定之程序與步驟按部就班地進行 (Leau et al. 2012)。相對地，敏捷開發模式不在初期行過度之計劃；而是先將專案解構成數個可於短期內完成的小目標，並在開發過程中隨時回顧檢視目標、修正計畫。其強調透過較短的週期內即可讓客戶看見實際軟體，並獲得具體回饋以行較具體之修正 (Leau et al. 2012; Saxena & Kaushik 2013)。

(二) 價值觀差異。敏捷軟體開發模式是屬於價值導向 (Value-Driven) 的開發方式。其強調每次都必須交付可實際運作的產出，且重視與客戶的協同合作並正面地看待變化。由於傳統軟體開發模式是計畫導向 (Plan-Driven) (Blankenship et al. 2011)，因此較強調合同、過程與文件的價值 (Wysocki 2013)。Deshpande 等人 (2011) 認為，若以專案管理的鐵三角 (Triple Iron) 觀點而言，傳統開發模式偏向於將功能範圍固定或議定，而時間與成本等為支援性管理機制，故可配合變動；相對地，敏捷軟體開發模式強調時程與成本必須先固定，但範圍則是可變動。

(三) 團隊規模：在傳統方式中，人員角色分工明確細分與高度結構化，因此區隔出系統分析、設計、程式撰寫、測試、系統建置等數個專業角色，是較適合較大型之團隊以求有效之分工。對敏捷開發而言，每一個團隊成員多屬於多工性質、且同時負責多項的專業任務。此外由於其強調合作與溝通 (見下一段之說明)，因此為減少溝通的複雜度，團隊人員往往介於 3 到 9 人之間 (Vijayarathy & Butler 2016)。

(四) 溝通。由於工作高度之結構化與專業分工，使得傳統開發方式嚴守分際、並著重於份內工作的確切完成與介面之嚴謹設計。敏捷開發模式則是強調客戶參與和務實溝通，但此意味著人際關係和社交溝通技能成為整個團隊順利運作與否的關鍵。若軟體功能毋須一開始即明確定義，則開發過程中團隊對於客戶提出的要求和變化需要更有效的溝通與協商技巧。當團隊的溝通能力有問題時，常導致客戶無法在後續的週期中提供準確的需求資訊。故敏捷模式對於溝通能力是有較高的依賴 (Saxena & Kaushik 2013)。

(五) 需求變更的處理方式。敏捷模式在開發過程中對於程序相對較不嚴謹，主要是為了隨時可以承受需求變動，且由於尚未將整個系統需求與設計完畢，因此對於變更得以快速地應變 (Leau et al. 2012)。相對地，由於傳統模式嚴格要求開發的程序性與系統的完整性，以致於開發的自由度降低—即要求專案或產品需先完整分析並期望不會有任何需求變動 (Rico 2008)。一旦發生變更，傳統的開發方式必須嚴謹執行變更管制程序，且由於所有需求與設計已完成加上牽動著大型團隊與諸多角色，變更常導致一連串的問題和影響並付出較高的代價。

二、 敏捷轉型過程

組織從傳統軟體開發方式轉換為敏捷模式，文獻上稱為敏捷轉型過程(agile transition process; ATP) (Gandomani & Nafchi 2016; Gandomani & Nafchi 2015b; Olsson et al. 2012; Armour & Kaisler 2001)。也因為前述的諸多差異，使得軟體開發團隊從事敏捷轉型過程常耗費大量的心力以及較長的時間，並衝擊團隊所處的企業環境與營運的諸多構面，因此其被視為是軟體發展的一場革命(Tolfo et al. 2011)。Abdelnour Nocera & Sharp (2012)指出，敏捷轉型是現今軟體組織在改革創新上的重要課題，但目前仍缺乏一套標準或眾所周知的模型或框架以讓組織有所依循。事實上，轉換至敏捷軟體開發模式涉及到很多改變組織行為，並且由於其變化的程度廣泛，企業通常會面臨許多障礙、問題和挑戰。其中包括流程、人員、管理等方面，以及組織文化和技術問題的轉換障礙(Ganesh & Thangasamy 2012)。

Gandomani 等人在 2013 年起所進行的一系列敏捷轉型研究中，彙整出軟體團隊與組織可能遭遇的數項重大障礙(Gandomani & Nafchi 2016; Gandomani & Nafchi 2015a)。其包括：(1)缺乏相關的知識：參與敏捷轉型的人員往往缺乏對於如何運作的正確認知以及不瞭解其與傳統作法的差異下即直接套用。其結果往往反映在開發團隊的執行效率以及士氣上，並回向造成對敏捷開發的誤解。(2)抵抗變革的阻力：由於人們傾向安於現狀的習性，使得執行人員都習慣於己身所熟悉的角色或活動而抗拒改變。(3)文化的衝擊：敏捷軟體開發模式強調團隊合作、集體決策、成員互信、積極溝通以獲得團隊最大利益。但在許多情況下人們首先會保護自己，而不是考慮到團隊的利益。(4)缺乏合作習慣：敏捷軟體開發模式強調團隊共同負責工作，而非個人。這跟前述傳統軟體開發模式強調專業分工的觀念完全相反。其進一步研究指出，習於分工的人通常較安處於自我的天地，而沒有足夠的信心去參加集體作業和群體決策(Gandomani et al. 2014)。在這種情況下，團隊的溝通自然就無法得到較好之效果。(5)錯誤的心態：對於敏捷軟體開發模式和其價值觀的錯誤心態是進行敏捷轉型過程的另一重大障礙。習慣於傳統軟體開發的人們認為指揮與控制是優秀的管理工作，因此要使這些管理人員能夠放下部份權力，授權(Empowering)讓團隊自我組織，對管理人員來說感到緊張與不安(Gandomani & Nafchi 2016)。

上述研究對於軟體組織轉換至敏捷開發所可能面臨的問題有一般性的了解。由於轉型是一個過程，而要能夠有效地整體衡量並解決過程中所形成的障礙以求最後的順利轉型，組織須要一套系統性的程序方法來辨識具體問題與不良效應、關聯問題、形成適切的改善行動、進而協助開發團隊順利轉型。因此，本研究從流程觀點著手，並嘗試運用限制理論來發展一個歷程模型以協助軟體組織順利突破轉型中的限制與障礙。接下來我們進一步探討限制理論及相關的應用。

三、 限制理論

限制理論(Theory of Constraints)是由 Goldratt 所提出(Goldratt 1999)，其並發展出一套思維程序，讓組織或系統能夠應用嚴謹的因果邏輯關係來獲得問題的改善(李榮貴 & 張盛鴻 2005)。限制理論主張一個系統可為一部機器、一套電腦軟體、一個企業或是一個部門，皆是由許多環節所組成，如同一條鏈子。因此，系統的整體績效往往是由最弱的環節所決定，而最弱的環節即是系統的「限制 (Constraint)」或「瓶頸 (Bottleneck)」(鄭勝璋 & 賴昱達 2013)；它往往形成問題因而阻礙系統達到更高的目標。為了能夠達成其更高的目標，組織必須聚焦於系統最弱的一環，亦即從限制或瓶頸著手，始可獲得

顯著的改善與突破，使組織整體發揮出最大的效率(盧昆宏 2004; 高義展 2010)。

限制理論主要用於了解並管理複雜的系統環境，主張提昇系統績效之關鍵方法就是破除限制(Umble et al. 2001)。一般而言，系統的限制可分為實體限制(Physical Constraints)與政策限制(Policy Constraints)(Goldratt & Cox 2016)。實體限制包括人力、設備以及原物料供應等實體作業資源；政策限制則是如法律、管理方法、制度、流程等無形枷鎖、且政策限制通常會導致實體限制的發生(Dettmer 2000)。為了提升系統績效並解決限制，限制理論建議兩項作法。第一項作法為「聚焦改善五步驟(Five Focusing Steps for On-going Improvement Process)」，其主張藉由循環的流程以幫助企業不斷破除限制進而成長；第二項則是針對管理與政策限制所使用的「思考流程(Thinking Process)」。

進而言之，聚焦改善五步驟為(Goldratt 1999)：(一)找出系統內受到限制的資源；(二)決定如何應用該限制；(三)調整所有系統內的非限制資源，以協助最大化系統的效率；(四)衡量系統限制，試著鬆綁它；(五)若前面步驟破除了系統原有的限制，則回到步驟(一)(Goldratt & Cox 2016; 岸良欲司 2009)。持續改善流程之五聚焦步驟所建構的是一種循環思考模式，促使企業之管理階層可以透過此模式不停地尋找組織的限制，持續地辨識阻礙系統達成目標的重大障礙因素，助其更有效地調適各種既有的資源，進而建構有效的循環改善機制。而為了能夠分析更複雜的系統、解決政策限制，限制理論於後續發展出一套邏輯工具，即為「思考流程」(Goldratt 1990; Kim et al. 2008; Watson et al. 2007)。思考流程指出，於解決系統限制時必須要針對三個問項進行分析(李榮貴 & 張盛鴻 2005; Dettmer 1997)：

(1)『決定要改變什麼(Decide what to change)?』即找出要改變的問題或不良效應為何、以及核心不良效應。

(2)『決定要變成什麼(Decide what to change to)?』即評估可能的解決方案。

(3)『決定如何造成改變(Decide how to cause the change)?』即為建立行動方針。

為了能夠協助組織回答上述三個問題，思考流程使用了五種圖形邏輯工具來分別針對不同的問題進行分析與對照(Goldratt 1994; Rahman 2002)，並且運用這五項工具來檢視系統運作的限制及與改善方向。由於每一個思考流程所運用的分析工具不同，各階段運用的工具對照整理於下表 1。

表 1 限制理論-思考流程階段所運用之邏輯分析工具對照 (Rahman 2002)

思考流程問題	目的	分析工具
什麼要改變?	尋找不良效應與核心問題	衝突圖、現況圖
要改變成什麼?	尋找、創造解決方案，並描述改變後的狀況	未來圖
如何造成改變?	激發解決方案、克服障礙與建立改善行動	必要條件圖 行動轉換圖

在系統歷經一次改善的過程後，系統可能因為實體環境條件的改變，進而產生了其他的限制，故思考流程可結合五個聚焦步驟，以進行持續的改善。因此，本研究將運用上述的概念，特別是思考流程與邏輯分析工具，以作為 CBAT 模式的發展基礎，而分析用之圖形工具的部分也將於稍後的個案內容中詳細呈現。

參、 研究方法

一、 個案研究

根據 Yin (2013; 2011)與 Eisehardt (1989)的研究，個案研究適合於探索新方法或管理新概念之務實性(Practicality)，而在探索型個案研究中，單一而深入的個案報導特別適合於在探討新方法或新概念是如何(How)形成、實施或應用的人為過程，以利模式或概念的 Know-how 之推廣。。由於軟體開發之關鍵因素為人(Darke et al. 1998; Seaman 1999; Acuna et al. 2006; Chen & Chong 2011)，以及限制理論應用於 ATP 是一項實務議題且著重於人性方面之探討，故本研究採用單一個案研究法來深入探索、也得以詳細說明企業如何運用所提出之方法於敏捷轉型的人為過程與運作成果，並可以具體提供其他軟體組織在推動敏捷轉型時參考。

由於從傳統軟體開發轉型為敏捷模式的過程中，個案企業的開發團隊必須思考如何有效排除限制與障礙，以克服因工作方式或觀念之改變所導致的各項問題、並能夠設定正確的發展目標。因此，本研究根據限制理論的思考流程為基礎，研究以下問題：

1. 團隊從傳統軟體開發轉型至敏捷開發模式時所遭遇的限制或障礙，
2. 分析所遭遇的各種障礙和問題之核心問題、並建立因果關係，
3. 以敏捷開發最佳實務為基礎，找尋適切的工具或做法以解決核心問題。

二、 研究過程與架構

本次個案研究過程分為三個階段：準備、執行與分析檢討。在準備階段中，研究團隊進行前置資料之收集與確立研究對象，並確認該組織對於敏捷開發之先備知識是否足夠。此外由於本研究欲針對實施成果進行效益檢討(即第三階段)，故於此階段時須先與個案企業研討關鍵績效指標，以及數據收集的方式。在執行階段中，即從 2015 年 10 月至 2016 年 2 月共約 20 周的期間，研究團隊於個案企業中透過參與該組織所執行的專案開發業務以觀察和紀錄所實際發生之事件，並蒐集組織成員對於投入敏捷開發模式的相關問題與回饋。而在最後階段，本研究針對 CBAT 於該企業的實施成果進行成效分析與檢討。

就研究架構而言，CBAT 模式運用限制理論中的聚焦改善五步驟與思考流程為基礎，並以思考流程中的三大問項為主軸來建立一個流程框架以引導企業進行 ATP。在個案組織的 Scrum 模式運作下，本研究利用每個敏捷循環週期(或所謂 Sprint，每個周期時間為兩週)中約 5~10%的時間來實施 CBAT 模式與團隊成員進行相關的引導活動。其中，CBAT 運用了思考流程的五個圖形工具以找出轉換過程中的不良效應與核心問題、並思索相對應的改善行動、進而執行改善行動。有鑑於改善需要永續進行以及軟體開發環境的高度動態特性，CBAT 模式可加入循環設計，藉以不斷檢視個案敏捷轉型過程中剩餘或新的限制，進行破除與改善的行動。CBAT 之架構如以下圖 1 所示。

進而言之，如圖 1 內的箭頭圖形所示，本研究先從系統限制觀點來彙整個案組織進行敏捷轉型過程中所遭遇之瓶頸與問題，將其視為不良效應，做為限制理論的思考流程之初始輸入資訊，並透過建立現況圖，尋找出本個案之核心問題。此即思考流程的第一問項：「決定要改變什麼？」。再則，本研究透過所發現的核心問題依現有資源(實體限制)與政策限制等兩觀點來建立衝突圖以發展解決核心衝突之策略構想、同時亦建構未來圖以描繪改變後的狀況。此即思考流程的第二問項：「決定要變成什麼？」。最後則透過建置必要條件圖以識別實行目標中可能發生的障礙，藉以發展改善之中間目標，並且依據目標發展行動轉換圖，以建立相關之改善活動。此即是思考流程的第三問項：「決定如何造成改變？」。本研究並藉由欲解決個案核心問題所建立之行動轉換圖來勾勒出本次個案企業進行敏捷轉型之歷程模式。

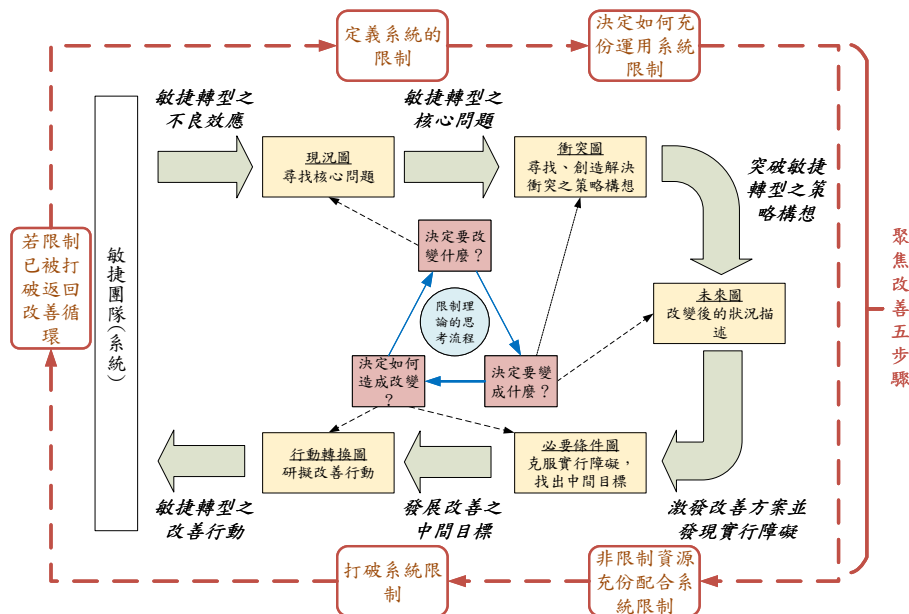


圖 1：CBAT 模式架構

肆、個案研究

一、 個案介紹

SS 公司為台灣某資訊服務集團旗下之子公司。該集團為台灣資訊服務產業的龍頭企業，位居台灣前一百大服務業，為跨足兩岸三地及東南亞的亞洲區域級資訊服務集團公司，並在台灣、中國、香港、新加坡與印尼設有 48 個營運據點。集團中的 SS 成員企業則擁有員工人數約 200 人，其資本額 5.5 億元，總公司位於台北，並於新竹、台中、高雄均設有營業據點，主要業務為銷售各式國內外原廠軟硬體，此外其主要業務為提供多種 IT 整合服務與軟體客製化之服務。

對於 IT 整合服務與軟體客製化之服務來說，SS 公司依據其業務人員所承接的專案客戶所提出之需求，發展出符合期望的軟體與系統產品。SS 公司在早期均採用傳統瀑布型開發模式以執行各類軟體發展專案。但由於資訊環境與客戶的商務環境之快速變遷，SS 公司在開發模式之過程中面臨以下的狀況：

(1) 專案的時程縮短：近年來由於企業對於資訊系統功能所要求的交付時間縮短，專案開發階段由早期是按月來衡量，縮減至幾周甚至數天內就須要完成一個階段。軟體開發團隊因此常被迫要加速工作節奏或工作方式以因應軟體快速交付的要求。

(2) 客戶的需求變化頻繁：由於專案時程的縮短，原本在傳統軟體開發模式所適用的長期、預先規劃的做法顯格格不入；而客戶自身也無法在短期內提供出詳細的需求，導致客戶提出需求變更的頻率變高。因此使得一向採用傳統軟體開發模式的團隊無法快速因應異動頻率的需求異動，進而造成開發時程延遲、交付的軟體品質不佳。

(3) 先求成果、再行修正：在傳統開發模式下，個案組織皆是先透過文件型式確定客戶需求之後，再發展詳細設計，待完整設計確立後才進行後續的步驟。然而如此正規的工作順序不適用現今客戶期望先看到部分成果再做討論的情況。因此常造成 SS 公司無法如期交付軟體產出，或是產品往往皆需要再行大量修改而前期成本無謂增加。

由於業務目標為『在期限內交付符合需求的軟體』、加上前述所面臨的狀況，面對無法順利完成的專案時，SS 公司往往投入大量人力以期能夠趕上進度。但此方式僅是短期止血的做法，且額外投入之人力是來自於其他專案的人員，進而也影響其他專案、形成惡性循環。為了解決此一困境，SS 公司開始尋求更適用的軟體開發方式。而敏捷開發模式所提倡因應需求快速變化的軟體開發能力、強調軟體團隊與客戶之間的緊密協同合作、緊湊而自我組織型的團隊等特色，成為了該組織嘗試突破困境的解決方案。由於 SS 公司所有部門均採用傳統軟體開發模式執行業務，考量到不影響各部門現行專案的運作，避免因全面轉換，公司成員不了解敏捷軟體開發模式的運作而無法有效執行，公司主管擇定該公司之軟體開發 A 組（以下簡稱：團隊）開始實施敏捷轉型之過程。並期望透過個案組織的轉型過程（以下簡稱：本研究個案），建立 SS 公司進行敏捷轉型之典範，供其他部門做為後續實施敏捷轉型之參考與諮詢對象。

二、 應用限制理論於敏捷轉換過程

在許多敏捷軟體開發模式中，SS 公司選擇了 Scrum，並依據 Scrum Guide (Schwaber & Sutherland 2011)來規劃團隊成員角色以及軟體開發週期、活動與產出。由於個案成員從未接觸過敏捷開發，也未使用過 Scrum，因此在轉型過程中遭遇到許多問題。SS 公司了解尚需一個流程模式來導引轉型，因此運用 CBAT 以及其圖 1 之架構來逐步釐清團隊在實踐目標過程中的這些障礙，並運用思考流程的五種分析工具進行推演以找出適切的改善行動。其過程進一步說明如以下小節：

（一）、 尋求核心問題

就限制理論思考流程的第一個問項（「要改變什麼」）而言，即是在於找出不良效應、進而界定核心問題與所造成的行動衝突(Operational conflicts)。首先，如圖 1 的第一步驟，本研究運用德菲法(Delphi Method)與成員先進行半結構化式之訪談，以求一級資料的豐富與完整性並能符合研究範圍。經初步彙整訪談內容後，該組織在進行敏捷轉型過程所產生之不良效應(Undesirable Effects，簡為 UDE)如下所列：

1. 成員感受不到敏捷軟體開發模式帶給他們的正向改變 (UDE-01)
2. 成員為了維持和諧，不會反映組織所面臨的問題 (UDE-02)
3. 部份成員不喜歡共享資訊，不喜歡參與協同合作 (UDE-03)
4. 成員不知道如何有效進行工作規劃 (UDE-04)
5. 程式開發的範圍不能確定 (UDE-05)
6. 開發的程式品質不佳 (UDE-06)
7. 部份管理人員習慣性提出決策，而成員會視其為命令 (UDE-07)
8. 對於目標的切割與短促的交期感到不適應 (UDE-08)

在條列出不良效應之後，透過「表 1-限制理論-思考流程階段所運用之邏輯分析工具對照」所介紹之邏輯工具-現況圖，來分析不良效應之間的關聯、進而找出核心問題。組成現況圖有三個要件(Youngman 2003)：(1)不良效應—即影響系統達到目的各種現象；(2)核心問題—造成不良效應的共同原因(Common cause)；及(3)中間效應(Neutral Effects)—用以了解不良效應之間的因果關係。也就是說，現況圖是以因果關聯邏輯推論形成不良效應的所有原因，再循序地尋找原因背後更底層的不利因素直至根本原因浮現為止。另外，現況圖中的表示方法為：箭頭的起始端為「原因」、終端為「結果」，此結果也可能是另一個不良效應的原因；若有兩個以上之原因造成結果，則利用一個橢圓形符號表示彙總。圖 2 即是個案組織透過上述建立之現況圖，並經分析與討論後界定核心

問題為『對於目標的切割與短促的交期感到不適應』。

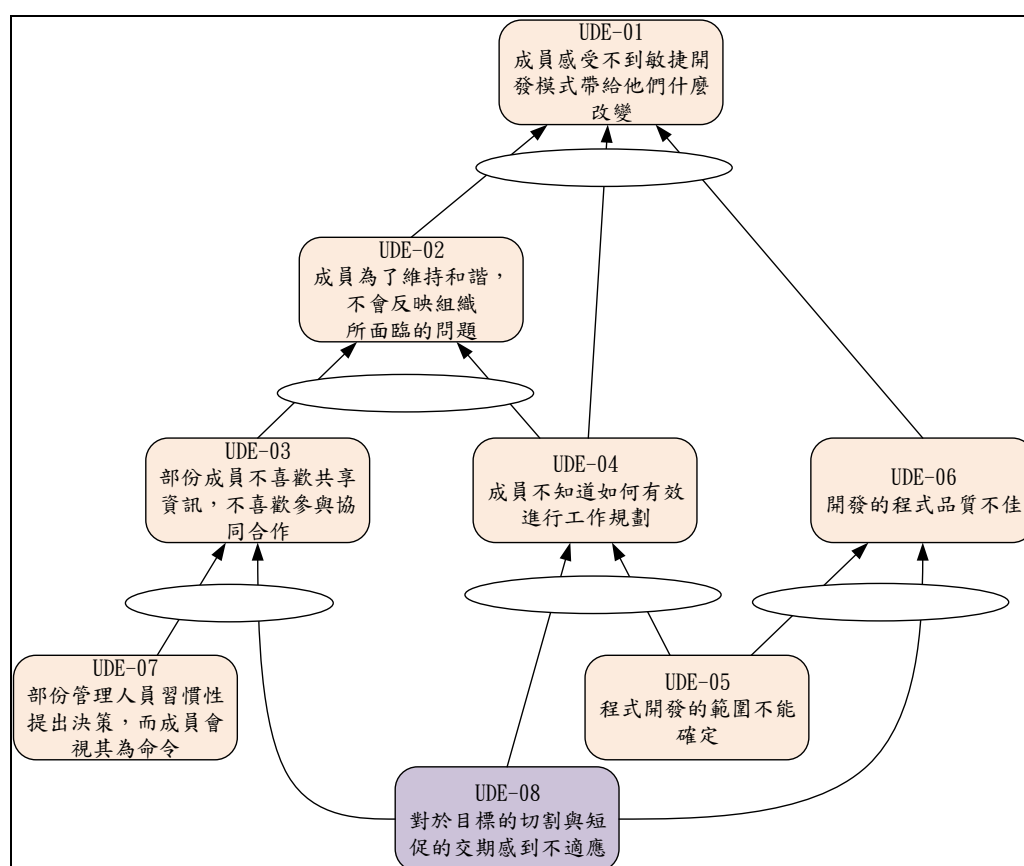


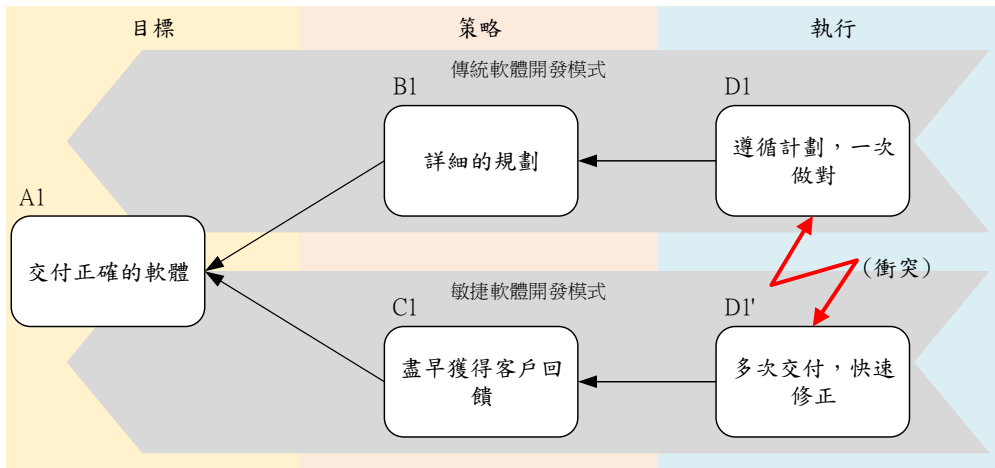
圖 2：現況圖

在辨識出不良效應與核心問題後，本研究參考 Guangyuan 等人 (2010)之建議，先聚焦於三項較為重要的不良效應進而辨識核心問題。至於如何界定較重要之不良效應，本研究建議可從組織目標著手。因此，就個案組織目標—『在期限內交付符合需求的軟體』而言，經訪談的收斂程序後得到三項重要不良效應為 UDE-04、UDE-06 與 UDE-08。接下來，本研究透過傳統開發與敏捷模式之差異對比以分析不良效應在這兩造模式下所造成的衝突。圖 3 即針對上述三項不良效應依：目標—策略—執行等三層面之衝突分析，並以衝突圖呈現因為不同開發模式所造成在執行上的衝突。

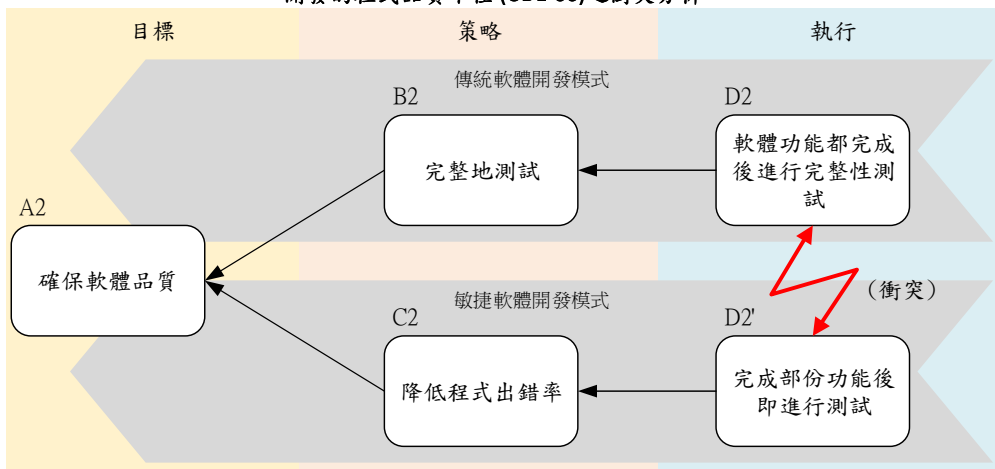
表 2：不良效應衝突共通表

衝突分析項目	共通點
A1、A2、A3	A. 在期限內交付符合需求的軟體
B1、B2、B3	B. 詳細的工作規劃及執行完整的測試
C1、C2、C3	C. 依目標持續調整計劃、少量交付以盡早獲得客戶回饋
D1、D2、D3	D. 既深又廣的工作計劃、依序進行開發工作
D1'、D2'、D3'	D'. 快速驗證客戶需求，迭代且增量地進行開發

成員不知道如何有效進行工作規劃 (UDE-04)之衝突分析



開發的程式品質不佳 (UDE-06)之衝突分析



對於目標的切割與短促的交期感到不適應 (UDE-08)之衝突分析

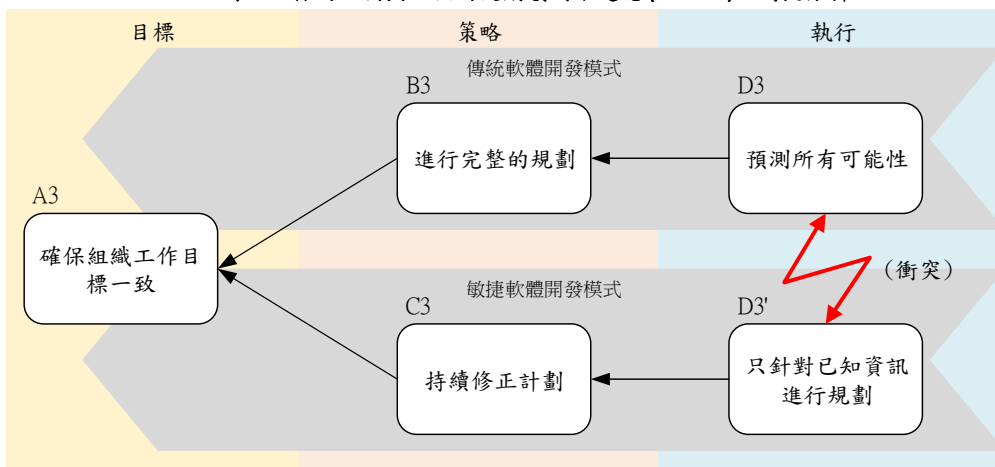


圖 3：三項不良效應之衝突分析

經由上述不良效應衝突圖過程後，進而找出圖 3 之三種不良效應衝突圖共通點，經由整理後如表 2 之不良效應衝突共通表來顯現三種不良效應的共同方向，進而彙整出如圖 4 之不良效應衝突彙整圖。

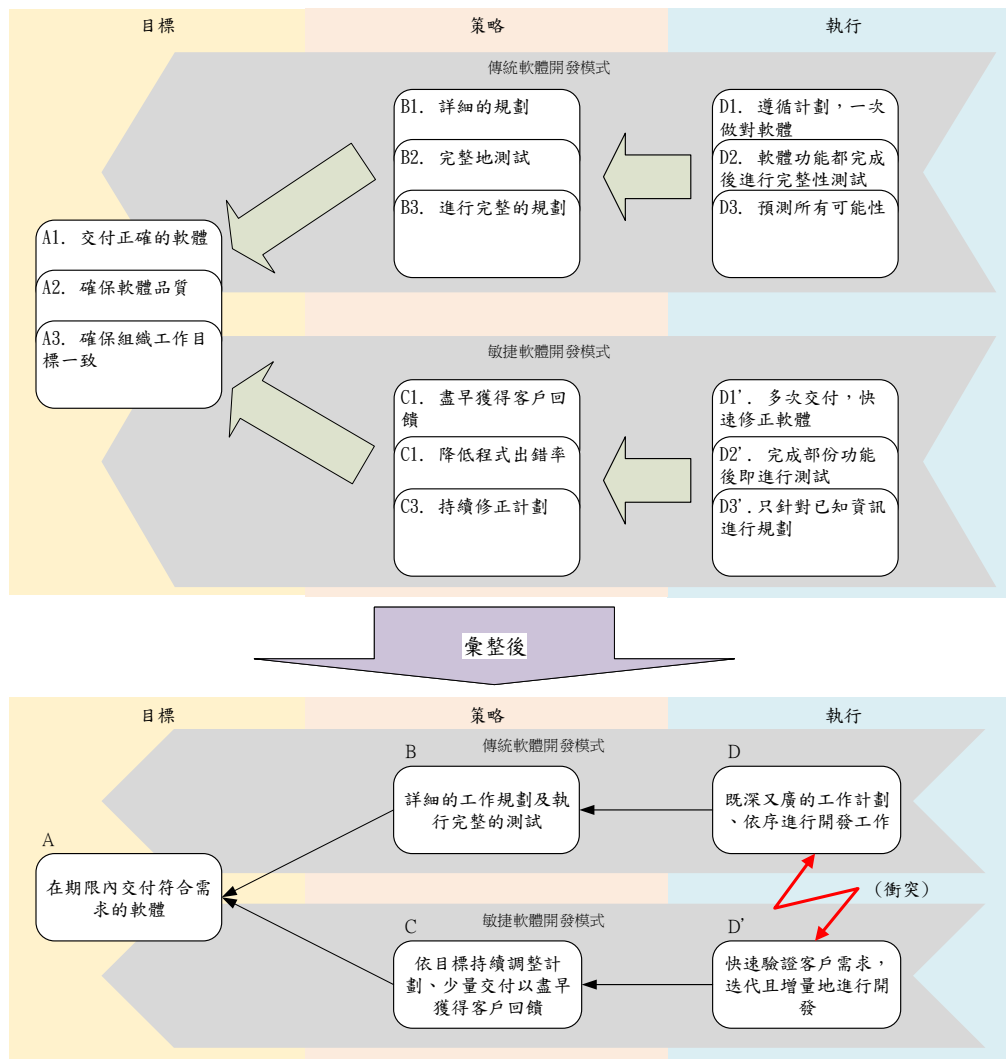


圖 4：不良效應之衝突彙整圖

其中，圖 4 的上半部為從上述三個不良效應與兩種開發模式來看個案組織所形成的策略與行動之脈絡；而圖的下半部則是進一步將策略與行動加以彙整與抽象化以界定核心衝突，其內容並進一步說明如下。由於個案之專案目標為【在期限內交付符合需求的軟體(A)】，為了達到此目標，組織在訪談中表示在以往的開發模式下會有【詳細的工作規劃及執行完整的測試(B)】的中間策略；然而敏捷模式則是有【依目標持續調整計劃、少量多次交付以盡早獲得客戶回饋(C)】的策略。就執行層的專案團隊而言，若要達成(B)之需求，團隊則須要著重【既深又廣的工作計劃、依序進行開發工作(D)】之行動。就敏捷模式來說，如要實踐(C)之策略，團隊則須著重【快速驗證客戶需求，迭代且增量地進行開發(D')】之行動。由於(D)著重在於利用目前已知部份來預測所有未知部份，而(D')則是著重於以迭代且增量進行探索與調整，也就是先做後再行檢討。此兩種執行方式是有所衝突的，也因為此行動衝突而造成個案組織在敏捷轉型過程中遭遇阻礙。

在完成核心問題與衝突分析之後，CBAT 運用思考流程中的現況圖以分析各問題之間的因果關係與檢視個案之核心問題。為了確認個案的核心衝突，本研究與個案組織成員討論與分析影響個案組織進行敏捷轉型過程的不良效應以及核心衝突之間的因果關係，並將傳統軟體開發模式及敏捷軟體開發模式視為因果關係的分析前提，進而整理成現況與衝突之關係圖，如圖 5 所示。圖中的箭線表示項目之間的因果關係。例如若【A-

交付客戶符合需求的正確軟體】成立且【採用傳統瀑布式軟體開發模式】也成立，則會有【B-詳細的工作規劃及執行完整的測試】之結果。圖 5 其他重要說明如下：就個案組織而言，雖然傳統模式與敏捷模式存在著衝突，但其以共同目標為起始，並以所產生的需求要件做為前提條件下對核心衝突進行檢驗，進一步發現個案之核心問題為【UDE-08：對於目標的切割與短促的交期感到不適應】。本研究透過圖 5 下半部份衝突圖的彙整後之衝突與不良效應的相關性建立現況圖，並運用因果關係將不良效應與衝突圖之衝突行動連結，確認核心問題確是造成本研究個案造成衝突之原因。

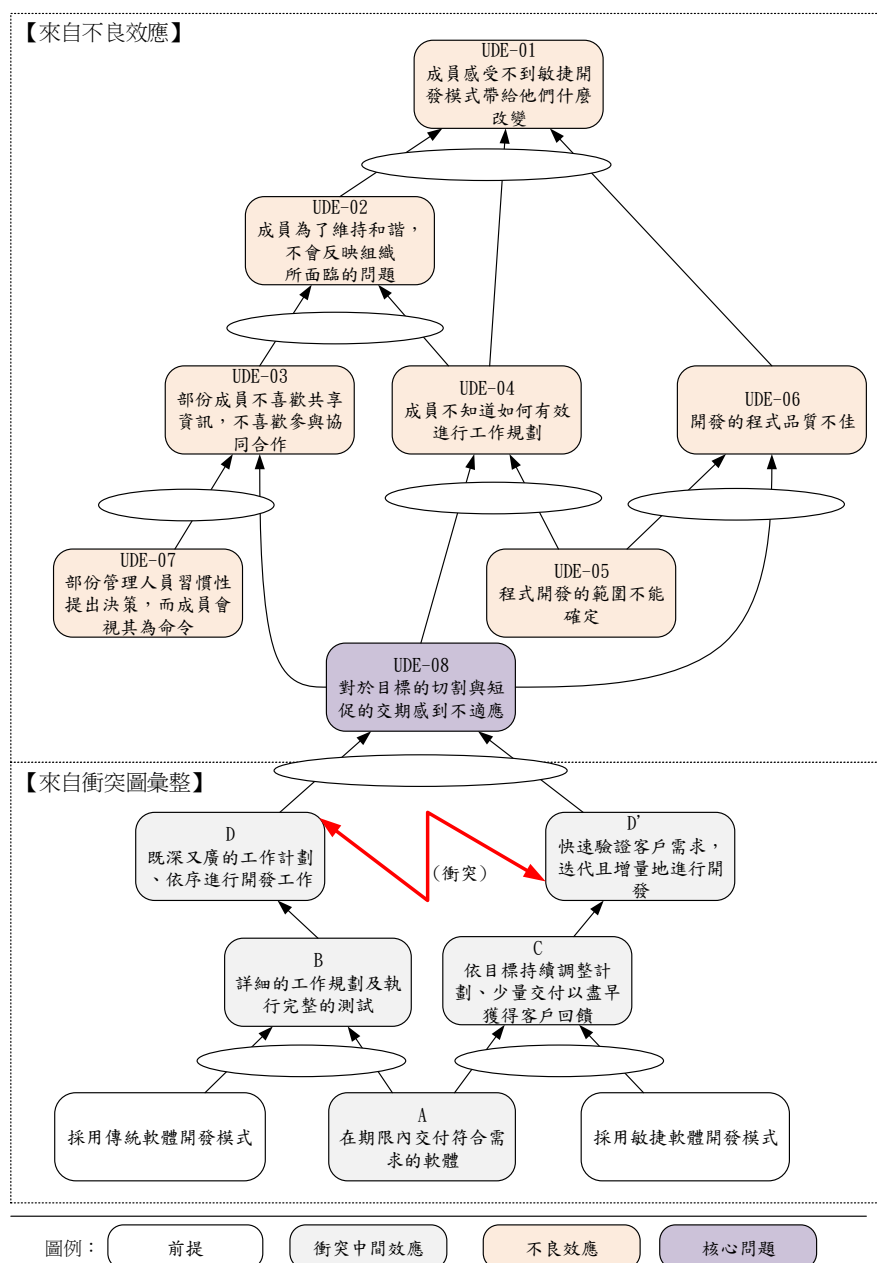


圖 5：現況與衝突之關係圖

(二)、 建立策略構想與改善方案

在完成衝突分析之後，就策略層面而言，個案組織依據現況共同研擬策略構想 (Strategic concept) 與改善作法，以建構「要改變成什麼(即思考流程之第二問項)」之未來圖。首先為發展策略構想，其位於衝突圖之中，表示透過該構想可以消除需求之間的衝

突；在建立策略構想與改善方案過程中，CBAT 主要是針對組織在轉型中所發生的實際衝突來定義組織所要解決與改善的問題，因此延續衝突圖來發展策略和作法，並且依據衝突圖中之共同目標、透過協同過程來共同激發解決衝突的策略構想，如圖 6 所示：

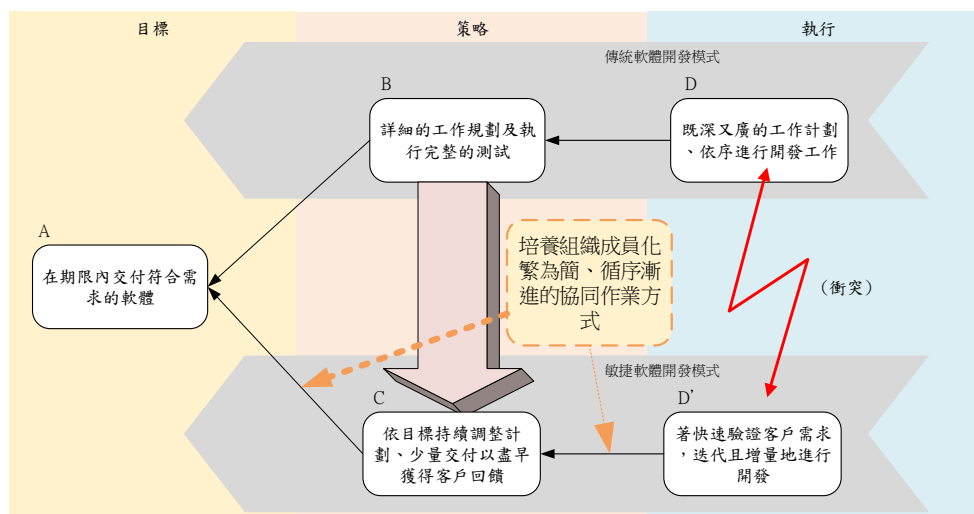


圖 6：從衝突圖中發展策略構想

就 CBAT 而言，策略構想的建構主要是從組織目標(在期限內交付符合需求的軟體)著手。因個案組織期望能以敏捷軟體開發模式進行後續之軟體開發，而該模式又著重於快速驗證客戶需求、迭代且增量地進行開發。經與團隊成員討論後發現，個案組織對於造成核心問題的主要原因為在面對複雜且繁多的批次需求時不知如何整理成數個前後適量開發之 Sprints 與快速驗證之方法。在瞭解上述原因後，個案組織於此一衝突之上之發展相對應的策略構想，即圖中的【培養組織成員發展化繁為簡、循序漸進的協同作業方式】，來使得轉換過程更為順利。而這一策略構想也呼應 Scrum 敏捷方法所提倡的將整個複雜系統內容拆解成以數個較短衝刺週期來迅速交付可執行的部分軟體功能。此外專案團隊也透過 IT 工具的協助，例如透過智慧型手機隨時取得資訊與支持的社群軟體，並促進建立共同決策的習慣，以建立策略構想中所需要之協同過程。

在發展解決核心問題之策略構想之後，CBAT 依不良效應來建立解決問題之戰術目標，並利用表 1 邏輯分析工具中的未來圖進行從策略構想到理想目標之間的驗證。因此在個案中，針對其轉型過程的不良效應(圖 2)為需要破除的障礙，個案組織以策略構想為基礎來發展並推演出解決各個不良效應的具體之戰術目標(tactical objective; TO)，如表 3 所示，並將其投入在團隊之日常作業(內化)，以模擬在此戰術目標為手段、不良效應為因子之下建立可能產生的預期效應(desired effects; DEs)與共伴事實(facts; Fs)。依據上述的預期效應、戰術目標、以及各個預期效應在戰術目標的運作之下的共伴事實等進一步建構出如下圖 7 之未來圖，以推演出邁向共同目標的對策與脈絡。

表 3 不良效應、戰術目標及預期效應表

不良效應(UEs)		戰術目標(TOs)		預期效應(DEs)	
UDE-01	成員感受不到敏捷軟體開發模式帶給他們的正向改變	TO-01	積極解決組織面臨的問題	DE-01	成員感受到敏捷軟體開發模式所帶來的正向改變
UDE-02	成員為了和諧不會反映組織所面臨的問題	TO-02	創造組織成員願意提出回饋的環境	DE-02	成員開始提出回饋，並且針對問題尋找解決方案
UDE-03	部份成員不愛共享資訊，不喜歡協同合作	TO-03	提供公開具透明度的組織訊息交換空間	DE-03	成員開始共享資訊並揭露自己的工作狀況

UDE-04	成員不知道如何有效進行工作規劃	TO-04	共同設定短期工作目標	DE-04	成員可以針對短期目標進行工作規劃
UDE-05	程式開發的範圍不能確定	TO-05	需求有效分解	DE-05	組織可確認 Sprint 週期內須開發之程式功能範圍
UDE-06	開發的程式品質不佳	TO-06	驗收準則明確化	DE-06	組織能夠依據驗收準則開發出一定品質的軟體
UDE-07	部份管理者習慣性提出決策，而成員會視其為命令	TO-07	鼓勵成員自我組織與決策	DE-07	成員可以自行決定軟體開發方式
UDE-08	開發模式轉換的不適應	TO-08	強化組織對敏捷模式的了解	DE-08	組織能夠開始以敏捷軟體開發模式進行軟體開發

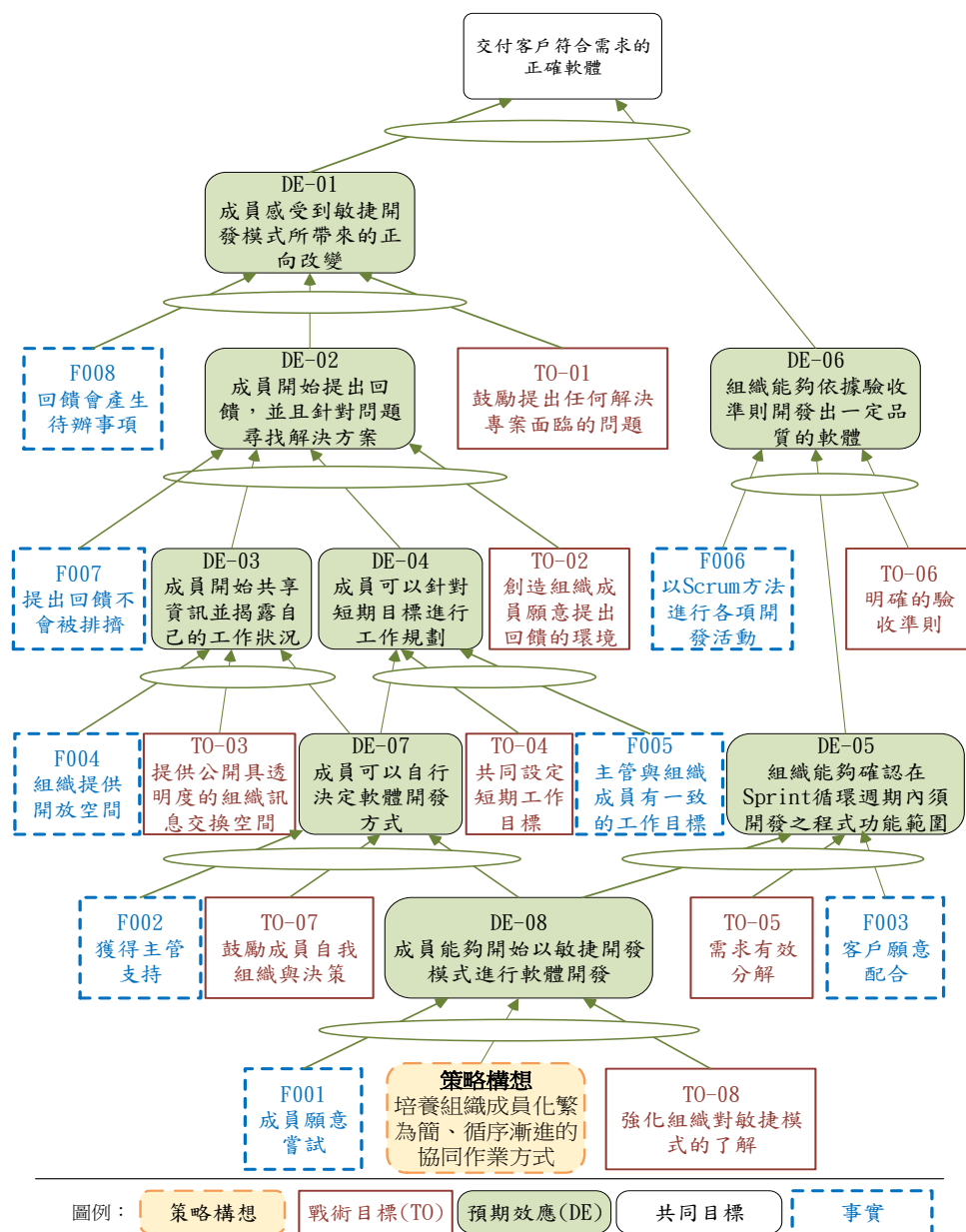


圖 7：敏捷轉型後之未來圖

在限制理論中，未來圖的目的是為了要測試或模擬在衝突圖中發展出的策略構想，

能夠在最終達到想要的共同目標。故透過策略構想開始分析策略構想如何產生各項預期效應；首先將現況圖內的不良效應反向改為預期效應，並且避免因此而可能衍生的所有負面分歧，一路推演到所欲達成的目標。故未來圖中包含四個組成要件(即如圖七的內容所呈現)：(1)策略構想，來自於衝突圖所發想之解決策略；(2) 預期效應，用來表達一些尚未發生的事實；(3)事實，為滿足預期效應所必須存在的事實；(4) 戰術目標，解決不良效應達成預期效應的構想。

(三)、 發展改善行動

CBAT 的第三階段主要為發展出為達成目標而在執行層面之改善行動，也就是探討如何造成改變(思考流程之第三問項)。CBAT 建議兩程序以發展改善行動：首先即是針對策略層面中的各個戰術目標以辨識其可能面對的障礙(Obstacles)；再則由於改變常無法一蹴可幾，故接下來即是推演出達成目標之區間步驟，以利減小目標(理想或概念性質)與行動(實際環境與執行力)之差距。這些區間步驟均對應特定之戰術目標，且其在思考流程稱之為「中間目標 (intermediate objectives; IO)」。以下為個案組織的戰術目標、阻礙與中間目標之對應表如表 4 所示：

表 4：中間目標障礙對應表

戰術目標(TOs)	障礙(Obstacles)	中間目標(IOs)
TO-01：鼓勵提出任何解決專案面臨的問題	1. 問題沒有後續追蹤	IO-01-1：成員可以檢視問題處理狀況
TO-02：創造成員願意提出回饋的環境	2. 擔心主管對回饋成員產生預設立場	IO-01-1：能以對事不對人的原則處理問題回饋
TO-03：提供公開具透明度的組織訊息空間	3. 電子化的訊息空間無法有效交換訊息	IO-03-1：能以視覺化的工具展示訊息
TO-04：共同設定短期工作目標	4. 組織決定的工作目標與客戶/主管期望有落差	IO-04-1：找出利害關係人的共同衡量指標
TO-05：需求有效分解	5. 如何滿足單一週期少量交付？	IO-05-1：簡化需求規格及相關短期工作流程
TO-06：驗收準則明確化	6. 成員不清楚如何產生驗收準則	IO-06-1：與客戶共同確認驗收準則
TO-07：鼓勵成員自我組織與決策	7. 管理者擔心組織決策結果與目標不符	IO-07-1：建立管理者與組織的共識
TO-08：強化組織對敏捷模式的了解	8. 部份成員擔心對敏捷軟體開發模式有誤會	IO-08-1：完成敏捷軟體開發模式相關訓練

依據上表 4 中間目標障礙對應表，研究由衝突圖所產生之目標或策略構想所可能遭遇之障礙，以便後續找出行動方案來解決這些阻礙，達成所擬定之目標；本研究運用上述資訊建構出所需之必要條件圖，來有邏輯性地了解目標如何被實現及其過程中的障礙。進而言之，必要條件圖的功能是用來模擬系統導入戰術目標所可能遇到的障礙、從現況轉移至未來時所必須事先克服的問題或是須先具備的條件，這些須要事先克服的問題或是必須先具備的條件就會形成中間目標，以做為發展行動轉換圖的基礎。

必要條件圖的組成包括三個要件：(1)戰術目標、(2)中間目標(intermediate objective; IO)—欲達成策略構想之前的各個中間步驟與(3)障礙(Obstacle)—即達成中間目標過程可能遇到的障礙。圖 8 為個案組織的必要條件圖。必要條件圖的表示方法與其他圖形有不同的邏輯思維，圖中箭頭所代表為前後關係，起頭起始處代表先有這個戰術目標或中間目標達成，才會造成後面的戰術目標或中間目標的發生，而障礙所指向的箭頭則表示先

後之戰術目標或是中間目標之間的障礙。此外在上一節中，我們透過發展未來圖以發展出策略面的戰術目標。而發展必要條件圖則是針對戰術目標辨別出實行的障礙，模擬系統導入戰術目標所可能遇到的障礙，進一步設定中間目標，做為後續發展改善行動的基礎。比較未來圖與必要條件圖的差異在於，未來圖中所發展之戰術目標其主要用以滿足預期效應，最終達成共同目標；而必要條件圖所發展出的中間目標，其目的在於確保戰術目標能夠達成，籍以能夠支持未來圖的規劃。

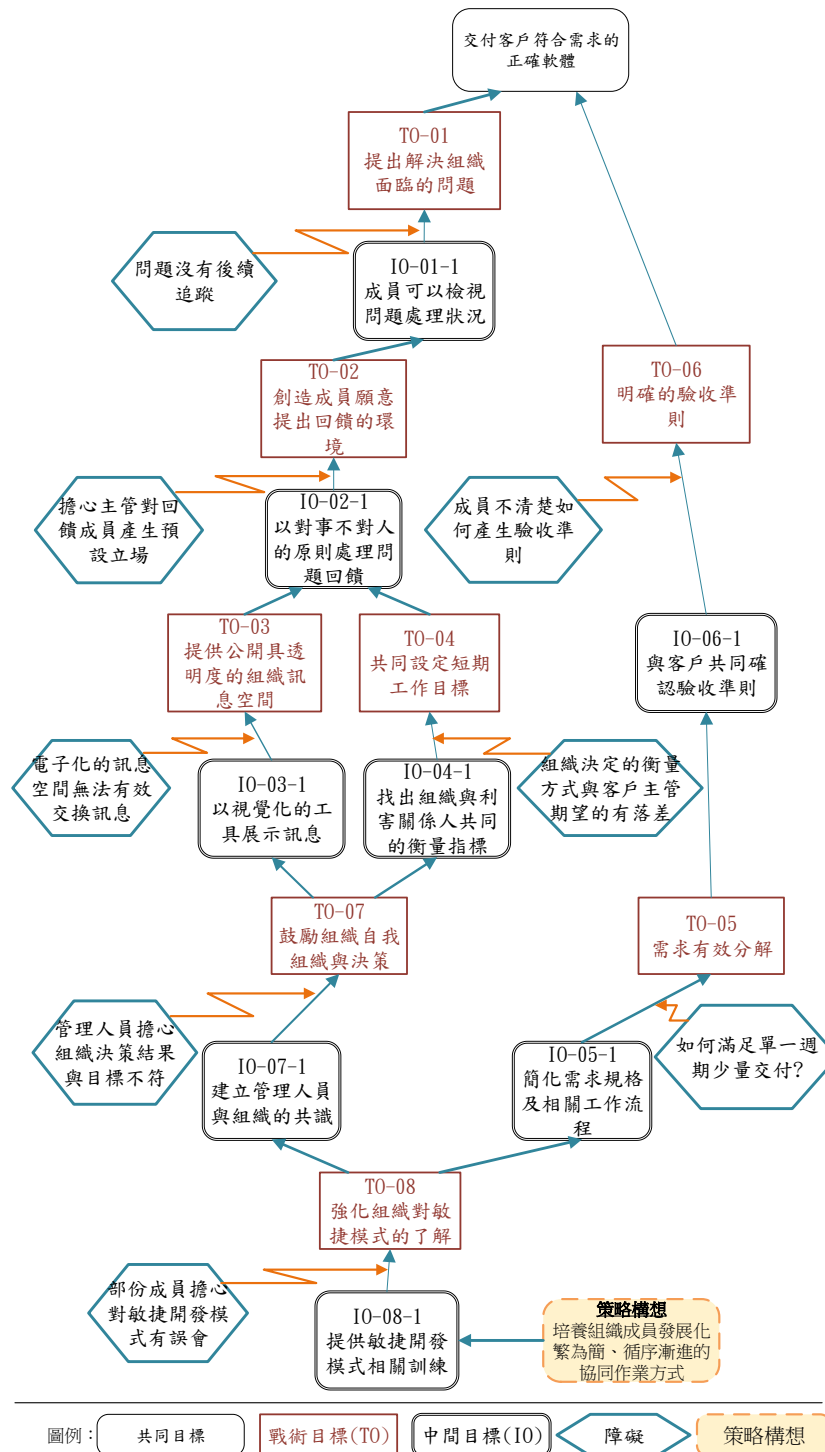


圖 8：必要條件圖

利用必要條件圖勾勒出達成目標的障礙之後，組織即針對上述的三種目標(中間目

標、戰術目標與預期效應)擬定確切的改善行動。將每一個目標擬定確切的改善行動，以顯示出為了成功地達成目標的作為，利用「現實、期望、行動」三要素來進行思考，了解現實的狀況利用改善行動來執行，這些改善行動透過熟悉敏捷模式之業界專家協助，以敏捷軟體開發模式的最佳實務進行規劃及實施，以達到所設定的各個目標。表 5 即是個案組織為達成目標所建立的 22 項改善行動。而在這些擬定出的改善行動之中，個案組織應用以下 Scrum 開發模式之最佳實務(Dahmann et al. 2013; Sander 2014; Schwaber & Sutherland 2011)：

(1) 建立團隊公約。讓組織成員自行決定公約，從定義 Scrum 固定開會時間及會議要求，到程式開發規範，都由成員共同決定，並在每次的回顧會議中重新檢討公約內容。

(2) 定期舉行需求梳理會議。讓 Product Owner 與 Development Team 在每一個 Sprint 循環之執行過程中抽出 5%到 10% 的時間來檢視在 Product Backlog 項目，然後挑選一些下一個 Sprint 準備要施工的需求。

(3) 建置實體工作白板。採用 Scrum 方法的團隊通過一個白板和便利貼來完成使用者故事以及工作的追蹤管理。為了使團隊成員能夠面對面有效的溝通，個案組織決定採用實體白板做為軟體開發過程中溝通的工具。

(4) 建立燃盡圖及進度衡量指標。個案組織利用工作燃盡圖以了解團隊在 Sprint 的工作進行是超前或是落後；並利軟體版本燃盡圖了解軟體系統的開發進度。

(5) 以使用者故事與驗收準則取代需求規格。個案組織的使用者故事有以下的規範：以企業規定之格式撰寫、必須至少有一項驗收準則、該故事經 Product Owner 估算過其價值、有 UI 草稿、故事的限制說明。而驗收準則中必須至少有一條基本的測試情節。

(6) 協助團隊自行發展工作完成準則：在運行敏捷軟體開發模式中較具成效的團隊，均提倡團隊須自行定義出工作完成準則(definition of done; DoD)。在個案組織當中，所定義的工作完成準則包括：程式開發工作必須執行單元測試才能視為完成、製作的使用者故事必須有驗收準則、程式碼必須完成才能簽入至版本管控系統等基本準則。

表 5 目標與改善行動對應表

改善行動(Action)	滿足目標	目標分類
Action01：建立團隊公約	DE-01：成員感受到敏捷模式帶來的改變	預期效應
Action02：待辦事項有負責人	TO-01：積極解決組織面臨的問題	戰術目標
Action03：將成員提出的回饋展開成待辦事項	IO-01-1：成員可檢視問題處理狀況	中間目標
Action04：客戶共同參與 Sprint Review	DE-06：組織能夠依據驗收準則開發出一定品質的軟體	預期效應
Action05：成員輪流主持 Retrospective	DE-02：組織成員開始提出回饋，並且針對問題尋找解決方案	預期效應
Action06：回顧會議	TO-02：創造組織願意提出回饋的環境	戰術目標
Action07：以使用者故事與驗收準則取代現行需求規格	TO-06：驗收準則明確化	戰術目標
	IO-05-1：簡化需求規格及相關工作流程	中間目標
Action08：與客戶進行溝通	IO-06-1：與客戶共同確認驗收準則	中間目標
Action09：要求成員準時參與 Daily Scrum	DE-03：成員開始共享資訊並揭露自己的工作狀況	預期效應

Action10：與客戶確認短期工作目標	DE-04：成員可以針對短期目標進行工作規劃	預期效應
Action11：評估在 Sprint 週期內可完成功能範圍	DE-05：組織能夠確認在 Sprint 循環週期內須開發之程式功能範圍	預期效應
Action12：協調組織開放空間	TO-03：提供公開透明的團隊訊息空間	戰術目標
Action13：確認指標衡量規則	TO-04：共同設定短期工作目標	戰術目標
Action14：需求梳理會議	TO-05：需求有效分解	戰術目標
Action15：建置實體工作白板	IO-03-1：以視覺化的工具展示訊息	中間目標
Action16：建立燃盡圖及進度衡量指標	IO-04-1：找出組織與利害關係人共同的衡量指標	中間目標
Action17：協助組織發展工作完成準則	DE-07：成員可自行決定軟體開發方式	預期效應
Action18：協助組織自行決策	TO-07：鼓勵成員自我組織與決策	戰術目標
Action19：尋求管理人員支持	IO-07-1：建立管理人員與組織的共識	中間目標
Action20：以 Workshop 方式進行敏捷模式訓練	DE-08：成員能夠開始以敏捷軟體開發模式進行軟體開發	預期效應
Action21：外部專家協助訓練	TO-08：強化組織對敏捷模式的了解	戰術目標
Action22：成員共同處理問題	IO-02-1：對事不對人原則處理問題回饋	中間目標

在預期效應、戰術目標、中間目標、障礙、與改善行動等建立之後，CBAT 將其匯集，根據上述的脈絡並運用「表 1-限制理論-思考流程階段所運用之邏輯分析工具對照」所描述之的行動轉換圖加以連結而建立敏捷轉型歷程模式，如圖 9 所示。行動轉換圖的功能在於規劃出改善行動，用以改善必要條件圖上所標示的障礙，以朝向策略構想邁進。行動轉換圖共有二個組成的要件：(1)目標(Objective)，可以為中間目標、戰術目標或策略構想；(2)改善行動(Specific Action)。根據預期要達到的預期目標、中間目標或預期效應，擬訂出能滿足這些目標的行動準則，並利用圖形方式驗證是否在現實的狀況中，採取這些行動準則，就能夠滿足所要達成的目標。完成此行動轉換圖之後，個案組織依據模式中所列示之(1)『目標』，來了解轉型的戰術目標、中間目標與預期效應之間的關聯；(2)為達成目標所可能遭遇的『障礙』，並藉以可進一步擬定所應建立的具體措施；(3)『改善行動』，即是掃除敏捷轉型障礙的具體措施與 Best Practices；以及(4)『脈絡』，以了解作為整體轉型過程的指引。本研究稱之為歷程模式是在於此歷程完整呈現案例組織在嘗試進行敏捷轉型中所歷經之過程與解決方案，亦是運行整個 CBAT 模式後的一個總結性結果，以供其他從事敏捷轉型的軟體組織參考。

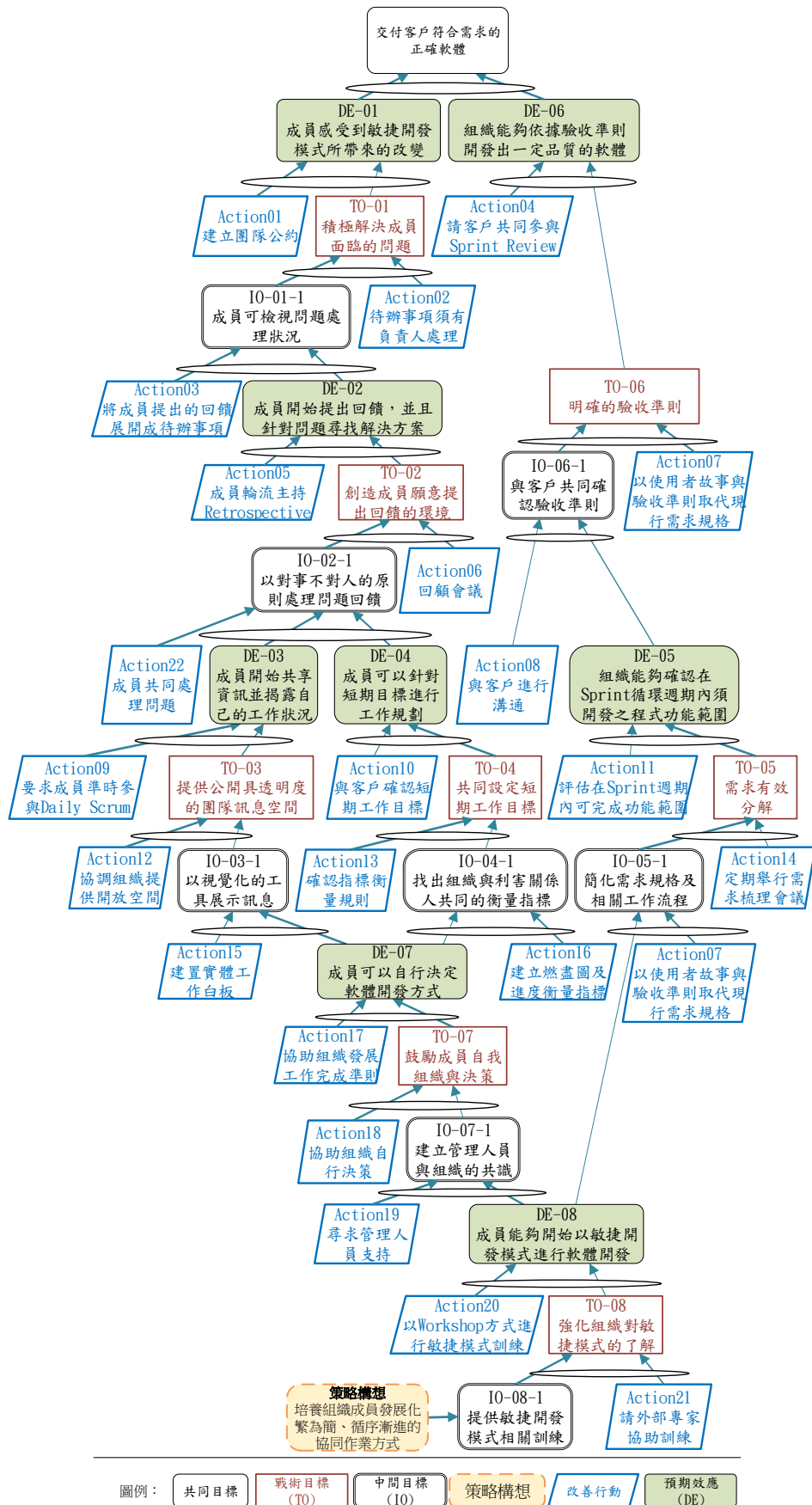


圖 9：行動轉換圖—個案組織之敏捷轉型歷程框架

伍、研究結果與討論

本研究於前一節中提出一個敏捷轉型歷程模式，並針對個案公司所遭遇之實例進行分析、進而發展出解決問題之行動方針。本節進一步針對 CBAT 模式導入之效益加以檢討，以期能推廣並提供給其他專案所運用與參考。本節計有兩個小節來說明與討論結果：(1)關鍵績效指標(KPI)所顯示之改善成果，以呈現本模式對於協助組織實施敏捷轉型的量化效益；(2)訪談分析：訪談參與本個案之團隊成員，從中分析團隊成員實施改善過程後所帶來之質性影響。本節最後則針對研究發現與現有類似文獻加以分析比較。

一、關鍵績效指標

敏捷軟體開發主張以客戶/使用者為依歸、在規劃的時間內完成預定的需求做為主要目標之一。此外個案組織亦期望能透過導入敏捷發展來降低專案開發的風險。因此就量化而言，本研究依據個案組織所著重的三項關鍵績效指標(KPI)來衡量 CBAT 對於個案組織從事敏捷轉型過程中的助益。這三項 KPI 包括：(1)循環週期需求完成比率、(2)風險/議題處理完成比率、及(3)未及時確切提出的細部需求數。其定義描述如下：

(1) 循環週期需求完成比率：個案組織透過檢視每一循環週期中的需求完成比率，來檢驗是否開始順利轉型為敏捷團隊的一項關鍵績效指標。該比率的計算方式如下：

$$\text{循環週期需求完成比率} = \frac{\text{當期循環週期需求完成數量}}{\text{當期循環週期需求計畫數量}} \times 100\%$$

(2) 風險／議題處理完成比率：風險與議題之識別與處理是軟體開發中重要的一環。而敏捷軟體開發模式強調溝通和客戶參與，透過密集地溝通應能確保風險及議題能被識別且及時處理。故本研究透過檢視風險／議題處理完成比率來確認個案組織是否能夠有效審視風險與議題。該比率的計算方式如下：

$$\text{風險／議題處理完成比率} = \frac{\text{當期風險／議題處理完成數量}}{\text{當期風險／議題剩餘數量}} \times 100\%$$

(3) 未及時確切提出的細部需求數：敏捷軟體開發模式在初期並不提倡過度計劃，並且在執行軟體開發的過程中，隨時回頭審視工作目標並做出適當的修正，並且在過程中陸續發掘出使用者在一開始較模糊的需求之細部內容。敏捷發展法提倡在每次開發循環中，專案團隊應挖掘未及時確切分析好的需求並與客戶確認之，以提升開發效益。由於對於待開發需求的評估，個案組織在過程中採用使用者故事點數(User Story Point)做為估算方式。故本研究透過審視個案組織在每一循環週期中是否有進行需求之點數調整與評估調整來驗證此一開發效益。

在本個案中，我們利用 6 周的時間針對個案之背景與環境進行 CBAT 的導入、與大約 14 週時間做為 CBAT 之實施。這 20 週的時間是配合個案公司一個正進行中的軟體開發專案，以能實際透過敏捷軟體開發模式來實施 CBAT。該專案過程中訂每二週為一個 Scrum 的循環週期，稱之為 Sprint，故有 10 個 Sprints。因此在採用 CBAT 方式之前，團隊已進行三個 Sprint 循環週期之專案工作，為使團隊盡快進行轉型問題檢視及改善規劃，研究者與團隊決定於第四個週期開始採行 CBAT 方式。在完成十個 Sprint 等量(開發數量與難度均等劃分)循環週期的觀察後，本研究彙整了此十次循環之關鍵績效指標數據，前三次為導入改善行動前之數據；後七次循環為導入後之數據，如表 6 所示：

表 6 關鍵績效指標表

Sprint 循環週期	循環週期 需求完成比率	風險/議題 處理完成比率	未及時確切提出的 細部需求數	說明
1	33.33%	0%	0	解決方案導 入前
2	30.00%	13%	0	
3	26.67%	0%	0	
4	50.00%	38%	2	解決方案導 入後
5	56.67%	40%	15	
6	70.00%	67%	18	
7	80.00%	100%	15	
8	90.00%	100%	18	
9	96.67%	100%	13	
10	100.00%	100%	23	

在經過十個等量循環週期的觀察，相關關鍵績效指標的變化分析如下：

(1) 循環週期需求完成比率：在導入解決方案前，個案組織在每一次循環週期結束進行週期審查時，實際完成比率均未滿 30%；在導入解決方案後，需求完成比率均高於 50%，且完成比率逐步上昇，並於所統計的最後循環週期結束時，完成比率到達 100%。分析其中變化的關鍵，我們發現透過限制理論的思考流程所規劃【TO-04：共同設定短期工作目標】、【TO-05：需求有效分解】、【TO-06：驗收準則明確化】等三項戰術目標以及所對應的改善行動對於個案組織產生了正向的效果，並解決個案組織對於如何確定開發範圍、如何有效進行工作規劃、開發程式品質不佳之問題。

(2) 風險／議題處理完成比率：導入解決方案前，個案組織在每一次循環週期結束進行週期審查時，處理風險或議題的數字幾近於零；經檢討後發現，個案組織在前三個循環週期中，團隊尚處於切換原軟體開發模式與尚未熟悉敏捷開發，因此對於專案中的風險或議題，團隊成員較不熟悉如何以敏捷強調的精神來處理或解決問題。也因為是長期習於傳統軟體開發模式，團隊在一開始並不積極發掘問題並即時與客戶確認，這也是文中提到團隊面臨的轉型不良效應。因此結果顯示前三周期並無或甚少發掘與解決問題，故表中為 0%或較低的數值(13%)。而在導入 CBAT 模式與提倡執行改善行動之後期，團隊開始會在每一次進行 Scrum 相關活動時，留意風險或議題的處理狀況，故可以發現在後七個週期的風險/議題完成比率的數字逐漸提昇，在第七個週期之後，團隊成員已能夠在每一個開發階段中有效處理各項風險及議題。在與個案組織檢討分析其原因後發現，【TO-01：積極解決團隊面臨的問題】、【TO-02：創造團隊願意提出回饋的環境】、【TO-03：提供公開具透明度的團隊訊息空間】、【TO-07：鼓勵團隊自我組織與決策】等四項戰術目標使個案組織成員之間的溝通變得頻繁且透明，而個案組織也逐漸演變成具備能夠自我決策的組織。

(3) 未及時確切提出的細部需求數：於個案中，客戶對開發目標與範圍已議定，但對如何發展其解決方案等細節並未進行詳盡之規劃。在擬定戰術目標與導入解決方案之後，團隊在每一次循環週期即進行需求梳理會議，以針對即將開發的需求進行細部功能規劃。此外團隊也嘗試在每一個 Sprint 中立即提交一版軟體供客戶使用，而客戶在試用之後則會提出細部修正之想法。這些作法的結果則反映在此關鍵績效指標中，即未及時確切提出的細部需求數之增加。雖然傳統的開發方式在初期即能詳列出需求內容，而敏捷開發看似過程中需求數量增加，但在多變與人本的商務環境下，經過個案組織的實際

運作後發現，這些在過程中增加的細部需求反而促進了雙方對於系統內容能夠即時了解與產生共識，並讓客戶得以在過程中見到實際系統可再提出更具體的看法，也涵蓋了因為不確定性所造成的變更，而避免了產品在完成後的變動。經與個案組織檢討分析其原因後發現，【TO-08：強化組織對敏捷模式的了解】此戰術目標使個案組織逐漸演變成具備調適變化的軟體開發團隊。

二、 訪談分析

除了上述的量化效益之外，本研究透過深入訪談方式、從人文觀點來檢討組織成員對於所採用的轉型模式的經驗與感受，以提供其他類似專案所運用與參考。本次訪談採取半結構化以及同時涵蓋形成性與總結性的提問方式，來獲得較深入之見解與了解組織的整體質性轉變。在訪談過程中經受談者同意後本研究將對話過程完整記錄(一級資料)，以利本節稍後的評論(二級資訊)之整理與分析。此外訪談問題為中性呈現，以避免導引效果或是因題目之態度與訪談人不同而使得回答內容有所保留。

為了完整地涵蓋相關關鍵人員，訪談對象則包含專案團隊與管理等不同階層、以及專案內不同角色與不同程度資歷之五位關鍵人員代表，如下表 7 所示。

表 7 訪談人員表

訪談人員	組織/團隊角色	工作年資
郭協理	個案公司技術部門主管	20 年
詹經理	個案組織主管	15 年
陳先生	資深研發工程師 (Development Team Member)	8 年
吳先生	系統分析人員(Product Owner)	20 年
林先生	開發人員 (Development Team Member)	2 年

訪談內容經整理與組織後呈現如下：

(1)對於建構衝突圖以找出策略構想之感想。由於在敏捷轉型的過程中，個案組織所遭遇到的核心問題是在於對於團隊成員對於開發模式轉換的不適應，而透過限制理論的思考流程中所發展之衝突圖策略構想「培養組織成員發展化繁為簡、循序漸進的協同作業方式」，受訪者皆同意其精神意涵，但在未具體發展出行動方針之前，其對此策略也多保持著觀望的態度。而在組織具體導入行動方針於數個循環週期之後，五位成員對於其轉型成效則表高度肯定，認為導入之行動方針具有相當成效。除了專案團隊有正面的總結性感受之外，管理者進一步回應如下。

「部門成員習慣以傳統開發模式進行軟體開發，而在轉換至敏捷軟體開發模式的初期，成員們對於反覆及增量方式的實踐方式確實是較無法理解，造成每次在進行 Sprint Review 時，目標達成比率總是過低。……經採用此方法(CBAT)以先找出核心問題並規劃出解決衝突之策略構想後，部門逐漸能夠掌握到敏捷軟體開發模式的務實做法，而在後續進行 Sprint Review 時，目標的達成比率也有顯著提昇」(郭協理)

(2)對於運用思考流程之邏輯工具找出解決方案之感想。CBAT 運用限制理論中思考流程之邏輯工具以發現核心問題並逐步、有脈絡地尋求解決方案與改善行動。此作法對於個案組織成員而言是全新的嘗試；但在瞭解思考流程的邏輯工具之運用方式後，組織成員均表示能夠快速地發現與了解核心問題，並對戰術目標及相應的改善行動的邏輯關

聯性皆表肯定。本研究認為這或許是由於個案組織成員從事軟體專案開發工作，對於邏輯推論過程並不感陌生難以接受。而專案主管亦進一步表示圖形化的呈現也有所助益：

「普遍來說，人對於新嘗試與新習慣較被動接受，也不太願意主動去了解。因此部門內的成員由於缺乏對於敏捷軟體開發模式的積極認識，使得在進行轉型過程中便遭遇到許多障礙，進而讓成員對於這樣的轉型過程有所困惑、感覺問題接踵而來，而沒有進一步想到去分析這些障礙的關連性。此方法特色在於強調關聯、以及建構關聯的圖形過程，讓團隊能具體明白，原來所有轉型中發生的問題都有因果關係，也透過這樣的過程，能夠檢驗這些不良效應的背後，是否都是因為核心問題所延伸出來。」(詹經理)

在組織中擔任 Product Owner 的吳先生也表示：「在剛開始實施敏捷轉型時，同事們其實都不太了解執行轉型之目的，也不清楚應該怎麼做才能達成所設定的目標，也不知道如何才能準時交出有品質的程式。經過一連串的問題探討過程以及幾種圖形的引導，讓我們對於之前發生的不良效應能有更多的認識，也讓我們明白原來這些不良效應都有其因果關係。」

(3)組織在敏捷轉型過程中的轉變。對於此一總結問項，成員的回答為兩大方面：自我決策能力的提升，以及快速應變能力之建立。成員們提到，組織依據行動轉換圖所發展的改善行動，配合敏捷軟體開發模式的最佳實務進行規劃及實施，做為滿足其中間目標、戰術目標、以及推導預期效應的解決方案。其並表示透過參與規劃改善行動的過程，除了對於限制理論的思考流程有更深入地瞭解外，並能夠使團隊考慮到自己的意見或是他人說服自己的疑慮，因此具相當高的認同感，這樣的支持提升了組織自我決策之能力。對此本研究認為在形成上有一項重點改變因素，即實施團隊參與與營造正面霍桑效應(Chen & Chong 2011)。此外而在實施改善行動的過程中，成員表示雖然不能夠馬上讓所有問題獲得立即性的改善，但透過多個循環週期的執行，成員能感受到改善行動項目的逐項完成，也讓團隊自我逐漸達成原先所設定的關鍵績效指標。對於自我決策能力的感想，本研究並特別關注較新進人員之感受，其回答如下：

「剛開始參與敏捷轉型的過程，個人都不太能夠適應工作模式的轉變，也對於集體作業及群體共同決策沒有信心。後來在找尋核心問題的過程中，我發現或許是因為習慣於之前專業分工模式加上本身經歷不足，因此較習慣於被動地受主管的指揮。在發現問題制定改善行動後，且獲主管充份授權與成員的相互勉勵下，我開始能夠與其他成員互動，主動提出自己看法以利形成共識，相互合作來完成程式的開發，同時在審查會議中，我們也因有先達成決策共識而相互 back-up，表現一致最後並得到客戶的肯定與鼓勵，也讓我開始認同敏捷軟體開發模式能夠幫助我們有效達成目標。」(林先生)

快速應變能力是敏捷發展的重要訴求，但成員們普遍認為這對於他們習於規劃與按部就班的舊作業是有相當大的差異，因此一開始很不習慣。然而透過所採用的模式之後，反成為他們最能感受到的最大效益與轉變。成員們指出，快速因應並非僅是迅速形成行動決策，其中也包含了問題脈絡與思考層次的建立，以求找出根本原因與共同原則，才能快速因應又能治本。不同資歷的成員對於這樣的轉變也補充了個人看法：

「我認為我們剛開始在敏捷轉型過程中，最大的挑戰是在於我們沒有辦法在每一個 Sprint (開發週期)內交付出客戶希望獲得的程式功能。透過此流程框架(CBAT)，我們擬定出『需求有效分解』的戰術目標，並考量現實的環境找出可能的障礙後，找出『簡化需求規格及相關工作流程』的中間目標，並依據敏捷軟體開發模式的實務，並『以使用者故事與驗收準則取代現行需求規格』做為改善行動之指導原則，這樣的思路十分清楚。

在實施行動準則的過程中，也會遭遇到其他的困難，但是都能夠透過其他擬定出的行動準則，快速地、一併地因應與克服。」(陳先生)

除上述之摘要探討外，本研究也將訪談結果與個案分析中提出的敏捷轉型所面臨之問題彙整成改善結果如表 8 所示，並提供給從事敏捷轉型的企業參考：

表 8 敏捷轉型過程不良效應改善結果

面臨的問題	改善結果
成員感受不到敏捷軟體開發模式帶給他們什麼正向改變	透過「積極解決組織面臨的問題」之戰術目標，透過短期的成果交付以獲得客戶的回饋，成員對於軟體的產出成果感到踏實與具體成長，也讓成員較具信心去因應變化。
成員為了維持和諧，不會反映組織所面臨的問題	透過「創造成員願意提出回饋的環境」之戰術目標，依據現實狀況實施改善行動之後，成員開始能以對事不對人的方式反映問題，並且共同找出問題的處理方法。
部份成員不喜歡共享資訊，不喜歡參與協同合作	透過「提供公開具透明度的組織訊息空間」之戰術目標，經由公開、視覺化的工作看板，成員開始進行溝通與協同作業。
成員不知道如何有效進行工作規劃	透過「共同設定短期工作目標」之戰術目標，使成員可以在短時間將目標完成，並在 Sprint 循環週期結束後，針對所規劃的內容進行評估與檢討。
程式開發的範圍不能確定	透過「需求有效分解」之戰術目標，成員與客戶共同將原本龐大、模糊的需求功能，分解成較小、具體的產品待辦清單項目(Product Backlog Item)，除了能協助客戶更加清楚需求的實現方式，也使得團隊成員更容易達成工作目標。
開發的程式品質不佳	透過「驗收準則明確化」之戰術目標，使每一個需求在進行開發之前，至少對應到一個明確可測試的驗收準則，開發人員在開發階段就能夠清楚了解如何驗證程式功能。
部份管理人員習慣性提出決策，而成員會視其為命令	透過「鼓勵成員自我組織與決策」之戰術目標。以及導入 Scrum 方法中的 Daily Scrum 例會和視覺化工作看板，管理人員開始能夠讓組織成員自行和客戶進行溝通達成共識，不再提出決策性的建議；而成員也能夠透過參與共同決策的過程，提出對工作設定上的意見與建議，最後依據共同決策的結果執行工作。
開發模式轉換的不適應	透過「強化組織對敏捷模式的了解」之戰術目標，組織成員在接受組織規劃之敏捷軟體開發模式之培訓(Training)、輔導(Coaching)和指導(mentor)過程後，對於敏捷軟體開發模式有更具體的了解。

另外，我們將本次研究所發現的敏捷發展轉型時所遭遇之限制、障礙、核心問題與因果關係等與現有文獻加以分析與比較。如本文的 2.2 節所述，在文獻上有關敏捷轉型過程的探討主要是以(Gandomani & Nafchi 2015a; Gandomani & Nafchi 2015b; Gandomani et al. 2015c; Gandomani et al. 2016)為主，故我們綜合 Gandomani 等人的系列研究與本研究發現，並針對認知與實作落差、人員抗拒、組織文化、團隊合作與心態等方面加以比較與分析如下表所示：

表 9 敏捷發展轉型時所遭遇的限制、障礙、核心問題與因果關係之差異分析

	Gandomani 等人的研究發現	本個案發現
認知與實作落差	參與敏捷轉型的人員往往缺乏對於如何運作的正確認知以及不瞭解其與傳統作法的差異下即直接套用。其結果往往反映在開發團隊的執行效率以及	透過外部專家協助團隊進行敏捷模式訓練，雖對其模式有所認識，但在實際運用時，本研究仍發現執行上有困難之處。此時輔以 CBAT 模式可協助團隊成員發現轉型當中造

	士氣上，並回向造成對敏捷開發的誤解。	成困難的核心問題，瞭解認知與實作之差異，進而發展出改善行動，於做中學、以解決問題。
抵抗變革的阻力	人們傾向安於現狀的習性，使得執行人員都習慣於己身所熟悉的角色或活動而抗拒改變。	在適當的激勵政策配合下，改善行動會在一段時間後有所改變。另外，參與轉型過程的人員往往不喜歡重覆的嘗試錯誤。
文化的衝擊	敏捷軟體開發模式強調團隊合作、集體決策、成員互信、積極溝通以獲得團隊最大利益。但在許多情況下人們首先會保護自己，而不是考慮到團隊的利益。	在實施轉換過程中，確實有部份成員發生保護自己的現象。我們發現此狀並非單靠改善行動即有效果，而是要透過組織或管理者給予適當的協助與授權，讓團隊成員不害怕犯錯，並勇於反映各種問題。
缺乏合作習慣	敏捷發展強調團隊合作而非個人。這跟傳統軟體開發模式強調專業分工的觀念差異甚大。其進一步指出，習於分工的人易處於自我天地內而無足夠信心去參加集體作業和群體決策。	基於左述之上，我們發現的策略構想中的「培養組織成員發展化繁為簡、循序漸進的協同作業方式」，與其延伸出的 22 項改善行動等能進一步協助團隊成員開始建立集體作業和群體決策的合作習慣。
錯誤的心態	對於敏捷軟體開發模式和其價值觀的錯誤心態是進行敏捷轉型過程的重大障礙之一。習於傳統開發的人認為指揮與控制是優秀的管理工作，因而產生自我矮化與防禦的心態。	管理者透過指揮與控制以提高專案能見度並管控風險。若管理者能以幫助與互惠的角度讓團隊放下防禦心態，雙方即能正面互動，並提高能見度。此外暢通的回饋意見管道以及開放的態度、加上工作檢討時對事不對人的態度，是管理者與團隊須共同注意。

三、 研究限制

本研究有以下數項研究限制。首先，本研究是以系統整合商之軟體專案開發團隊為探討對象與範圍，且由於個案組織配合敏捷發展的作法而將開發團隊規模設定為 3-11 人；然而若團隊人數不在此範圍內或是不同的系統型態之開發，或許會有不同之研究結果。再則，本研究之貢獻在於所創建的轉換歷程模型、並且完整提供實際發生的實例以供其他企業或研究者參考。雖如此，歷程與實例內容如不良效應、衝突與核心問題等係為個案公司之所有，其仍會因不同企業而有所差異。此外，由於本研究為探索型之個案研究且針對單一企業以深入與詳細探討模式內容與運作過程，因此研究結果較無法一般化。且依據本研究引用之限制理論的思考流程來推導，若行動有衝突，在其脈絡上的策略甚至目標也可能有衝突；透過本研究探討的衝突是在行動與執行層面—即專注於受資源、人員與工作分配上的行動限制下之衝突，故僅針對執行面加以探討。在未來，本研究建議可進一步延伸實證 CBAT 與組織特色的關聯，並擴大與多家個案公司進行整合比較。

最後，在本次研究中，個案組織為首次從事敏捷轉型，也是首次採用限制理論的思考流程做為改善敏捷轉型過程的方法。由於是離型研究與研究範圍限制，本研究僅探討轉型過程與所導入模式的成效，對於成效的維持或許可運用 CMMI(capability maturity model integration)的永續化設計(Institutionalization design)與能力度分級概念(Capability)來進一步永續化所建立的導入成效。

陸、 結論

本研究運用限制理論來建立一個敏捷轉換過程模式並實施於某系統整合商之軟體開發部門。隨研究的結束，我們獲得以下結論。首先，個案團隊從傳統開發模式轉型至

敏捷開發模式時延伸出許多執行層面的限制或障礙。透過本研究提出的 CBAT 模式，個案團隊分析所遭遇的各種障礙與核心問題、建立因果關係；並以敏捷開發最佳實務為基礎來發展出協助該團隊排除限制或障礙的改善行動。再則，敏捷轉型用於軟體產業，在本質上屬於工作流程與觀念的改變。對於軟體企業而言，開發團隊為其命脈；而改動開發流程等於是改變了組織人員根深蒂固的工作慣性。因此在轉型之前，組織可運用本模式來具體建立行動方針—包括對技術與客戶要求迅速做出反應的新思維、及將此反應迅速轉化為行動的實踐力。

任何工作方式的轉變，都會影響組織原有的效率，甚至造成組織士氣低迷、成員抵制。在本研究個案中，個案組織原習慣於詳盡計劃，然後各就各位、依序進行的傳統開發模式。在透過研究模式的協助下，組織成員改變成化繁為簡、循序漸進的協同作業方式，同時針對其不良效應能建立戰術目標、中間目標、以及相應之改善行動，以協助順利完成轉型。最後，在軟體工程實務與研究領域中，敏捷開發是一項重要之課題。就敏捷轉型來說，在目前相關研究領域並沒有提出標準、模式之情況下，本研究所提出的轉換模式為一個創新的嘗試，其改善過程與成果或許可提供其他研究進一步發展之參考。

參考文獻

- 盧昆宏 (2004)，『整合性生產決策模式之建構與實證』，第一屆台灣作業研究學會研討會暨 2004 年科技與管理研討會，台北，台灣，11 月 12 日，頁 886-894。
- 李榮貴、張盛鴻 (2005)，*TOC 限制理論：從有限走向無限*，中國生產力中心。
- 岸良欲司 (2009)，*問題不能拆開來看*，張凌虛、鄭曉蘭譯，時報文化，台北。
- 高義展 (2010)，應用限制理論建構適性化高雄都會區的成人數位學習系統，*城市學學刊*。1(1)，頁 85-113。
- 鄭勝璋、賴昱達 (2013)，『限制理論思維應用於整合型郵件系統之建構—以核能研究所為例』，*TANET 2013 臺灣網際網路研討會*，台中，台灣，10 月 23 日，頁 1-6。
- Abdelnour Nocera, J. and Sharp, H. (2012), 'Understanding conflicts in agile adoption through technological frames', *Int. J. Sociotechnol. Knowl.* Vol. 4, No. 2, pp.29-45.
- Abrahamsson, P., Conboy, K. and Wang, X. (2009), 'Lots done, more to do: the current state of agile systems development research', *Euro. J. of Info. Syst.* Vol.18, No. 4, pp. 281-284.
- Acuna, S.T., Juristo, N. and Moreno, A.M. (2006), 'Emphasizing human capabilities in software development', *IEEE software*, Vol.23, No. 2, pp.94-101.
- Armour, F.J., Kaisler, S.H. (2001), 'Enterprise architecture: Agile transition and implementation', *IT professional*, Vol.3, No. 6, pp.30-37.
- Blankenship, J., Bussa, M. and Millett, S. (2011), *Pro Agile .NET Development with Scrum*, Apress.
- Carvalho, F., Azevedo, L.G. (2013), 'Service Agile Development Using XP', *7th IEEE International Symposium on SOSE*, Redwood, CA, USA, March 25-28, pp.254-259.
- Chen, C.Y. and Chong, P. (2011), 'Software engineering education: a study on conducting collaborative senior project development', *J. Syst. Soft.*, Vol. 84, Issue 3, pp.479-491.

- Dahmann, J., Gregorio, D. and Modigliani, P. (2013), 'Systems engineering processes for agile software development', *2013 IEEE Int'l Conf. on Systems*, Orlando, FL, USA, April 15-18, pp.351-355
- Darke, P., Shanks, G. and Broadbent, M. (1998), 'Successfully completing case study research: combining rigor, relevance and pragmatism', *Info. Syst. J.*, Vol.8, No. 4, pp.273-289
- Denning, S., (2012). 'How Agile can transform manufacturing: the case of Wikispeed', *Strategy & Leadership*, Vol. 40, No. 6, pp.22-28 .
- Dettmer, H.W., (1997), *Goldratt's Theory of Constraints – A System Approach to Continuous Improvement*. ASQC Quality Press, Milwaukee, WI, USA.
- Dettmer, H.W., (2000), *Constraints Management*, Quality America Inc., Tucson, AZ USA.
- Dybå, T. and Dingsøyr, T. (2008), 'Empirical studies of agile software development: a systematic review', *Information & Software Technology*, Vol. 50, No.9-10, pp.833-859.
- Gandomani, T.J., Zulzalil, H., Ghani, A.A., Sultan, A.B.M. and Sharif, K.Y. (2014). 'How human aspects impress Agile software development transition and adoption.' *International Journal of Software Engineering and its Applications*, Vol.8, No.1 (2014), pp.129-148
- Gandomani, T.J. and Nafchi, M.Z. (2015a), 'Identifying some important success factors in adopting agile software development practices', *J. Syst. Soft.*, Vol.82, No.11, pp.1869-1890.
- Gandomani, T.J. and Nafchi, M.Z. (2015b), 'An empirically-developed framework for agile transition and adoption: A Grounded theory approach', *J. Syst. Soft.*, Vol.107, No. C, pp.204-219.
- Gandomani, T.J., Zulzalil, H., Ghani, A.A.A., Sultan, A.B.M. and Parizi, R.M. (2015c), 'The impact of inadequate and dysfunctional training on agile transformation process: A grounded theory study', *Info. & Soft. Tech.*, Vol.57, pp.295-309.
- Gandomani, T.J. and Nafchi, M.Z. (2016), 'Agile transition and adoption human-related challenges and issues: a grounded theory approach', *Computers in Human Behavior*, Vol.62, pp.257-266.
- Goldratt, E.M. and Cox, J. (2016), *The Goal - A Process of Ongoing Improvement*, Routledge, London, UK.
- Goldratt, E.M. (1999). *Theory of constraints*, North River Press. Great Barrington, MA, USA.
- Goldratt, E.M., (1994), *It's Not Luck*, North River Press, Great Barrington, MA, USA.
- Goldratt, E.M., (1990), *The Haystack Syndrome: Sifting Information Out of the Data Ocean*, North River Press, Croton-on-Hudson, NY, USA.
- Guangyuan, S., Chang S.H. and Zhang, W. (2010), 'Challenges of Beijing road transportation system: An extended application of TOC think process', in *Advances in Human Factors, Ergonomics & Safety in Manufacturing & Service Industries*, CRC Press, Boca Raton, Florida, USA, pp.264-274.

- Kim, S., Mabin, V.J. and Davies, J. (2008), 'The theory of constraints thinking processes: retrospect and prospect', *International Journal of Operations & Production Management*, Vol. 28 No. 2, pp.155-184.
- Leau, Y.B., Loo, W.K., Tham, W.Y. and Tan, S.F. (2012), 'Software development lifecycle agile vs. traditional approaches', *The 2nd International Conference on Information and Network Technology*, Chennai, India, April 28-29, pp.162-167.
- Nikiforova, O., Nikulsins, V. and Sukovskis, U. (2009), 'Integration of MDA framework into the model of traditional software development', *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems*, Vol. 187, No. 5, pp.229-239.
- Noreen, E., Smith, D. and Mackey, J. (1995), *The Theory of Constraints and Its Implication for Management Accounting*, North River Press, Great Barrington, MA, USA.
- Olsson, H.H., Alahyari, H. and Bosch, J. (2012), 'Climbing the stairway to heaven --A multiple case study exploring barriers in the transition from agile development towards continuous deployment of software', *The 38th IEEE Euromicro Conf. on Software Engineering and Advanced Applications*, Cesme, Turkey, Sept. 5-8, pp.392-399.
- Pikkarainen, M., Salo, O., Kuusela, R. and Abrahamsson, P. (2012), 'Strengths and barriers behind the successful agile deployment-insights from the three software intensive companies', *Emp. Softw. Eng.* Vol. 17, No.6, pp.675-702.
- Rahman, S. (1998), 'Theory of constraints a review of the philosophy and its applications', *International Journal of Operations & Production Management*, Vol.18, No. 4, pp.336-355.
- Rahman, S. (2002), 'The theory of constraints thinking process approach to developing strategies in supply chains', *International Journal of Physical Distribution & Logistics Management*, Vol. 32, No. 10, pp.809-828.
- Rico, D.F. (2008), 'What is the ROI of agile vs. traditional methods? An analysis of XP, TDD, pair programming, and scrum', *Tick IT International*, Vol. 10, No. 4, pp.9-18.
- Sander, B. (2014), 'Challenges in the transition from waterfall to scrum- A case study at port base', *20th Twente Student Conf. on IT*, Enschede, Netherlands, June 23-25, pp.1-10.
- Savolainen, J., Kuusela, J. and Vilavaara, A. (2010), 'Transition to agile development-rediscovery of important requirements engineering practices', *The 18th IEEE Int'l Req. Eng. Conf.*, Sydney, Australia, Sept 27-30. pp.289-294.
- Schova, Z., (2010), 'Hidden gap in agile methods adoption', *The 9th International Conference on New Trends in Software Methodologies, Tools and Techniques*, Yokohama, Japan, Sept 29-October 1, pp.346-356.
- Schwaber, K. and Sutherland, J. (2011), *The Scrum Guide*, available at <https://www.scrumalliance.org/community/articles/2011> (accessed 25 May 2016).
- Saxena, P. and Kaushik, M. (2013), 'Software development: techniques and methodologies', *International Journal of Software & Hardware Research in Engineering* Vol.1, No.3, pp.48-52.

- Seaman, C. (1999), 'Qualitative methods in empirical software engineering', *IEEE Transactions on Software Engineering*, Vol.25, No. 4, pp.557-572
- Srinivasan, J. and Lundqvist, K. (2010), 'Agile in India: challenges and lessons learned', *The 3rd India Software Engineering Conference*, Mysore, India, Feb 25-27, pp.125-130.
- Thangasamy, S. (2012), 'Lessons learned in transforming from traditional to agile development', *Journal of Computer Science*, Vol.8, pp.389-392.
- Tolfo, C., Wazlawick, R.S., Ferreira, M.G.G. and Forcellini, F.A. (2011), 'Agile methods and organizational culture: reflections about cultural levels', *Journal of Software: Evolution and Process*, Vol.23, No. 6, pp.423-441.
- Umble, M., Umble, E. and Deylen, L.V. (2001), 'Integrating enterprise resources planning and theory of constraints: A case study', *Production & Inventory Management*, Vol.42, No. 2, pp.43-48.
- Vijayasarathy, L. and Butler, C. (2016), 'Choice of software development methodologies-do project, team and organizational characteristics matter?' *IEEE Soft.*, Vol. 33, No. 5, pp.86-94.
- Watson, K.J., Blackstone, J.H. and Gardiner, S.C. (2007), 'The evolution of a management philosophy: the theory of constraints', *J. Oper. Mgt.*, Vol.25, No. 2, pp.387-402.
- Wysocki, R.K. (2013), *Effective Project Management -Traditional, Agile, Extreme*, John Wiley & Sons, New York, NY, USA.
- Yin, R.K. (2013), *Case study Research: Design & Methods*, Sage Thousand Oaks, CA, USA.
- Yin, R.K. (2011), *Applications of Case Study Research*, Sage, Thousand Oaks, CA, USA.
- Youngman, K.J. (2003), *A Guide to Implementing the Theory of Constraints*, available at <http://www.dbrmfg.co.nz/Thinking%20Process%20CRT.htm> (accessed 25 May 2016).