# [[Học nghề] Giới thiệu](https://codegym.vn/blog/2018/07/26/hoc-nghe-gioi-thieu/#respond)



*Học nghề tạo nên sự khác biệt bởi vì nó thấm nhuần một niềm đam mê trọn đời để làm chủ nghề thủ công, và khi đó, mở đường cho những người thợ học việc trở thành những bậc thầy chế tác.*

*-Pete McBreen, Nghề Thủ công Phần mềm-*

Cuốn sách này được viết dành cho những người học nghề thủ công phần mềm – những người có niềm yêu thích với nghề phát triển phần mềm và mong muốn tiến xa hơn nữa, nhưng lại đang cần một chút chỉ dẫn. Không quan trọng bạn đã có trong tay một tấm bằng tốt nghiệp ngành Khoa học Máy Tính danh giá hay hoàn toàn là tự học, thì bạn đều cảm nhận thấy rằng ngoài kia có rất nhiều những nhà phát triển phần mềm vĩ đại, và bạn ao ước đạt được tới đẳng cấp giống như của họ. Cuốn sách này được viết hoàn toàn dành cho bạn – không phải cho ông chủ, không phải cho trưởng nhóm, cũng không phải cho giáo sư của bạn. Có nhiều cuốn sách khác phù hợp hơn cho những người ở vị trí đó, nhưng cuốn sách này xin được dành tặng cho tất cả những người đang chập chững bước vào cuộc hành trình.

Trong khi viết cuốn sách này, chúng tôi chịu ảnh hưởng sâu sắc từ các nguyên tắc và tư tưởng của nghề thủ công phần mềm (software craftmanship). Thật vậy, tiêu đề của cuốn sách cũng đã phản ánh điều đó. Khái niệm về học việc vốn được dựa trên mô hình làm nghề thủ công thời trung cổ, nơi các nhóm nhỏ học viên làm việc cùng nhau và những thợ học việc thiếu kinh nghiệm sẽ chạy việc vặt giúp cho các thợ bạn và thợ cả. Một trong những mục tiêu ở cuốn sách này là truyền cảm hứng cho những người yêu thích kiến tạo phần mềm tập trung vào kỹ thuật thủ công của họ. Cuộc hành trình đang được bàn đến tại đây sẽ bắt đầu với “Hello world!”, nhưng kết thúc của nó ở đâu? Hầu hết là nó kết thúc với một đề bạt lên quản lý cấp trung. Nhiều tài năng đã chớp lấy cơ hội đó không do dự và nhận ra sau vài năm rằng họ muốn rời khỏi vị trí mình không còn hứng thú. Nhưng đối với những người có niềm đam mê phát triển phần mềm và yêu thích quá trình học hỏi, phát triển phần mềm là một nghề nghiệp có thể làm tới cuối đời, và với họ, chuyến hành trình đó có khi quá đỗi tuyệt vời.

Nhưng trước khi chuyến đi có thể bắt đầu, chúng tôi sẽ kể cho bạn câu chuyện của Dave và đưa ra một vài định nghĩa. Câu chuyện của anh cho thấy cách mà các khuôn mẫu học việc có thể kết hợp lại với nhau, để tạo ra một sức mạnh tuyệt vời trong việc phát triển bản thân bạn như là một nhà phát triển phần mềm. Mặt khác, các định nghĩa được đưa ra ở đây là những khái niệm làm nền tảng cơ sở định nghĩa nghề thủ công phần mềm, và cả để trả lời một số câu hỏi thường gặp về những khái niệm đó, mà chúng tôi đã cố gắng gom tập lại.

### ***Câu chuyện của Dave (được kể kèm theo một vài khuôn mẫu học việc)***

*Ngôn ngữ lập trình đầu tiên của tôi là Perl, nhưng đó là sau hai lần thử học lập trình thất bại. Lần đầu là khi tôi 12 tuổi, tôi đã thử tự học BASIC trên chiếc Apple IIe sau khi xem bộ phim Tron và được truyền cảm hứng rằng có cả một thế giới tồn tại bên trong chiếc máy tính của mình. Tôi đã mua một cuốn sách hướng dẫn BASIC được xuất bản bởi Apple, nhưng tôi không thể tìm được cách nào làm ra một thứ gì hấp dẫn với ngôn ngữ đó. Cuối cùng khi nhận ra rằng mình chẳng thể làm ra gì khác ngoài mấy trò chơi từ các chữ cái, tôi đã bỏ cuộc. Đến lúc 25 tuổi, tôi lại thử tự học Java bằng cách đọc một cuốn sách cho những tay “thế mà đần”, và từ từ thực hành thông qua các bài tập, tạo các ứng dụng Java nhỏ. Nhưng rồi tôi cảm thấy như là mình đần thật, và cũng phải từ bỏ bởi vì mọi thứ thật sự trông quá khó khăn. Đã không có gì thay đổi cho tận tới lúc 26 tuổi, tôi tìm được hai người hướng dẫn và họ kéo tôi vào với ngôn ngữ đầu tiên của mình. Tôi gặp họ tại Edventions LLC ở Skokie, Illinois, trong thời kỳ đỉnh điểm của đợt bong bóng dot-com. Irv Shapiro, người sáng lập công ty, sau khi biết được rằng tôi muốn trở thành một lập trình viên (ông ấy vốn thuê tôi để làm nhân viên chỉnh sửa nội dung trực tuyến) đã yêu cầu tôi học Perl. Ông ấy ném cuốn Programming Perl (http://oreilly.com/catalog/9780596000271/) xuống bàn tôi và mô tả về một trò chơi xếp gạch mà tôi có thể xây dựng như là một bài luyện tập. Một vài ngày sau, tôi đã ngốn hết cuốn Programming Perl, mặc dù nó hơi nặng đô cho tay mơ như tôi. Thế là tôi tiếp tục bằng cách đọc thêm một cuốn, dễ tiếp thu hơn, từ danh sách đọc của mình – cuốn Visual QuickStart Guide. Một người khác nữa đã hướng dẫn tôi lúc ấy là Steve Bunes, CTO của Edventions, người thường xuyên trực tiếp chỉ tận tay, dạy cho tôi một số kỹ năng tầm lỗi lợi hại mà cho đến giờ tôi vẫn còn đang sử dụng. Mô hình khó nhằn nhất được áp dụng để cho ra phiên bản đầu tiên của trò xếp gạch đã phơi bày sự non yếu của tôi so với các lập trình viên Perl giàu kinh nghiệm cũng như mấy tay quản trị hệ thống ngồi làm việc quanh đó. Nhưng nó cũng đáng bớt chút sĩ diện thôi bởi vì họ đã chỉ cho tôi một ít về shebang và phân quyền trên Unix, phần nào giúp tôi nhanh chóng hoàn thành được trò chơi xếp gạch và làm Irv cùng Steve thực sự bất ngờ.*

*2 năm sau, tôi đã có ý định tìm kiếm thêm các cơ hội để phát triển sự nghiệp ra ngoài ngôn ngữ Perl yêu thích (nhưng đang dần không được đánh giá cao) của mình và trau dồi thêm nhiều kĩ năng mới. Tôi đã mở rộng băng thông của mình bằng cách đắm mình vào lập trình cực hạn (XP) và phát triển linh hoạt Agile, vốn đang nổi lên rất nhanh vào thời điểm đó. Tôi đã dành vài ngày tham gia hội nghị XP/Agile Universe, trực tiếp hóng theo từng chút thông tin nóng hổi. Gặp gỡ và lắng nghe chia sẻ của những người như Ron Jeffries, Martin Fowler, “chú” Bob Martin, Alistair Cockburn, và Kent Beck là một trải nghiệm tuyệt vời, và tôi đã chính thức ra về từ hội nghị như một tín đồ của lập trình hướng đối tượng và lập trình cực hạn. Tôi phát hiện ra cuốn Refactoring to Patterns lúc đó đang được viết bởi Joshua Kerievsky nghe có vẻ thú vị, thế là tôi đi tìm một người bạn cùng chí hướng để cùng nghiên cứu nó. Nhưng rồi chúng tôi sớm nhận ra bản thân đang đốt cháy giai đoạn, khi mà tôi thậm chí còn đang không biết tái cấu trúc hay mẫu thiết kế là gì. Vậy nên tôi đã tìm các cuốn sách khác phù hợp với trình độ của bản thân hơn. Rồi định mệnh dẫn dắt tôi đến với Object-Oriented Software Construction và A Pattern Language. Nhưng tôi vẫn muốn có ngày quay về nghiên cứu tiếp Refactoring to Patterns, do đó tôi đã cất lại nó vào danh sách đọc của mình.*

*Tôi bắt đầu học Ruby vào năm 2002, vẫn cùng với người bạn của mình, nhưng tôi không cách nào ứng dụng nó vào công việc của mình cho đến khi có Ruby on Rails. Năm 2005 tôi tiếp tục chọn Ruby với nỗ lực tìm kiếm cơ hội làm việc với nó hằng ngày Tôi đã sử dụng Ruby để xây dựng một game xếp gạch, nhưng nhận thấy bản thân lại đang suy nghĩ quá giống với một lập trình viên Perl. Bất cứ lập trình viên nào đã thành thạo ngôn ngữ đầu tiên của mình cũng sẽ rất dễ bị cám dỗ quay lại với những tiêu chuẩn và thành kiến của ngôn ngữ đó trong lúc học một ngôn ngữ mới. Ruby nổi tiếng với sự tạo nhã và đơn giản, vậy mà những dòng code tôi viết ra thì thật là xấu xí và vụng về, tôi nghĩ chắc hẳn tôi đã làm sai điều gì đó. Tôi đưa ra quyết định thật nghiêm túc về việc đeo được chiếc đai trắng, vậy nên tôi đã bỏ qua hết những hiểu biết vốn có của mình về Perl và tập trung đào sâu vào các tài liệu Ruby. Ngay sau đó, tôi đã tìm thấy những gì tôi cần và tái cấu trục lại toàn bộ phần mã lộn xộn của mình về thành một lời gọi phương thức gọi đẹp đẽ, chuẩn chỉnh (String#scan, dành cho những tín đồ Ruby đang đọc đến đây và tò mò). Để giúp những thông tin mới được giữ lại trong đầu, tôi quyết định giãi bày hết thiếu sót của bản thân bằng cách ghi chép lại những gì đã học được lên trang web của tôi cho tất cả mọi người cùng xem.*

### **Nghề Thủ công Phần mềm là gì?**

Các định nghĩa theo từ điển của các từ đơn giản như thủ công, nghề thủ công, học việc, thợ bạn, và thợ cả là không đủ với nhu cầu của chúng ta ở cuốn sách này. Chúng thường định nghĩa lòng vòng (với “thủ công” được định nghĩa là kĩ năng mà một thợ thủ công có, “thợ thủ công” lại được định nghĩa là một người làm nghề thủ công, và “nghề thủ công” thì lại được giải nghĩa là “công việc đặc trưng của các người thợ cùng nhau làm thủ công truyền thống”), hiếm khi được ghi lại trong lịch sử hệ thống phường nghề của các quốc gia, và thường dùng để mô tả chung chung cho bất cứ sản phẩm nào được tạo ra một cách khéo léo. Nói tóm lại, những định nghĩa này không giải thích rõ được bất cứ điều gì và lại còn đánh đồng tất cả mọi thứ với nhau. Chúng ta cần nhiều hơn như thế.

Google liệt kê 61.800 tham chiếu tới cụm từ “nghề thủ công phần mềm”, nhưng chỉ một vài trong số chúng cung cấp những định nghĩa hữu ích có thể sử dụng được cho những ai đang tìm định hướng nghề nghiệp. Đáng buồn thay, rất nhiều bài được viết bởi những lập trình viên đầy thiện chí, những người nhận ra rằng có thứ gì đó hữu ích đằng sau tập khái niệm nhập nhằng này, nhưng không thể nào cắt nghĩa chính xác được.

Cuốn sách Software Craftsmanship của Pete McBreen là một nỗ lực để đưa ra một tuyên ngôn mới về một cách tiếp cận khác cho công việc phát triển phần mềm, dành cho những người không cho rằng phát triển phần mềm là một ngành kĩ thuật hay khoa học khô cứng. Nhưng ngay cả công trình đầy cảm hứng của McBreen cũng có thiếu sót. Nó không phân biệt được nghề thủ công phần mềm đang được vận hành trong thực tiễn với nghề thủ công phần mềm mà ông ấy đang hướng tới. Ông cũng không làm tách biệt rõ ràng tầm nhìn của mình với các khái niệm thời trung cổ coi thủ công là các ngành công nghiệp có tay nghề cao được giám sát bởi các phường hội bí mật. Ông ấy đã gặp lỗi khi định nghĩa nghề thủ công phần mềm đối nghịch với ngành công nghiệp phần mềm, và yêu cầu độc giả của mình phải chọn lựa giữa 2 khái niệm đó. Chúng tôi nghĩ rằng mô hình thủ công có thể được định nghĩa theo một cách tích cực mà không cần phải chối bỏ đi giá trị nào đấy của những người đang cố xây dưng nên một khuôn phép của ngành công nghiệp phần mềm hiện nay.

Mô hình mà chúng tôi lấy cảm hứng đã từng rất được phổ biến tại Châu Âu thời trung cổ cho đến khi cách mạng công nghiệp bắt đầu (The Craftsman, trang 52–80). Trong mô hình đó, các phường nghề sẽ quản lý các thợ cả và các thợ cả lại quản lý chính những người

quyết định tuyệt đối trong đó. Ngay dưới họ trong hệ thống phân cấp nghiêm ngặt này là làm việc và sinh sống trong phân xưởng của mình. Thợ cả sở hữu xưởng nghề và có quyền những người thợ bạn. Họ thường là thợ thủ công chưa tới được khả năng làm ra những “kiệt tác” để chứng tỏ tay nghề của mình đã đạt tới tầm cỡ của thợ cả.

Thợ bạn là những người nay đây mai đó, và chính họ là những người đã góp công mang các kỹ thuật mới từ vùng này sang vùng khác. Cùng với đó, họ sẽ giám sát các hoạt động hàng ngày của thợ học việc. Các thợ học việc sẽ làm việc cho 1 thợ cả trong nhiều năm đến khi họ tốt nghiệp để trở thành thợ bạn, bằng cách chứng minh bản thân đã tiếp thu được các kỹ năng và giá trị cơ bản cho tay nghề thủ công của họ. Người không nằm trong hệ thống cấp bậc của phường nghề không có quyền được thực hành tay nghề của họ.

Bạn có thể tưởng tượng rằng, hệ thống này đã được mở rộng tới mức làm dụng và phi thực tế, nếu không muốn nói là bất hợp pháp, ở trong thế giới ngày nay. Chúng tôi không mong muốn đi vào “vết xe đổ”, nếu bê nguyên mô hình đó tới xã hội hiện đại bây giờ. Thay vào đấy, chúng tôi tin rằng hoàn toàn có thể bác đi mô hình cổ xưa phi thực tế đó, và ủng hộ một cách làm hiện đại, nơi chúng ta có thể tự do cải tiến chứ không chỉ bắt chước lại những gì đã qua.

Một bài học chúng tôi rút ra được từ phong trào phát triển linh hoạt Agile đó là chỉ làm việc theo chỉ dẫn của người khác thì không thể tạo ra được sự thay đổi lâu dài và bền vững. Khi người được bạn hướng dẫn gặp phải một tình huống không được chuẩn bị sẵn trước đó, họ sẽ lạc lối. Tuy nhiên, nếu cũng những người đó tiếp thu được những giá trị cơ bản của các nguyên tắc, họ sẽ có thể tìm ra được các nguyên tắc mới phù hợp với từng hoàn cảnh khác nhau. Mục tiêu của chúng tôi ở đây không chỉ đơn giản là đưa cho mọi người một cuốn sách đầy quy tắc mà còn cho họ khả năng tạo ra cách giải quyết mới cho các hoàn cảnh mới, từ đó thúc đẩy sự rèn luyện của quá trình phát triển phần mềm tiến lên phía trước.

Viễn cảnh về nghề thủ công phần mềm của chúng tôi phần nào là sự chưng cất các giá trị của những cá nhân có tay nghề cao mà chúng tôi đã phỏng vấn cho cuốn sách này, phần nào là bức tranh của một loại cộng đồng chúng tôi đang muốn phác họa nên. Những ý tưởng trong cuốn sách này là điểm bắt đầu cho viễn cảnh đó. Vì vậy, khi chúng ta sử dụng cụm từ thủ công phần mềm, chúng ta đang nói về một cộng đồng thực hành thống nhất và được xác định bởi các giá trị chồng chéo, bao gồm:

* Một sự liên kết với nghiên cứu của Carol Dweck, cái được gọi là “tầm suy nghĩ tăng trưởng”. Điều này đòi hỏi một niềm tin ở bản thân rằng có thể làm mọi thứ tốt hơn và tất cả đều được cải thiện nếu bạn đang chuẩn bị để thực hiện nó. Theo như cô ấy nói, “nỗ lực là điều làm cho bạn thông minh hay tài năng” (Mindset, trang 16), và sự thất bại chỉ là động lực để thử làm lại điều đó trong lần tới. Nó trái ngược với niềm tin rằng tất cả chúng ta đều sinh ra với tài năng có sẵn, và thất bại là dấu hiệu cho thấy bạn không làm được điều gì cả.
* Cần phải luôn luôn thích nghi và thay đổi dựa trên phản hồi bạn nhận được từ thế giới xung quanh bạn. Atul Gawande đề cập đến điều này như là một sự sẵn sàng để “nhận ra những bất cập trong thực tế và tìm ra các giải pháp cải thiện” (Better, trang 257).
* Một khát khao thực dụng chứ không phải là giáo điều. Nó liên quan đến việc sẵn sẵn sàng hạ thấp giá trị của những lý thuyết xuông, hay nói cách khác ủng hộ bắt tay hoàn thiện mọi thứ ngay hôm nay.
* Một niềm tin rằng tốt hơn là chia sẻ những gì chúng ta biết chứ đừng giữ lại nó cho riêng mình. Nó thường liên quan đến sự tham gia của cộng đồng Phần Mềm Mã Nguồn Mở.
* Sẵn sàng thử nghiệm và có thể bị chứng minh là sai. Có nghĩ là chúng ta thử làm một công việc đầy thách thức. Chúng ta thất bại. Và rồi chúng ta sử dụng những bài học từ thất bại đó cho lần trải nghiệm kế tiếp. Như Virginia Postrel đã nói: “Không phải mọi thử nghiệm hay ý tưởng đều tốt cả, nhưng chỉ bằng cách thử những ý tưởng mới, chúng ta mới khám phá ra những cải tiến thực sự. Và luôn có nhiều việc phải làm. Mọi cải tiến đều có thể được cải thiện; mọi ý tưởng mới vẫn có thể tạo ra được sự kết hợp mới hơn”(Future Enemies, trang 59).
* Một sự cống hiến cho cái mà các nhà tâm lý học gọi là một địa vị kiểm soát. Điều này liên quan đến việc kiểm soát và chịu trách nhiệm về số phận của chúng ta thay vì chỉ đợi người khác trả lời.
* Tập trung vào cá nhân chứ không phải nhóm. Đây không phải là một phong trào với các trưởng nhóm và những người đi theo. Thay vào đó, chúng ta là một nhóm người muốn nâng cao kỹ năng của mình và đã phát hiện ra rằng tranh luận, bất đồng quan điểm và bất đồng ý kiến còn hơn là mù quáng với một thẩm quyền tự xưng để quyết định mọi thứ. Chúng tôi tin rằng tất cả chúng ta cùng đi trên hành trình và sự thay đổi mà chúng ta tìm kiếm là ở chính chúng ta chứ không phải trên thế giới. Đây là lý do tại sao cuốn sách này không tập trung vào cách khắc phục đội nhóm của bạn, mà là cách cải thiện kỹ năng của bạn.
* Một cam kết để mọi mọi thứ trở nên công bằng hơn. Chúng tôi không chối bỏ quá trình phát triển phần mềm doanh nghiệp, khoa học máy tính hay kỹ thuật phần mềm (trên thực tế, Ade có từ “kỹ sư” trong chức danh hiện tại của ông). Thay vào đó, chúng tôi nghĩ rằng một hệ thống hữu ích có thể xác định và tiếp thu những ý tưởng tốt nhất từ tất cả các ý kiến đóng góp của cộng đồng phát triển phần mềm.
* Chúng tôi tập trung vào kỹ năng hơn là tập trung vào quá trình làm việc. Đối với chúng tôi, có tay nghề cao quan trọng hơn là sử dụng quá trình “đúng”. Ý tưởng này có một hệ quả. Gawande hỏi: “Liệu y học là nghề thủ công hay là một ngành công nghiệp? Nếu y học là nghề thủ công, thì bạn tập trung vào việc giảng dạy các bác sỹ để có được một kỹ năng thủ công …. Bạn nghiên cứu để tìm các kỹ thuật mới. Bạn chấp nhận rằng không phải ai cũng sẽ làm ra những sản phẩm giống nhau” (Better, trang 192). Ý tưởng này cho thấy rằng không có quá trình hoặc công cụ nào sẽ làm cho mọi người thành công như nhau. Mặc dù ai cũng có thể cải thiện bản thân, thì vẫn luôn có sự khác biệt về trình độ kỹ năng giữa tất cả chúng ta.
* Một niềm hứng thú đặc biệt với cái mà Etienne Wenger gọi là “học tập tùy cảnh”. Đây là một ý tưởng rằng cộng đồng phần mềm đã cố gắng nắm bắt với các khuôn mẫu như Expert in Earshot. Bản chất của nó là chỉ ra cách tốt nhất để học là khi trong cùng một phòng với những người đang sử dụng kỹ năng mà bạn muốn học để hoàn thành mục tiêu của họ.

Hệ thống giá trị này tạo ra các vai trò khác nhau với các trách nhiệm khác nhau, mà sẽ được thảo luận trong các phần sau.

### **Trở thành một Người Học Việc có nghĩa là gì?**

Khi thảo luận về vấn đề này, Marten Gustafson, một trong những người được chúng tôi phỏng vấn, đã thực sự gây ấn tượng khi trả lời rằng:”Tôi đoán về cơ bản nó được hiểu là có một thái độ tin tưởng rằng sẽ luôn có cách tốt hơn/thông minh hơn/nhanh hơn cho những việc mà mình đã và đang làm. Học việc là một trạng thái/quá trình của việc phát triển, tìm những cách làm tốt hơn và tìm kiếm những con người, những công ty hay tình huống buộc bạn phải học được những cách tốt hơn/thông minh hơn/nhanh hơn đó”. Chúng tôi nghĩ rằng có rất nhiều ý nghĩa trong việc sử dụng năng lực bản thân mà không nhờ vào giải pháp được đưa ra của người khác, nó sẽ khiến bạn học được cách đối mặt với các vấn đề. Giống như Dweck viết:”Nó không phải là những gì tự nhiên sẵn có được nuôi dưỡng bởi những thành công dễ dàng và mài mòn bởi những thất bại. Nó không phải thứ mà chúng ta có thể đưa cho mọi người bằng cách khen ngợi sự thông minh của họ. Nó là thứ mà chúng ta trang bị cho họ - bằng cách dạy họ đề cao việc học tập hơn là ỉ lại vào trí thông minh sẵn có, để có thể “nhấm nháp” những thử thách và biết cách biến sai lầm thành những con đường để tiến tới sự tinh thông” (Self theories, trang 4).

Trong khi tình huống thông thường là bạn sẽ ở trong một nhóm nhỏ với những người cùng học việc, thợ bạn, và thợ cả, thì với những gì chúng tôi biết, lại không đòi hỏi một sự sắp xếp như vậy. Sự học việc của bạn phải do bạn làm chủ, và thành quả thiết yếu cuối cùng chính là trách nhiệm của bạn Khi bạn đã xác định được đường lối và tiến trình học việc của bạn rồi, thì sự cởi mở và chất lượng của người hướng dẫn cũng sẽ có ảnh hưởng cuối cùng quyết định tới tay nghề thủ công của bạn.

Học việc là sự khởi đầu của cuộc hành trình trở thành một thợ thủ công phần mềm. Trong khoảng thời gian này, bạn sẽ chủ yếu tập trung vào bản thân bạn, rồi dần dần phát triển tay nghề thủ công của mình. Dù sự quan tâm của bạn học và cả những người lập trình viên có kinh nghiệm sẽ giúp ích cho bạn, thì bạn vẫn phải học cách tự phát triển bản thân, học cách tiếp thu kiến thức. Sự tập trung vào bản thân và nhu cầu để phát triển này chính là bản chất của ý nghĩa trở thành một người học việc.

Một người học việc cuối cùng rồi cũng sẽ trưởng thành từ một vị trí đơn giản ít trách nhiệm cho đến một vị trí cao hơn, tiềm tàng nhiều trách nghiệm hơn, và chúng ta có thể tin tưởng rằng, sự chuyển đổi đó chỉ có thể được thấy rõ ràng nhất trong một thời điểm khi chúng ta nhìn lại tất cả. Vào một lúc nào đó, một thầy thợ cả hay một bác thợ bạn sẽ tiến lại gần và nói với bạn rằng công việc và vai trò của bạn ở đây chẳng khác nào một người thợ lành nghề cả. Khi ấy, bạn phải hiểu rằng điều ấy có nghĩa là lúc trước bạn đã bắt đầu đảm nhận nhiều trách nhiệm hơn, giống như một con ếch được luộc dần dần, chứ không phải bỗng dưng một con ếch sống có thể trở thành một con ếch luộc. Sự chuyển đổi này có thể mất nhiều thời gian đối với một số người. Thậm chí nó còn có thể dài hơn cả sự nghiệp chuyên nghiệp của họ.

### **Trở thành một Người Thợ Bạn có nghĩa là gì?**

Khi bạn tiến bộ qua các giai đoạn của nghề thủ công, bạn giữ lại những gì đã học được của các giai đoạn trước đó. Vì vậy, giống như người học việc, người thợ bạn và thợ cả vẫn sẽ duy trì sự tập trung vào bản thân để học hỏi và phát triển tay nghề của mình. Tuy nhiên, một người thợ bạn lại có thêm một điều nữa cần tập trung. Đó là việc kết nối những người học việc, giao tiếp với mọi người trong và ngoài team của mình. Theo truyền thống, một người thợ bạn sẽ chuyển từ xưởng nghề này qua xưởng nghề khác, trên con đường truyền bá kiến thức giữa vô số các team. Nhưng với quy trình phát triển phần mềm ngày nay, bạn có thể sẽ phải làm việc khá lâu ở trong một team, và điều cần làm khi ấy là tập trung vào việc cải thiện các mối quan hệ trong tập thể đó. Sự tập trung này cuối cùng sẽ mở rộng thành trách nhiệm để hướng dẫn những người xung quanh bạn và để giao tiếp với phần còn lại của ngành công nghiệp.

Những người thợ bạn sẽ ngày càng tập trung vào việc xây dựng những hồ sơ năng lực để triển lãm sự tiến bộ trong sản phẩm của mình; Họ di chuyển giữa nhiều dự án và các thầy thợ cả, tìm cách đa dạng hóa và nâng cao chuyên môn hồ sơ năng lực; Họ mong muốn nâng cao vị thế của mình trong cộng đồng; và cố gắng để sẵn sàng trở thành một thầy thợ cả.

Trách nhiệm của một thợ bạn lớn hơn nhiều so với một người học việc. Cùng với đó sai lầm của họ có thể dẫn tới nhiều mối nguy hiểm hơn. Một số mô hình chúng ta bàn luận sắp tới sẽ không phù hợp với vị trí thợ bạn, tất nhiên rồi vì thợ bạn có trách nhiệm lớn hơn đối với những người thậm chí còn đang coi họ là thầy hướng dẫn.

### **Trở thành một Thầy Thợ Cả có nghĩa là gì?**

Công việc của người thợ cả bao gồm thực hiện tất cả các vai trò của người học việc và thợ bạn cũng như tập trung vào việc phát triển tiếp ngành công nghiệp này. Vấn đề “hoàn thiện kỹ năng và kỹ thuật” (The Creative Habit, trang 167) chỉ là khởi đầu. Sự tinh thông đòi hỏi thu nhận các kỹ năng đó và đồng thời biến nó thành một chiếc kính lúp để có thể khuếch đại trình độ của những người xung quanh theo từng lớp kính. Điều này tạo nền tảng để làm ra những công cụ mới đi thẳng vào bản chất của quá trình phát triển phần mềm, hoặc là cơ hội đào tạo nên một lứa thợ bạn có chất lượng đồng đều, và thậm chí còn vượt trội hơn chính người thầy của họ. Hoặc nó sẽ có thể làm được những điều mà chúng ta chẳng có thể tưởng tượng được. Nói tóm lại, các thầy thợ cả xem việc thu nhận, sử dụng, và chia sẻ kĩ năng cao cấp là phần quan trọng nhất của quá trình trở thành một thợ thủ công phần mềm.

Các định nghĩa về người học việc, thợ bạn, và thầy thợ cả được thảo luận ở đây là những thứ bạn sẽ không thể tìm được ở bất cứ từ điển nào Đó là một cái gì đó mới mẻ. Nhưng chúng tôi tin rằng các giá trị cố hữu trong tầm nhìn của chúng tôi về nghề thủ công phần mềm sẽ giúp bạn thành công như bạn mong muốn, cho dù bạn nắm lấy chúng, trân trọng chúng, từ chối chúng, hoặc đi theo một con đường hoàn toàn khác.

### **Quá trình Học việc là gì?**

*Tình huống học tập cơ bản đó là học hỏi bằng cách giúp việc cho những người thực sự biết điều họ đang làm là gì.*

-Christopher Alexander et al., A Pattern Language, trang 413-

Một hình ảnh thường thấy về học việc được phổ biến trên rất nhiều cuốn sách, trong đó có cả cuốn Fifteen Craftsmen On Their Crafts xuất bản năm 1945 (trang 69), là hình ảnh một cậu bé tuổi vị thành niên dính đầy nhọ trên mặt, đang làm việc tại cửa hàng của một người thợ rèn. Người thợ rèn ấy, một thợ thủ công cộc cằn, giàu kinh nghiệm, rèn đồ của ông ta với sự hỗ trợ của cậu bé. Đôi khi, cậu bé tích cực tham gia vào công việc, đôi khi cậu đang lau dọn cửa hàng, nhưng cặp mắt thì vẫn luôn chăm chú hướng theo người thầy đang làm việc. Thông thường, việc học nghề của cậu bé sẽ kéo dài vài năm và ngoài kiến thức, kinh nghiệm, thức ăn, chỗ ở, cậu còn được nhận một chút lương nữa. Cuối cùng, cậu sẽ thu thập đủ kỹ năng để thực hiện những dự án của riêng mình, thậm chí có thể là rời khỏi người thầy đầu tiên của cậu để nhận một công việc nhiều trách nhiệm hơn trong cửa hàng khác. Khi quá trình học việc của cậu bé kết thúc, cậu có thể kiếm tiền trang trải cuộc sống như một người thợ rèn, dựa vào khả năng của bản thân để kiếm chỗ ở, thức ăn, vật dụng cho mình. Trong cuộc sống hiện đại, quá trình học việc bao gồm một lập trình viên lành nghề và một tay mơ cũng có chút tương đồng so với khuôn mẫu trên kia. Vậy chúng ta đang hiểu quá trình học việc này là gì, và làm sao để vượt qua được sự lặp lại sáo rỗng kia?

Cần phải làm rõ rằng, trong cuốn sách này chúng tôi không hề cố gắng mô tả quá trình học việc lý tưởng của chúng tôi cho một người mới bước đến đến ngành phát triển phần mềm. Nếu chúng tôi viết nó cho các trưởng nhóm và quản lý dự án, chúng tôi sẽ cung cấp các hướng dẫn để tạo ra quá trình học việc hiệu quả, bởi vì họ là những người có đủ năng lực để thực hiện kiểu công việc đó. Nhưng cuốn sách này dành cho những người mới, những người trong các chiến hào đang cố gắng để tìm xem họ cần học thêm điều gì để đạt được các mục tiêu như có một công việc (tốt hơn), hoàn thành dự án của họ, hoặc trở thành một lập trình viên vĩ đại. Vì hầu hết kinh nghiệm của những người mới không giống với “lý tưởng” của quá trình học việc, khái niệm hiện đại về học việc chủ yếu là một khung cảnh: bạn nhận ra rằng bạn vẫn còn đang ở điểm bắt đầu, ngay cả khi bạn đã làm lập trình trong nhiều năm, và bạn đang chuẩn bị để thực hiện các bước đột phá trong sự nghiệp của mình.

Hầu hết mọi người sẽ không có cơ hội được làm việc theo kiểu học việc đàng hoàng, nơi họ được hướng dẫn bởi thợ thủ công phần mềm. Trên thực tế, mọi người sẽ phải mài dũa và tôi luyên quá trình học việc của họ vượt quá những tình huống khó khăn. Họ có thể phải đối mặt với những người quản lý kém cỏi và/hoặc không đủ năng lực, những đồng nghiệp thiếu nhiệt huyết, những deadline vô lý và môi trường làm việc mà coi những lập trình viên mới vào nghề như tay sai, nhét họ vào trong các buồng nhỏ chữ nhật cùng với một chiếc máy tính và đường truyền Internet “tậm tịt”. Tất cả bài học của cuốn sách này được đúc rút từ kinh nghiệm của những người (như chúng ta) đã phải vượt qua những khó khăn đó để đạt đến được cấp độ tiếp theo. Cho đến khi nào ngành công nghiệp của chúng ta có thể lắng nghe những lời khuyên sau đây của Pete McBreen, nếu không thì những người mới vẫn sẽ tiếp tục cần những cuốn sách như thế này để giúp họ tạo ra những cơ hội học tập cho riêng mình.

*Chúng ta cần dành thời gian để đào tạo những lập trình viên học việc bởi vì chúng ta đang phải đối mặt với vấn đề về sự phong phú, hơn là khan hiếm … Hiện này chúng ta có nhiều lập trình viên hơn mức cần thiết, nhưng lại có ít những người giỏi thực sự.*

-Pete McBreen, Nghề Thủ công Phần mềm, trang 93-

Học việc là một cách để học trở thành một nhà phát triển phần mềm chuyên nghiệp. Cụ thể, đó là một cách để hướng bản thân đến giống với những lập trình viên lão luyện nhất mà bạn có thể tìm được. Nó liên quan đến việc tìm kiếm giáo viên giỏi, và tận dụng các cơ hội để học hỏi bằng cách làm việc bên cạnh họ. Đây là bước đầu tiên trên con đường tiến tới trở thành một kiểu chuyên gia phần mềm khác – một người muốn có nhiều thứ hơn là sự thạo việc.

### **Khuôn mẫu Học việc là gì?**

Một khuôn mẫu học việc có mục đích đưa ra hướng dẫn cho một ai đó đang làm việc kiểu thủ công với những cách có thể cải thiện tay nghề của họ. Tất cả các khuôn mẫu của chúng tôi được đúc kết từ kinh nghiệm của chính chúng tôi và của những người mà chúng tôi đã phỏng vấn. Cũng như những bộ sưu tập khuôn mẫu tốt khác, nó không gây hại gì cho bạn cả bởi vì chính những người xung quanh bạn cũng đang sử dụng. Một giá trị khác nữa mà các khuôn mẫu này đem lại đó là sự sáng tạo. Mỗi lần bạn áp dụng chúng bạn sẽ nhận được những kết quả khác nhau, và nếu chúng được sử dụng ở những hoàn cảnh phù hợp, môi trường làm việc của bạn sẽ được cải thiện. Đây không phải là các giải thuật đảm bảo trả về cùng một kết quả trên mỗi lần thực hiện. Thay vào đó, nó là những công cụ giải quyết một loạt vấn đề tương tự nhau và tạo ra những vấn đề mới. Bí quyết là sử dụng phán đoán của bạn để lựa chọn những vấn đề bạn thích.

Cuốn sách này được sắp xếp như một ngôn ngữ kiểu mẫu. Một ngôn ngữ kiểu mẫu là một tập hợp giải pháp cho các vấn đề phổ biến trong một lĩnh vực cụ thể. Ngôn ngữ kiểu mẫu ban đầu được viết bởi Christopher Alexander trong cuốn A Pattern Language, nơi ông đã mô tả hơn 250 mẫu cho việc thiết kế mọi thứ từ nhà bếp, nhà ở cho đến các thành phố và ngay cả xã hội. Ward Cunningham và Kent Beck đã giới thiệu các ngôn ngữ kiểu mẫu này cho ngành công nghiệp phần mềm vào những năm 90, dẫn đến vô số bài báo, sách và thậm chí các hội thảo tập trung vào các khuôn mẫu thiết kế. Tác phẩm nổi tiếng nhất về khuôn mẫu thiết kế phần mềm là cuốn Design Patterns của “The Gang of Four” mặc dù cuốn Refactoring của Martin Fowler là một ví dụ tốt hơn về một ngôn ngữ kiểu mẫu. Nhưng cần phải nói rõ ràng, cuốn sách bạn đang đọc không phải là một cuốn sách về thiết kế phần mềm. Thay vào đó, nó là một cuốn sách về thiết kế sự khởi đầu trong sự nghiệp phát triển phần mềm của bạn và khiến những thứ bạn đang làm trở nên tuyệt vời và ý nghĩa hơn.

### **Những Khuôn mẫu này từ đâu mà có?**

Một trong những nguyên tắc để thiết kế framework phần mềm tốt đó là lấy ra framework từ chính một hệ thống đang hoạt động. Tương tự như thế, những khuôn mẫu thiết kế phần mềm đều được rút ra từ rất nhiều hệ thống làm việc mà đã sử dụng cùng một cách giải quyết cho nhiều vấn đề giống nhau. Cuốn sách này nguyên bản được trích từ những câu chuyện về quá trình học việc của Dave, sau đó được thử nghiệm và bổ sung với những kinh nghiệm của Ade, và cuối cùng được kiểm tra lại trên khoảng 30 học viên có kinh nghiệm từ vài năm lên đến nhiều thập kỷ.

Chúng tôi đã phỏng vấn những con người này để kiểm nghiệm rằng liệu những khuôn mẫu ấy có thực sự là những cách giải quyết tối ưu cho những vấn đề thường gặp hay không, cùng với khai thác các khuôn mẫu khác mà chúng tôi vẫn chưa được biết. Chúng tôi cũng tham gia rất nhiều cuộc hội thảo (PLoP 2005, Agile Atlanta, và ThoughtWorks) để có thể tập trung vào việc nâng cao cấu trúc và tính chính xác của cái mà chúng tôi tin rằng đó là những khuôn mẫu học việc Và cuối cùng, chúng tôi công bố tài liệu này trực tuyến miễn phí để thu thập ý kiến phản hồi từ cộng đồng.

### **Chúng ta sẽ đi đâu từ đây?**

Khi bạn bắt đầu học về các khuôn mẫu, hãy nhớ rằng bạn có quyền được chọn, được kết hợp, và ráp chúng vào tình huống cụ thể của bạn theo vô vàn những cách khác nhau. Bởi vì những khuôn mẫu này đều được viết cho một đối tượng cụ thể trong một hoàn cảnh cụ thể nào đó. Trong vài năm sau, sẽ có một vài khuôn mẫu phù hợp với bạn, nhưng cũng có một số thì sẽ không. Học việc là một thời kỳ trong sự nghiệp của bạn khi mà bạn tập trung vào việc phát triển bản thân hơn bất cứ thứ gì khác. Đây là thời điểm mà bạn nên tạm gác giấc mơ tối ưu thu nhập để phát triển tối đa cơ hội học hỏi của mình. Chính vì thế, nó cũng là một thời điểm mà chúng ta ích kỷ cho bản thân nhiều hơn. Và một khi thời kỳ đó qua đi, những sự ưu tiên của bạn cũng cần phải điều chỉnh lại. Bạn sẽ không còn là một người học việc, cho dù sự thật bạn vẫn luôn phải học hỏi thêm nữa, và bạn phải tập trung đến những thứ khác hơn như là: khách hàng, đồng nghiệp và chính cộng đồng xung quanh bạn.