序

make 公用程式是一個令人滿意的僕人，它總是隨伺在側、總是給人方便。就像是對你而言不可或缺的伙伴，make 起初是一個未能充分發揮潛力的職員，你將一些臨時的工作丟給它做，然後它漸漸掌控整個企業（這是許多小說和電影中常見的情節）。

就在每個專案都被我改成以 make 為基礎之際，我的老闆也就是本書第一版的作者 Steve Talbott 注意到了我的狂熱行為，邀請我為本書撰寫第二版。在我的人生歷練中這的確帶來關鍵性的成長（也是相當大的冒險），我因而進入 O'Reilly 的美妙世界，但是我們實在不知道第二版能在市場上存活多久？這個版本總共存活了 13 年。

下面摘要列出了自本書第二版發行以來 make 的發展：

- 本書第二版發行當時，GNU 版的 make 已經是大多數程式員的選擇，儼然成為業界的標準。
- GNU/Linux 的興起讓 GNU 編譯器工具鏈的使用更為廣泛，其中包括 GNU 版的 make。舉例來說，Linux 核心本身就相當倚重 GNU 版 make 所提供的延伸，正如本書第 11 章所描述的那樣。
- 以 BSD 的變體（Darwin）為內核（core）的 Mac OS X 繼續朝向 GNU 工具鏈以及 GNU 版的 make 邁進。
- 有越來越多的訣竅被發現，讓你得以健全、無錯誤、可移植及具彈性地使用 make。大型專案上常見的問題已經在社群中逐漸形成標準的解決方案。已經到了將這些口耳相傳的解決方案立言成書的時候了，這就是本書要做的事情。
- 尤其是，出現了將 make 應用在 C++ 和 Java 語言的新需求，make 發明當時這些語言尚不存在。以下的例子可以說明 make 出現的年代：最初的 make 具有兩項特殊的功能，一個是支援 FORTRAN 的兩個變體（這個遺跡還在！），一個是與 SCCS 不怎麼有用的整合。

即使有諸多限制，make 仍舊是所有電腦開發專案中最重要的工具，恐怕當年對 make 的批評或洞察都沒預料到這一點。這十三年來，有意取代 make 的新工具如雨後春筍般不斷推陳出新，它們都想超越 make 設計上的限制，其中也不乏眾多值得讚賞的巧思。不過，make 的簡單易用仍使它立於不敗之地。

當我觀察到這些趨勢之後，十年前為本書撰寫新版的想法又湧上心頭。不過我意識到自己的經驗淺薄並不足以擔負此重責大任。最後，找到了 Robert Mecklenburg，他的專業能力受到 O'Reilly 所有同仁的肯定。能將這本書交由他全權負責實在太好了，我則退居幕後成為本書的編輯，這讓我的名字又可以出現在本書的版權頁上。

Robert 對他的博士學位保持低調，不過他思考的深度和精確度卻在本書展露無遺。或許更重要的是，他把焦點擺在實用性上。他會告訴你如何執行得更有效率，以及讓你知道如何進行除錯。

這是一個重大的時刻：O'Reilly 最早期和最持久的一本書出新版了。坐下來，瞭解一下，一個保守的小工具何以有此能耐讓幾乎每個專案都要使用它。不要安於老舊而無法令人滿意的 *makefile* — 開發你的潛力就在今朝。

-Andy Oram
Editor, O'Reilly Media
August 19, 2004