Tăng trưởng năng lực của tổ chức

Tác giả**: John Vũ**

Người dịch và biên tập: **Ngô Trung Việt**

Hà Nội, 6/2013

Nguồn tư liệu: John Vu, Carnegie Mellon University
http://www.science-technology.vn

Mục lục

1. Lịch sử CMM: tăng trưởng năng lực tổ chức

(Bài giảng của giáo sư John Vũ cho lớp đào tạo giảng viên SEGVN tại CMU năm 2009)

1. Khởi đầu CMM

Vào khoảng thời gian 1980, thập niên những năm 1980, chính phủ Hoa Kì, cụ thể là Bộ quốc phòng có một số chương trình khá lớn. Từ trước tới nay các chương trình quốc phòng có rất nhiều, làm rất nhiều về phần cứng, phần mềm là phụ. Nhưng đến giai đoạn đó phần mềm trở nên quan trọng hơn và có một số kế hoạch bị hỏng. Lỗi hỏng của phần mềm quá lớn, quá nhiều thành ra chính phủ yêu cầu phải có một giải pháp để giải quyết tình trạng phần mềm hỏng. Vệ tinh bay lên mà phần mềm hỏng thì vệ tinh bay lạc mất. Con số thống kế các phần mềm hỏng chiếm tới 70-80 phần trăm. Có hơn 100 dự án bị hỏng lung tung, lúc đó họ đi tới kết luận là phần mềm trở nên quá quan trọng, cần tìm giải pháp cho vấn đề phần mềm đó.

Khi đó Bộ quốc phòng đến trường Carnegie Mellon yêu cầu nhà trường thành lập một viện nghiên cứu về phần mềm, chuyên môn để giải quyết các vấn đề về chất lượng phần mềm. Lúc đó họ thành lập viện Software Engineering Institute (SEI). Viện Software Engineering Institute hình thành vào năm 1984, xây cất xong vào khoảng năm 1986. Khi xây cất xong cái đó, họ

đưa một người có thể nói là số một về phần mềm của Mĩ lúc đó là ông Watts Humphrey, lúc đó đang là chủ tịch của IBM. Ông Watts Humphrey là người chịu trách nhiệm về phần mềm của công ti IBM, lúc đó ông ấy đang ở giai đoạn sắp sửa về hưu. Họ đưa ông Watts Humphrey với địa vị của ông ấy, đưa xuống Viện Kĩ nghệ phần mềm Software Engineering Institute, để tìm giải pháp.

Ông ấy có đưa ra một đề nghị với chính phủ. Thường thường công việc làm với nhà nước, với chính phủ, các nhân viên làm việc cho chính phủ, trên nguyên tắc, ở địa vị rất cao nhưng thường thường khả năng về kĩ thuật lại không giỏi cho lắm. Ông Watt Humphrey đến từ công ti IBM, là một công ti tư, ông ấy nêu ra một vấn đề, lúc ấy tôi có thể gọi đó là một sự điều đình. Ông ấy không muốn Bộ quốc phòng, ông ấy không muốn chính phủ dính vô. Ông ấy bảo "Viện của tôi phải độc lập." Thành ra ông ấy yêu cầu mỗi công ti lớn ở Hoa Kì, mỗi công ti lớn bên này, mỗi công ti nổi tiếng gửi cho ông ấy một đến hai người, người giỏi nhất công ti để thành lập một hội đồng gồm những người làm phần mềm giỏi nhất. Ông ấy lựa chọn ra một số công ti, mỗi công ti được yêu cầu đưa người đến, những người đó phải làm việc với ông ấy trong vòng 2 năm để thành lập, cải tổ toàn diện lại tất cả cách làm phần mềm từ trước tới nay.

Khi đó tôi đang làm giám đốc tại Motorola, công ti đó cử tôi đến, bảo "Ngày xưa anh học ở Carnegie Mellon, anh không làm gì với cái trường này. Hơn nữa anh nắm về phần mềm." Lúc đó tôi nắm toàn bộ phần mềm của Motorola. Tôi được cử đến, là một trong những người đó. Sau khi mình được cử tới thì ông Watts

Humphrey chọn, có khoảng ba mươi mấy người, cuối cùng ông ấy chọn tất cả 16 người, trong đó có tôi. Lúc đó là thành phần đầu tiên vào ngồi cùng với ông ấy. Sau đó ông ấy mang đến một số người, là các khoa học gia, kiểu người được giải thưởng Nobel, nói cho vui. Những người đứng nhất về phần mềm và tên tuổi những người đó có trong các sách giáo khoa, Crosby, Richard, Mary, đại khái những người như vậy... Ông ấy đưa đến để ông ấy phác hoạ một kế hoạch về làm phần mềm.

Khi đó chúng tôi ngồi lại với nhau, gồm có 16 người trong các nhân viên lúc đó, ngồi bàn cách làm thế nào. Ông ấy bảo "Với kinh nghiệm của các anh, nếu các anh đi khảo sát phần mềm, các anh thấy cái gì?" Mỗi người đều đóng góp ý kiến. Sau khi đóng góp tất cả các ý kiến xong xuôi, họ tổng hợp thành mô hình. Từ mô hình đó họ đưa ra một phương pháp để đi kiểm định phần mềm. Khi đó trong nhóm cũng hình thành một dự án, trong dự án đó mỗi người lo một phần. Người thì lo về architecture (kiến trúc) người thì lo về đủ mọi thứ hết. Nhưng cuối cùng chốt lại cái khung. Cuối cùng đến năm 1988, từ năm 1986 mất 2 năm, đến 1988 chúng tôi hình thành mô hình đầu tiên, danh từ gọi là Software Process Assessement. Lúc đó mô hình đầu tiên là như vậy.

Trong nhóm đó chia ra làm ba phần. Một phần chuyên môn về chi tiết, một phần chuyên môn về kiến trúc, một phần chuyên về khảo sát. Khi đó tôi đứng đầu nhóm đi khảo sát, tức là làm assessement, đi kiểm định lại xem công việc ra sao. Phương pháp năm 1988 gọi là SPA: Software Process Assessement, người director trông về assessement lúc đó là tôi. Làm cái đó mình phải nắm vững tất cả, mình đi mình phải xem họ làm thế nào.

Khi đến nơi lúc đó giải pháp vẫn còn sơ sài lắm. Mình đi các công ti, mình xem xét nghiên cứu, thời gian mất độ một đến hai tuần. Phương pháp lúc đó là đưa ra một số câu hỏi, câu hỏi thuộc loại chuẩn, khoảng độ 160 câu hỏi. Mình phải gặp từng người một để hỏi. Thí dụ gặp ông giám đốc, "Thưa ông, ông làm ơn trình bày cho tôi biết làm như vậy, như vậy làm thế nào." Sau khi kiểm điểm tổng kết tất cả lại, thống kê lại mình cho công ti kia biết kết quả, "Đây là ưu điểm của anh, đây là khuyết điểm của anh. Đây là những nơi anh cần thay đổi."

Làm việc đó thì trên nguyên tắc cũng không có vấn đề gì mấy. Sau khi làm cái đó xong đem về trình lên, Bộ quốc phòng xem xét phương pháp đó họ nói là "Nếu anh nói công ti có ưu điểm này có khuyết điểm này, nó hay thế này, nó dở thế này, nói thì là như vậy. Nhưng bây giờ phải lựa chọn giữa năm bảy công ti thì sao? Công ti nào cũng có ưu điểm khuyết điểm thì tôi lựa chọn ai đây?" Họ muốn có một tiêu chuẩn chặt chẽ hơn. Lúc đó chúng tôi lại ngồi lại với nhau. Bây giờ làm như thế nào?

Ông Crosby là một trong những người rất nổi tiếng về quality - chất lượng, Crosby là một trong những người đã từng làm việc với ông Mary, ông Crosby nói rằng là "Bây giờ chỉ có một cách duy nhất là phân chia thành các thứ tự, liệt kê các công ti vào thứ tự một, hai, ba bốn, năm gì đó. Liệt kê nó ra. Công ti này với ưu điểm khuyết điểm thế này thì xếp vào hạng số một, hạng thấp nhất, công ti này với ưu điểm khuyết điểm thế này thì xếp hạng vào số hai, số ba gì đó." Dựa trên ý kiến của ông Crosby, ông Watts Humphrey thu thập tài liệu lại, xác lập một mô hình mới. Mô hình ấy gọi là Capability Maturity Model (CMM) - Mô hình tăng trưởng năng lực.

Trong khi làm cái đó cũng có nhiều ý kiến rất khác nhau. Thứ nhất capability là khả năng, thứ hai maturity tức là trưởng thành. Coi một công ti không có gì cả, giống như một người trưởng thành từ nhỏ tới lớn. Lúc đầu họ như vậy, từ từ họ trưởng thành lên. Lúc đó mình nói từ một thanh niên, mới bắt đầu đi học, rồi đi làm rồi từ từ trưởng thành lên. Tôi nghĩ trong lúc chúng ta dùng chữ maturity đã có yếu tố thời gian trong đó. Vì có yếu tố thời gian, do đó anh không thể nhảy từ mức nọ sang mức kia được. Như một người hai mươi tuổi không thể nhảy lên thành người năm mươi tuổi được. Trưởng thành như một người năm mươi tuổi, vậy cần yếu tố thời gian. Tôi rất thích chữ maturity, đa số mọi người về sau dùng chữ capability mà không dùng chữ maturity. Tôi rất thích maturity vì nó là một trong những yếu tố đầu tiên, bắt đầu từ quan niệm đầu tiên cho tới khi thành mô hình. Thành ra tôi biết rõ yếu tố đó. Nhưng trong khi làm việc đó có một cái mà thực sự do mình ngây thơ mà không biết.

Sau khi mô hình làm xong, ông Watts Humphrey chuyển sang mức khác, ông ấy đi vào giai đoạn nghiên cứu. Giao việc chương trình làm việc cho ông khác, ông Bill Curtis. Ông Bill Curtis lúc đó là CIO của At&T. Ông Watts Humphrey theo chúng tôi là một người đầu óc rất sáng, đúng là một người có vision, tầm nhìn rất cao, rất là giỏi. Nhưng ông Watts Humphrey là một nhân viên quản lí rất tồi, nói thẳng như vậy. Ông ấy không phải là người manager giỏi. Do đó khi làm việc trong nhóm đó, mọi người cãi nhau lung tung, với ý kiến của ông ấy, không cãi nhau trước mặt ông ấy, nhưng khi ông ấy quay mặt đi là mọi người trong nhóm cãi nhau. Lúc đó tôi là một người nhân viên quản trị của công ti

Motorola, chính tôi sắp xếp việc điều hành với ông ấy, tôi cũng hơi bực mình với ông ấy. Nhưng với uy tín của ông ấy mình không nói được gì. Nhưng khi ông ấy vắng mặt, trong nhóm cũng có vấn đề xích mích với nhau.

Khi mang một người khác đến, ông Bill Curtis đến để trông cái đó, ông Bill Curtis đã là một CIO của AT&T. Khi ông Bill Curtis về, ông ấy là một nhân viên quản trị rất giỏi, vào đến nơi là ông ấy sắp xếp anh nào ngồi đâu là ngồi đó, không có cãi nhau nữa. Lúc đó ông ấy đổi nhóm lại một chút. Chúng tôi là những người tự nguyện, được các công ti cử đến do đó không ai bảo được ai cả. Mặc dầu tôi là giám đốc của Motorola nhưng bây giờ nói chuyện với ông giám đốc của AT& T mình đâu có cãi nhau được. Ngồi làm việc chung với nhau người nào cũng điều hành cả. Đến khi anh Bill Curtis về, anh ấy là người làm CIO, một chuyên viên rất là cao, anh ấy tái tổ chức lại nhóm này. Khi tổ chức lại nhóm này anh ấy có một ý khác. Vì trong nhóm cãi nhau rất nhiều, người nào cũng muốn là "tôi làm phần này tôi làm phần kia," cãi nhau lung tung, nên anh ấy đặt ra một luật mới, luật chơi mới, nói rằng, "Tôi biết các anh là những người đã sáng chế ra phương pháp này. Mười sáu người mười bẩy người gì đó. Nếu bây giờ viết phương pháp này ra rồi để tên người vào thì ông nào cũng muốn mình là đứng đầu không ai nhường ai cả." Thành ra Bill Curtis đưa ra trò mới là, "Tất cả các anh là contribution, người đóng góp. Còn tôi sẽ mướn những người học trò mới ra trường để những người đó nắm lấy ý tưởng, get ideas. Đầu óc của các anh, chất xám của các anh, nhưng viết xuống là những người học trò họ viết xuống."

Bill Curtis mướn 6 người học trò mới ra trường Carnegie ở Pittsburgh đây thôi, giao cho những người học trò đó trách nhiệm viết nó xuống. Lúc chúng tôi làm là các khái niệm concepts, ý tưởng ideas lung tung, nhưng mà khi ngồi xuống viết thành bài thành bản, thì giao cho các em học trò đó viết. Lúc đó các học trò đó viết xong, ông ấy bảo, "Những người học trò này sẽ được để tên, coi như là tác giả của những phần này. Còn các anh là người đóng góp, contribution, hơn nữa các anh không làm việc, không ăn lương, các anh tự nguyện từ các công ti khác gửi đến. Thành ra tất cả danh sách các anh đều là advisors - cố vấn." Là vì trong nhóm có xung khắc với nhau. Đó là ý kiến để giải quyết vì ông nào cũng có sự cãi nhau khi làm việc chung. Chẳng hạn như nói là "Phần tôi làm 80%, tôi muốn để tên tôi ở đó."

Tôi thì tôi không coi thành vấn đề cãi nhau về tên tuổi nhưng mà những người khác cãi nhau rất nhiều. Tôi khi ngồi họp chung tôi chỉ cười thôi, tính tôi không thích tranh luận các vấn đề tên tuổi. Lúc đó mình đang làm giám đốc ở một công ti, mình chỉ muốn làm sao xong việc để trở về vị trí của mình, làm việc cho công ti của mình chứ mình đâu có muốn ngồi ở đây. Nhưng mà những người đó có tham vọng, để về sau tôi sẽ kể câu chuyện này. Là vì mình không biết ý kiến của người ta. Tôi cũng có nói chuyện với một số anh em, tôi bảo chứ "Anh làm giám đốc ở AT&T, anh cần gì cái tên cái tuổi ghi vào bài vở sách giáo khoa cho học sinh, anh để cho chúng nó ngồi chứ." Nhưng mà mình không biết thâm ý của họ. Thành ra khi họ nói chuyện với tôi, tôi bảo, "Không cần, mặc dầu tôi làm 80-90 phần trăm nhưng ai muốn làm sao thì làm."

Khi làm như vậy thì trong nhóm cũng có bất mãn khá lớn. Sau sự việc đó, mô hình đó hình thành vào năm 1989, ông Bill Curtis nói, "Cám ơn tất cả các anh. Bây giờ tất cả các anh thành một cái gọi là Advisor Board. Còn tất cả các em học trò này nó ngồi nó viết dưới sự kiểm soát của tôi, sau đó tôi đưa mô hình này ra, tự nó sẽ là đủ. Bởi vì bây giờ đông quá, cãi nhau lung tung, trong khi nhóm viết chỉ có 4 người." Ba người học trò và ông Bill Curtis, ba người kia là học sinh mới ra trường. Khi làm chuyện đó xong, họp lại, mọi người đồng ý, bởi vì không ai nhường ai cả, thành ra khi đưa cho học trò làm, mọi người thấy có lí rồi. Thành ra bốn người, với ba em học trò đó trở thành tác giả, mặc dầu đằng sau đó thì đây là những người tham gia khác, là những contribution, những người đóng góp cho việc này. Đến đây giải pháp đó coi như hoàn tất. Chúng tôi ngồi đây đều trở thành cố vấn.

Trong nhóm cố vấn đó có ba người, tôi, ông Ron Radis, vice president của Hughes Aircraft và ông Ray Dion là Director tại Raytheon Electronics. Ba người chúng tôi đều ở địa vị rất cao trong công ti, chúng tôi chỉ mong hết việc để trở lại công ti làm việc. Thành ra ba chúng tôi công việc xong là chạy về công ti làm việc ngay, không để ý gì cả.

Khi mô hình này được đưa ra, chính phủ rất thích, mang mô hình này ứng dụng vào trong Bộ quốc phòng, tìm được rất nhiều lỗi, sơ suất ở trong đó. Mang đi áp dụng vào các công ti làm việc với chính phủ, tìm ra rất nhiều lỗi. Thành ra mô hình năm 1990 rất là thành công. Lúc đó mới chia ra anh ở level 1, level 2, level 3, cấp 1, cấp 2, cấp 3, cấp 4 gì đó. Khi đó tất cả các chương trình

đó, chỉ có một chương trình duy nhất được cấp 5. Lúc đó anh Grerior tức là vice president của ... đến khảo nghiệm chương trình của NASA. Chương trình NASA phóng phi thuyền con thoi lên mặt trăng, chương trình đó là nơi duy nhất được cấp 5, ngoài ra không ai được cấp 5, không ai được cấp 4, không ai được cấp 3. Tất cả đều cấp 1 và cấp 2 hết. 1 với 2 là thấp nhất. Tóm lại nó cho mình thấy với tình trạng làm phần mềm lúc đó: tất cả các công ti làm phần mềm, trên nguyên tắc, theo phương pháp này, đều rất thấp.

Mang sang Âu châu để thử bên Anh, không một công ti nào bên Anh đạt mức 2, đừng nói chuyện mức cao hơn. Mang sang Đức, ai cũng ở mức đó, chính phủ mới nhận ra là phần mềm, việc huấn luyện về phần mềm từ trước tới nay có vấn đề. Từ xưa đến này dạy phần mềm dựa trên phương pháp là của khoa học máy tính computer science, chỉ có làm lập trình. Đến khi chúng tôi thiết kế cái chương trình này thì chính phủ lựa chọn trường Carnegie Mellon, trường số một về kĩ nghệ phần mềm, software engineering. Khi chúng tôi làm, chúng tôi đưa tất cả những cái gì hay nhất của software engineering vào mô hình này. Lúc đó chính phủ Mĩ nói rằng là có khác biệt rất lớn giữa khoa học máy tính và kĩ nghệ phần mềm. Lúc đó sự phát triển của software engineeing nổi lên. Đấy là vấn đề nó như vậy.

Thế nhưng bây giờ nếu mình lùi lại, lúc chúng tôi trở về làm việc thì một năm sau vẫn yên lặng không có gì cả. Khi những công ti họ được khảo sát bởi Viện kĩ nghệ phần mềm, Software Engineering Institute, họ được đánh giá. Thí dụ lấy một công ti, thí dụ công ti ABC, đánh giá công ti ABC cấp số 1, cấp thấp nhất. Ông chủ

công ti nói, "Vâng tôi đồng ý với các anh, tôi đang ở mức một. Nhưng bây giờ làm sao tôi lên mức thứ hai? Ông có mô hình đánh giá, ông phải cho chúng tôi biết chứ." Khi đó mới nảy sinh tình trạng cần có tư vấn, tư vấn chỉ ông phải làm thế này, ông phải làm thế kia. Khi đó tôi mới vỡ lẽ ra lí do mười sáu người đó cãi nhau là vì họ có thâm ý là họ nhảy ra họ làm tư vấn. Trong khi ba người trong nhóm chúng tôi không nghĩ tới chuyện đó. Thứ nhất là vì địa vị của mình rất là cao, lương bổng của mình rất là khá, mình không nghĩ tới chuyện mình đi làm tư vấn bên ngoài. Trong khi những người kia trong thâm ý của họ là họ làm cái này, với cái tên tuổi đó thì họ nhảy ra làm tư vấn kiếm nhiều tiền hơn.

Đó là một điều mà theo ý kiến của tôi, nhìn lại quá trình làm việc thấy mình còn rất là ngây thơ trong khi những người kia họ đã tính trước rồi. Bởi thế mới có chuyện cãi nhau chứ bình thường thì có chuyện gì đâu. Bây giờ quay lại, ba người chúng tôi, Ron Radis, Ray Dion, sau đó năm 1991 ngồi lại nhìn, thì mười lăm người kia nhảy ra làm tư vấn. Họ đến các công ti được đánh giá số 1 số 2, họ tư vấn cho các công ti đó để lên cấp cao hơn. Mà lên cấp cao hơn thì trúng thầu chính phủ sẽ hơn. Lúc đó tôi nhớ rằng Bộ quốc phòng nói rằng, "Nếu anh không ở mức 3, chúng tôi không cho anh đấu thầu nữa." Thành ra công ti nào cũng muốn nhảy lên mức số 3. Tôi nhớ lúc đó ông bộ trưởng quốc phòng có hỏi ông Watts Humphrey, "Mức nào làm được?" Ông Watts Humphrey nói là "Phải cỡ mức 3 trở lên mới khá được, chứ mức 2, mức 1 thấp quá." Thành ra bộ quốc phòng mới ra chỉ thị là "Nếu các anh không đạt được mức 3 thì đừng có hòng mà thầu." Thầu với chính phủ đây, mỗi cái thầu chương trình với bộ quốc phòng là mấu trăm triệu, ngon lành

đấy. Thành ra các công ti rất mong tư vấn giúp họ. Thành ra khi đó anh tư vấn kiếm tiền bội phần.

Mình không hình dung ra chuyện đó. Lúc tôi về Motorola chương trình đó đang quay rất mạnh. Thành ra tôi về Motorola tôi làm việc tiếp. Các anh chị cũng biết, khi mình ở địa vị khá cao, mà công ti đưa mình ra khỏi vị trí để làm việc ở nơi khác khoảng 2 năm, nội bộ trong công ti mình không biết nữa. Khi mình trở về người khác có thể đã lấy công việc đó rồi. Các anh chị cũng hiểu chuyện đó. Khi tôi trở về Motorola, lúc tôi vắng, họ đã đưa một người giám đốc khác lên. Bây giờ mình trở về, hai ông giám đốc ngồi chung một chỗ thì cũng là việc như vậy. Khi mình đi như vậy là sự hi sinh, công ti họ cũng biết như vậy. Khi tôi trở về, công việc đã chạy theo chiều hướng khác. Thành ra đó cũng là bước của mình. Đáng lí ra thì cũng phải công nhận là mình quá sốt sắng, mình muốn cải thiện phần mềm, mình bỏ công việc cũ của mình, nhận một sự phân công đặc biệt để đi sang làm việc thành lập cái Software Engineering Institute.

Khi nhìn trở lại lúc mô hình bắt đầu ra, ngay khi đó tôi thấy là trường hợp đó có sự thay đổi rồi. Chương trình đó rất thành công với bộ quốc phòng. Khi đó chúng tôi bắt đầu mang chương trình đó áp dụng vào các công ti phần mềm của khu vực không phải của quốc phòng. Cũng rất thành công. Thành ra theo thời gian, tôi là người dạy chương trình đó tại Motorola, một công ti hoàn toàn tư nhân không dính gì tới quốc phòng. Tôi mang sang Microsoft, tôi mang sang các công ti phần mềm khác. Đó là vấn đề dạy như vậy. Năm 1991 mô hình đó được công bố trên toàn thế giới, lúc đó mọi người bắt đầu biết tới CMM là năm 1991. Từ năm 91 tới

năm 95 CMM phát triển rất mạnh trong lĩnh vực quốc phòng, lĩnh vực của Mĩ. Nhưng đến năm 95 bắt đầu có các giao ước quốc phòng giữa Hoa Kì và các quốc gia khác. Nếu anh ở bên Anh và anh không đạt yêu cầu thì tôi cũng không giao thầu. Bên đó cũng phải làm như vậy. Điều đó tạo ra tình trạng khan hiếm tư vấn.

2. CMMI và tư vấn đánh giá

Nếu nói về tư vấn biết CMM này thì trên cả thế giới lúc đó chỉ mới có 15 người thôi. Lúc đó các công ti lớn như IBM, Acenture, họ rất nhạy về tư vấn này, thành ra họ mới làm ra vụ kiện. Họ bảo "Đây này, có 15 anh làm cái này. 15 anh này xuất thân từ Carnegie Mellon, từ Software Engineering Institute. Mà các anh ấy nắm độc quyền thế này à? Không ai biết nó là cái gì cả." Anh kiếm tiền thì các công ti lớn nó nhìn thấy ngay, nó muốn nhảy vào chứ. Nó mới tung ra một vụ kiện như vậy. Các anh không giấu nghề được, các anh phải dạy dỗ người khác. Cho đến năm 1995, mọi người chỉ biết là có một số người rất ít, 15 anh có thể tư vấn được, bởi vì 15 anh này là những người làm cái đó. Ngoài ra không có dạy dỗ gì, cũng khó cho mọi người biết cái đó. Mọi người chỉ biết ông ở phần 1, phần 2, cấp 1, cấp 2, cấp 3, nhưng nội bộ bên trong không ai biết. Lấy lí do đó họ có một vụ kiện lớn. Khi có vụ kiện đó nhà trường mới nói chúng tôi không dính dáng tới chuyện đó. Nó bắt phải đem phương pháp CMM ra giảng dạy cho tất cả mọi người. Và giảng dạy xong, những người học xong cái đó thì phải qua một kì thi thì lúc đó mới có quyền đi làm tư vấn.

Đấy là quá trình tôi trình bày cho các anh chị như vậy. Khi đó cái nhóm 15 người kia ngồi lại mới bảo, "Có bao nhiêu người mình dạy, khi làm CMM?" Khi pha đầu tiên chuyển sang CMM, chúng tôi là những người làm xong, chuyển cho các em học trò viết, chúng tôi trở về công ti của chúng tôi. Ba người chúng tôi, Gorbinis, Dominique và tôi trở về công ti. Còn 15 anh kia vẫn ngồi đấy. Nó mới hướng dẫn các anh học trò này viết lại. Nó đã giấu bớt những cái then chốt, bí mật, kiểu giấu nghề đó. Nó sử dụng các danh từ rất mơ hồ, nó sửa các câu chữ, mất đi 20%. 15 anh đó, với 3 anh học trò mới ra trường làm việc đó. Ba anh học trò này hoàn toàn trong trắng, vừa mới ra trường sau 4 năm đại học đã biết gì đâu, không có một kinh nghiệm gì cả. Nó bảo sửa cái này, sửa cái kia, thì cứ nghe lời người ta thôi. 15 anh này giấu khoảng 20%. Nhưng nó giấu với nhau thôi, khi nó đi làm tư vấn cho mọi người không hiểu gì thì nó mới chỉ tài liệu cho.

Đến khi công ti nó mạnh lên, năm 1995 anh phải bung ra anh dạy tất cả mọi người thì 15 anh tư vấn ngồi lại với nhau bảo, "Bây giờ mình phải dạy nơi khác, nó thành tư vấn thì kinh doanh của mình không tốt." Nó mới đổi lại một cái nữa, đây hoàn toàn là việc thay đổi, xào xáo lại bên trong, thay đổi lung tung tất cả lên. Nó đưa ra một mô hình mới rồi mới nói với chính phủ là ngày xưa mô hình CMM chỉ nhắm tới phần mềm thôi. Bây giờ môn tích hợp cả phần mềm lẫn phần cứng lẫn phần xanh phần đỏ gì đó thì đưa ra một cái mới gọi là integration CMMi, thêm chữ i vào bên trong đó. Đối với chính phủ thì integration càng hay, nghe nói tới tích hợp ai cũng thấy là hay, nhưng thực sự nó xáo trộn nội bộ

không còn ai hiểu cái gì nữa. Thế là nó ra cái mô hình đó.

Khi đó 3 người chúng tôi thấy tình hình như vậy, chúng tôi thấy vẫn còn trong ban Advisor Board, chúng tôi trở lại những cái cũ, hỏi sao sửa hết những cái thế này. Tôi mới gọi những người kia lại bảo," Bây giờ phải rõ rệt 1 là 1, 2 là 2, 3 là 3. Bây giờ nó thành cái gì không à." Họ bảo "Các anh không làm cái này lâu rồi các anh không hiểu đâu." Lúc đó mình mới vỡ lẽ ra là như vậy. Chúng tôi, ba người chúng tôi viết một lá thư yêu cầu không được làm như vậy, đây là trường hợp vi phạm rất là lớn, chúng tôi đưa vấn đề lên. Nhưng trong một nhóm Advisor Board, gồm có 20 người, 3 người chống còn 17 anh kia không chống thì cũng chẳng đi đến đâu. Lúc đó anh Ron Radis tức lắm, nói "Thôi từ nay chuyển giao không dính dáng đến cái này nữa. Bỏ cái đó đi, không dính tới nữa." Nhưng mà tôi lúc đó đã là giáo sư ở đây, tôi cảm thấy mình có trách nhiệm phải nói lên tiếng nói của mình.

Tôi nói vui, trận đánh này rất là đơn độc. Hai ông kia trở về là phó chủ tịch, vice president của các công ti lớn, quyền hạn và uy tín của họ rất là lớn, họ bảo "Chúng tôi không quan tâm, không cãi nhau chuyện này. Chúng tôi về trông công ti của tôi, chuyện này không là gì." Họ đã lên chức rất là cao, lúc đó tôi mới chỉ là kĩ sư trưởng, chief engineer và giáo sư đại học. Tôi coi vấn đề này là đơn độc và đứng lên nói những cái đó. Khi đó họ cũng rất khó chịu với mình. Họ bảo, "Ngày xưa làm CMM, anh là Advisor Board, nhưng sau CMM anh chả biết gì về cái này cả, xin mời anh ra khỏi cái board luôn." Khi đó họ đưa tôi ra khỏi cái board luôn. Lúc đó họ càng

lộng hành thêm nữa, tôi nói thực sự nó là như vậy, họ càng xào xáo thêm nữa. Trong nội bộ của họ lúc đó cũng có sự chia rẽ. Người này giấu cái này người kia giấu cái kia thành ra không ra đâu vào với đâu cả.

Đến khi họ mang ra giảng dạy, rất nhiều điều mơ hồ. Tôi biết IBM họ vào họ xem. Tôi nhớ là hôm đó tôi ăn cơm với ông Rinoprino hiện giờ đang làm president của IBM. Sau khi ăn tôi cười cười mới bảo "Công ti IBM rất là lớn, tiền bạc rất nhiều, uy tín rất cao. Các anh biết thế này, làm thế này, mỗi lần làm lượm bạc cắc, hai chục nghìn, ba chục nghìn, trong khi công ti ông thu bạc triệu. Ông mà làm sai, quốc phòng vào nó mang tiếng, ông lãnh đủ. Nó giấu đầu giấu đuôi rồi thì không cách chi ông vô làm mà biết được. Nhân viên của ông được huấn luyện cách mấy đi chăng nữa thì vô cũng hỏng." Tại vì khi xưa họ nhìn thấy có bộ ăn tiền ngon, nhưng sau đó ngồi lại tôi bảo nếu không tin ông ra xem công ti khác ông hỏi thêm đi. Lúc đó ông Dominique đang làm vice president, ra hỏi ông president, xin ý kiến luôn. "Ý kiến của người ta như vậy mà người ta còn bỏ cái board người ta đi thì mình dính cái đó làm chi." Lúc đó Rinoprino phụ trách IBM trở lại liên lạc với anh Ron Radis của công ti Hughes Aircraft và anh Ray Dion, của công ti Raytheon Electronics. Hai người đó nói ngay "Đúng rồi, John Vũ nói đúng rồi, cái chuyện này bây giờ không ra cái gì nữa. Đi sâu vào là lãnh đủ." Thì IBM mới rút ra không làm cái đó nữa.

Lúc đó nhóm người làm tư vấn họ mới huấn luyện xong bung ra làm tư vấn. Vào khoảng năm 1995 đến khoảng năm 2000 số người tư vấn họ huấn luyện được quãng 400 người. Cái quality vẫn còn rất chặt chẽ, mặc

dầu volumn đã bị xào xáo rồi nhưng quality vẫn còn chặt chẽ. Đến khi vào khoảng 1997 -1998 người Ấn Độ bắt đầu vào. Những anh Ấn Độ đang làm Y2K, anh ấy đánh hơi thấy cái này coi bộ ăn tiền được thì anh ấy nhào vào làm. Người Ấn Độ có một cái hay là khi đến nơi mua chuộc rất kĩ, tiền trong tiền ngoài mời ông ấy ăn mời ông ấy uống rồi chuyện cửa trước cửa sau. Thành ra những anh nắm được cái then chốt đó đều mở cửa. Tự nhiên số người làm tư vấn của chính phủ Hoa Kì vào khoảng 200 người, nhưng người làm tư vấn Ấn Độ khoảng 400 người. Anh Ấn Độ bảo chúng tôi không kinh doanh ở đây là thị trường mở. Đúng là những người Ấn Độ đi vào thị trường Hoa Kì này, người tư vấn Mĩ nói, họ chỉ nắm thị trường thế này thôi, chứ còn ra thị trường thế giới chúng tôi không cần. Anh Ấn Độ bảo chúng tôi chỉ nắm thị trường thế giới, không nắm thị trường của anh, không có đụng chạm nhau. Tức là người Ấn Độ sẽ không vào thị trường Hoa Kì, nhưng những anh kia không nghĩ chuyện đó, không nghĩ chuyện thị trường bên ngoài là khá.

Nói đại để như vậy, để 400 anh Ấn Độ đi làm tư vấn cho CMM ở ngoại quốc. Anh Ấn Độ làm ngay một trò. Nếu đi đúng theo phương pháp thì mất rất lâu. Anh Ấn Độ bàn với mọi người là đổi CMMi thành CMMi version 2. Version 1 thì đã bị sửa nát bét ra rồi, bây giờ đổi version 2. Mà muốn đổi sang version 2 để làm tư vấn thì phải chuyển từ cái gọi là Capables, năng lực, sang cái thứ hai gọi là Documentation, tài liệu. Các anh hiểu cái đó không ạ. Một là anh chứng minh cho tôi thấy khả năng của anh, hai là nếu anh có tài liệu thì tôi chỉ coi tài liệu chứ không cần khả năng của anh. Tức là đang đi từ cái mức là tôi xem khả năng của anh thì chuyển thành tôi

xem bằng cấp của anh. Thí dụ như mình khảo sát một sinh viên học sinh xem có thi đậu hay không, cứ đưa bài ra là được rồi. Người Ấn Độ đổi phương pháp đó là vì khi đó Ấn Độ có rất nhiều cách họ đẩy vấn đề thành documentation. Thành ra CMM biến thành một hình thức là nếu anh có tài liệu thì anh là được. Thành ra nó đổi phương pháp khảo sát thành documentation. Và cuối cùng nó thành version 2, documentation oriented, hướng theo tài liệu, không phải là quality oriented, hướng theo chất lượng nữa.

Ngay khi đó những người chúng tôi phải đứng lên, lúc đó là nói tiếng nói cuối cùng. Chúng tôi đứng lên họp tất cả mọi người lại đưa ra tiếng nói. Có hai em học sinh tác giả của CMM cũng đứng lên chung với chúng tôi. Cái này đã bị sai quá rồi, bây giờ phải dừng nó lại. Phải làm lại hết. Đến khi đó thì nhóm tư vấn ngồi trong Advisor Board, ba bốn trăm người ngồi đó rồi, làm thế thì bể nồi cơm của họ. Họ yêu cầu tất cả những người đã làm, kể cả những em học sinh ngày xưa họ đã thuê, người authors đó, cũng phải trong nhóm luôn. Thành ra cuối cùng mô hình đó trở thành, tôi nói vui, là một món ăn ngon lành biến thành nồi cơm heo rồi.

Đó là sự thực về cái mô hình này. Trong suốt 20 năm ở Hoa Kì, tôi là người chịu trách nhiệm về phần mềm, khi nào lên số 4, hay số 5, gần như tất cả các phần mềm cần quyết định ở đẳng cấp số 4 số 5 đều giao cho tôi hết. Và tôi là người luôn luôn xem xét các công ti đó. Khi tôi về tôi làm cho Boeing, uy tín của tôi rất là lớn. Thành ra trách nhiệm ở trong đó, ngay cả ở các công ti họ đạt đẳng cấp số 4 số 5, họ cũng mời tôi đến. "Tư vấn chỉ cho tôi như vậy nhưng mà anh đến anh xem lại xem

có đúng thế không." Họ vẫn rất là tôn trọng mình. Nhưng mà về sau người tư vấn làm bê bối quá tôi mới nói thế này, tôi bảo "Hai mươi năm nay, cả một nước Mĩ, làm về phần mềm, suốt 20 năm được 7 nhóm, 7 công ti, không phải toàn công ti đâu, chỉ 7 nhóm đạt được đẳng cấp thứ 5." Bên này nó không xét company, nó xét organization. Chẳng hạn như chương trình phi thuyền con thoi của NASA, là một chương trình trong NASA chứ không phải toàn cõi NASA. Ngay công ti Boeing chúng tôi cũng có 2-3 nhóm làm cái đó, nhưng cả công ti Boeing thì không được cái chuẩn số 5, nhưng mà một group trong đó được cái đó. Đó là cái khác biệt, chúng tôi quan niệm đó là organization chứ không phải là company. Độ 17 công ti, 17 nhóm đạt được mức số 5. Trong khi đó sang Ấn Độ, chỉ có 2 năm thôi, vào khoảng năm chục cái số 5 và đến bây giờ gần như là 400 đến 500 công ti đạt mức số 5. Trung Quốc mới biết đến mô hình này khoảng độ 3 năm nay thôi. Tôi nhớ là lần tôi sang Trung Quốc lần đầu tiên giới thiệu mô hình này, lúc đó năm 2003. Cả Trung Quốc chưa ai biết CMM là cái gì. Đến năm nay là 2008 thì họ cũng đã có 300 đến 400 công ti đạt tới mức số 5.

Tôi trình bày cái đó để các anh chị thấy như vậy. Cuối cùng thì đi tới Việt Nam đâu 7-8 công ti đạt mức số 5. Tôi đã đi và nói chuyện, khi tôi sang Trung Quốc, tôi ở Thượng Hải gặp một công ti mức số 5 họ cho tôi coi giấy chứng nhận. Tôi sang Thành Đô, tức là tuốt phía bên kia, cũng gặp công ti khác, họ cũng khoe số 5. Mà con số của họ chính xác đến mức độ bên này đo lường ra sao thì bên kia đo lường cũng vậy. Công ti nào cũng giống hệt nhau, qui cách giống nhau, các con số giống hệt nhau. Bất kì cái gì tới từng chi tiết cũng giống nhau.

Hai công ti hoàn toàn tách biệt mà nó giống nhau từng con số, từng dấu chấm, từng dấu phẩy, khác mỗi tên công ti. Và bây giờ sang Việt Nam, tôi không nói tên công ti ở Việt Nam. Tôi sang một công ti ở Việt Nam tôi ngồi nói chuyện, họ cũng trình bày những dữ kiện như vậy. Mà dấu chấm dấu phẩy của họ cũng không khác công ti Ấn Độ, không khác công ti Trung Quốc bao nhiêu hết. Tóm lại, theo quan niệm của tôi, có một người nào đó đưa bài viết sẵn ra, nó chỉ đổi tên công ti thôi. Đây này rõ ràng tất cả các tài liệu đã có sẵn rồi, ở cái mức đó.

Tôi trình bày cho các anh các chị về lịch sử như vậy để các anh các chị nắm vững vấn đề. Đó là lịch sử của cái làm qui trình CMM. Hiện giờ thì nó đã đến mức gọi là tệ hại rồi. Đây là quan niệm cá nhân của tôi, tôi trình bày. Tôi không đứng trên quan niệm nào. Thứ nhất là tôi không hề làm tư vấn. Thứ hai tôi cũng không giảng dạy cái này cho ai hết. Tôi chỉ giảng dạy trong trường đại học. Rất nhiều công ti rất nhiều nơi họ mời tôi đến, tôi nói "Không tôi không làm chuyện này." Khi anh Ron Radis, về sau bị một bệnh rất là nặng, từ chức, về dưỡng bệnh. Trên giường bệnh anh ấy nói là "Bây giờ trên chiến tuyến này chỉ có 3 người chúng ta, anh, tôi và Ray Dion, chúng ta ráng lên tiếng nói trung thực, dù tiếng nói của mình người ta không nghe, vẫn phải nói như vậy." Tôi thì tôi rất cảm những người vậy, anh phải biết là những giờ phút cuối cùng của con người, lời nói rất thành thật. Anh ấy nói là, "Tôi trở về công ti của tôi, tôi cấm tất nhân viên của tôi làm cái này. Làm thì làm cho tử tế. Ai cũng biết là trong nhóm làm việc, mỗi người làm một phần. Cái phần khảo sát, assessement, anh là số một." Ai ở đâu trong công ti nào cũng mời tôi đến, nhiều

khi mình tư vấn cho họ rồi, nhưng những công ti làm ăn đứng đắn như công ti IBM, họ đã có người tư vấn làm được, nhưng khi họ đạt mức số 3, số 4 họ cũng gọi tôi đến. "Nhờ anh coi giùm xem chúng tôi có làm được thực sự hay không."

Họ nói như vậy cho nên tôi sẽ giúp các anh các chị một cách nhìn nó không giống như cách nhìn của người giảng dạy, nó không giống cách nhìn của người tư vấn. Tôi sẽ nêu lại, hôm qua có một số ý kiến trao đổi, năm ngoái cũng đã nói qua đề tài rồi, hôm nay có một số anh chị mới, tôi nói lại vấn đề này. Và bài giảng của tôi ngày hôm nay tôi cũng đưa hết cho các anh chị. Các anh chị có thể mang bài này về giảng dạy trong nước rất là giản dị. Ngoài ra tôi thêm một phần nữa.

Đến năm 2005 công ti Boeing được giải thưởng cao nhất về phần mềm là IEEE. Đây là giải thưởng trên thế giới, gồm có một uỷ ban đi xét tất cả các công ti trên thế giới xem những công ti nào làm giỏi nhất, chất lượng cao nhất. Tôi đạt giải đó, dựa trên CMM tôi đạt giải đó. Để đạt cái giải đó mình phải có tất cả những dữ kiện. Tất cả những dữ kiện gì mà mình có thể trình bày được, tôi cũng mang vào đây. Bài đó tôi đã trình bày trước hội đồng hàn lâm viện bên Mĩ này. Tôi đã trình bày ở một số conferences khác nhau. Thành ra các anh chị thấy như vậy việc đạt được giải thưởng IEEE Award không phải là chuyện dễ. Gần như dính tới vấn đề quality thì nó cần tất cả các dữ liệu đo, vấn đề chất lượng cho một nhóm người. Mà tôi cũng nhấn mạnh trong công ti Boeing chỉ có 3 nhóm đạt được mức số 5 thôi. Công ti tôi có khoảng 400- 500 organizations chỉ có 3 organizations đạt được. Tôi cho các anh chị thấy là tôi đặt vấn đề chất lượng, vấn

đề con số đo đạc rất là chính xác, rất là kĩ lưỡng, chứ không có làm bừa làm bãi.

Thành ra hiện giờ nó vẫn là một cái thước đo của những công ti đứng đắn. Thành ra Nhật, công ti Toyota sang xem cái này nó và bảo "Anh có những cái chúng tôi cần phải học. Chúng tôi dùng cái đó làm benchmark để nếu các công ti khác hỏi nếu anh đạt mức số 5 thì anh phải làm thế này." Ngày xưa tôi làm cho Motorola thì làm Six Sigma. Sau này Toyota sang cũng xem lại. Bây giờ các công ti đứng đắn họ cũng xem, họ dùng cái đó làm benchmark cho công ti của họ. Tôi trình bày với các anh chị như vậy.

Câu hỏi: Thầy nói là vào thời điểm CMM được phổ biến ra bên ngoài thì tổ chức nào ở trong nước Mĩ cấp chứng nhận?

Cái Software Engineering Institute đó - Viện kĩ nghệ phần mềm. Đã cấp khoảng hơn 1000 giấy phép. Anh trả tiền cho họ thì anh được giấy phép. Trường học không có cấp cái đó. Việc lấy chứng nhận cũng như chứng nhận bằng lái xe vậy thôi. Tôi vẫn nói đùa với mọi người như thế này. Tôi nói rằng ai cũng có thể có bằng lái xe. Nhưng người lái xe đua với người lái xe mới tập là khác nhau. Một em 17-18 tuổi mới bắt đầu đi làm bằng lái xe và một tay lái xe đua Innova 500, cái khác nhau ở chỗ đó. Thành ra mọi người bảo tôi, có một số anh Việt Nam hỏi tôi, "Nếu bây giờ tôi muốn là người làm cái đó thì có được không?"

Tôi bảo "Dĩ nhiên là làm được chứ. Bây giờ anh vào anh học 3 lớp họ dạy. Sau đó anh đậu kì thi. Và anh đóng một số tiền. Hiện giờ cơ quan cấp giấy phép gồm 8

người, trong đó 6 người Ấn Độ. Họ đòi mỗi người mỗi năm phải đóng cho họ hai chục nghìn đô la. Họ sẽ cấp cho anh giấy phép hành nghề. Anh muốn hành nghề ra sao thì hành nghề." Tôi nói đùa đùa, "Bây giờ Ấn Độ nó làm chỉ có hai ba mươi nghìn thôi. Anh có hai chục ngàn đô la anh đóng cho nó một năm, tức là tối thiểu tóm lại anh phải làm 3 cái assessement anh mới thu hồi được vốn." Lúc đó tên tuổi của anh nó nhận ra. Tôi khuyên mọi người không nên dính vào chuyện đó. Trước sau gì một thời gian nữa cũng có sự thay đổi. Và cái gì cũng vậy, nếu mình không ngăn được thì cứ để nó làm vậy đi. Đến khi nào nó mất hết giá trị đi rồi thì nó đổi. ISO cũng vậy. Thực sự những cái đó gần như không còn giá trị nào nữa. Vì chúng tôi biết rất rõ ISO, với ITIL với CMMI, bây giờ những cái đó người ta chỉ trưng bày mà thôi. Chính tôi đi sang bên Ấn Độ tôi nhìn thấy công ti bán gạo bảo "Chúng tôi cũng ISO đây," công ti bán trái cây "Chúng tôi cũng ISO đây," rồi công ti phần mềm cũng CMM level 5. Ở đâu cũng vậy, người nào cũng có cái đó. Mà như thế thì còn giá trị gì nữa. Mà thực sự họ cũng không hiểu cái đó là cái gì. Bây giờ các anh chị đọc các quyển sách họ viết cũng không hiểu nó nói cái gì. Tôi sẽ giải thích rất rõ rệt cho anh chị nó là cái gì. Còn câu hỏi nào nữa không?

3. Điểm then chốt của CMM: trưởng thành năng lực

Điểm quan trọng nhất của CMM là process, qui trình. Chất lượng của một sản phẩm phần mềm dựa vào chất lượng của qui trình làm ra nó. Nếu mình muốn có

chất lượng sản phẩm của mình tốt, mình phải cải thiện qui trình làm việc. Tức là muốn cải tiến sản phẩm, improve product thì phải cải tiến qui trình, improve process. Thay vì ngày xưa theo phương pháp bình thường làm phần mềm tức là làm xong, hỏng lại sửa, các anh chị thấy không ạ: viết code xong test, hỏng đâu sửa đó. Bây giờ phương pháp này không làm như vậy được là vì anh chị thấy những dữ liệu tôi đưa ra, các yêu cầu, requirements, nếu sửa chữa phần mềm khi nó đã thành một sản phẩm rất là tốn kém. Do đó phải đặt vấn đề chính làm sao để qui trình mình làm việc cho thật tốt, thật hay. Đây là cái nguyên tắc, principal của ông Watts Humphrey ông ấy đưa ra ngay từ những ngày đầu ngồi xuống làm chương trình CMM. Tôi vẫn coi đây là cái then chốt, cái tinh hoa của giải pháp này. Tức là muốn cải thiện, nâng cao chất lượng của sản phẩm thì phải cải tiến, cải thiện qui trình làm ra nó. Câu này tôi luôn luôn đặt lên đầu.

Qui trình, nó là cái gì? Cái process hay cái qui trình đó, nó là tập hợp của từng bước một khi mình làm việc gì đó. Nó là những cái phương pháp làm việc, nó là những cách thức làm việc, và đồng thời là khả năng của người làm việc. Tức là nó làm việc psycho-process-tool-technology (tâm lí-qui trình-công cụ-công nghệ). Tất cả những cái đó cộng lại để làm. Vấn đề tôi tóm tắt ngay ở những bài ở đây là qui trình là cái người ta làm việc, chứ không phải là cái người ra ghi ra giấy tờ. Cái này là cái các anh hiện đang đi làm gọi là tư vấn phần mềm thấy nhức nhối với câu đó. Tôi nói ngay là bài giảng của tôi, tôi dạy đại học, tôi không làm tư vấn. Thành ra nếu các anh chị đến một công ti giấy tờ sổ sách rất tốt, các anh chị cũng nên cẩn thận. Nói là nhân viên phải làm thế này

thế này nhưng mà họ có làm thế hay không? Công ti nào cũng có những cái policies, chính sách làm việc phải không ạ. Có những phương pháp làm việc ghi nhận bằng giấy tờ, phải làm việc này, phải làm việc này, phải làm việc này. Giấy tờ thì nói như vậy nhưng người ta có làm việc đó hay không?

Do đó tôi đặt vấn đề đó. Process là cái người ta làm chứ không phải cái ghi nhận trên hồ sơ. Nếu chỉ là ghi nhận trên hồ sơ thì người ta bảo là anh làm phần mềm, anh phải trắc nghiệm phải làm cái này, làm cái này, ai không nghi ngờ vào những điều đó. Hiện giờ đến người trách nhiệm phần mềm và hỏi anh có policies không, anh có procedure không, anh có cái này không, nếu có thì cho tôi xem. Còn nhân viên có làm hay không tôi không cần biết. Anh chị có biết khác biệt nó là cái gì không? Nếu anh chị làm với người tư vấn, người tư vấn sẽ nói như vậy. Tôi đến công ti tôi chỉ xem là bây giờ CMM nó đòi phải có cái này cái này cái này, anh có cái đó hay không? Cứ xem cái đó xong là xong rồi. Nó không cần biết người thực hành như thế nào. Lí thuyết là như vậy nhưng người thực hành là nó không hỏi cái đó. Nếu nó hỏi, sẽ lòi ra ngay. Cho nên tôi nhấn mạnh quản trị cho ai, tài liệu để làm gì, for whom, what that for documentation? Đó là điểm tôi nói thẳng vào điều đó. Nếu chúng ta nắm được vấn đề này thì tất cả mọi việc sẽ rất là rõ rệt.

Bây giờ chúng ta nhìn vào qui trình phần mềm, có tính chất là trình tự thôi. Chúng ta có hai vấn đề. Thứ nhất, là khi làm phần mềm chúng ta phải tạo ra sản phẩm. Qui trình làm phần mềm ai cũng biết rồi, không có gì là đặc biệt cả. Còn qui trình phần mềm đây nó dựa

vào nhu cầu thị trường và đòi hỏi của khách hàng. Phân biệt rất là rõ, khi mà cải tiến phần mềm, phải dựa trên yếu tố chính của công ti đó, tức là lời lãi. Đây là vấn đề nhiều người vẫn chưa nắm được sâu sắc hẳn. Ngay từ lúc đầu, người khách hàng yêu cầu chúng ta làm sản phẩm phần mềm. Dựa trên nhu cầu của người khách hàng, dựa theo thị trường đòi hỏi, người ta làm cái này. Cái này ai cũng làm, nhưng dựa trên mục tiêu của mình, của công ti xí nghiệp đó, chúng ta mới cải thiện. Lí do tại sao? Cái này ai làm cũng được, nhưng mà lời lỗ sao không thành vấn đề. Và không có công ti nào muốn làm lỗ hết. Do đó người chủ công ti nói rằng nếu chúng tôi muốn giảm chi, chúng tôi muốn tăng thu thì chúng tôi làm thế rất là giản dị. Công ti nào khi làm việc cũng vậy, tôi giảm chi phí xuống, tôi tăng tiền vô, thì muốn làm cái đó họ phải cải thiện cách làm việc của họ, để họ đạt được mục đích đó, có phải không ạ?

Do đó phải phân định hai yếu tố đó khác nhau. Cái cải tiến improvement phải dựa trên mục tiêu cải thiện kinh doanh, business improve objectives của công ti. Nếu công ti bảo bây giờ chúng tôi phải cắt giảm ngân sách, giảm tiền vô thì họ phải làm thế nào? Nếu công ti bảo chúng tôi tăng khả năng bảo hiểm lên là như thế nào? Dựa trên yếu tố này, cả chương trình cải thiện phải dựa trên yếu tố này. Thành ra khi làm như vậy chúng ta không có vấn đề cấp 1, cấp 2, cấp 3. Cấp là vô nghĩa. Bây giờ công ti bảo, năm nay tiền lời của chúng ta là, thí dụ 100 đồng. Năm tới chúng ta muốn tiền lời lên 300 đồng thì chúng ta phải có phương pháp nào đó, áp dụng gì đó để đạt được cái mục đích từ 100 lên 300. Thì như vậy mới có lí phải không ạ? Nếu chúng ta đang cấp 1 chúng ta nhảy lên cấp 2, thì cấp 1, cấp 2, cấp 3 không có

nghĩa lí gì cả. Tôi hoàn toàn phủ nhận vấn đề nói chuyện cấp. Nếu ông chủ công ti bảo chúng ta đang ở mức số 1, chúng ta phải lên mức số 5. Các anh chị nghĩ đi, mức thứ 5 có nghĩa lí gì với thương mại của công ti đó? Thành ra rất nhiều người lầm lẫn ở chỗ đó, tư vấn nó chỉ bậy. Anh đang thế này, anh đạt được cấp 1 đến cấp 5, cấp 1 cấp 5 không nghĩa lí gì đối với vấn đề business hết. Có đúng không ạ?

Do đó khi chúng ta bắt đầu làm CMM, chúng tôi cho là mục đích để đạt đến cái gì, phải đặt mục tiêu ra. Phải dựa trên mục tiêu đó mà sự cải thiện phải nhắm tới làm sao đạt được cái mục tiêu đó, tức là business improve objectives. Tôi lấy một ví dụ khác đặt cơ sở trên nền giáo dục. Bây giờ các anh chị bảo học sinh chúng ta phải đạt trình độ như vậy. Chúng ta phải làm sao nâng chất lượng của học sinh chúng ta lên mức độ nó cao hơn. Thì cái này là mục đích chỉ đạo. Tất cả những chương trình, những improvement cải thiện phải làm sao bắt được mục đích đó. Do đó ra khỏi vấn đề kì thi. Tôi không cần biết thi tập gì hết nhưng mà làm sao nâng nó lên. Quality mới là quan trọng chứ đâu phải bằng cấp mới là quan trọng. Có đúng thế không ạ. Làm phần mềm cũng phải như vậy. Tôi nói trực diện vấn đề này. Tôi đổi lại quan niệm tư duy của những người này. Tôi đang ở cấp 1 nhảy từ cấp 1 lên cấp 3, cấp 4, cái cấp bậc đó không có nghĩa lí gì cả.

Bây giờ chúng ta đi sâu thêm vào vấn đề một chút các anh chị sẽ thấy đây là vấn đề. Đây là cái then chốt, cái chìa khoá mở cánh cửa mà làm sao những người tư vấn vẫn mắc. Thì nó có 5 cấp độ từ 1 đến 5. Trước cái 1 là qui trình làm việc ở trong công ti lộn xộn, mạnh ai nấy

làm. Không ai bảo được ai hết. Đấy là chương trình số 1. Mạnh người nào người đó làm, không ai bảo được ai. Project này chạy đường này, project này chạy đường này, ông giám đốc nói ông giám đốc, nhân viên nói nhân viên, không ai bảo được ai, cãi nhau lung tung cả lên, không ra cái thể thống gì cả. Đó là cấp 1, impossible.

Cấp thứ 2, muốn từ cấp 1 sang cấp 2 thì phải có quản trị, tức là management. Quản trị cái gì? Thì việc đầu tiên giải quyết vấn đề, trong bất kì công ti nào có rất nhiều dự án. Một công ti có thể có cả trăm dự án. Việc cải thiện đầu tiên là việc cải thiện từ cái việc căn bản, việc then chốt bắt đầu từ cải thiện dự án, project. Khi chúng ta cải thiện được dự án, thì ta đi đúng bước. Khi công ti cải thiện được cách làm việc của dự án, họ đạt đến mức 2. Mức thứ hai là họ đã nắm vững được yếu tố quản lí dự án, project management. Mức thứ hai là mức nắm vững được project management. Thế thì chúng ta thử tưởng tượng một công ti có 100 dự án. Nếu đạt được mức 2 thì tất cả các dự án đó phải thành công. Dự án dưới sự kiểm soát và điều hành của người quản lí dự án có thể khác nhau. Thí dụ bây giờ thầy Ban quản lí dự án của thầy Ban, và anh Sơn quản lí dự án của anh Sơn, mỗi người làm việc khác nhau, nhưng mà cả hai đều thành công. Đó là mức thứ hai.

Bây giờ mà ông chủ ông ấy bảo rằng là "Dự án nào cũng thành công, nhưng tôi muốn là bảo đảm trong tương lai dự án không thuộc sự chỉ huy của thầy Ban hay của anh Sơn, bất cứ ai làm dự án cũng thành công." Anh Ban có cái tài điều khiển dự án, nhưng nếu anh Ban bước ra khỏi và người khác vô làm thì có thể nó hỏng. Thế thì dự án là ở mức thứ hai. Vẫn dựa vô cái khả năng của

người quản lí dự án. Có đúng không ạ. Thí dụ nếu anh Ban là người quản lí, anh rất thành công bây giờ anh Ban bảo tôi đi hãng khác tôi làm, để người khác làm không có gì đảm bảo dự án thành công. Do đó bắt đầu đến mức thứ ba là tiêu chuẩn hoá, standardization. Tiêu chuẩn hoá tức là phải lập ra một người, hay một nhóm người đến hỏi anh Ban, "Anh Ban ơi, anh làm dự án rất thành công. Thế thì anh làm cái gì hay, cái gì dở anh chỉ cho em." Anh Ban bảo "Tôi làm như thế này, tôi làm như thế này đây." Rồi đến hỏi anh Sơn, "Anh Sơn ơi, anh làm như thế nào mà dự án rất thành công. Anh làm sao, anh làm thế này, thế này..." Ghi chép lại những cái gì mà anh Ban và anh Sơn thành công, ghi chép lại cái đó. Cái đó biến thành cái gọi là organization assets, tài sản của công ti. Dựa trên tài sản của công ti đó, nếu chúng ta mang ra phân tích, chúng ta sẽ thấy rằng giữa anh Ban và anh Sơn làm việc, nó có rất nhiều điểm tương đồng. Và dĩ nhiên cũng có những điểm khác biệt, chứ không ai giống hệt ai. Điểm tương đồng là phải có những nét chính để nó làm cho dự án đó thành công. Điểm khác biệt là do khả năng điều khiển và kinh nghiệm của người làm việc, nó có cái hơi khác một chút.

Và với tất cả những điểm tương đồng, thí dụ anh Ban làm A anh Sơn cũng làm A thì điểm A là điểm tương đồng vì cả hai cùng là A hết. Vậy A sẽ được đưa thành tiêu chuẩn mẫu mực mà tất cả mọi người đều phải đi theo. Vậy nó trở thành một cái là common process, qui trình tương đồng. Qui trình tương đồng tất cả mọi người đều làm giống nhau. Và khi làm việc có khả năng riêng của mỗi người, nó có những chi tiết không giống nhau. Cái chi tiết đó là unique process, qui trình duy nhất. Khi chúng ta có điểm tương đồng là common

assest thì tất cả mọi người trong công ti phải được huấn luyện để làm cái việc đó. Anh Ban làm cũng thành công, anh Sơn làm cũng thành công. Tất cả mọi người trong công ti phải tuân theo cái standard, cái tiêu chuẩn. Ai cũng phải làm như vậy hết. Tất cả mọi người đều phải tuân theo cái common process. Khi theo common process đó thì common process đó trở thành standard process. Tiêu chuẩn, mọi người đều phải theo tiêu chuẩn hết. Từ cái tiêu chuẩn standard process đó, nó phân biệt ra có một số điểm không giống nhau, unique, cái unique đó nó gọi là tailoring, đo may, phải không ạ. Anh lấy cái điểm tương đồng đó, nhưng mà riêng cái phần của anh cũng sửa đi một chút, cái đó nó gọi là tailoring, hay là customization. Sửa đổi lại, nó là cái riêng. Tôi sẽ đi sâu vào cái vấn đề đó, thêm chi tiết. Nếu mà các anh chị nắm vững bài số 1 thì toàn bộ CMM các anh chị không có thể sai được chỗ nào. Mình không nắm vững chi tiết đầu, về sau khi mình đi vào chi tiết nó rất là phiền phức. Tôi muốn tránh đi vào chi tiết. Là vì những người đi vào chi tiết họ đã sửa đổi nhiều quá rồi. Đây là điểm then chốt từ 1988 đến 1990 chúng tôi ngồi, ngày nào cũng nói về cái này. Các anh chị thấy không ạ?

Khi chúng ta đạt được cái tiêu chuẩn hoá, toàn bộ cho công ti làm việc theo chung của cái gọi là common processes, standard process thì tất cả mọi dự án trong công ti đều tuân theo một qui trình nhất định. Dĩ nhiên cũng có sửa đổi trong đó, nhưng mà tuân theo qui trình nhất định đó, thì đạt được mức thứ ba, tức là mức organization. Khi đạt được mức thứ ba tất cả các dự án trong công ti đều tuân theo qui trình chung, và cho phép họ sửa đổi tuỳ theo người khách hàng, tuỳ theo hoàn cảnh, có quyền sửa đổi. Khi sửa đổi thế thì có cái được

quyền sửa đổi và mà có những cái không được quyền sửa đổi. Cái common process không được đụng vô. Nhưng mà có một số thứ tôi sẽ đi sâu vào chi tiết hơn.

Thí dụ, tôi lấy một thí dụ rất giản dị, tất cả mọi thứ dự án phần mềm đều phải đi theo một method, phương pháp. Nhưng mà cái customization thì anh này làm agile, anh kia làm chu trình thác nước, có đúng không ạ. Một cái là tradition, một cái là agile. Phương pháp làm việc có thể khác nhau nhưng phải tuân theo một method. Nhắm mắt làm bừa làm bãi là không được. Cái luật của công ti, cái common đó là "You must follow the methodology. You have the choices. The choose is tradition or agile." Cái đó là customize, tailoring - may đo đấy. Phương pháp có thể khác nhau, nhưng mà phải đi theo một phương pháp chứ không có nghĩ bừa trong đầu óc một cái gì đó là không được. Tức là không làm bừa. Tất cả mọi phương pháp làm việc phải có một cái metric, phải được đo đạc. You must have measurement. Nhưng mà anh chọn, measurement dựa trên line of code hay là function points hay là objects, cái đó là tailoring. Các anh chị có thấy khác biệt không ạ? Và việc đo đạc phải được đo, nhưng mà đo bằng cách gì, đo bằng từng dòng lệnh một, hay đo bằng từng object một, hay là đo bằng function points thì tuỳ anh. Cái đó là cái khác biệt, anh có quyền thay đổi, nhưng mà phải đo, cái đo là cái standard, tiêu chuẩn cao. Còn phương pháp đo thì tuỳ người quản lí dự án lựa chọn. Nó khác biệt ở chỗ lựa chọn. Các anh chị nắm vững vấn đề đó chưa ạ. Khi làm việc như vậy là ta ở mức thứ ba, measurement, standard.

Bây giờ từ mức thứ ba muốn lên mức thứ bốn vấn đề là tất cả qui trình làm việc, process làm việc, từng

bước, từng bước một, phải đo. Và cái sản phẩm phần mềm cũng phải đo. Thí dụ bây giờ cái mình đo được là đo qui trình - process measurement, và cái mình đo được là đo sản phẩm - product measurement. Bây giờ chúng ta biết được sản phẩm phần mềm đó nó có dữ kiện, và qui trình làm việc phần mềm nó cũng có dữ kiện. Bây giờ chúng ta mới phối hợp giữa sản phẩm phần mềm và qui trình phần mềm đó để tạo ra một cái là optimization. Tức là bây giờ nếu tôi làm theo phương pháp 1, 2, 3, 4 thì chất lượng của tôi là x. Nếu tôi làm 1-2-3-4 thì tôi được x. Nhưng bây giờ nếu từ cái x đó, nếu tôi muốn được cái y cao hơn x, thì cái 1-2-3-4 chúng tôi phải tune up, tức là phải điều chỉnh, điều chỉnh sơ sơ lại một chút. Để đạt được cái mức kia thì sự phối hợp giữa product measurement và cái process measurement, tức là sự đo lường của sản phẩm và qui trình. Thí dụ năm nay chúng tôi lời 10 đồng, ông chủ bảo "Mười đồng không đủ, phải làm sao năm nay làm được 15 đồng." Thì mình đã có tất cả dữ kiện 1-2-3-4 làm sao để đạt tới 10 đồng rồi. Bây giờ mình sửa thế nào để nó lên mức 15 đồng. Anh chị có thấy khác không ạ? Cái đó nó gọi là điều chỉnh lại, cái đó là mức thứ tư.

Mức thứ tư không nằm ở mức organization nữa. Mà mức thứ tư nó nằm ở mức business. Mức thứ tư là mức lời lãi trong công ti, business của công ti, làm ăn của công ti, thị trường của công ti. Bây giờ tôi bán sản phẩm trong Việt Nam, tôi được như vậy, bây giờ tôi bung sản phẩm này ra bán khắp thế giới, tôi phải làm cái gì? Sản phẩm của tôi là như vậy, nếu tôi làm ở trong Việt Nam tôi bán được, nhưng nếu tôi bán sang bên nước khác chưa chắc đã bán được. Bởi vì tôi phải thay đổi sản phẩm làm sao để cho nó thích hợp với hoàn cảnh thế

giới. Thế thì nó không còn nằm ở vấn đề sản phẩm nữa, nó ở vấn đề organization. Có đúng thế không ạ? Mức organization đây nó phải chỉ ra. Đây là điểm then chốt những người tư vấn họ muốn giấu xuống.

Để tôi giải thích lí do làm sao như vậy. Là vì anh tư vấn, anh nói lắp bắp trong vấn đề phần mềm, không ai hiểu nói cái gì hết. Anh lên business, anh phải nói với ông chủ công ti. Một người họ nắm công ti của họ, họ đã thành công bao nhiêu năm nay rồi, một anh nào vô nói bậy nói bạ, không dễ. Thành ra tất cả những người tư vấn không bao giờ chấp nhận cái business. Họ nói rằng chúng ta làm phần mềm chúng ta ở trong địa vị phần mềm thôi. Đừng nói business, đừng nói cái chuyện đó, chuyện đó là không đúng đâu. Thành ra hiện giờ nếu các anh chị đọc kĩ CMMI đó, qui trình số 4 là qui trình phần mềm, không phải là qui trình business, nó vẫn nằm ở mức organization. Tôi đưa cho các anh chị nghiên cứu.

Các anh chị đi từ dự án, project lên tổ chức, organization, mức 4 cũng organization, mức 5 cũng organization. Anh làm cái gì ở organization nữa đây? Có đúng không ạ? Nếu chúng ta muốn cải thiện thì chúng ta phải đi lên chứ. Cái mô hình nó xây dựng phải là từ project lên organization rồi phải lên business chứ. Có đúng không ạ? Chứ nếu mình ở organization nó cứ chạy mãi ở mức thứ ba này. Sửa tới sửa lui nó cứ nằm ở mức thứ ba này thì dính dáng gì tới business? Đó là cái bí mật của CMM. Là vì cái thứ nhất, có một số anh đi nói với ông chủ hãng, "Ông phải cải thiện đi," anh là thằng làm phần mềm, anh biết cái gì về business mà anh nói. Khi đó mới lòi cái dốt của anh ra. Thành ra họ biết điều đó,

họ không nói ra vấn đề này. Đây là điểm then chốt mà tôi muốn trình bầy cho các anh các chị.

Thành ra khi mình làm việc, mức thứ 4 không đo đạc như mức ở phần mềm nữa, mức thứ 4 là đo đạc cái business của công ti. Business gì không thành vấn đề, bất kì business nào cũng không thành vấn đề. Đến mức này, đưa tất cả vào business, process như vậy, product như vậy, bây giờ muốn cái product improved thì cái process phải được thay đổi ra sao. Tức là tôi gọi là, tôi dùng cái danh từ cho xe hơi, gọi là tune up. Điều chỉnh để đạt được mục đích và mục tiêu kinh doanh, business goals and objectives. Như vậy improve là improve business chứ không phải improve cái khác, có đúng không ạ. Phần mềm anh có tốt đến đâu mà bán không ai mua thì business cũng hỏng. Chất lượng anh có cao chừng nào mà mục đích của anh không đạt được thì cũng hỏng. Do đó vấn đề phần mềm cộng với inprove phải là business, không phải là phần mềm nữa. Ra ngoài lãnh vực phần mềm, bước sang lãnh vực business. Đó là mức thứ tư, còn ai có thắc mắc gì nữa không?

Khi đạt được mức thứ tư rồi, chúng ta cần phải đạt được cái thứ ba nữa, bước thứ 5 là optimization. Tức là đạt được cái đó rồi thì chúng ta nhìn rộng ra, tất cả những cái ở dưới này phải hoàn toàn optimize, tối đa. Chúng ta đạt được cái business objectives, chưa đủ, chúng ta phải optimize business đó. Nếu năm nay chúng ta mới làm được 5 đồng, và nếu cố gắng chúng ta làm được 10 đồng, vậy thì cái 10 đồng mới là optimize. Mà nếu được cái 10 đồng chúng ta phải lên được 15 đồng. Lên được 15 đồng đến khi mà không thể lên được nữa thì lúc đó mới là optimization. Một trong những

optimization ở đây tôi thấy, theo nghiên cứu bao nhiêu năm nay của tôi, tôi thấy là: Ví dụ, công ti phần mềm rất thành công và tốt đẹp bán trong nội địa. Đến lúc này có thể bán được ra ngoài thị trường thế giới. Khi mình tiến xa, mình thấy là hai bước trên không còn nằm trong phạm vi của phần mềm, nó nằm trong phạm vi của business. Return of investment, thu hồi theo đầu tư, anh bỏ vô 5 đồng, anh được lời 10 đồng. 10 đồng đã là tối đa chưa? Nếu 10 đồng chưa là tối đa thì anh phải tối đa nó là bao nhiêu? Anh bảo "Tôi bỏ 5 đồng tôi phải được lời 10 đồng." Nhưng mà bây giờ 10 đồng chưa đủ, phải lời 15 đồng. Muốn lời 15 đồng anh phải thay đổi cái gì đó để đạt được mức 15 đồng. Cái mức cuối cùng ở đây là hoàn thiện, perfect, optimize để đạt tới mục tiêu. Đó là điểm rất quan trọng, nhưng mà khi nêu lên 2 mức trên này, thì nó vượt hơn khỏi lãnh vực của phần mềm.

4. Thực trạng tư vấn CMMI

Đó là những nhà tư vấn không biết gì về business, chưa từng quản lí công ti lớn, họ không dám nói tới vấn đề đó. Thành ra nó rất là mơ hồ. Thành ra khi tôi hỏi, tôi nói chuyện, tôi bảo, "Vâng tôi xin lỗi anh, công ti anh ngày xưa đang ở CMM số 1. Bây giờ công ti của anh là CMM số 5, vậy thì anh thấy khác biệt ra làm sao?" Ông chủ hãng bảo, "Tôi chả thấy có gì khác biệt." Tôi hỏi nhân viên, "Hồi xưa anh làm ở mức số 1, bây giờ anh đạt ở mức số 5, anh có thấy cái gì không?" Họ bảo, "Chúng tôi không biết. Có thằng nó vào nó cho chúng tôi cái certificate số 5, tôi chả thấy khác nhau chỗ nào." Đó là

câu hỏi thông thường nhất mà tôi nghe. Không biết bao nhiêu người nói thế này.

Chính tôi ngồi ăn cơm cùng với một ông chủ hãng bỏ rất nhiều tiền, tôi hỏi "Thế anh ba năm trước đây anh ở mức số 1, lời lãi bao nhiêu tôi không biết. Bây giờ anh có số 5, anh có thấy được lời nhiều không?" Ông ấy ngồi rồi ông ấy bảo, "Chả thêm được đồng nào." Tôi bảo, "Anh trả tiền cho những người tư vấn, không biết bao nhiêu tiền, anh được cái gì?" Ông ấy ngồi ông ấy ngẫm nghĩ rồi bảo "Dẫu sao thì tôi cũng được một cái certificate số 5."

Tôi bảo, "Ông trả cho cái certificate này bao nhiêu tiền?" Ông ấy ngồi ông ấy bảo, "Nói thì tôi không nói ra được, nhưng mà cũng phải mất mấy trăm ngàn."

Tôi bảo, "Thế thì lỗ quá rồi. Thế tạo sao ông không ngồi ông viết ra tờ giấy, viết số 5 rồi ông certify cho ông, tự động có cần gì ai certify cho ông, ông dán lên tường số 5. Ông muốn số 7, số 8, số 9, số 10, muốn được cái gì đều được hết. Chỉ mất một cái poster thôi."

Ông ấy ngồi ông ấy ngẩn người ra hỏi, "Nhưng lí do gì tôi certify tôi."

"Cái thằng tư vấn nó certify ông chứ có trường đại học nào certify ông đâu."

Ông ấy hỏi, "Thế Carnegie Mellon không certify à?"

"Nó là trường đại học chứ nó có phải là công ti tư vấn đâu. Trong trường học này có ai tư vấn cái gì đâu. Không thể có chữ certification. Đây là university. Not consultanting company. Vậy thì người nào certify ông?"

"Tôi thấy nó đề chữ SEI mà."

"Đứa nào nó thêm chữ SEI mà chẳng được. Thế ông hỏi nhà trường đi, ông hỏi cái trường Carnegie Mellon này xem nó có biết công ti của ông là ai không?"

Tôi nói với ông ấy là, "Đó là cái mà phải nói là tư vấn nó nói sai. Tư vấn nó không biết. Chứ tôi làm ở chỗ ấy tôi còn lạ gì nữa. Trường học chứ đâu có phải là tư vấn. Ông đến trường học ông hỏi xem có ai làm tư vấn không. Còn thằng tư vấn nó đi nó kể lung tung, nó bịa đặt ra đấy là chuyện của nó, tôi không biết."

"Ơ, mà tờ giấy của nó có chữ SEI, chữ Carnegie mà,"

"Ông đến ông hỏi đi. Trường Carnegie là chỗ để học, nó đâu có làm tư vấn. Thằng làm tư vấn, tên tuổi nó không ai biết. Nó ở đâu nó ra."

Thì lúc đó họ mới bật ngửa người ra. Ông ấy bảo, "Chúng tôi đã tốn quá nhiều tiền, không nghĩ nó bịp tôi."

Tôi bảo "Tôi không dùng chữ bịp, chữ đó hơi nặng một chút, nhưng mà tôi nghĩ ông bị lừa." Nói nhẹ đi một chút thôi.

Câu hỏi: Nhưng tư vấn cũng phải học ở đâu chứ?

Tư vấn học ở đâu? Nhà trường đâu có làm tư vấn. Nó vào học ở SEI, tư vấn dạy cho tư vấn. Tư vấn lại cấp giấy phép hành nghề cho tư vấn khác. Bây giờ nó bỏ chuyện đó rồi, bỏ từ lâu rồi. Nó thấy bê bối quá nên dẹp cái này đi từ lâu rồi. Chính phủ không đòi hỏi cái này. Nó không nhận cái này. Bây giờ chỉ có một nhóm tư vấn nó hoạt động. Chính phủ Mĩ hoàn toàn không dính dáng gì tới cái này nữa. Chính phủ nói, Bộ quốc phòng nói "I

am not to do with that." Nhà trường nói "I am not to do with that." Bây giờ một nhóm, 400 người tư vấn đó, 300 anh Ấn Độ nó lộng hành nó múa may lung tung. Chính phủ bảo "I have nothing to do with that." Trường học bảo "I have nothing to do with that." Những người khác nói "I have nothing to do with that." Bây giờ chỉ những người đó làm chuyện đó. No problem any more.

Tôi đưa vấn đề này ra để cho mọi người thấy. Bây giờ nó mơ hồ nó lặp đi lặp lại cái chuyện như vậy. Tôi đặt vấn đề, một vấn đề rất then chốt. Bây giờ anh chị có thể verify được cái chuyện đó rất dễ dàng. Anh chị đặt một câu hỏi, "Carnegie Mellon và Software Engineering Institute có certify cái công ti đó đạt mức 4 mức 5 hay không?" Câu trả lời là "We are not certification organizations. We are not in certification business. We are university, we are pure education business, have not to do with that." Vậy thì ai là người certify? No, không hề có certify. Đó là nói như vậy. "You are appraised" Anh được thẩm định, không có ai cấp bằng gì cho anh hết. Thẩm định thì người tư vấn nó bảo, anh làm giỏi lắm tôi cho anh cấp 7 cấp 8 gì đó, giữa người thẩm định với anh thôi. Nhà trường không có chuyện đó. Tôi muốn nhắc lại vấn đề như vậy.

Đó là một số vấn đề tôi gặp một số anh em tôi nói chuyện, tôi nói ra, họ cũng mập mờ vậy thôi. Một số người tư vấn họ rất là than phiền, "Anh làm phiền phức cho tôi. Tôi kiếm bao nhiêu tiền, mà bây giờ anh lại nói ra hết sự thực." Sự thực thì phải nói ra chứ sao sự thực lại không nói ra. Đó là tôi nói lại vấn đề này. Bây giờ các anh chị thử nhìn lại cái sự kiện này. Toàn nước Mĩ này, toàn thể nước Mĩ này, hai ba mươi năm nay, bao nhiêu

organizations, tôi không dùng chữ công ti, organizations, được mức thứ 5. Mà tại sao Ấn Độ bốn năm trăm, Trung Quốc ba bốn trăm, chỉ có một năm hai năm nó nhảy lên. Như một cậu bé mới sinh ra, đùng một cái trở thành ông lão tám mươi. Không biết tại làm sao. Nếu các anh chị nắm vững tất cả những vấn đề đó thì tất cả các vấn đề sau đơn giản. Đối với tôi vấn đề chính yếu là ở chỗ đó: Đây là cái mô hình original. Về sau họ sửa lại, nó không còn hai cái mức business và project nữa. Tất cả nó ngừng ở mức software.

5. Trưởng thành của tổ chức

Bây giờ trở lại thêm vấn đề maturity nó là cái gì, tôi nói thêm một chút nữa. Sự trưởng thành nó là thế này. Anh từ mức thứ 1 đùng một cái nhảy một cái lên mức thứ 5 là không thể được. Đây là cái rule, đây là cái luật của model này. Không có bỏ qua các mức được. Không thể từ 1 mà nhảy lên 3 hay 1 nhảy lên 5 được, không có.

Đây là vấn đề làm việc. Trước khi làm việc chúng ta phải tuân theo từng bước một. Bước thứ nhất phải có một sự commitment, quyết tâm. Quyết tâm phải là từ trên xuống dưới. Quyết tâm không thể nào là người dưới quyết tâm mà người trên không quyết tâm được. Do đó trong công ti phải có thượng tầng rõ rệt, cái phương hướng, direction, cái đường hướng, cái chủ trương rất là rõ rệt. Thứ hai là phải có một chương trình làm việc. Thứ ba, khi chúng ta quyết tâm rồi, chúng ta phải có tiền bạc và thời gian. Ông chủ hãng bảo tôi muốn số 5 mà không có thì giờ thì làm sao mời được tư vấn vô để lấy số 5.

Giả tiền cho nó. Chuyện đó xảy ra rất thường. Do đó tôi nắm chắc vấn đề, trước khi làm bước số 1 phải rõ là chính sách phải rõ rệt, phải có chương trình làm việc, phải có người, có tiền. Thiếu ba yếu tố đó, không hội đủ điều kiện thứ nhất, không làm được.

Cái thứ hai khi có tiền, có người, thì phải có infrastructure, nòng cốt phải xây dựng căn bản chứ không chỉ một người hai người làm như vậy.

Thứ ba nữa ấn định trở lại giới hạn làm việc. Tôi lấy một thí dụ công ti Boeing bây giờ có gần 500 organizations. Vậy thì cái scope có thể là toàn bộ công ti được không? Chắc chắn là không. Đó thì phải giới hạn là từng khúc một, từng organization một, chứ không thể toàn bộ cái gì quá lớn được. Thí dụ trong công ti Boeing, rada là một nhóm làm về rada, nhóm đó được đánh giá, assessed, nhóm đó được mức 5. Chế tạo có phân chia ra cấu trúc khác, nó phân chia ra các thể loại. Nếu tôi làm ở công ti phần mềm, tôi chia theo miền, chẳng hạn nhóm làm agile là một group, nhóm làm traditional là một group. Trong cái nào lớn quá, tôi lại phân biệt nó ra. Thì nhóm này làm theo phương pháp này, finance là một group nhé, HA là một group nhé, những nhóm khác, kĩ nghệ vào một nhóm. Mình cứ theo miền thì dễ hơn vì tiêu chuẩn, giữa cái tài chính và kĩ nghệ nó khác nhau. Nó không cùng một tiêu chuẩn được. Đó là cách phân chia theo miền. Nếu có một phòng ban tài chính, nó quá lớn, như công ti tôi tài chính có 10 nghìn người chẳng hạn, thì trong cái tài chính đó lại chia nhỏ hơn, kế toán vô một nhóm khác, makerting vô một nhóm khác, thuế đi một nhóm khác. Mình phải phân chia theo các nhóm nhỏ.

Theo tôi cái phạm vi tối đa là vào khoảng ba ngàn người. Vào khoảng đó, vì đó là cách tiếp cận, theo ý tôi, nếu nhóm dưới 150 người hay là dưới 200 người thì làm được. Tối ưu, theo ý tôi tối ưu là được, nhóm nó lại vào khoảng 300 người. 300 người đến 3000 người là kích cỡ trung bình. Còn nếu hơn 3000 người thì khó quá. Nếu nhỏ hơn 100 thì theo tôi không nên làm. Theo tôi phạm vi vào khoảng từ 300 tới 3000 người là tối đa. Nếu lớn hơn thì phải phân nhỏ ra. Nếu chúng ta làm một cái 10 ngàn người hay 100 ngàn người thì vô lí. Vì không thể 100 nghìn người làm hệt như nhau, đúng như nhau. Chuyện đó không thể có được. Dù anh muốn đến đâu đi chăng nữa chuyện đó cũng không thể xảy ra. Thứ hai, một công ti quá lớn như vậy anh tài chính làm tài chính, anh kĩ nghệ nữa, hai phương pháp làm việc nó khác nhau hoàn toàn, làm sao giống như nhau được. Thành ra tôi tính vấn đề miền, domain, cách tôi chia trong công ti của tôi, cách tôi áp dụng tại Motorola, tôi áp dụng tại GE tôi áp dụng tại Boeing đều đi theo cái chuẩn như vậy.

Sau khi phân nhóm được rồi, một vấn đề khác là mình phải có những người chuyên môn. Đây là vấn đề chuyên môn. Phải có những người chuyên môn được huấn luyện để làm tác nhân thay đổi, change agents, những người là yếu tố thay đổi, cải thiện công ti. Đây là một việc làm của những người professionals, chuyên nghiệp. Đây không phải là ông chủ chỉ đạo người đó làm. Người nào làm change agents đều phải được huấn luyện tất cả. Đó là vấn đề tư vấn nó vô nó nói cả ngày , để anh chị biết rằng là câu chuyện xảy ra rất thường.

Nói cho nó vui nhé. Bây giờ thử tưởng tượng, tôi đi làm việc rất là bận. Ông chủ hỏi tư vấn, tư vấn bảo

"Phải có một người làm thay đổi công ti." Ông chủ nhìn tới nhìn lui bảo, "Thằng nào cũng bận hết. Chỉ có anh đó mới ra trường ngày hôm qua, tôi mướn nó vô, không có việc gì làm cho nó cả, cho mày làm change agents mày đi mày sửa đổi công ti đi." Tức là một anh mới ra trường, tuổi đời chưa được bao nhiêu, đến bảo "Chị ơi chị, chị làm thế này sai rồi. Chị phải làm theo CMM nhé." Chị ấy bảo, ""Tao làm ba chục năm nay rồi, mày là cái thằng gì mà đến bảo chị mày thay đổi?" Anh chị có thấy cái nghịch lí đó không ạ? Thành ra tôi luôn luôn đặt vấn đề người change agent này thứ nhất phải được huấn luyện, thứ hai phải là người rất kinh nghiệm, thứ ba phải là người rất có uy tín, thuyết phục, thì nói người ta mới nghe chứ. Đó là cái lỗi lầm căn bản của rất nhiều công ti.

Tóm lại tôi đưa ra vấn đề là không có commitment. Anh muốn làm việc mà chuyện này không thành thật, anh bảo, "Làm qua loa cho nó xong đi." Thế thì câu này, khi tôi nói chuyện với các công ti, họ nói thế này, họ bảo, "Khổ quá, trong công ti của tôi người làm giỏi nhất, có uy tín nhất. Tôi thí dụ đây, anh Sơn là người làm giỏi nhất, uy tín nhất, nhưng anh ấy bận lắm, anh ấy làm bao nhiêu việc quan trọng, đi họp, bây giờ bảo anh ấy làm cải tiến, chuyện đó là vô lí. Anh nào cũng cải tiến thì hỏng hết công việc. Anh là người quá quan trọng nên tôi không thể nào để anh làm việc đó được. Đây có anh Phú mới ra trường, anh làm việc đó đi nhé." Sẽ không có quyết tâm.

Tôi hỏi ông chủ, tôi nói, "Vâng, thưa anh, anh Sơn là người quan trọng rất có uy tín. Bây giờ tôi hỏi anh là, liệu công việc hàng ngày của anh ấy có quan trọng hơn công việc cải thiện hay không? Nếu có, thì tôi ra, Okay.

Đúng không ạ. Là vì nếu ông làm ông hỏng, ông tốn tiền. Bây giờ mình đưa anh Phú, anh Phú mới ra trường, tôi nói vui nhé. Anh Phú mới ra trường chưa biết gì cả, vừa mới tốt nghiệp đại học. Tốt nghiệp đại học CMU chương trình của Văn Lang. Đưa ra yêu cầu thay đổi tất cả mọi thứ. Anh Phú gặp thầy Dũng mới bảo là "Thầy Dũng ơi, thầy có biết CMM là cái gì không? Phải làm thế này, làm thế này thôi." Anh Phú được bảo ngay, "Cửa đây, anh đi ra đằng kia chơi." Có đúng không ạ. Người làm phải đi đôi với uy tín. Anh có invest, anh có đầu tư không. Nếu công ti của anh, những người nhân viên giỏi nhất không thể sử dụng được, tức là anh ở mức thứ mấy? Mức thứ 1, mức thứ 2. Bận rộn bù đầu bù cổ, không ngửng đầu lên nhìn được cái gì nữa thì anh cải thiện làm chi? Đúng không ạ? Công việc nó vẫn chạy, người nào người nấy bận rộn. Khi mình muốn cải thiện mình phải bỏ thời giờ, mình phải bỏ công sức, mình phải bỏ tiền bạc, để mình cải thiện nó tốt hơn. Mà nếu anh không chịu đầu tư, cái vấn đề anh chỉ chính đáng, thì anh chỉ tốn tiền vô ích thôi, chắc chắn anh sẽ không thành công. Thí dụ một người còn chưa có uy tín, nói không ai nghe, bắt người khác phải thay đổi, người ta sẽ càng không nghe nữa. Thay đổi là khó khăn vô cùng, không thể thay đổi một cách nhanh chóng được, chuyện đó không thể có được.

Muốn cải thiện phải đi từ từ. Bởi vậy người ta dùng chữ maturity, trưởng thành. Nó không dùng chữ acceleration, tăng tốc đâu. Có ai lo cái chuyện đó được đâu. Cái duy nhất mà cải thiện rất là nhanh là đột phá, break through. Đột phá chỉ xảy ra khi nào có thay đổi rất là lớn. Thường thường nó là công nghệ, technology. Bây giờ nói với chúng ta là cái phương pháp này, thí dụ anh

nào làm như Intel, anh chế ra con chip chạy nhanh gấp 10 lần mà nhỏ gấp 10 lần, nó làm cái đột phá, nó đổi cái máy. Bây giờ có anh nào chế ra được cái gì hay hơn cái Microsoft Windows rất là rẻ, rất là hay, anh cho không chẳng hạn. Cái gì cũng chạy tốt 100 phần trăm thì bảo đảm một thời gian ngắn là anh Microsoft sập tiệm thôi. Cái đó không là thay đổi, cái đó là đột phá. Internet chẳng hạn, là một cái đột phá. Điện thoại di động là cái đột phá. Từ chỗ không có điện thoại di động, bây giờ 99 phần trăm mọi người có. Đồng loạt. Bây giờ có ai dùng điện thoại thường nữa đâu, đúng không ạ? Từ khi có internet rồi bây giờ có bao nhiêu người dùng thư gửi tem, bao nhiêu người gửi thư dùng email. Cái đó là cái đột phá, break through. Bưu điện bây giờ lỗ be lỗ bét ra. Người ta không viết thư nữa, người ta gửi email. Những cái đó là cái đột phá, break through, những sự thay đổi nó dựa trên công nghệ. Chứ còn cải thiện vấn đề thì nói bình thường, mình muốn làm, anh phải quyết tâm, commit, anh phải có kết cấu nền, infrastructure.

Đây là bước thứ hai, đây là điều kiện ắt có và đủ. Phải có mới làm được. Điều kiện thứ ba, phải đo được. Đa số công ti làm cải thiện, quên yếu tố đo. Quá chú trọng vào sản xuất, phải chứng minh bằng con số. Con số không nói dối mà con người ta thì nói dối. Mọi người bảo, "Con số có thể nói dối được." Nếu anh là chủ nhân, ai nói dối anh đây? Vấn đề là mình có là ông chủ không. Tiền lời tiền lỗ ông ấy phải biết chứ. Công việc của ông, business của ông mà. Nhân viên có thể nói dối ông, nhưng ông không thể tự nói dối ông ấy được. Vậy thì cá nhân con người phải nêu ra, phải đo lường được, yếu tố chỉ có thể đo được.

Cho đến khi đạt được ba cái này rồi thì mới bắt đầu áp dụng. Không đạt được ba yếu tố kia, không nên làm.

Câu hỏi: Có thể mời một chuyên gia ở ngoài công ti tới giúp làm CMM không? Công ti không biết cách làm cải tiến thế nào thì mời một chuyên gia ở công ti khác đã làm được CMM tới giúp làm cải tiến được không?

Để tôi hỏi anh việc này nhé: Người công ti khác biết cái gì về công việc của anh. Anh có thể mang người thật giỏi vô làm tư vấn, ok. Những tác nhân thay đổi, change agents không phải là tư vấn. Tư vấn là người hướng dẫn thôi, giảng dạy cho anh, chỉ bảo cho anh thôi. Người thay đổi phải là người nội bộ của anh chứ. Đúng không ạ? Vậy thì anh phải có tác nhân thay đổi change agents nội bộ của anh. Người đó không có kinh nghiệm thì phải đi học từ người tư vấn. Nhưng người làm phải là người trong công ti của anh chứ. Bây giờ tư vấn làm sao nó vô nó thay đổi công ti của anh dược? Vậy thì tác nhân thay đổi khác với người tư vấn. Tác nhân thay đổi là người làm cái sự thay đổi đó, người hiện thực cái đó, người đổ mồ hôi nước mắt để làm cái đó. Còn ông thầy ông ấy giảng dạy bên ngoài, anh học của người đó, rồi anh về anh làm việc áp dụng vào công việc của anh. Học bên ngoài là việc của người tư vấn. Nó khác nhau như vậy. Người tư vấn không bao giờ là tác nhân thay đổi. Tôi khẳng định vấn đề đó. Anh tư vấn chỉ bảo cái này anh sai ở chỗ đó đó, anh sửa chỗ đó như vậy. Họ có kinh nghiệm họ chỉ như vậy. Nhưng mà người làm phải là anh chứ. Chứ người tư vấn họ vô họ có làm đâu.

Bây giờ anh mang một người giỏi từ công ti khác vào, thứ nhất người đó biết gì về công ti của anh? Người

đó có uy tín với công ti hay không? Một người chân ướt chân ráo mới bước vào mà chỉ trỏ lung tung thế nào cũng bị ghét lắm. Anh bảo một người lạ mới vô chân ướt chân ráo không biết gì cả mà cứ vô chỉ trỏ, "anh không biết để tôi chỉ cho anh làm." Chẳng thể nào thành công được. Người ngoài không thể nào làm thay đổi được. Phải người trong, người trong cũng phải nghĩ chín rồi. Anh đòi thay đổi cách thức làm việc, anh đòi thay đổi một nếp làm việc có từ bao lâu nay rồi, không phải chuyện dễ. Lí do công ti, bây giờ nói vấn đề cải thiện qui trình, process improvement, trên nước Mĩ này chỉ có độ 3 công ti số 1. Boeing là một trong những công ti số 1. Mọi người hỏi tôi là "Tại sao nói tới chương trình CMM thành công với những con số với những dữ kiện thật là ghê gớm. Tại sao tôi thành công?" Tôi chỉ trả lời một câu hỏi rất giản dị, "Tất cả những người thay đổi đều là nhân viên quản trị." Trong công ti tôi tất cả những tác nhân thay đổi change agents đều là nhân viên quản trị. Tối hậu là ông chủ tịch, president, là người bỏ tiền ra, là người chỉ huy. Bây giờ tôi nói trong công bi Boeing, tôi chỉ nói về engineer thôi, tôi không nói về manufacturing, các chuyện khác. Tôi là kĩ sư trưởng, chief engineer, người chỉ huy về kĩ thuật cao nhất là tôi. Trên tôi là ông CEO, ông CEO bỏ tiền ra, tôi chịu trách nhiệm về chuyện này. Tôi bắt tất cả các ông tác nhân thay đổi phải là người quản lí không phải là kĩ sư. Đó là yếu tố thành công.

Nhân viên quản trị thay đổi được. Nhân viên cấp dưới không thay đổi được. Nếu nhân viên quản trị thay đổi thì nhân viên khác theo ngay. Cấp quản lí phải là người đổi trước thì nhân viên đổi sau. Nhân viên đổi, cấp quản lí không đổi, cũng không đổi được gì. Các anh chị hiểu chỗ đó không ạ. Thành ra những công ti khác họ gọi

là benchmarking, họ làm không thành công, họ đến tôi họ xem. Họ bảo là "Ai là người sponsor?" "Ông CEO." "Ai là người trông chương trình improvement?" "Tôi." Anh là ai?" "Tôi là kĩ sư trưởng. Tất cả các kĩ sư làm việc dưới quyền tôi." "Vậy thì ai là tác nhân thay đổi?" "Mọi mức quản lí, từ cao xuống thấp." Vậy thì ai là người làm việc cấp quản lí làm? Kĩ sư có làm không? Tôi bảo "Không, không, kĩ sư không làm, cấp quản lí bảo làm sao anh ta làm vậy. Không phải kĩ sư thay đổi mà là cấp quản lí. Khi cấp quản lí thay đổi, công ti thay đổi. Vậy quyền hạn thay đổi phải là một người có uy tín. Người có uy tín nhất là người quản lí. Thí dụ anh chỉ huy 50 nhân viên, anh bảo "Tôi muốn làm như thế này," thì trong 50 nhân viên đó dù thích hay không thích, nó phải nghe lời chỉ huy chứ. Làm gì có chuyện anh nhân viên bảo "Tôi không làm như vậy." Có đúng không ạ? Mọi người bảo, "Nhưng mà cấp quản lí rất bận. Không có thì giờ làm." Thế thì thôi đừng làm nữa, làm việc khác đi. Anh đầu tư, anh bỏ tiền, anh bỏ công, anh bỏ của, anh bỏ thời gian, thì anh phải nhắm cho nó thành công chứ. Anh lại làm nửa chừng ba hồi có làm ba hồi không thì làm làm chi. Thành ra đa số các công ti tới tôi, tới chín mươi phần trăm không làm.

Yếu tố thành công không có, kinh nghiệm thành công không có, tôi mang bài tập của tôi đi giảng dạy khắp tất cả mọi nơi, cho không. Anh nào sau khi học xong cũng bảo, "Tôi học với thằng tư vấn, thấy dễ mà tôi học với thầy thấy khó quá." Tại sao, tại vì mình bắt họ phải làm. Tôi dùng một thí dụ thế này. Ở bên Mĩ có cái mảng là thế này, những người ăn nhiều quá thì họ cũng hơi béo. Tôi nói đùa tôi bảo, "Bây giờ anh béo, anh tập cách dễ, anh tập phương pháp này, uống viên thuốc này

là nó tiêu ba chục cân ngay lập tức. Đổ xô đi mua thuốc và không bao giờ nó hết. Anh phải vô, anh phải làm việc, anh phải tập thể thao." Họ bảo "Tập thể thao mệt lắm, bây giờ có thuốc nào tôi chỉ uống vô có một viên là xong." Có đấy chứ. Ba bốn chục năm nay rồi không biết năm nào cũng cả ngàn viên thuốc, mà tóm lại nó là gì, nó chỉ là bột thôi. Anh bỏ ra rất là nhiều tiền. Năm trước vừa bỏ ra mua thuốc A, uống vô, nó mượn một cô có hình mẫu rất đẹp như cô tài tử xi nê. Xưa cô ấy nặng lắm, uống vô là cô gầy ngay. Uống thuốc này vô, bỏ tiền mua xong, năm tới mọi người thấy không công hiệu, hãng thuốc nói "Thuốc này dở, bây giờ mới có thuốc mới." Năm nào cũng có người đi mua thuốc, năm nay thuốc A, sang năm thuốc B. Ba mươi năm nay người ta vẫn mắc cái chứng đó. Là vì người ta lười.

Tôi bảo có 2 phương pháp, một là ăn ít đi, hai là tập thể thao. Chứ không thể nào uống viên thuốc mà người gầy người đẹp lên được. Trừ phi có cách nào khác thì không nói. Phương pháp này cũng vậy. Cho dù đôi khi nói khôi hài một chút tôi bảo là, "Các anh phải biết không thể có viên thuốc nào uống vô người đẹp lên mà lại xuống cân, đẹp như vậy tôi chắc chắn là không có." Mà bao nhiêu ngàn bao nhiêu triệu người vẫn nuôi ảo vọng có viên thuốc như vậy. Không có. Nếu anh muốn gầy, anh muốn khoẻ mạnh, anh phải tập thể thao hoặc là anh ăn ít đi. Chỉ có hai giải pháp đó thôi. Ăn thật nhiều mà không tập thể thao rồi uống viên thuốc mất ba bốn chục cân, chuyện đó không thể nào có được. Thành ra lí do là vì thế mà kĩ nghệ bán thuốc nó vẫn còn.

Tôi muốn nhấn mạnh là khi muốn thành công cần phải hội đủ bốn yếu tố sau đây. Nếu không làm được bốn

yếu tố này thì theo ý tôi là không nên làm. Vì làm mất thì giờ vô ích, mất công, mất của, mất thời gian mà nó không đạt được kết quả mong muốn.

Lúc nãy có một số anh em hỏi tôi là có phải phương pháp này áp dụng cho phần mềm nhưng có áp dụng cho các phần khác được không. Tôi nói là đến mức này, cái mô hình này nó đang là cái mô hình toàn diện, nó chưa đi vào phần mềm. Thành ra tất cả cái này đều áp dụng cho bất kì cái gì cũng được. Các anh chị có thể thay đổi bất kì cái gì dựa trên phương pháp này vì hiện nay chưa nói gì về phần mềm hết. Đã nói gì tới software đâu.

Tôi lấy một ví dụ cho nó vui. Nếu các anh chị nghĩ là mình mang một phương pháp làm việc để mình thay đổi, thí dụ như trường học. Tôi có thể phân tích như thế này. Đây là một cái đại học. Cái khởi đầu này, mình coi một trường học, mạnh phân khoa nào, mạnh giáo sư nào thì dạy theo phương pháp đó. Chúng ta đi đến project, tức là ở level 2, những người giáo sư dạy tốt, mỗi một người giáo sư có thể dạy rất là hay, theo ý mình, nhưng lớp học rất là tốt. Tôi coi project là một lớp. Bây giờ dựa theo cái tài liệu các giáo sư giảng dạy chúng ta thống nhất hoàn chỉnh lại, để cho mọi người cùng dạy, cùng trao đổi làm việc với nhau. Thì cái organization đối với tôi nó là một cái phân khoa. Từ phân khoa nó lên businees tức là nó lên college. Tôi không biết danh từ Việt Nam phân biệt university với college thế nào. Ở đây phân biệt rất là rõ rệt. Engineer là một college, một school, engineer là một cái, business là một cái, finance là một cái. Mình gọi đây là cái viện. Bây giờ lên cái cuối cùng đây là trường học đây, đây là university đây, viện trưởng nằm ở đây. Có đúng không ạ. Cao nhất là ông

president nằm ở đây, xuống đến đây là dean rồi. Dean of college, từng phân khoa một. Xuống đến đây nó là college, chẳng hạn trong engineer nó có 7-8 thứ engineers có đúng không. Xuống đến đây là giáo sư dạy từng lớp một. Thì mô hình này có thể áp dụng cho trường học được như với các mô hình khác. Từ trước tới nay mọi người nghĩ mô hình này áp dụng cho phần mềm. Nhưng mà chưa đi đến phần mềm.

Tất cả những cái này, đây là tôi coi là công trình của Crosby và Deming và Watts Humphrey. Ba người đó làm việc với tôi trong suốt 2 năm đầu. Khi mà cái mô hình thành công, ông Watts ông ấy chỉ huy xuống và ông ấy bảo các anh đi vô chi tiết viết cho phần mềm. Ngay lúc đó trong đầu tôi đã bảo nếu mình đi vô chi tiết thì viết cho phần cứng như thế nào, nếu mình đi vô chi tiết thì viết cho không phải phần mềm như thế nào. Ông Crosby trả lời rằng là "Cái mô hình này là tổng quát, generic, áp dụng cho rất nhiều phương pháp, nhưng mà vì chúng tôi được đưa vào để làm cải thiện phần mềm, thì các anh là những người phần mềm, chuyên gia phần mềm, các anh đưa chi tiết vô. Cái khuôn khổ, cái khuôn khổ trưởng thành áp dụng cho mọi thứ." Thành ra dựa trên kinh nghiệm đó, sau khi thành công chương trình phần mềm này, tôi với Bill Curtis ngồi làm việc với nhau và ra cái mô hình mới gọi là People CMM cải thiện con người trong công ti, cải thiện nhân viên. Sau đó, sau thời gian đó, chính tôi là người làm cái chương trình nữa, cũng dựa trên cái mô hình này cải thiện cái e-business và e-government. Sau nữa tôi làm thêm cái mô hình nữa là cải thiện cái khoán ngoài, outsourcing. Vì tôi là người nắm rất vững qui trình này, do đó tôi sáng chế ra một số cái. Hiện giờ tôi đã làm xong thành công 4-5 thứ mô

hình rồi nhưng tôi không muốn nó bị lệ thuộc vào vấn đề tư vấn, cho nên tôi giữ những cái tôi làm ở trong trường chỉ dạy cho sinh viên trong trường nhưng cấm không được đi ra ngoài làm tư vấn, tôi gạt cái vấn đề đó ra. Thành ra những cái mô hình sau chúng tôi chế ra một số giáo sư chỉ áp dụng cho một số lĩnh vực giới hạn. Vì mang ra ngoài là bị tư vấn nó vào nó làm lệch đi, tôi không muốn như vậy. Thành ra chương trình về e-business, chương trình về outsourcing, chương trình về business, chương trình về làm cái khác, được dạy trong trường như một môn học, nhưng không có đưa vấn đề đánh giá, cắt chương trình ra. Thong thả tôi sẽ nói thêm về những phần khác.

Tôi nắm rất vững những vấn đề này. Lúc mà làm với Crosby, với Deming và Watts, lúc đó có 16 người thôi. Tôi thì lúc đó ngày đêm ngồi làm việc sát với Crosby, văn phòng ngồi bên cạnh nhau. Thành ra tôi biết là chúng ta có thể áp dụng cái mô hình này, nó mang tên là mức độ trưởng thành, maturity level, không phải là phần mềm, software. Mô hình trưởng thành thế này, cái mô hình trưởng thành có thể áp dụng trong nhiều phương diện khác nhau. Cho đến khi áp dụng vào phần mềm thì đưa phần mềm chi tiết, khi áp dụng vào chỗ khác thì đưa thứ khác vô, có thể áp dụng chỗ nào cũng được.

Và nếu áp dụng thì phải hội đủ bốn điều kiện, hội đủ bốn bước này. Tôi gọi đây là điều kiện ắt có và đủ. Phải có, phải đủ cái này thì nó mới được. Không có đủ bốn cái bước này thì sẽ không làm được. Đây là bốn bước đầu, bốn bước cơ sở của nó. Đây cũng là cái lộ trình rất quan trọng. Như chúng ta cải thiện, chúng ta cải

thiện cái gì? Chúng ta phải nhìn lại. Bất kì một cái nhóm, một hội, một tổ chức nào cũng đều có mục đích, cái goals. Mục đích gì? Với mục đích đó, bây giờ mình dựa vào cái goals của CMMi này, mình dựa vào cái mô hình maturity này. Cái này là hướng dẫn, guidance, nó không phải là chuẩn, standard. Nó chỉ là hướng dẫn, các anh chị nhớ nó khác với chuẩn. Standard là tiêu chuẩn phải theo. Đây nó là guide thôi, đó là vấn đề người tư vấn với chúng tôi cãi nhau rất nhiều. Họ bảo đây là tiêu chuẩn, standard mình phải theo. Tôi bảo không, đây là hướng dẫn, guidance. Bản đồ thôi. Khi mình làm mình phải có một cái plan. Mình phải có một chương trình làm việc. Thế thì chương trình này nó dựa theo khả năng của chúng ta đang ở mức nào. Mức độ trưởng thành của chúng ta đang ở mức nào.

Cái CMM này nó giúp cho chúng ta biết chúng ta đang ở mức nào. Chúng ta đã học mức 1, mức 2, mức 3, mức 4 thí dụ như vậy. Dựa vào bước này chúng ta tạo ra một dự án để làm việc. Từ cái dự án này ta phân nó ra làm nhiều phần nhỏ. Từ những phần nhỏ này chúng ta lập tức có hai phần. Một phần là áp dụng nó, implementation, và một phần là đo đạc nó, để xem cái hiệu quả của thực hiện là như thế nào. Và đây là cái bản đồ. Khi chúng ta có cái này, và chúng ta áp dụng cái này, chúng ta phải có huấn luyện. Thành ra đối với tôi môi trường huấn luyện training rất là quan trọng. Mình không huấn luyện cho người ta, người ta không biết làm. Do đó khi mình đưa xuống nhiệm vụ này, ngay cái trang này, rằng anh phải làm cái này, việc này, việc này. Đêm ngày chúng ta phải có huấn luyện ra làm sao. Huấn luyện người ta làm cái việc này ra làm sao. Hai cái này hợp với nhau mới thành cái implementation. Từ cái

implementation này rồi cái áp dụng này nó mới đo. Tất cả các dữ kiện đo lường được phải nằm ở cái database. Và con số này phải được phân tích đi ngược lại so với điều chúng ta có đạt được mục đích này hay không. Đây là điểm then chốt, tôi gọi đây là tinh hoá của mô hình này, model. Thành ra đây như cái bản đồ, nếu chúng ta làm đúng theo cái này, thì chắc chắn chúng ta đạt được.

Thứ nhất, chúng ta phải biết chúng ta đang ở đâu. Bây giờ mình đi từ Sài Gòn ra Hà Nội. Bây giờ mình đang ở Sài Gòn hay mình đang ở Vũng Tàu, hay là mình đang ở Nha Trang, hay là mình đang ở Huế. Mình phải biết mình ở đâu thì mình mới biết được mình còn xa hay còn gần. Chứ còn không biết mình đi đâu, không biết mình đang ở đâu hết, là bị kẹt. Ai cũng biết cái mục đích của mình, nhưng mà không ai biết mình đang ở đâu. Người ta lạc đường không phải là người ta không biết người ta đi đâu, người ta lạc đường là vì không biết là người ta ở đâu. Tôi nhắc lại điều đó, không có người nào lạc đường mà người ta không biết người ta đi đâu hết. Người ta lạc đường, người ta không biết người ta đang ở đâu. Có đúng thế không ạ. Các anh chị đi từ Sài Gòn ra Hà Nội, mình biết mình đi đến đâu chứ. Mình biết mục đích của mình là đi ra Hà Nội. Nhưng mà bây giờ lạc đường, không biết mình đang ở đâu, mình đang ở Nha Trang, mình đang ở Đà Nẵng, hay mình đang ở Huế hay mình đang ở Sài Gòn. Mình không biết mình đang ở đâu thì mới lạc đường chứ. Mình biết mục đích của mình chứ. Thành ra ai cũng nhầm ở chỗ đó, người nào cũng biết mục đích của mình, nhưng mà họ không biết cái khả năng của họ. Vì không biết khả năng mà rối rắm lung tung. Phương pháp này nó xác định cho, giải quyết cái khả năng của mình, mức độ triển khai của mình, khả

năng phát triển của mình. Mình phải biết được mình có thể làm được gì. Từ cái chuyện mình biết được mình đang ở đâu đây, trong cái mô hình trưởng thành này, mình mới tham khảo tài liệu, rồi mình vẽ lại cho mình cái bản đồ. Đây là cái bản đồ, từ cái bản đồ này chúng ta phát triển từng bước một. Thí dụ chúng ta đi từ Sài Gòn ra Hà Nội, chúng ta phải đến Vũng Tầu trước đúng không ạ. Rồi sau đến Nha Trang, xong đến Đà Nẵng, xong đến Huế, lúc đó mới ra đến ngoài kia chứ đâu có phải nhảy một cái là bay đến ngoài kia đâu, trừ phi bay tầu bay thì không kể. Đây là từng bước một đấy. Mà mỗi bước này đi cần có sự hướng dẫn, sự huấn luyện. Phối hợp giữa sự huấn luyện với các bước này mới đi đến cái áp dụng. Trong áp dụng thì phải có đo lường. Đi phải đi trái đi tiến đi lui. Chúng ta phải đo lường cho chính xác thì mình mới biết mình có đạt tới mục đích hay không.

Tôi làm tạm một cái bản đồ, các anh chị đã thấy chưa, có thắc mắc gì nữa không ạ. Thành ra phần đầu tôi dạy rất kĩ về chỗ này. Còn thực sự khi các anh chị nắm vững vấn đề này, tất cả những cái về sau không thành vấn đề. Và cái này vẫn là áp dụng CMM, chưa phải cho phần mềm, chưa phải software. Trong CMM nó có một cái tôi gọi là tinh hoa, mình gọi nói chung, tức là cái bí mật. Nếu mà nói theo danh từ truyện kiếm hiệp thì đây là bí kíp. Chùa Thiếu Lâm giữ bao nhiêu kho sách, thì đây là bí kíp của họ. Đây là bí kíp của phương pháp này. Nó nắm bí kíp này thì nó thành các mức gì. Các anh chị đọc truyện chưởng thì các anh chị biết rồi đấy. Bí kíp nằm đây, Cửu âm chân kinh.

Đây là bí kíp, bí kíp có 7 phần thôi, chỉ có 7 câu bí kíp thôi. Nếu các anh chị nắm được bí kíp này thì luyện

cái môn gì cũng thành công hết. Tôi xin giải thích về bí kíp. Thứ nhất là bây giờ chúng ta nói là chúng ta thay đổi, chúng ta cải thiện, improve. Việc đầu tiên là cải thiện cái gì. Định nghĩa. Bây giờ chúng tôi muốn thay đổi qui trình làm việc, cái qui trình làm việc phải được định nghĩa ra, nói nó ra, đúng không ạ. Phải định nghĩa nó ra, nói nó ra. Tóm lại, muốn làm phần mềm phải làm như thế này, phải làm như thế này, phải làm như thế này. Phương pháp làm việc phải nói nó ra. Và tôi nói lại, đến bây giờ vẫn chưa là phần mềm. Đến bây giờ vẫn là mô hình trưởng thành, maturity model chứ chưa phải là trưởng thành phần mềm, software maturity. Do đó nó có thể là hardware, nó có thể là e-business, nó có thể là e-government, nó có thể là bất cứ điều gì, nó vẫn dựa trên cái này. Đây là tâm huyết của Crosby và Deming. Hai năm trời tôi với ông ấy đêm ngày ngồi uống bia nói chuyện với nhau trên trời dưới biển, chỉ nói mỗi vấn đề này mà thôi. Thành ra tôi có thể nói đây là tâm hồn của Crosby và Deming nằm ở đây. Tôi là người đi sát với các ông ấy nhất. Hai năm trời gần như ăn ngủ ngồi nói chuyện với nhau. Ông ấy huấn luyện cho mình để mình làm cái điều này.

Thành ra tôi vẫn gọi, thứ nhất chúng ta phải xác định, define, định nghĩa nó ra. Thứ hai phải làm tài liệu, document, phải viết nó ra. Định nghĩa nó xong không đủ, phải viết nó ra. Tới vấn đề thứ ba, phải huấn luyện người ta, train, phải dạy người ta làm. Họ được huấn luyện để làm cái việc đó. Vấn đề thứ tư phải áp dụng, phải dùng, use, hay là thực hành, practice. Nói mà không làm thì không chạy. Phải làm chứ, tức là phải dựng lên. Đúng không ạ. Làm vẫn chưa đủ, phải đo lường, xem làm có tốt hay không, measure. Đo đây cũng không đủ. Phải có

verify, có người trắc nghiệm lại. Sau cùng, dù mình làm cả sáu yếu tố này không có nghĩa là nó hoàn hảo. Do đó luôn luôn phải cải thiện, continuous improvement. Mà trong cái continuous improvement nó đòi hỏi yếu tố thời gian. Yếu tố thời gian tối thiểu là từ 9 đến 12 tháng. Lúc đó nó mới đạt tới độ trưởng thành là mature. Tất cả những gì mình làm trong cái chương trình CMM level 5 hội đủ 7 yếu tố này. Anh ghi nó là bí kíp đi. Cửu âm thần công gì đó, đại khái thế.

Bây giờ tôi nói cái này để anh chị thấy rõ. Khi anh chị gặp những người tư vấn, các anh chị hỏi danh từ institutionalization, thể lệ hoá. 99 người tư vấn sẽ trả lời là "It depends on what you want - Còn tuỳ ông muốn gì." Không có người nào dám nói cái này hết. Cái này nó giấu biến đi mất rồi. Năm 1989 nó đã biến mất ra khỏi danh từ CMM rồi. Giấu chứ. Giấu kín để lần không ra. Cái này là cái đầu tiên mà nhóm tư vấn, origin, 14-15 người nó giấu. Cái đầu tiên nó giấu, cho nên bây giờ không ai tìm ra được tung tích của nó ở đâu nữa. Thành ra khi tôi mang cái này ra, tôi nêu cái này ra, 15 anh kia tới than phiền với tôi, bảo, "Nồi cơm của chúng tôi nằm ở đây mà ông không giấu đi à." Khi nó dạy lại cho Ấn Độ, nó dạy hai phần thôi. Giấu hết. Vì sao các tư vấn phần lớn không dạy? Các anh chị thấy không ạ. Bây giờ bảo thể lệ hoá, institutionalization, nói nó ra là biết vướng ở trong rồi. Bọn Mĩ nó cũng không đến nỗi dại đâu. Nó than phiền với tôi nhưng nó không làm được gì. Bởi vì mình không đụng chạm tới họ. Tôi dạy trong trường đại học tôi dạy ở Carnegie mà, anh nào học thì học, không học thì thôi. Nhưng khi nó dạy cho Ấn Độ, và bây giờ anh có vô Software Enginerring Institute nó cũng dạy là, "everthing but document." Do đó viết xuống

rất là dễ, thành ra nó không quan tâm là ai dạy dỗ, ai có áp dụng hay không. Thành ra nó đi nó làm cái vấn đề đánh giá. Người đánh giá đến công ti hỏi công ti có chính sách không? Có thủ tục không, có cái này không? Nó chỉ xem cái tài liệu đó thôi. Có xong rồi là kì thi xong. Nó chấm qua cái tài liệu, nó không chấm qua cái khả năng, nó chấm qua tài liệu documentation chứ không chấm qua năng lực thay đổi, change capability.

Người tư vấn Ấn Độ nó nghĩ ra, chấm qua, bắt người ta viết cái này, cũng mất công quá, ít dùng, bán. Bán tài liệu: hai chục ngàn level 2, bốn chục ngàn level 3, năm chục ngàn level 4, bẩy chục ngàn level 5. Thành một cái giá. Mà bây giờ theo thời buổi lạm phát giá nó đi xuống. Năm ngàn, mười ngàn level 2, hai chục ngàn level 3, level 4 chi đó. Thành ra nó đến các công ti nó bán. Khi tôi làm khoán ngoài tôi đến các công ti, công ti nào cũng có cái document này, giống như hệt nhau từ mức số 1 đến mức số 5, mọi thứ, hỏi là nó có. Chính xác tới mức độ con số đo metric mà nó còn giống nhau. Một công ti ở Thượng Hải, một công ti ở Thành Đô, hai công ti phần mềm khác nhau, giống nhau đến từng con số, defects 0.6 trên một ngàn dòng mã. Level 2 từ 0.4 đến 4. Nó giống đến cái mức như vậy, project nào cũng con số như vậy. Xuống Hongkong cũng như vậy, sang Đài Loan cũng như vậy, xuống Việt Nam cũng như vậy. Cả mức số 5, con số giống hệt nhau trên toàn thế giới. Cùng một khuôn mẫu, cùng một người tư vấn. Anh nào cũng con số 5, cũng dán một con dấu là CMM số 5, kí tên người tư vấn đó. Một công ti tư vấn chứ không phải là một người. Bán hết tất cả những cái đó đi. Bởi vì họ có cái đó rồi thì tự nhiên trên thế giới nó nảy sinh ra gần 1000 chuẩn số 5 trong khi toàn nước Mĩ chỉ có 17. Nước Mĩ được 17

nhóm đạt số 5, nhưng mà thế giới lại 1000 cái số 5. Tức là tóm lại tất cả các nước khác đều giỏi hơn Mĩ, tức là chất lượng của phi thuyền con thoi đó không bằng chất lượng của một công ti phần mềm nhỏ ở Việt Nam độ 100 người. Thành ra đó là lí do tại sao mình không muốn nói gì hết, mình im lặng. Nhiều người bảo "Tôi số 5 đây mà sao mãi chưa thấy anh đem việc về." Mình cũng im thôi.

Nếu anh làm cái mô hình trưởng thành, maturity phải có nhiều cải tiến chỗ này. Đây, chỗ này đây. Coi như là tất cả những cái gì quan trọng nhất tôi đã nói ra rồi. Các anh chị từ sáng đến giờ tất cả những cái gì hay nhất của CMM model các anh chị nắm hết rồi. Phần sau chỉ còn nói chơi, phần sau bắt đầu áp dụng cho software, phần trước là hoàn toàn mở rộng ra, áp dụng cho bất kì cái gì cũng được. Có ai còn thắc mắc gì nữa không ạ.

6. Cải tiến qui trình phần mềm

Bây giờ tôi bắt đầu đi vào software, tôi đi rất là nhanh. Từ số 1 đến số 5. Trường hợp số 1 là như thế nào? Số 1 là không có cái gì, lộn xộn, mạnh người nào người đó làm, cãi nhau lung tung. Bên Mĩ nó không dùng từ cãi nhau, nó dùng từ là creatibility, phát huy sáng kiến. Sở dĩ tôi không theo anh vì tôi phát huy hay hơn anh. Do đó tội không thể theo ý anh. Không ai tuân theo một tiêu chuẩn gì cả, mạnh người nào người đó làm. Muốn giải quyết những việc đó, thì việc đầu tiên là phải cải thiện cách làm việc, cách tổ chức làm việc, project management, quản lí dự án. Muốn quản lí dự án thì phải tuân theo 7 phần này.

Đầu tiên là requirement management - quản lí yêu cầu, project value - giá trị dự án, project monitoring - giám sát dự án, configuration management - quản lí cấu hình, processes and products - qui trình và sản phẩm, quality assurance - đảm bảo chất lượng, software environment - môi trường phần mềm, management and measurement - quản lí và đo. Cái này nó rất là giản dị thôi. Nếu các anh chị để ý, chính anh chị sẽ thấy tất cả những cái này nó chỉ là vấn đề của người quản lí dự án, không có gì khác. Bất cứ dự án phần mềm nào, sự đòi hỏi cũng có thay đổi. Nếu chúng ta không kiểm soát sự thay đổi đó, thay đổi lung tung, thì chúng ta không thể kiểm soát được. Do đó vấn đề chính gọi là kiểm soát sự thay đổi, change management requirement, chúng ta đã học một tuần qua rồi.

Thứ hai, bất kì dự án nào chúng ta cũng phải hoạch định nó ra. Khi hoạch định xong chúng ta phải theo dõi, kiểm soát nó. Sau khi kiểm soát theo dõi rồi, chúng ta phải luôn luôn kiểm soát sự thay đổi. Phiên bản 1, phiên bản 2, quản lí thay đổi - change management, ban thay đổi - change board. Sau đó rồi thì có những người kiểm lại chất lượng của cả qui trình lẫn sản phẩm. Nếu chúng ta làm việc với thuê khoán nước ngoài chúng ta thấy rằng cả những thoả thuận với người thầu lại, subcontract, những người bán - vendors hay những người cung cấp - suppliers. Và chúng ta phải đo, theo dõi con số. Hoàn toàn tất cả những cái này là nó khả năng của người quản lí dự án. Không có cái gì ra ngoài quản lí dự án. Một người làm quản lí dự án phải nắm rất vững bẩy yếu tố này. Không có gì mới lạ, thành ra tôi đi rất nhanh.

Câu hỏi: nhà cung cấp bên ngoài và bên trong thì sao?

Một công ti lớn có rất nhiều nhà cung cấp, cả bên ngoài và bên trong. Thí dụ như tôi có bộ phận kĩ nghệ. Nhà cung cấp của mình có thể là bộ phận CNTT, có thể là trung tâm dữ liệu. Nó cũng là nhà cung cấp phần mềm, nội bộ.

Phần sau các anh chị cũng có thể coi được, tôi cũng sẽ chỉ đi lướt qua rất nhanh. Bởi vì những phần này không phải là phần chính, như các anh chị thấy. Nhìn lại, tôi vẽ cái đồ hình này, cái chính là lập kế hoạch dự án. Yêu cầu chúng ta vừa học qua rồi, đây là các chi tiết thôi, về các anh chị tham khảo các anh chị đọc, không phải nói qua nói lại. Chia ra hai phần chúng ta vừa mới học cuối ngày hôm qua.

Phát triển. Số 2 chú trọng vào quản lí, số 3 chú trọng vào phát triển. Tại vì nó phải giải quyết những vấn đề nào, ưu tiên giải quyết thì giải quyết sự thay đổi trước khi giải quyết sự thành lập. Tại vì nếu chú trọng tới cái này mà không chú trọng tới cái này thì ở các dự án thường gọi là ad hoc, không thể thức, mạnh người nào người đó làm, thay đổi lung tung. Kiểm soát sự thay đổi trước, đó là lí do tại sao quản lí làm trước, phát triển làm sau. Ngoài ra không có gì thay đổi cả, các anh chị về coi qua cái này, tôi nghĩ rằng cũng không có gì thay đổi.

Đo là đây. Mọi người hỏi về đo lường yêu cầu thế nào. Có ba cách đo lường, anh có bao nhiêu yêu cầu thay đổi, cái gì anh đã giải quyết xong rồi, cái gì anh chưa giải quyết xong, đại khái như vậy. Ít nhất mình có thể coi, không có gì cả. Ngoài ra tôi thêm một số checklists

có thể áp dụng. Như tôi đã nói, tôi hoàn toàn dùng checklist rất là nhiều. Đây chỉ là một số ví dụ. Đo, đây là một cách trích ra. Khi chúng ta thành lập dự án, chúng ta đo, chúng ta lập độ đo, metric. Metric là gì ạ, đo đạc phải không ạ. Khi chúng ta đo, chúng ta phải biết chúng ta đo vào cái mục đích gì. Nhiều người bảo chúng ta cứ việc đo mà không biết để làm gì. Tôi muốn nhấn mạnh là phải có một mục đích, sau đó mục đích đó phải được định nghĩa rất rõ rệt. Sau đó chúng ta phải dùng nó một cách cẩn thận, kiểm soát cẩn thận và phân tích cẩn thận.

Chúng ta trở lại vấn đề đo lường, đo đạc. Đo thì việc đo đây, chúng ta phải biết chúng ta thu thập các dữ kiện gì. Đây là cái qui trình. Trước hết chúng ta tìm xem cái giới hạn đo thế nào, qui trình phục vụ mục đích gì. Sau đó chúng ta có thể định nghĩa cách thức đo là thế nào. Sau đó thu thập các dữ kiện đó vào, phân tích các dữ kiện đó. Dựa trên các dữ kiện chúng ta phân tích, chúng ta phải làm việc gì để thay đổi. Dùng các dữ kiện đó để chúng làm các việc sau, cải thiện vấn đề làm phần mềm. Thì đấy là vấn đề rất giản dị không có gì cả. Các anh chị có thể đọc qua vấn đề này, nếu có gì thắc mắc thì chúng ta có thể trao đổi sau. Tôi lưu ý đây là vấn đề project, mục đích của project này là đo xem có hiệu quả không. Thường cho một dự án thế này, cũng tốt thôi, xác định mục tiêu. Đây, cái data dữ kiện thế nào đây. Từng những dữ kiện này mình mới đưa tới ứng dụng. Thay đổi dự án đây, dòng mã là bao nhiêu đó, tốn mất thời gian bao nhiêu đó, tốn bao nhiêu tiền, đây là những điều rất là giản dị. Việc đo không có gì hết.

Thông thường thì các sách giáo khoa nói về đo, thường thường có rất nhiều cách đo. Theo riêng tôi khi

chúng ta bắt đầu thu thập dữ kiện, tối thiểu chúng ta cần 3-4 tháng để cho các dữ kiện được ổn định. Khi chúng ta lấy dữ kiện lần đầu tiên nó còn chưa stable, nó chưa vững, tối thiểu phải thu thập dữ kiện đó 3 đến 4 lần. Thông thường tôi thu thập dữ liệu, trong vòng 3-4 tháng đầu tôi bỏ. Lúc đầu mới đo đôi khi nhân viên không cung cấp đúng, chính xác dữ liệu, thành ra tối thiểu cũng phải 3 tháng đầu dữ kiện mới thu thập được chưa vững chắc lắm đâu. Thí dụ mình xuống mình bảo, "Anh cho tôi biết là anh có bao nhiêu lỗi?" nhân viên nó báo cáo lên, lúc đầu chưa có vững chắc, lúc đầu nó không rõ. Thường thường tôi thu thập khoảng 6 tháng thì 3 tháng đầu tôi bỏ.

Vấn đề thứ hai là mỗi dự án không nên có quá nhiều cách đo. Tôi sử dụng thông thường là 3. Đừng bao giờ nhiều hơn 4. Vì nhiều quá nó rối rắm không làm được cái gì cả. Đó là kinh nghiệm trong thời gian đầu. Các anh chị thấy rồi, trong khi chúng ta đo luôn luôn có hai phần, phần thứ nhất đo lường cái qui trình kĩ thuật, phần thứ hai đo lường cái sản phẩm. Hai cái đó cộng lại cho ta cái dữ kiện này, từ dữ kiện này chúng ta phân tích nó ra, rồi sau đó chúng ta làm quyết định. Đây là ví dụ. Các anh chị có thể xem qua ví dụ này. Từ đây là dữ liệu thô, những dữ kiện này. Những người này đang làm ở một project, họ đang làm ở phần nào, họ tốn công bao lâu. Từ đó, từ chỗ này chúng ta suy ra khả năng họ làm được bao nhiêu phần trăm. Đại khái như vậy. Những cái này rất là bình thường và rất là căn bản, không có gì mới lạ cả.

Đo thì dễ nhưng mà thực hành thì khó. Bây giờ mình thấy trục trặc chỗ này mình sửa thì khó chứ đo thì dễ không có gì cả. Tôi đi qua rất là nhanh phần này.

Câu hỏi: Đo lường và phân tích mình có cần các công cụ hỗ trợ không?

Cách thức đo ngày xưa thì dùng giấy tờ. Thông thường một dự án, người quản lí dự án họ có một người so sánh xem xét công việc, rồi báo cáo với họ. Còn trường hợp của tôi, trong công ti của tôi, tôi có một website. Trong website đó hàng tháng, hàng tuần mỗi một người quản lí dự án vào website đó, cho các con số vào đó. Rồi có một nhóm nhân viên của tôi họ chuyên môn họ theo dõi các con số đó, họ vẽ ra các đồ hình rồi họ báo cáo lên. Với một công ti lớn rất dễ làm chuyện này. Ngày xưa báo cáo bằng giấy tờ lên thất lạc thông tin nó phiền phức lắm. Bây giờ tôi thấy rất là giản dị, là trong công ti tôi tất cả mọi thứ đều nằm ở cái website. Cứ đến ngày 1 đến ngày 15 hàng tháng, ngày 1 đến ngày 5 những dự án đó, những người quản lí dự án lại cho các con số vô. Từ ngày 5 đến ngày 10 là mình chỉ đợi các dự án đó. Đến ngày 15 là tối thiểu. Ngày 15 là khoá chót. Ngày 14 là khoá sổ. Ngày 12 anh nào còn chưa nộp đủ con số, nhóm làm về việc đo họ gửi email đến nói, ông quên là chưa nhập số sao. Và đến ngày 14 là khoá sổ, ngày 15 là bắt đầu phân tích. Ngày 20 là họ phải vẽ thành đồ hình, nổi lên hết. Có một cái ban nó phân tích và xem xét. Ngày 25 tôi có một cái dashboard các con số chính. Đến cuối tháng là nhân viên quản lí cấp cao hơn nhận được đầy đủ các chi tiết về đo lường, tất cả các dự án về kĩ nghệ. Thành ra mình khép cái như vậy. Đến cuối tháng là họ xem được hết các thứ trong tháng. Mình

đo chứ, mình có 15 ngày thu thập dữ liệu đo, mình có 10 ngày để mình làm phân tích mọi sự và cho nó thành cái đồ hình. Cứ đến trước ngày cuối tháng là họ có được báo cáo tổng hợp. Đó là phương pháp tôi làm việc trong công ti.

Câu hỏi: Em đọc sách thấy người ta có một số cái người ta hỗ trợ đo lường, cái hệ thống OLAP. Cái đó cũng là công cụ?

Vâng, OLAP phải có cùng cái dashboard. Chúng ta đã qua cái OLAP, năm ngoái đã học về OLAP ở Information system management. Vấn đề chính tôi thấy là các sách giáo khoa họ nói quá nhiều về vấn đề đo. Họ nói rối rắm chi tiết trong khi mình đo rất giản dị, không có gì cả. Thứ hai nữa, các nhà hàn lâm, khi đo được là đo tối đa thôi. Đo được cái gì là đo thôi. Còn một người quản lí dự án tôi thấy là họ không bao giờ dùng quá 3 đến 4 thứ. Chẳng hạn như anh quản lí dự án, anh muốn đo cái gì? Thông thường cái thứ nhất mình biết là mình có bản kế hoạch, mình theo dõi kế hoạch đó xem có đúng kế hoạch hay không. Tức là đo plan vers actual, đúng không ạ. Mình dự định nó là như vậy mà hiện thời nó thay đổi như vậy thì tôi phải xem cái kế hoạch mà tôi đã dự định đó nó có chính xác hay không. Tôi ước chừng là 15 ngày thì nó phải xong cái này. Bây giờ đến ngày thứ 13-14 rồi nó có đạt được tới mức đó hay không. Chậm quá hay là nhanh quá, plan vers actual, cái đó gần như ai cũng làm.

Cái thứ hai là mình đo vấn đề về lỗi, có bao nhiêu lỗi. Thông thường cái đó là cái rất căn bản. Mình không cần quá nhiều cái đo. Thành ra quan niệm của tôi là mỗi một người quản lí dự án cho tôi 3 cái metrics, thế thôi, 3

hay 4 cái. Mục đích các anh áp dụng cái đó để quản lí dự án của các anh. Còn tôi thì tôi cần chẳng hạn như là có bao nhiêu dự án chậm trễ như thế nào. Khi tôi báo cáo lên trên có 400 dự án thì 10 cái chậm 1 tháng, 20 cái chậm 3 tháng , thí dụ như vậy. Người quản lí ở trên họ đâu cần cái chi tiết quá làm gì. Và tổng số nỗ lực - lỗi là bao nhiêu. Chứ còn người vice president người ta hơi đâu mà nhìn xuống cái dự án A, dự án B chạy ra làm sao. Mình phải tổng kết nó lại. Tôi đưa ra một cái vấn đề là, tôi dùng danh từ là các chi tiết. Còn thì thông tin tóm tắt để nêu vấn đề cho người trên quyết định.

Thỉnh thoảng tôi cần ông phó chủ tịch xuống nói vài ba câu. "Sao các anh làm lỗi nhiều thế này?" chẳng hạn. Nhiều khi mượn cái oai của ông ấy dễ nói cho với người cấp dưới của mình. Mình là kĩ sư trưởng thôi, mình chưa tới mức mình nói họ nghe được thì đôi khi mình cũng phải nhờ người trên xuống nói. "Năm nay làm sao mà defects nhiều quá, lỗi nhiều quá." Dĩ nhiên là khi ông ấy nói với người dưới, tôi giả bộ như vậy. Tôi mời ông vice president ông xuống nói vài ba câu. Thì ông ấy họp với các nhân viên. Thường với các hãng rất là lớn, đôi khi họ không gặp trực tiếp được tất cả mọi người. Ông ấy vào một cái phòng, cái phòng đó có các máy tự động quay phim. Ông ấy vào đó bấm máy và ông ấy đứng nói độ 15 phút, 10 phút đồng hồ. Nó thu cái đó vào rồi thảy vô cái web của hãng. Nhân viên sáng tới bật lên là có cái thông điệp đây "letter from the vice president." Chương trình chiếu lên ông ấy ngồi ông ấy nói. Đó là phương pháp truyền thông rất hữu hiệu. Chúng tôi làm cái đó rất là nhiều. Khi tôi cần gửi tới tất cả kĩ sư, ngày xưa phải họp tất cả mọi người, kĩ sư ở rải rác khắp nơi. Gửi một email mình nói với nó mà nhiều

khi nó không nghe cho nên mình cần phương pháp trực tiếp. Mình bước vào phòng đó. Phòng đó là phòng tự động. Máy đã có sẵn ở đó rồi, không có ai chỉ huy cái phòng đó. Màn ảnh, hình ảnh, camera có sẵn hết, chỉ bấm máy trong phòng rồi mình nói 5 phút đồng hồ. "Tháng này các anh làm lỗi nhiều quá, phải coi chừng." Đôi khi mình đoán là hình như trong tháng hãng có thể sa thải bớt người làm lỗi nhiều, chúng tôi cần biết làm lỗi nhiều là những ai. Quản lí dự án cần chỉ mặt những người làm lỗi nhiều. Đại khái như vậy, quản lí. Tôi nghĩ như vậy nhưng thường thường để ông kia nói chứ mình nói nó không nghe. Ông phó chủ tịch xuống nói thì nó sợ. Đại khái như vậy. Cho nên đưa bài lên web, bật nó lên là đủ.

Thành ra tôi thấy là vấn đề level 2 rất là giản dị, không có gì. Lập kế hoạch dự án cũng là vấn đề rất giản dị, không có gì quan trọng. Tôi lướt qua vấn đề này thật nhanh, chỉ đi vào những vấn đề then chốt chính thôi. Một cái dự án là cái gì, project là cái gì đây, định nghĩa dự án. What is project? What is the project management? Quản lí dự án là gì? Làm kích cỡ dự án như thế nào, công cụ dự án, mình làm thế nào. Yêu cầu, mình chia nhỏ nó xuống thế nào, mình ước lượng nó, lớn nhỏ thế nào, nó gồm có những chức năng, nào, nó cần bao nhiêu người, thời gian bao nhiêu, đại khái như vậy, rủi ro thế nào. Tôi nghĩ rằng một trong những yếu tố rất quan trọng mà tôi thấy là hơi thiếu sót mà tôi quan niệm phần lớn công ti phần mềm thất bại ở đó. Lí do chính cho thất bại có hai cái. Thứ nhất là chức năng, lấy yêu cầu là số một, requirement là number one. Số hai là quản lí dự án. Phần lớn những người quản lí dự án không được huấn luyện một cách tốt đẹp. Đa số những người quản lí dự án, theo

công trình nghiên cứu của tôi, họ là những người lập trình rất giỏi. Họ biết lập trình rất hay, họ được thăng chức lên quản lí dự án. Thành ra phần lớn đều thất bại.

Tôi luôn luôn nói với các chủ nhân trong khoá dạy về CIO, "Ông cho một người nào biết lập trình hay nhất, cho vào làm quản lí dự án mà không có huấn luyện cẩn thận thì ông sẽ mất một người lập trình rất là hay, và ông sẽ được người quản lí dự án rất là tồi." Tôi đặt vấn đề là tất cả những nhân viên, trước khi làm quản lí dự án, phải được huấn luyện rất là kĩ lưỡng. Bởi vì nếu quản lí dự án thất bại thì công ti thất bại, bởi vì nó là cấu trúc căn bản nhất của bất cứ một doanh nghiệp nào chốt lại là quản lí dự án. Thành ra tôi đặt rất nặng việc project management, quản lí dự án.

Tất cả nhân viên dưới quyền tôi được chọn làm quản lí dự án tối thiểu đều phải học qua khoá học về quản lí dự án. Phải làm phụ tá quản lí dự án ít nhất trong vòng ba tháng. Sau khi mình huấn luyện xong, mình trao cho người đó chức quản lí dự án, mình phải đưa người đó phụ cho một người quản lí dự án khác. Anh ấy vào anh ấy ngồi trong vòng ba tháng học nghề đó. Xem người kia người ta quản lí thế nào để mình học. Thành ra tôi dùng cái phương pháp gọi là mentor, kèm cặp. Đôi khi tôi nói với người quản lí dự án rất giỏi, tôi bảo "Trước khi tôi phong cho anh lên một chức khác, anh phải huấn luyện cho anh này trong vòng 3 tháng. Khi anh này giỏi làm được việc rồi thì tôi mới đưa anh đi." Ông kia hết lòng truyền nghề cho ông này trước khi ông ấy được bổ nhiệm vào một chức vụ khác. Tức là nếu mình đưa người ta vô mà người ta ngồi đấy lơ mơ tức người ta không làm được, tôi bảo "Anh chịu trách nhiệm

cho việc này. Anh này anh ấy sẽ vô phụ tá cho anh cho công việc này, anh sẽ tập trung vào công việc khác. Trước khi anh qua công việc khác tôi cho anh 3 tháng để anh truyền nghề. Có cái gì thì chỉ dạy hết cho anh kia đi. Khi anh kia bảo tôi đủ sức tôi quản lí cái dự án này rồi, thì lúc đó anh mới được chuyển sang việc khác." Đó là cách của tôi thu xếp với nhân viên.

Thông thường thì người làm lập trình khi họ mới nghe được họ sẽ làm quản lí dự án, đấy là vấn đề được thăng chức. Thăng chức thăng lương ai cũng thích. Nhưng phải nói là 80 phần trăm những người làm lập trình, sau khi được làm quản lí dự án, lại muốn trở lại làm lập trình. Không ai muốn làm quản lí dự án. Hai lãnh vực này nó quá khác nhau. Thứ hai là lúc mình ngồi làm lập trình, mình cãi nhau bằng những vấn đề là "anh sai anh đúng" cũng như "một cộng một là hai" rất dễ dàng, mọi thứ là logic thôi. Nhưng làm quản lí dự án thì logic có 10 phần trăm thôi, 90 phần trăm không có logic, đúng không ạ. Thành ra cái vấn đề đụng chạm nó không phải là logic. Thành ra nhiều người họ rất là giỏi, họ không thích làm, nhất là những người làm kĩ thuật giỏi, họ không thích làm quản lí.

Nhưng mà người làm quản lí dự án giỏi, họ rất là thành công. Thành ra bây giờ đưa một người quản lí dự án giỏi, đưa lên vai trò gọi là quản lí chứ không phải là quản lí dự án nữa, thì phải qua một giai đoạn huấn luyện nữa. Đối với tôi, từ quản lí dự án, bước tiếp là chuyển lên quản lí loạt dự án - portfolio management. Portfolio management là người quản lí nhiều dự án. Trong một miền, người quản lí dự án, thí dụ 5 dự án về kĩ nghệ về software, lúc đầu họ quản lí một dự án, về sau cho họ lên

portfolio quản lí 5 dự án, 10 dự án gì đó. Đó là portfolio management. Khi mà họ quản lí thành công cái portfolio rồi thì bước tiếp họ lên cái gọi là quản lí chương trình. Quản lí chương trình là trông sản phẩm, tức là product. Họ trông một chương trình, thí dụ như chương trình là quản lí dự án phần mềm cho động cơ tầu bay. Chuyên trị về động cơ, về engine. Đó là một cái chương trình. Trong cái động cơ đó có hàng chục cái dự án nhỏ nhỏ khác nhau. Như vậy trông một cái lớn hơn. Khi họ lên được cái gọi là quản lí chương trình, bước tiếp họ lên quản lí, tức là trông một khu, miền. Thí dụ như động cơ cũng là khu, trong đó có mấy chục loại động cơ khác nhau. Lúc đó thì vấn đề ra khỏi kĩ thuật mà lên quản lí. Lúc đó, mỗi lần lên như vậy, họ phải trải qua một giai đoạn tôi gọi là thử việc. Tức là mình đưa người đó làm việc dưới hướng dẫn của một người khác, thông thường là khoảng độ 3 tháng tới 6 tháng. Lên quản lí chương trình thì tối thiểu cũng phải 1 năm. Thành ra luôn luôn có sự huấn luyện, training, và có giai đoạn để tiếp thu, giai đoạn để học nghề.

Tôi thấy nhiều công ti cứ thuyên chuyển người này từ chức vụ này sang chức vụ khác mà không cho người ta giai đoạn để người ta học. Thành ra tôi thấy sự đó rất thường đem lại rất nhiều thất bại. Đây là phương pháp làm việc tôi học được qua nhiều năm kinh nghiệm làm việc tại nhiều công ti khác nhau. Cuối cùng rốt cục điều mình thấy được đem ra chia sẻ với các anh chị. Mong rằng sau này các anh chị cũng áp dụng được những điều này trước khi mình nâng một người lên, phải nói cho họ biết. "Tôi đưa anh vào đây nhưng tôi cũng chuẩn bị thu xếp huấn luyện cho anh làm quen, anh làm phụ tá cho một người khác trong một thời gian." Họ biết là họ chỉ

làm phụ tá cho một người khác trong một thời gian thôi rồi họ mới được giao cho việc đó. Họ học việc trong thời gian đó, họ cố làm, họ hết sức họ học. Người kia trước khi lên chức vụ khác phải huấn luyện người thay thế cho mình cũng trong thời gian đó. Họ chưa hết nhiệm vụ, anh này không xong thì họ cũng không đi được sang nhiệm sở khác. Thành ra đó là cái vấn đề mình gọi là training. Tôi có cái kĩ thuật như vậy.

Monitor là kiểm soát. Mình có một chương trình mình đưa ra. Cái dự án nào cũng phải có kế hoạch hết phải không ạ. Mình xem xem là mình hoạch định như vậy có đúng không và nó có theo đúng chương trình mình nghĩ như vậy không. Mình phải theo dõi, theo dõi xem nó nói vậy nó có làm được như vậy không. Trong cái monitoring có nhiều phần, tôi chỉ lướt qua cái này các anh chị có thể vào tìm sách đọc. Nhiều người có kinh nghiệm rồi nên tôi chỉ nói qua thôi. Quản lí phiên bản dự án chúng ta cũng đã học rồi, kiểm soát sự thay đổi thôi chứ không có gì hết. Cái này năm ngoái chúng ta đã học qua rồi trong chương trình Software Engineering.

Đảm bảo chất lượng cũng vậy. Có một sai lầm mà nhiều người nghĩ rằng người kiểm soát chất lượng là người chịu trách nhiệm chất lượng. Cái đó là cái không đúng. Người kiểm soát chất lượng xem người ta có tuân thủ theo qui trình đó và cái chất lượng có đạt được mức mong muốn hay không. Người trách nhiệm cho chất lượng là người làm chứ không phải người kiểm soát. Tôi thấy sách giáo khoa hay lầm lẫn, người làm chất lượng phần mềm hay đảm bảo chất lượng qui trình hay sản phẩm không chịu trách nhiệm về chất lượng. Người làm, tức là người lập trình phần mềm, người software làm,

chịu trách nhiệm về chất lượng sản phẩm, người kia là người kiểm soát. Người đảm bảo chất lượng phần mềm không thể nào làm chất lượng lên được, nó có làm đâu.

Nhưng mà khó khăn thứ hai mà tôi thấy nhiều công ti vấp phải: họ không có coi trọng chức vụ của người đi kiểm soát chất lượng. Thành ra đa số công ti, khi họ làm, khi họ mời tôi đến xem về cách thức làm việc của công ti, một trong những câu hỏi của tôi là "Cho tôi nói chuyện với người đảm bảo chất lượng." Tôi hỏi người đó, kinh nghiệm thế nào, làm ra sao. Tôi đi nhiều công ti, họ đưa người làm kiểm soát chất lượng là người không có kinh nghiệm. Là vì họ quan niệm thế này: trong phần mềm người lập trình là quan trọng, do đó anh nào càng làm lập trình giỏi thì cứ ngồi cứng mà làm lập trình thôi. Còn anh nào lập trình dở quá cứ viết sai hoài thì cho đi làm quality assurance. "Thôi, anh làm không được thì anh đi kiểm soát người ta đi." Còn người làm được thì cứ làm hoài. Đó là một sai lầm rất lớn.

Thành ra khi tôi đến các công ti rất là giỏi, họ đạt mức số 4, họ đạt mức số 5, họ muốn có thêm một người khác đến xem họ làm có được không, những công ti đó phần lớn họ liên lạc với tôi rất nhiều. Khi tôi đến, một trong những câu hỏi của tôi là: cho tôi gặp người kiểm soát chất lượng trước. Nếu người đó là người rất có nhiều kinh nghiệm, làm việc rất có uy tín, thì tôi tin là chất lượng nó tốt. Nếu người đó là một người không có kinh nghiệm gì cả, hoặc là một người mới ra trường 2-3 năm kinh nghiệm gì mà đưa ra làm việc đó thì tôi bắt đầu có sự nghi ngờ. Cái này tôi cũng học được khi tôi làm việc với Toyota, tôi thấy là người lương cao nhất của Toyota là người kiểm soát chất lượng. Không một người

kiểm soát chất lượng nào mà dưới 20 năm kinh nghiệm. Thành ra tất cả những người kĩ sư làm việc cho Toyota lên chức quản lí dự án, lên chức quản lí miền, rồi mới lên chức quản lí chất lượng. Đảm bảo chất lượng phần mềm của Toyota lương cao hơn người quản lí, phần lớn họ là những người quản lí trông về một miền rồi họ mới trở thành đảm bảo chất lượng. Đến lúc đó tôi mới bảo, "À, thì ra thảm nào cái chất lượng, nhất là cái xe Lexus, thảm nào cái công ti này họ thành công lớn vì họ nắm được cái yếu tố đó."

Thành ra anh lên làm manager đã, anh quản lí đã, rồi anh lên quản lí một miền, quản lí một cái máy động cơ, rồi lúc đó mới đi làm quản lí chất lượng. Thứ nhất họ làm bao nhiêu năm làm quản lí, họ rất có uy tín rồi, mọi người rất nể họ rồi. Bây giờ họ nhường cái chức đó cho một anh thay thế họ, họ trở về việc quản lí chất lượng. Khi đó ông ấy đi đến đâu mọi người cũng nể ông ấy hết. Vì ngày xưa đó là việc của ông ấy. Khi họ làm manager, tôi lấy thí dụ họ làm manager của phần mềm trên cái xe hơi. Họ làm cái đó trong vòng bao nhiêu năm rồi. Bây giờ họ chưa được đưa vào dự án thay thế, họ trở thành người quản lí chất lượng. Ông ấy biết rất rõ mọi công việc, ông ấy chỉ đâu là đúng đó, không ai cãi được hết. Đó là yếu tố thành công của Toyota. Thành ra bí mật của Toyota nó nằm ở chỗ đó. Công ti đó có một phương pháp tổ chức tôi nghĩ là không mấy công ti theo được, rằng họ đặt điều là chất lượng là cao nhất trong công ti.

Khi tôi qua viếng thăm Toyota, tôi thấy công ti này có một vision. Vào văn phòng của ông chủ tịch, có treo cái khẩu hiệu, khi vào cửa là thấy treo ngay khẩu hiệu: "We want to be the only transportation company in the

world." Chúng tôi muốn trở thành công ti vận tải duy nhất trên thế giới. Đó là kỉ niệm 100 năm của ông Toyota. Ông Toyota là người sáng lập ra công ti Toyota đã viết câu ấy. Tôi nhìn thấy 2 chữ tôi hơi nhột: thứ nhất là "only company," tức là chúng tôi muốn trở thành công ti duy nhất. Tức là trong cái tiến trình của họ, họ muốn đánh dẹp tất cả các công ti khác. Only là độc nhất mà. Chúng tôi muốn trở thành công ti độc nhất. Thứ hai là công ti transportation, không phải car company nhé, không phải xe hơi nhé, mà là transportation. Thành ra tôi nhìn và tôi cứ cười cười. Tôi hỏi ông chủ tịch, "Ông ơi ông, cái chuyện ông muốn là only company thì hơi kiêu căng một chút, nhưng ông nói chữ transportation này nó có tầu bay ở trong đó không đấy, hay nó là xe hơi?" Ông kia ông ấy bảo là "Anh nghĩ sao cũng được. Nhưng mà tất cả các nhân viên trong hãng của tôi sẽ phải nhắm vào cái mục đích này." Đó là cái mục tiêu của họ.

Thứ hai, the chất lượng tốt nhất có thể được trên thế giới - best quality possible in the world. Người sáng lập ra công ti đó để lại cho hậu thế, cho đến nay, câu đó vẫn đúng như thường. Bao giờ họ đạt được mức đó là cái chuyện của họ, mình không nói. Nhưng cái thứ hai mà tôi không thích, cũng hơi hợm một tí, only transportation company, công ti độc nhất đứng về vận tải giao thông. Người chủ họ nhắm cái đó, mà các nhân viên trong hãng cũng nhắm cái đó. Và thứ hai là the best quality possible. Thành ra các anh các chị nhìn thấy chất lượng của họ rất là cao. Cái thứ hai trong cách tổ chức của họ thì mình nêu cái cách gì đó rồi mình làm có chất lượng. Đó là lí do tại sao họ làm ra chất lượng tốt. Tốt hơn những người khác. Và bây giờ vượt qua cả General Motors, vượt qua cả Chevelet. Dư âm thì nó còn. Đặt hai cái xe Lexus và

... cạnh nhau, xe ... thua xa lắc xa lơ, thua rất là xa, tôi nói thẳng là như vậy.

Cái câu này của họ viết bằng tiếng Anh, đúc chữ treo ngay trước cửa. Tức là sự ghê gớm của họ, họ viết cái câu đó từ một trăm năm về trước. Người sáng lập ra công ti đó, viết câu đó ra. Thế còn về sau cái chuyện họ có làm được hay không là chuyện khác. Nhưng mà tóm lại, tôi thì tôi nhìn chữ only, tôi bảo thế này kiêu căng quá. Tôi hỏi chữ transportation, anh nói anh làm xe hơi, anh có làm tầu bay không đây. Ông ấy cười cười ông ấy bảo "Anh nghĩ sao cũng được." What do you think? Không nói ra.

Khi mà sang thăm viếng công ti Toyota, mình sang mình làm việc với họ, thấy họ cái bảng đó gắn ở tổng hành dinh của họ, ngay trước cửa văn phòng ông chủ tịch. Tôi thấy có hai chữ tôi không thích. Thứ nhất là chữ only, tức là có nghĩa thanh toán người khác. Tôi là người duy nhất nhìn vô cái đó vì đó là sự kiêu căng. Nhưng mà kiêu căng thì cũng được. Nhưng mà mình hơi lo ngại về chữ transportation, như vậy có nghĩa là ông cũng định làm tầu bay. Mà nếu ông định làm tầu bay thì có thể ông ấy vượt qua mặt mình, mình cũng mệt. Nhưng tôi nghĩ rằng nếu ông ấy có làm tầu bay thì cũng phải năm chục năm nữa. Lúc đó thì chắc mình phải đi làm cái khác. Mình nghĩ như vậy nên mình cười cười mà hỏi "Ông nói ông làm transportation ông có nghĩ rằng ông sẽ làm tầu bay hay chỉ xe hơi hay làm cái gì khác?" Hãng làm tầu bay mình biết rồi, nhưng ông còn cái gì khác không. Ông ấy cứ cười cười, bảo rằng "Anh muốn nghĩ sao cũng được." Thì nói vậy thôi.

Tôi chỉ nói qua về vấn đề người kiểm soát chất lượng. Cách làm việc trong hãng đó, người quản lí cao nhất là người quản lí chất lượng. Lương của nhân viên quản lí chất lượng rất là cao, uy tín của họ rất là cao, cho nên chất lượng cao. Thành ra tôi đến lúc đó tôi rút xuống, thành ra khi tôi đi thăm các công ti khác, tôi muốn biết lương của người quản lí chất lượng là bao nhiêu, chức vụ của họ trong công ti là như thế nào. Thí dụ, tôi lấy một thí dụ nữa. Bây giờ tôi tới thăm một công ti phần mềm mà họ bảo người quản lí chất lượng là một anh mới ra trường, chưa biết gì cả, không có kinh nghiệm gì cả, thì rõ ràng chất lượng đối với người chủ công ti đó là không quan trọng. Không có nghĩa là họ không có, nhưng mà không quan trọng. Một anh lương thấp nhất, địa vị thấp nhất không có ý nghĩa gì cả, thì nói ai nghe, đúng không ạ. Do đó với chúng ta vấn đề là địa vị. Ở bên Mĩ này địa vị là do số lương, nhiệm vụ càng cao thì lương càng nhiều. Do đó nó hỏi cái lương của nhau để xem địa vị của anh cao hay thấp. Chức vị thì nó mơ hồ, ông nào cũng xưng chức vị rất lớn, nhưng mà thực sự người ta có quyền hay không. Cái đó đi cùng với số lương của người ta. Nó có tiêu chuẩn về lương.

Nhưng đôi khi lương không theo tiêu chuẩn. Đôi khi con ông cháu cha thì lương lớn, họ hàng xa xôi thì lương nhỏ. Lương của các cấp dưới thì công bố chuẩn. Thí dụ một sinh viên mới ra trường thì thí dụ lương năm chục ngàn đến bảy chục ngàn. Lương cụ thể trong cái giới hạn là bao nhiêu thì mình không biết, nhưng mà có cái chuẩn thu nhập nó là như vậy. Họ cho con số như vậy, nhưng mà bắt đầu từ giám đốc trở lên thì con số phải nói rõ ra. Nhân viên ở dưới, kĩ sư ở dưới cả trăm cả ngàn người, tốt nhất nếu chức vụ của anh là kĩ sư, anh

chỉ ở trong cái mức đó thôi, sáu chục ngàn bẩy chục ngàn gì đó mình không biết. Nhưng mà giữa hai con số đó, lương anh chỉ ở đó, anh không thể vượt quá cái đó được. Nhưng mà khi anh lên người quản lí thì cũng vậy, khoảng cách như vậy. Người quản lí là tám chục ngàn tới một trăm ngàn, thí dụ như vậy. Nhưng mà khi anh lên đến mức gọi là cao nhất, điều hành, đồng lương công bố rất rõ rệt. Ngoài ra, ngoài đồng lương thì có những cái mà mình gọi là bổng lộc, phụ cấp, lương trách nhiệm, phải kê khai ra hết. Cổ phần được bao nhiêu cũng phải được công bố hết. Bên này nó rất rõ rệt từng đồng từng cắc một.

Thí dụ tôi nói cho nó vui. Tôi hay đùa nói với mọi người rằng "Anh có biết là lương các anh các chị hơn lương Bill Gates không ạ?" Anh có biết không? Lương các anh các chị hơn lương của Bill Gates, lương có 1 đồng một năm à. Lương Bill Gates một đồng một năm nhưng mà cổ phần của Bill Gates tại Microsoft là 80 phần trăm. Bill Gates nó than mỗi lần thị trường chứng khoán rơi xuống một điểm hai điểm là tôi mất mấy trăm triệu đô la. Lương có 1 đồng thôi. Lương của Bill Gates tại Microsoft là 1 đô la một năm. Nhưng mà ngoài ra tôi không biết có bao nhiêu cổ phần. Cho nên tôi nói đùa tất cả các anh các chị ngồi đây chắc chắn tất cả đều hơn lương Bill Gates. Nhưng mà cổ phần thì không bằng. Có đúng không ạ. Warren Buffett lương có 100 ngàn một năm. Cái người giầu nhất thế giới, lương ông ấy có một trăm ngàn, nhưng mà cổ phần của ông ấy thì khủng khiếp. Thành ra nhiều người bảo rằng lương một trăm ngàn nhưng mà cổ phần thì bạc tỉ. Nhưng mà họ phải công bố, những người đó họ phải công bố. Nhân viên dưới trong tất cả các công ti ở bên này, nhân viên từ cái

mức nó gọi là director trở xuống, mình gọi là lính, nó cho một cái mức, nhưng mà lên trên mức đó, mức quan, là phải công bố hết. Thành ra những người cao của công ti, đồng lương bao nhiêu, được bonus bao nhiêu, được thưởng bao nhiêu và tiền thuế bao nhiêu đều phải công bố hết, rất rõ rệt.

Câu hỏi: Người có cổ phần có phải đóng thuế không?

Cổ phần anh chưa chuyển thành tiền thì không phải đóng thuế. Cổ phần mà anh chuyển thành tiền thì phải đóng thuế. Cổ phần chỉ là tờ giấy thôi. Khi anh chưa chuyển thành tiền, anh còn nằm đó, thí dụ Bill Gates hiện có 1 triệu cổ phần của Microsfot, ông ấy không phải đóng thuế. Nhưng mà khi anh bán ra thị trường thì phải tính thuế. Nói cho vui, ở Mĩ có một tổng thống chưa bao giờ đóng thuế. Lương tổng thống là hai trăm ngàn, ông ấy cúng cho từ thiện 199 ngàn, ông chỉ giữ 1 ngàn thôi. Thì không phải đóng thuế. Đây là một ngàn một năm. Nhưng mà tổng thống thì còn có những cái khác, đây là câu chuyện nói cho nó vui. Có những người họ làm việc có quĩ đen quĩ đỏ gì đó mình không biết. Họ có những cách thức khác mình không biết. Nước nào cũng vậy.

Bên Nhật có vấn đề hơi khác là lương ông thì ít lương bà thì nhiều. Tức là vợ của những ông chủ xí nghiệp, được một chức khác, support psycho, hỗ trợ tâm lí, lương rất là lớn. Vì nếu ông chủ hãng mà lương thấp thì tất cả nhân viên trong hãng lương thấp hết. Có đúng không ạ? Nhưng bà chủ thì không dính dáng gì tới nhân viên cả. Ông chủ ông ấy cho bà vợ cái chức là bà ấy supports tôi về tâm lí. Trong một công ti khổng lồ không người nào làm việc đó được. Đó là cái đường lối và

chiến thuật có tác dụng như vậy. Đây là nói vui, anh chủ công ti, lương anh có 100 ngàn đô la thì sống làm sao. Không lương tôi 100 ngàn nhưng vợ tôi lương có một triệu thôi. Vậy bà vợ làm chức gì không? Không vợ tôi làm psychology supporting. Tôi làm việc rồi về nhà bà ấy lo cho tôi.

Bây giờ chúng ta tiếp tục lại. Lí do của các báo cáo không chính xác. Thứ nhất, mình bắt nó báo cáo mà mình không xem báo cáo. Nếu mình không xem báo cáo, thằng đó báo cáo lên một lần, hai lần, ba lần, đâm ra nó bảo "Làm báo cáo làm chi." Tôi kể một chuyện vui. Câu chuyện trong đời hồi mới đi làm, mới bắt đầu đi làm quản lí dự án. Có cái anh chuyên môn đi thu lượm tin tức để báo cáo. Tuần nào đầu tháng anh ấy cũng gọi phôn xuống, "Thế nào, anh báo cáo con số với tôi." Mình rất chịu khó mình xuống mình đo, hôm nay thế này hôm nay thế kia. Mình chịu khó mình xem. Đầu tháng nào anh ta cũng gọi. Có tháng tôi không lấy. Không biết con số ra làm sao cả. Anh ta gọi xuống, "Thế nào, con số defect rate của anh là bao nhiêu?" Chết rồi, tháng này mình không biết. Mình ngồi mình nghĩ nghĩ, nó thì nó ngồi ở văn phòng ông vice president nó gọi xuống. Mình mới nghĩ, tháng trước con số là số 6 thì phải. "Phải rồi, tháng trước anh là con số 6, lần này là bao nhiêu?" "Lần này số 3." "Thanh you," cúp phôn. Mình nghĩ giản dị, nó chả hỏi gì cả, mình hỏi tuần trước nó cũng nói vậy, ok. Tháng sau nó gọi, mình nói bừa, "số 4," nó cũng chẳng nói năng gì, cúp phôn. Mình ngồi mình nghĩ, thằng ấy làm cái gì đây. Sau này mình lên một anh bạn cũng làm quản lí. Tôi bảo tôi đo, anh ấy bảo "Mày đo làm cái gì? Thằng ấy chỉ báo cáo, có ai đọc đâu." Gần 2 năm sau mỗi lần gọi phôn, tôi bảo, "Tháng trước mình nói số 3,

bây giờ nói số 4, tháng tới mình nói số 5, rồi tháng tới mình trở lại số 3," nó cũng chả hỏi cái gì cả. Không có người nào phân tích hết. Báo cáo để đó có ai đọc đâu. Thì mình hơi đâu mình làm chuyện đó làm chi cho mất công của mình. Mình còn bao nhiêu việc mình làm. Thế là từ đó trở đi suốt 2 năm trời toàn báo cáo láo hết. Mà mình báo cáo láo, những người khác báo cáo láo, ông lớn ở trên ông ấy cũng chả biết. Cũng không sao cả. Thằng kia về sau tôi gặp tôi hỏi nó bảo, "Nhiệm vụ của tao được trả lương để lấy các báo cáo. Còn việc đó xong đấy là nhiệm vụ của người khác. Còn tao phải lấy báo cáo của tụi mày lên để tao làm báo cáo, chứ nếu không thì tao không có việc. Việc của tao chỉ là báo cáo lên, ai làm thế nào thì làm." Mình thấy công việc như vậy thì mình cũng cười.

Cho đến khi tôi đi qua công ti khác tôi làm hoàn cảnh khác đi. Mà mình quen cái thói báo cáo láo đó rồi. Thế sang công ti kia cũng quản lí một cái dự án. Cũng là vui. Hôm ấy các nhân viên nó làm, chả biết cái tay kia nó làm ra sao. Hôm đó trên văn phòng gọi xuống, tôi cho ngay một con số. Mười lăm phút sau có lệnh gọi tôi lên. "Anh làm thế nào, tôi vẽ cái đồ hình nó lệch thế này. Anh giải thích tại sao nó lệch thế này đi. Rõ ràng cái khớp đây này, anh thấy không, con số nó chạy đều đặn thế này, tự nhiên nó lệch ra bên ngoài. Anh giải thích đi." Mình đứng mà mồ hôi chảy ròng ròng, mình nói bậy nó biết ngay, phó chủ tịch nó biết liền. "Đây này, xác suất rõ ràng như thế này, cận trên đây, cận dưới đây, con số của anh làm lệch thế này, anh giải thích đi." Mình đứng gãi đầu gãi tai biết nói làm sao đây? "Tôi cũng không hiểu làm sao nó lệch thế." "Anh làm quản lí dự án, anh phải theo dõi con số này chứ. Tự nhiên con số nó

lệch thế này thì anh phải biết lí do tại sao. Anh phải giải thích cái lí do. Tôi không nói con số sai con số đúng. Lí do. Giải thích đi." Thì không giải thích được. Ông kia ông ấy cũng nhìn mình, mình cũng mới vô, ông ấy cười cười cười, biết chắc thằng này lưu manh nói bậy. Đây là tâm lí. Ông ấy bảo "Anh trở xuống anh coi lại đi. Thứ nhất, con số anh báo cáo có đúng không. Nếu nó đúng, giải thích lí do tại sao." Tôi chạy xuống tôi hỏi nhân viên hoá ra con số khác. Thế bây giờ không biết mình trả lời làm sao đây. Trót nói dối rồi thì phải làm sao đây? Sau khi mình tìm hiểu mọi sự rồi thì mình đành lên mình thú thật. Có hai cách, một là mình nói dối, mình nói dối, nó vặn, mình nói dối nữa, nó vặn nữa, trước sau thế nào cũng lòi ra thôi, tôi nói thật. Tôi lên tôi gặp ông ấy bảo, "Hôm qua hôm kia anh gọi tôi, tôi vội quá tôi nói bậy. Bây giờ con số là như thế này, tôi giải thích là thế này." Ông ấy ngồi ông ấy bảo, "Tao biết mà. Tao theo dõi nó hàng tháng rồi, con số nó xê xích thế này thôi, tự nhiên nó nhảy lên. Nó không có lí do gì để mà nhảy lên thế trừ phi có chuyện gì đó xảy ra ghê gớm lắm. Mà công ti mọi chuyện điều hoà thì không thể con số là vậy. Tao theo dõi hàng tuần một, con số này lạ quá đi. Tao đoán là mày mới vô làm chắc mày nói bậy." Mình lên mình thú tội trước. Ông ấy bảo "Không sao cả. Anh lên thế này rất hay." Tôi lên, tôi học được điều này. Tôi bảo "À thì ra ông giám đốc ông ấy theo dõi." Mà ông ấy theo dõi thì mình không dám làm láo. Đó là kinh nghiệm, về sau tôi lên chức giám đốc, tôi cũng làm y như vậy. Tôi chia sẻ với các anh chị cái kinh nghiệm này.

Cái chuyện làm việc mình chỉ có thể thích ứng với hoàn cảnh, chuyện nó xảy ra mình chỉ có thể giải quyết được vấn đề khi mình có đầy đủ dữ kiện. Tức là mình

phải luôn luôn cân đo. Mà cân đo đến đâu thì phải theo dõi đến đấy. Vẽ cái đồ hình nó khớp nó chạy thế nào đấy, lên xuống ra sao mình phải theo dõi. Luôn luôn tôi có cái phương pháp gọi là xác suất. Nó có cái cận trên, cận dưới, nằm trong giới hạn đó thì không sao. Thí dụ mình bảo "Tôi giao công việc cho anh ấy làm trong vòng 1 năm. Mỗi một tháng anh ấy báo cáo. Mỗi một tháng tôi đi nửa ngày, hụt một ngày, 3 giờ 5 giờ không sao cả. Anh hụt một hai giờ thì không phải lo. Trong phạm vi độ 4 tiếng đồng hồ thì không để ý đến. Nó ra ngoài 4 tiếng đồng hồ, nó ra ngoài 1 ngày thì phải để ý đến. Mà nó ra ngoài 1 tuần thì phải điều chỉnh. Nó có phương pháp làm việc như vậy. Khi mình theo dõi những cái đó thì không bao giờ công việc nó chậm được. Đó là phương pháp làm việc tôi áp dụng tối đa. Cho đến từng giờ, hàng ngày hàng tháng tôi vẫn áp dụng phương pháp đó. Tức là khi nó nằm trong phạm vi mình kiểm soát được, thì không để ý đến. Mà khi nó bước ra khỏi, mặc dù chỉ lệch đi 1 con số thôi, mình phải hỏi ngay lí do nó làm sao. Một người quản lí phải dựa vào chỗ đó.

Tức là khi mình đã lên trên rồi, mình ở dưới chăm cái dự án, thì để biết làm thế nào tất cả các anh phải báo cáo lên cho tôi xem. Nó thành cái dashboard, xem cái đồng hồ nó chạy thế nào. Nó chọn lọc ra sao. Nếu nó chạy đúng mức như vậy thì mình cứ để yên không cần hỏi. Nó lân ra ngoài là phải hỏi ngay. Khi mình bắt đầu mình hỏi là nhân viên phía dưới bắt đầu bối rối nếu báo cáo láo. Không hỏi con số tại sao nó như vậy. Mình hỏi là "Con số này hình như nó hơi lệch. Anh giải thích lí do, rationale, explanation. Khi đó nó biết là mình biết việc, lần sau không ai báo cáo láo. Chỉ hỏi 2 lần không ai dám báo cáo láo. Đấy là cách làm việc của tôi. Các anh chị

còn thắc mắc gì về cái này nữa không ạ? Hôm nay các anh chị nắm vững CMMI chưa?

2. Cải tiến qui trình của tổ chức

Qui trình là gì?

Qui trình là một trình tự có tổ chức các hoạt động để hoàn thành cái gì đó. Chẳng hạn: Dự án phần mềm. Trong trường hợp này, dự án là việc áp dụng tài nguyên vào qui trình đó. Tài nguyên là con người, công cụ và kĩ thuật mà bạn áp dụng khi tuân theo qui trình. Thuật ngữ "con người" cũng chỉ ra kĩ năng và kinh nghiệm của thành viên tổ. Trình tự là trật tự theo đó mọi sự được hoàn thành. Thuật ngữ "trật tự" nghĩa là bạn phải tuân theo nó "từng bước một" tương ức theo qui tắc. Chẳng hạn, bạn phải hiểu yêu cầu trước khi bắt đầu thiết kế; chỉ khi thiết kế được hoàn thành thì bạn mới có thể bắt đầu viết mã v.v.

Qui trình được đại diện bởi ba yếu tố:

1. Hiệu quả: Mối quan hệ giữa việc dùng tài nguyên và kết quả được hoàn thành.
2. Thời gian chu kì: "Tốc độ" của qui trình, tức là, thời gian cần để hoàn thành một qui trình.
3. Và Chất lượng: Chất lượng của qui trình như được xác định bởi người dùng như đáp ứng yêu cầu, không có lỗi v.v.

Tổ hợp của ba yếu tố này xác định ra năng lực của tổ hay tổ chức. Cải tiến liên tục là việc thay đổi hay nâng cấp từ năng lực mức thấp hơn lên năng lực mức cao hơn.

Không có đào tạo đúng, người phát triển phần mềm sẽ làm bất kì cái gì họ muốn chỉ để làm cho công việc của họ được thực hiện. Vì phát triển phần mềm là hoạt động "làm việc theo tổ," không phải là hoạt động cá nhân, điều quan trọng là mọi thành viên tổ đều tuân theo những qui tắc nào đó như tuân theo "Qui trình được xác định" cho dự án đó. Người quản lí dự án phải nhận diện "Qui trình được xác định" trong bản kế hoạch dự án và giám sát các hoạt động để đảm bảo rằng các thành viên tổ tuân theo nó để có được kết quả mong muốn như hiệu quả, chất lượng và tốc độ.

Một trong những vấn đề chính trong đào tạo khoa học máy tính là sinh viên có xu hướng làm việc cô lập. Điển hình, từng người được trao cho một vấn đề để giải quyết, một chương trình để viết mã, và từng người được cho điểm tương ứng theo thành tích cá nhân. Khi sinh viên đi làm, từng người sẽ tiếp tục làm bất kì cái gì có thể để làm cho việc làm của họ được thực hiện, giống như khi họ còn trong trường. Không có hiểu biết về làm việc tổ bằng việc tuân theo qui trình, nhiều dự án sẽ không chuyển giao được phần mềm cho khách hàng trong lịch biểu và có chất lượng. Nhiều dự án thường chậm và có chất lượng kém. Chúng thất bại vì thiếu sự phối hợp và làm việc tổ điều cho phép các thành viên tổ làm việc cùng nhau. Chúng thất bại vì một số thành viên vội vàng viết mã mà không thực sự hiểu các yêu cầu. Chúng thất bại vì có quá nhiều thay đổi trong dự án điều thường tới trễ và không có qui trình giải quyết thay đổi. Chúng thất bại vì người quản lí dự án không biết cách lập kế hoạch, tổ chức hay ước lượng thời gian, lịch biểu và nỗ lực được cần để hoàn thành dự án.

Qui trình được xác định là bản lộ trình cho dự án. Nó yêu cầu các thành viên tổ tuân theo "con đường" từng bước một. Từng bước phải có vai trò và trách nhiệm được xác định rõ ràng cho từng thành viên tổ. Khi thay đổi xảy ra, bản lộ trình nhận diện rõ ràng ai làm cái gì để cho tổ có thể làm cho công việc của họ được thực hiện có hiệu quả, chất lượng và đáp ứng lịch biểu dự án (tốc độ).

Qui trình phần mềm-1

Một sai lầm thông thường trong những người hành nghề phần mềm là ở chỗ phát triển phần mềm chỉ là lập trình. Khi họ nghĩ về lập trình, họ nghĩ tới các ngôn ngữ như C, C++, Java v.v. và chừng nào họ còn có thể viết mã trong các ngôn ngữ đó, họ có thể làm phần mềm. Thực ra, lập trình chỉ là một phần nhỏ của qui trình phát triển phần mềm. Có nhiều điều phải được thực hiện trước khi việc lập trình có thể xảy ra.

Qui trình của việc phát triển phần mềm bao gồm nhiều bước. Tổng của mọi bước được gọi là vòng đời phát triển phần mềm. Có nhiều vòng đời phát triển phần mềm như mô hình thác đổ, mô hình xoáy ốc, mô hình đưa ra tăng dần v.v. Tuy nhiên, tất cả các mô hình này đều bao gồm năm qui trình cơ bản tạo nên vòng đời phần mềm: Yêu cầu, Thiết kế, Thực hiện (viết mã), Tích hợp & kiểm thử, và Bảo trì.

Qui trình yêu cầu: Điển hình, tổ dự án nhận được đặc tả yêu cầu từ khách hàng. Tổ kiểm điểm, phân tích các yêu cầu này rồi gặp gỡ khách hàng để thảo luận và

làm hợp thức các yêu cầu này. Người quản lí dự án và một số thành viên then chốt phải dùng một số kĩ thuật để đánh giá "nhu cầu thực" của khách hàng. Sai lầm thông thường trong các sinh viên là coi đặc tả yêu cầu là đủ tốt để bắt đầu dự án cho nên họ không học phân tích và thẩm tra các yêu cầu này. Thực ra, phần lớn các đặc tả yêu cầu do khách hàng viết đều không tốt. Phần lớn có nhiều yêu cầu xung đột hay thiếu và không biểu thị "nhu cầu thực" của khách hàng. Nhiều yêu cầu chỉ là "ước muốn" chứ không phải là "nhu cầu" và một số thậm chí đã áp đặt giải pháp cho người phát triển. Đó là lí do tại sao tổ dự án phải phân tích, kiểm điểm và làm hợp thức chúng với khách hàng trước khi dự án có thể bắt đầu. Nếu qui trình yêu cầu không được thực hiện đúng, phần mềm cuối cùng có thể không hữu dụng cho khách hàng hay không đáp ứng nhu cầu của họ.

Đặc tả yêu cầu được khách hàng viết thường phản ánh cái nhìn của người dùng. Tổ dự án phải phân tích và hiểu nhu cầu của người dùng, cách nó sẽ được dùng, và biến đổi chúng thành cái nhìn của người phát triển, nơi họ có thể thực hiện. Nếu yêu cầu không được xác định rõ vì khách hàng có thể không biết điều họ thực sự muốn, tổ dự án có thể phải dùng kĩ thuật như làm bản mẫu nhanh theo đó một "bản mẫu" đơn giản được xây dựng mô phỏng chức năng của phần mềm cuối cùng được mong đợi. Bằng việc dùng "bản mẫu" này để trình diễn cho khách hàng, tổ dự án có thể thảo luận chi tiết với khách hàng để hiểu các sản phẩm cuối cùng sẽ được dùng và xác định "yêu cầu thực." Qui trình yêu cầu được hoàn tất khi bản đặc tả yêu cầu phần mềm được chuyển đầy đủ sang cách nhìn của người phát triển để cho tổ dự án có thể chuyển sang pha tiếp. (Lưu ý: Tổ dự án càng

để nhiều nỗ lực vào qui trình này, các yêu cầu sẽ càng ít có khả năng thay đổi và cơ hội cho dự án phần mềm đạt tới sự thoả mãn của khách hàng là cao.)

Qui trình thiết kế: Trong qui trình này, tổ dự án quyết định "cách" họ sẽ xây dựng sản phẩm phần mềm để cho nó đáp ứng các đặc tả được chấp thuận. Thông thường qui trình thiết kế trải qua vài bước từ mức cao (kiến trúc) tới các mức thấp hơn (thiết kế chi tiết). Tại mức kiến trúc, các yêu cầu được tổ chức thành các kiểu hay cách nhìn khác nhau. Cách nhìn là biểu diễn của tập các cấu phần hệ thống và mối quan hệ giữa chúng. Đây là chỗ cấu phần phần cứng, cấu phần phần mềm và cấu phần giao diện được nhận diện và được tổ chức về cách chúng sẽ được thực hiện. Qui trình này được gọi là "làm mịn dần" (từng bước một) vì nó cho phép tổ dự án quản lí độ phức tạp của phần mềm. Bắt đầu ở mức cao tại tầng cấu phần, tổ dự án có thể phân rã chức năng thành các nhiệm vụ đặc biệt hơn ở câu lệnh ngôn ngữ lập trình. Khi thiết kế được hoàn tất, nó được ghi lại trong tài liệu đặc tả thiết kế. (Lưu ý: Tổ dự án càng để nhiều nỗ lực vào qui trình này, sản phẩm cuối cùng sẽ càng trở nên có chất lượng cao hơn và cấu trúc tốt hơn. Các thuộc tính nào đó như tính hiệu năng, tính đổi qui mô, tính bảo trì v.v được nhận diện trong qui trình thiết kế)

Qui trình thực hiện (viết mã): Trong qui trình này, tổ dự án bắt đầu viết mã của phần mềm dựa trên cấu phần thiết kế. Phần mềm được chia thành các đơn vị tách rời được gọi là mô đun để giải quyết độ phức tạp của qui trình lập trình. Tổ phải thực hiện những mô đun này tương ứng theo chuẩn và thủ tục viết mã. Thành viên tổ cũng chịu trách nhiệm cho làm tài liệu đúng mô tả cho

mã của họ và cho kiểm thử mã để đảm bảo tính đúng đắn.

Thành viên tổ phải kiểm thử công việc riêng của họ để chắc chúng làm việc tốt (kiểm thử đơn vị) trước khi trao chúng cho qui trình tiếp. (Lưu ý: Qui trình này không yêu cầu nhiều nỗ lực nếu các qui trình trước đó được thực hiện đúng. Viết mã thường chỉ là thực hiện của thiết kế chi tiết. Nếu thiết kế được làm tốt, dự án không bao giờ có vấn đề với viết mã.)

Qui trình tích hợp & kiểm thử: Trong qui trình này, tổ dự án làm hợp thức và thẩm tra rằng phần mềm đáp ứng yêu cầu và làm việc như mong đợi. Vì các mô đun đã được phát triển tách biệt, kiểm thử là rất quan trọng để chắc rằng chúng tất cả đều cùng làm việc với nhau như đã lập kế hoạch. Ngay cả với thiết kế tốt, sự không tương hợp giữa các mô đun có thể xảy ra và chúng cần được nhận diện và sửa để làm đầy đủ việc tích hợp khi mọi mô đun riêng được tổ hợp để tạo nên sản phẩm phần mềm tích hợp. Có một số kiểm thử phải được thực hiện như kiểm thử chức năng, kiểm thử hệ thống, kiểm thử tích hợp, kiểm thử hiệu năng, kiểm thử an ninh v.v. (Lưu ý: Với dự án nhỏ, tổ dự án tiến hành tất cả những kiểm thử này. Với dự án lớn hơn, thường có tổ kiểm thử độc lập sẽ tiến hành các kiểm thử này. Mọi kiểm thử đều phải được lập kế hoạch, tổ chức và các trường hợp kiểm thử phải được kiểm điểm và chấp thuận trước khi việc kiểm thử có thể được thực hiện.)

Sau khi tất cả kiểm thử được hoàn tất thành công tổ dự án chuyển giao phần mềm cho khách hàng. Thường khách hàng sẽ tiến hành kiểm thử chấp nhận trên phần mềm để xác định liệu nó có đáp ứng đặc tả yêu

cầu hay không, điều cả hai bên đã đồng khi trong qui trình yêu cầu. Nếu phần mềm được chấp nhận, nó được cài đặt và dùng bởi khách hàng.

Qui trình bảo trì: Phần lớn các sản phẩm phần mềm là không tĩnh tại mà sẽ thay đổi. Trong bảo trì, tổ dự án tiếp tục cung cấp hỗ trợ bằng việc sửa lỗi (nếu có), thêm chức năng mới, thay đổi phần mềm được dùng cho nền mới, hay thích nghi phần mềm với công nghệ mới. Mặc dầu nhiều sinh viên tin rằng dự án hoàn thành chuyển giao cho khách hàng, công việc tổ đáng phải xong, nhưng trong thực tế, mọi sản phẩm phần mềm đều tiến hoá qua thời gian để đáp ứng nhu cầu thay đổi của khách hàng. Bảo trì có lẽ là qui trình lâu nhất trong vòng đời phát triển phần mềm, nó có thể kéo dài vài năm sau khi phần mềm được chuyển giao cho khách hàng.

Nếu bạn hiểu qui trình phát triển phần mềm này, bạn có thể áp dụng nó cho bất kì mô hình nào như thác đổi, gia tăng hay xoáy ốc tuỳ theo kiểu dự án bạn đang dùng.

Qui trình phần mềm-2

Tôi nhận được một email mà người gửi hỏi: “Khác biệt giữa qui trình phần mềm và qui trình được xác định là gì? Khác biệt giữ qui trình được xác định và qui trình cá nhân là gì?"

Theo định nghĩa, “Qui trình phần mềm” là trình tự các bước mà người kĩ sư phần mềm PHẢI TUÂN THEO để làm công việc của mình (cái gì và cái gì tiếp) nhưng KHÔNG xác định cách làm nó. Chẳng hạn bạn phải thiết

kế trước viết mã và viết mã trước kiểm thử. Nó KHÔNG cho bạn biết cách thiết kế hay cách viết mã.

"Qui trình được xác định" là trình tự các bước đã được làm tài liệu ĐƯỢC YÊU CẦU bởi một công ti trong thực hiện một việc làm. Chẳng hạn công ti yêu cầu mọi người phần mềm phải tuân theo vòng đời thác đổ và mọi kĩ sư phải bắt đầu dự án bằng yêu cầu trước khi thiết kế và thiết kế trước viết mã v.v. Nếu bạn bỏ qua trình tự này và bắt đầu với viết mã thì bạn vi phạm qui trình được xác định của công ti. Qui trình được xác định cung cấp cho kĩ sư phần mềm trình tự các bước cho mọi công việc mà người phần mềm phải tuân theo như lập kế hoạch, theo dõi, và quản lí mọi công việc phần mềm một cách đúng đắn và đầy đủ. Bằng việc tuân theo qui trình được xác định, người kĩ sư phần mềm biết cái gì cần làm và cái gì cần làm tiếp đó. Người kĩ sư phần mềm cũng có thể đo công việc của mình, theo dõi tiến độ của mình theo mục đích dự án để chắc chắn rằng người đó đang làm nó tương ứng với người khác trong dự án nên họ có thể phối hợp công việc của mình để tạo ra sản phẩm phần mềm tương ứng.

"Qui trình cá nhân" là trình tự được CHI TIẾT HOÁ các bước hướng dẫn cho người kĩ sư phần mềm làm công việc RIÊNG của người đó. Nó thường dựa trên qui trình đã được thiết lập như Qui trình được xác định nhưng được SỬA ĐỔI tương ứng với kinh nghiệm cá nhân riêng của người đó. Qui trình cá nhân cung cấp cho các cá nhân khuôn khổ để cải tiến công việc của họ và để nhất quán làm công việc chất lượng cao. Người kĩ sư phần mềm TẠO RA qui trình chi tiết RIÊNG CỦA MÌNH dựa trên dữ liệu người đó thu thập (đo) và làm

điều chỉnh tương ứng. Đây là cách RIÊNG của người đó tuân theo "Qui trình được xác định." Chẳng hạn nếu người đó đi theo qui trình phần mềm cá nhân riêng của mình và thấy nó tạo ra nhiều lỗi hơn thì người đó phải thay đổi nó cho tới khi nó cho người đó sản phẩm chất lượng tốt hơn.

Qui trình phát triển phần mềm

Nhiều sinh viên được đào tạo tốt trong viết mã vì đa số thời gian của họ ở đại học tập trung vào viết mã nhưng ít tập trung vào qui trình phát triển phần mềm. Một số sinh viên coi phát triển phần mềm là công việc cá nhân vì họ thường nhận được phân công cá nhân điều mỗi người phải làm và phần lớn nhiệm vụ được phân là về viết mã. Khi họ đi làm nhiều người có xu hướng bắt đầu viết mã ngay lập tức sau khi nhận được phân công mà không hiểu đầy đủ vấn đề. Vì yêu cầu thường xuyên thay đổi và họ phải sửa mã của mình, họ thường phạm sai lầm cần sửa. Chúng càng được sửa, mã của họ càng trở nên phức tạp hơn. Cuối cùng, sẽ quá khó không thể hiểu nổi cấu trúc mã của họ và logic của họ cho kiểm thử hay bảo trì. Nếu người phát triển chỉ viết mã và sửa bất kì cái gì họ muốn thì sẽ khó mà tích hợp mã của họ vào một sản phẩm phần mềm cố kết.

Khi sinh viên được đào tạo về qui trình phát triển phần mềm, họ hiểu rằng phát triển không chỉ là viết mã mà nó là nỗ lực tổ. Trước khi bắt đầu, tổ phải có hiểu biết chung về vấn đề mà họ phải giải quyết, mục đích dự án, hoạt động dự án nơi từng thành viên tổ có vai trò và trách nhiệm nào đó cho công việc. Qui trình phần mềm

là bản lộ trình chỉ ra cho tổ cách phát triển phần mềm từ mức cao tới mức chi tiết. Có một trật tự trình tự được gọi là vòng đời phát triển nơi toàn thể qui trình được phân chia thành vài pha và từng pha có vai trò, trách nhiệm nào đó cho các thành viên tổ.

Qui trình phần mềm bắt đầu bằng việc hiểu vấn đề qua việc thẩm tra các yêu cầu với khách hàng. Bởi vì khách hàng thường không mô tả nhu cầu của họ rõ ràng và thường đổi ý của họ, tổ phải chắc điều khách hàng đã đòi hỏi là điều họ cần. Chỉ khi các yêu cầu này được thảo luận và xác nhận, tổ mới bắt đầu làm tài liệu chúng thành đặc tả yêu cầu phần mềm. Bằng việc tuân theo qui trình, tổ hiểu rằng nguyên nhân thông thường nhất của thất bại dự án là các yêu cầu không ổn định hay xác định kém và bằng việc có các yêu cầu được kiểm nghiệm đầy đủ họ có thể giảm bớt rủi ro của yêu cầu thay đổi. Sau khi khách hàng đã chấp nhận tài liệu yêu cầu, tổ bắt đầu thảo luận riêng của họ để chắc rằng mọi thành viên của tổ hiểu vấn đề đủ rõ trước khi họ chia dự án thành những nhiệm vụ nhỏ hơn mà họ phải làm.

Bằng việc tuân theo qui trình phần mềm, tổ xác định kiến trúc của hệ thống qua nhiều cấu phần. Các thành viên tổ nhận diện giao diện giữa các cấu phần này và thế rồi thẩm tra rằng mọi cấu phần có thể được dõi vết trở lại các yêu cầu để cho sản phẩm sẽ đáp ứng nhu cầu của khách hàng. Điều này là quan trọng bởi vì bất kì lỗi nào trong kiến trúc cũng có thể tạo ra vấn đề trong việc tích hợp phần mềm trong pha sau. Khi dự án tiến từ pha yêu cầu sang pha kiến trúc, tổ sẽ khám phá ra rằng có nhiều "yêu cầu suy dẫn" mà cần được thực hiện. Thỉnh thoảng danh sách các yêu cầu suy diễn còn nhiều hơn

các yêu cầu nguyên gốc vì kiến trúc sư phải xem xét mọi thứ như hiệu năng, tính đổi qui mô, tính dùng được, và tính bảo trì mà khách hàng mong đợi nhưng không đòi hỏi. Dựa trên kiến trúc, tổ chọn công cụ phần mềm mà họ sẽ dùng để xây dựng hệ thống. Đồng thời, các thành viên tổ có thể bắt đầu tạo ra tập các kiểm thử đơn vị để được áp dụng một khi họ kết thúc viết mã. Nhiều người phát triển không nghĩ về kiểm thử mãi cho tới khi họ kết thúc viết mã nhưng nếu họ bắt đầu phát triển kiểm thử đơn vị sớm trong pha kiến trúc, họ có thể tránh được nhiều sai lầm mà thường xảy ra trong pha thiết kế. Bằng cách làm việc trên kiểm thử đơn vị sớm, họ hiểu các yêu cầu tốt hơn, điều giúp cho họ thiết kế sản phẩm tốt hơn.

Pha thiết kế là nơi logic nội bộ của từng cấu phần được tổ chức. Trong pha này, các thành viên tổ làm việc trên chi tiết về cấu trúc dữ liệu và thuật toán của từng cấu phần. Trong pha kiến trúc, hội tụ chính là vào nhận diện các cấu phần và mối quan hệ của những cấu phần này và cách chúng tương tác lẫn nhau. Trong pha thiết kế, hội tụ là vào thiết kế logic cho từng cấu phần trong những cấu phần này. Các thành viên tổ tuân theo qui trình và dùng tập các kĩ thuật và hướng dẫn như phân hoạch vấn đề và trừu tượng hoá. Việc dùng trừu tượng hoá cho phép các thành viên tổ hội tụ vào một nhiệm vụ mỗi lúc, không lo nghĩ về chi tiết của các nhiệm vụ khác. Khi mọi nhiệm vụ đều được thực hiện, tổ phải tuân theo qui trình trắc nghiệm bằng việc có cuộc kiểm điểm thiết kế của từng cấu phần. Một cách điển hình, người quản lí dự án, kiến trúc sư hệ thống, và đảm bảo chất lượng phải tham gia vào kiểm điểm để chắc rằng mọi việc đều được thực hiện tương ứng theo kế hoạch, kiến trúc hệ thống tổng thể, và trong tuân thủ theo qui trình chuẩn.

Khi thiết kế được thẩm tra đầy đủ và chấp thuận, các thành viên tổ có thể bắt đầu pha thực hiện (viết mã). Nhiều người thích viết mã ngay nhưng nếu họ tuân theo qui trình, họ phải chọn cấu trúc dữ liệu đáp ứng nhu cầu của thiết kế; giữ cho cấu trúc logic đơn giản nhất có thể được; lựa chọn các tên biến có nghĩa nhất quán với chuẩn. Bằng việc lựa cấu trúc dữ liệu trước, rồi bắt đầu với tổ chức logic của cấu trúc mã họ có thể tránh được nhiều sai lầm thường xảy ra trong pha này. Sau khi hoàn thành việc viết mã đầu tiên, thành viên tổ phải tiến hành buổi kiểm thảo (walkthrough) để kiểm điểm mã của họ để chắc chúng tuân theo chuẩn viết mã và cấu trúc của chúng là nhất quán với cách dự án được tổ chức và được lập kế hoạch. Từng thành viên tổ phải thực hiện kiểm thử đơn vị và sửa các lỗi đã được tìm ra trước khi chuyển sang pha tiếp.

Trong pha kiểm thử, các thành viên tổ chuyển mã của họ cho tổ kiểm thử. Kiểm thử là quá trình thực hiện một chương trình với ý định tìm ra lỗi. Kiểm thử tốt là kiểm thử có xác suất cao tìm ra lỗi. Bằng việc tuân theo qui trình, thành viên tổ hiểu rằng kiểm thử nên được lập kế hoạch sớm trong pha thiết kế và mọi kiểm thử đều phải dõi vết được về yêu cầu của khách hàng. Việc kiểm thử phải bắt đầu trong phần nhỏ nhất và tiến tới kiểm thử phần lớn hơn và cuối cùng tới toàn thể sản phẩm phần mềm. (Kiểm thử đơn vị, kiểm thử chức năng, kiểm thử tích hợp, kiểm thử hệ thống v.v.)

Qui trình cho dự án phần mềm

Tuần trước, một người bạn có sở hữu một công ti phần mềm gọi điện thoại cho tôi: “Sau khi làm việc cho một công ti phần mềm trong sáu năm, tôi bắt đầu công ti riêng của tôi. Tôi tuyển những sinh viên tốt nghiệp hàng đầu và trả lương hậu cho họ, tôi có một số khách hàng và công ti của tôi phát triển nhanh chóng. Tuy nhiên, hiện thời chúng tôi có vấn đề với lỗi nhiều và trượt lịch biểu. Nếu những vấn đề này tiếp tục, công ti của tôi sẽ lâm vào rắc rối. Là người chủ, tôi thường dành hầu hết thời gian của mình với khách hàng để đưa vào việc kinh doanh mới cho nên làm sao tôi có thể sửa được những vấn đề này và tiếp tục phát triển công ti của tôi?"

Tôi nói với ông ấy: “Ông không thể bỏ qua các vấn đề bằng việc dành thời gian cho khách hàng và để mọi người làm việc mà không có giám sát nào. Việc sửa lỗi yêu cầu phần lớn thời gian của người phát triển và có lẽ nó là lí do chính cho việc trượt lịch biểu. Điều khẩn thiết cần làm bây giờ là hội tụ vào việc tìm và sửa lỗi. Tôi nghĩ ông cần tiến hành kiểm điểm thường xuyên hơn để nhận diện và sửa chúng sớm nhất có thể được. Trong các cuộc kiểm điểm này, ông phải để mọi người phát triển và người kiểm thử tới và biết cách lỗi đã bị phạm phải thế nào và làm sao sửa được chúng."

Ông ta hỏi: “Sao lại mọi người? Tôi không để người không làm việc trong cùng dự án tham dự họp kiểm điểm được. Điều đó là phí thời gian.”

Tôi giải thích: “Về căn bản, những người phát triển trong dự án đều có kiểm điểm giữa họ để tìm và sửa lỗi. Đây là những lỗi 'không biết" với người phát triển khác,

người đã không tham dự buổi kiểm điểm và họ có thể phạm phải cùng sai lầm lần nữa. Lí do cho "kiểm điểm dành riêng" là người phát triển không muốn người khác tìm ra lỗi của họ. Tuy nhiên, là người chủ công ti, ông phải coi kiểm điểm như quá trình học tập cho nên mọi lỗi được tìm ra đều là cơ hội học tập. Người phát triển càng biết nhiều về nguyên nhân của lỗi, càng dễ tránh phạm phải chúng.

Ông ấy dường như thoả mãn: “Được, tôi có thể bắt đầu bằng kiểm điểm và để mọi người phát triển tới và học. Tôi có thể làm cái gì khác?"

Tôi tiếp tục: “Dự án phần mềm điển hình bắt đầu với ít người, rồi khi nó tiến triển, ông thêm người cho nó. Phần lớn các môn quản lí dự án bao giờ cũng dạy cách tiếp cận dần dần và nó có tác dụng tốt với doanh nghiệp xây dựng. Ông chỉ cần vài nhà kiến trúc lúc ban đầu nhưng khi các pha kiến trúc và thiết kế được thực hiện rồi, công ti đem nhiều công nhân vào xây dựng. Tuy nhiên, với dự án phần mềm cách tiếp cận này không có tác dụng tốt. Sẽ là tốt hơn nếu ông có thể bắt đầu với đầy đủ cán bộ từ đầu."

Ông ấy ngạc nhiên: “Sao chúng tôi cần nhiều người hơn ngay từ những pha đầu? Điều đó sẽ tốn nhiều tiền hơn.”

Tôi giải thích: “Đó là lí do tại sao nhiều dự án phần mềm có vấn đề. Ông bắt đầu chỉ với ít người làm việc với yêu cầu của khách hàng và rồi họ hiểu điều gì phải làm. Rồi họ bắt đầu kiến trúc và thiết kế hệ thống. Vì họ làm việc trên dự án từ đầu, họ hoà hợp tốt. Khi ông thêm nhiều người vào dự án về sau, người mới không có thời

gian để biết lẫn nhau hay hình thành cách làm việc tổ hiệu quả. Xung đột cá nhân có thể xảy ra trong thời gian này. Người mới không có thời gian hiểu yêu cầu nhưng họ phải thiết kế và viết mã ngay lập tức."

"Họ có thể không biết cách tích hợp công việc của họ với kiến trúc hệ thống. Họ có thể không hiểu chức năng chi tiết của hệ thống nhưng họ phải làm cho việc của mình được thực hiện trong lịch biểu bị giới hạn. Đây là chỗ nhiều sai lầm bị phạm phải. Người mới được mong đợi hình dung ra mọi thứ theo cách riêng của họ trong thời gian rất ngắn. Tất nhiên, họ có những câu hỏi và họ phải hỏi người khác, những người có đó từ lúc bắt đầu để giúp. Điều này có thể làm gián đoạn công việc dự án bởi vì thời gian để dạy người mới về dự án bao giờ cũng mất lâu hơn mong đợi, cho nên dự án có thể bị chậm. Do sức ép lịch biểu, mọi người phải vội vàng làm cho việc của mình được thực hiện và nhiều lỗi bị phạm phải."

Ông ấy gật đầu: “Tôi chưa bao giờ nghĩ về điều đó, ông có thể đúng. Có những vấn đề giữa người mới và người cũ, họ tranh cãi mọi lúc. Tôi cần làm gì khác?"

Tôi bảo ông ấy: “Vòng đời phần mềm truyền thống là cách tiếp cận tuần tự giống như dây chuyền lắp ráp trong công nghiệp. Trước hết, kĩ sư yêu cầu làm việc với khách hàng để làm tài liệu cho tất cả các yêu cầu. Kiến trúc sư hệ thống sẽ hình dung ra cách hệ thống làm việc dựa trên các yêu cầu này. Thế rồi người phát triển sẽ thiết kế và rồi viết mã. Tiếp đó, người kiểm thử sẽ kiểm thử chúng. Sau khi tất cả kiểm thử đã được hoàn thành, sản phẩm phần mềm được đưa ra cho khách hàng. Chúng ta hãy nhìn lại cách tiếp cận này. Kĩ sư yêu cầu

chỉ quan tâm tới điều khách hàng cần và hội tụ vào việc làm tài liệu cho chúng. Kiến trúc sư đặt mọi thứ dựa trên tài liệu yêu cầu và thiết kế hệ thống. Người phát triển hội tụ phần lớn vào viết mã dựa trên thiết kế. Người kiểm thử không chăm nom tới cách toàn thể hệ thống được thiết kế mà chỉ vào các trường hợp kiểm thử, nơi kiểm xem mã có qua được kiểm thử hay không. Điều gì sẽ xảy ra khi yêu cầu sai? Điều gì sẽ xảy ra nếu thiết kế bị thiếu chức năng nào đó? Điều gì sẽ xảy ra khi người kiểm thử kiểm vào các mã mà không được làm tài liệu trong tài liệu yêu cầu? Điều gì sẽ xảy ra nếu có những chức năng mới được thêm vào trong các phút cuối. Nếu người kiểm thử không biết về chức năng mới, họ không thể kiểm thử được nó. Đó là lí do cho cách tiếp cận kiểu dây chuyền lắp ráp này thực sự không có tác dụng tốt với phần mềm. Vòng đời thác đổ là tốt cho giảng dạy, nó dễ hiểu nhưng nó không có tác dụng tốt trong công nghiệp.

Bạn tôi ngần ngại một chốc: "Đó là điều chúng tôi vẫn làm sao? Ông gợi ý gì để chúng tôi làm khác đi?"

Tôi giải thích: "Đó là lí do tại sao đào tạo kĩ nghệ phần mềm yêu cầu rằng công ti phải xác định qui trình phát triển tích hợp đầy đủ mọi thành viên tổ sớm nhất có thể được. Qui trình được xác định không phải là một hoạt động theo chuỗi mà là tăng dần với mọi vai trò được tích hợp đầy đủ. Nó rất quan trọng để bắt đầu với đầy đủ cán bộ hay ít nhất với đa số người lúc ban đầu. Người quản lí dự án phải thảo luận về vai trò, trách nhiệm của từng thành viên tổ và điều từng người cần làm để thành công. Đào tạo làm việc theo tổ lúc bắt đầu sẽ tạo ra một số các thảo luận, nơi các thành viên tổ có thể nói về điều họ cần từ nhau. Rồi tổ trình bày nhu cầu của

họ, quan điểm của họ với người quản lí dự án. Sau vài tuần đầu trong dự án, tổ có thể đi tới qui trình mà mọi người đều biết nhiệm vụ của họ là gì, và cách để làm việc với chúng. Thông tin này nên được làm tài liệu như một phần của qui trình tổ dự án nơi mọi người đều biết ưu tiên dự án là gì, và cách ra quyết định để thoả mãn nhu cầu dự án.

Bạn tôi hỏi: “Sao ông cần vài tuần đầu để xây dựng dự án? Có quá nhiều không?"

Tôi trả lời: “Làm việc theo tổ là hoạt động quan trọng nhất trong dự án phần mềm. Ông không thể mong đợi rằng những người không biết lẫn nhau sẽ hài hoà với nhau trong vài ngày. Họ cần thời gian để biết lẫn nhau và đi tới qui trình được xác định mà mọi người sẽ đồng ý tuân theo. Thay vì qui trình tuần tự, ông nên yêu cầu tổ phát triển một qui trình song hành để xây dựng phần mềm. Chẳng hạn, kiến trúc sư xác định kiến trúc tổng thể của sản phẩm phần mềm, xác định các tính năng cho từng bản đưa ra và tiến hành kiểm điểm ban đầu nơi mọi thành viên tổ cần phải tham dự. Đây là lúc mọi người nên thực sự học về khía cạnh kĩ thuật và hỏi các câu hỏi nếu cần. Sau kiểm điểm ban đầu, nhóm những người phát triển sẽ bắt đầu thiết kế ngay lập tức khi nhóm khác làm việc trên kiểm thử hệ thống. Sau thiết kế, tổ phải tiến hành kiểm điểm thiết kế cùng với toàn tổ và nhóm kiểm thử phải chắc chắn rằng kiểm thử hệ thống của họ sẽ làm việc tốt với thiết kế này. Nếu mọi sự tốt lành, thì tổ có thể bắt đầu thiết kế chi tiết và thực hiện, đồng thời nhóm kiểm thử sẽ làm việc trên việc phát triển các kiểm thử chức năng. Mọi pha đều phải có kiểm điểm với sự tham dự của toàn tổ để chắc chắn họ phối hợp các hoạt

động của mình và công việc của họ được tích hợp đầy đủ. Đến lúc mã được viết ra và kiểm thử (đơn vị được kiểm thử), mọi người sẽ sẵn sàng cho kiểm điểm mã. Mọi mã đều phải trải qua kiểm điểm trước khi được đưa vào hệ thống quản lí cấu hình. Sau khi mã được kiểm, người kiểm thử sẽ bắt đầu kiểm và bất kì lỗi nào được nhận diện đều phải được sửa theo qui trình thay đổi. Về căn bản, mọi thứ trong dự án đều phải tuân theo qui trình song hành được xác định tốt và người quản lí dự án phải dùng độ đo để trắc nghiệm mọi thứ. Lịch biểu dự án nên được dõi vết theo ước lượng gốc, mọi yêu cầu đều phải được quản lí, cũng như mọi lỗi. Bằng việc để mọi người tham gia sớm và giúp xác định qui trình, các thành viên tổ hiểu công việc của họ và những cột mốc chính. Họ biết điều họ cần làm từng tuần và trắc nghiệm công việc của họ lẫn nhau. Trong nỗ lực cộng tác này, các thành viên tổ có thể lưu ý khi vấn đề xuất hiện và có hành động sửa chữa thay vì chờ đợi cho tới pha sau. Nếu tổ tuân theo qui trình được xác định tốt, họ có thể cải tiến cách họ giải quyết các lỗi và việc trượt lịch, và cuối cùng mọi dự án sẽ được cải tiến.

Bạn tôi nói: "Vậy ông nghĩ làm việc tổ và qui trình là nhân tố mấu chốt cho cải tiến dự án à?"

Tôi bảo ông ấy: "Vâng, dứt khoát rồi. Qui trình phần mềm có thể có vẻ đơn giản nhưng chúng biểu thị cho thay đổi mạnh mẽ trong cách mọi người làm công việc của họ. Với làm việc theo tổ, mọi người nhận ra rằng công việc của họ là nỗ lực cộng tác nơi mọi thứ đều phải được tích hợp. Đây là chìa khoá cho thành công của mọi công việc phát triển, không ai làm việc một mình. Với kỉ luật kĩ nghệ phần mềm, họ sẽ biết phải làm gì và

làm gì tiếp. Có kỉ luật là bước đầu tiên để phát triển công ti của ông. Bên cạnh những qui trình kĩ thuật này, ông có thể cải tiến qui trình quản lí của ông nữa. Chừng nào chưa có qui trình phát triển được được xác định tốt, sẽ khó mà biết ai đang làm gì hay ai là tốt và ai không tốt. Qui trình được xác định tốt cho phép người quản lí dự án hiểu công việc phát triển và vai trò, trách nhiệm của từng thành viên tổ. Điều này cho người quản lí cơ hội làm việc với những người có công việc không chấp chấp nhận được, và thưởng cho những người hoàn thành công việc xuất sắc.

Qui trình đơn giản cho dự án nhỏ

Hôm qua, một sinh viên đã tốt nghiệp vài năm trước tới gặp tôi. Anh ta nói: "Em đã đi làm cho một công ti phần mềm lớn trong vài năm, bây giờ em muốn bắt đầu công ti riêng của em. Em có đủ kinh nghiệm và một số tiền mà em đã tiết kiệm trong những năm qua. Em cũng có một vài người phát triển người sẵn lòng làm việc cho em. Em lập kế hoạch bắt đầu công ti của em bằng việc làm các dự án nhỏ và dần dần phát triển công ti. Em cần lời khuyên của thầy để thiết lập qui trình tốt để phát triển phần mềm có chất lượng với chi phí thấp nhất có thể được. Vì em chỉ có "vốn giới hạn," em không muốn phí hoài nói. Thầy nghĩ điều đó có thể được thực hiện không?"

Sau khi suy nghĩ một chốc, tôi bảo anh ta: "Điều tốt là bắt đầu từ các dự án nhỏ. Nếu bạn có người phát triển có kĩ năng, bạn có cơ hội tốt hơn để thành công và xây dựng danh tiếng cho công ti của bạn. Bắt đầu từ

những cái nhỏ cũng có thể tiết kiệm được tiền bạc bởi vì nếu bạn phạm sai lầm nó sẽ không là thảm hoạ và bạn có thể phục hồi được. Phần lớn mọi người thường bắt đầu công ti bằng việc hội tụ vào khía cạnh kĩ thuật. Đó KHÔNG phải là ý tưởng hay. Là người chủ công ti, bạn phải hội tụ vào khách hàng đầu tiên. Ưu tiên của bạn phải bắt đầu với việc có khách hàng và hiểu yêu cầu của họ. Đây là việc của bạn bởi vì không hiểu nhu cầu của khách hàng và không đáp ứng mong đợi của khách hàng là nguyên nhân chính cho thất bại doanh nghiệp. Bạn phải tiếp xúc với khách hàng, thu được yêu cầu, phân tích và làm tài liệu chúng cẩn thận. Bạn phải trắc nghiệm với khách hàng để chắc rằng bạn hiểu rõ yêu cầu và mong đợi của họ. Khách hàng phải chấp thuận tài liệu yêu cầu trước khi bạn bắt đầu làm bất kì việc nào. Trong khi có nhiều công cụ làm yêu cầu có tính thương mại bán sẵn trên thị trường, để tiết kiệm tiền, tôi gợi ý rằng bạn dùng Microsoft Word hay Excel cho việc làm tài liệu yêu cầu của bạn."

Anh ta đồng ý: "Điều đó là dễ dàng, phần lớn các máy tính đều tới với Window 7 và Microsoft Office nạp sẵn cho nêm em sẽ không phải chi thêm tiền phụ."

Tôi tiếp tục: "Nếu khách hàng chấp thuận yêu cầu và kí hợp đồng thì bạn có thể bắt đầu dự án. Với những dự án nhỏ bạn nên dùng cách tiếp cận phát triển Agile bởi vì nó ít rủi ro hơn các phương pháp khác. Bạn có thể tránh rủi ro của thay đổi yêu cầu thường xuyên điều thường xảy ra trong công nghiệp."

Anh ta gật đầu: "Một ý hay là dùng cách tiếp cận Agile, em biết "Phương pháp Scrum."

Tôi bảo anh ta: “Trong trường hợp đó, bạn phải chia yêu cầu thành những nhiệm vụ nhỏ hơn dùng kĩ thuật Cấu trúc phân việc (WBS). Bạn cần ước lượng công việc cần được thực hiện, phân công chúng cho người phát triển của bạn và xác định ngày chuyển giao. Với cách tiếp cận Agile dùng Scrum, bạn phải xác định công việc được yêu cầu cho từng đợt nước rút Sprint (2-4 tuần). Bạn có thể dùng Microsoft’s Excel để lập kế hoạch công việc và làm tài liệu cho mọi nhiệm vụ. Nếu bạn có dự án lớn hơn, bạn có thể dùng công cụ "Project" của Microsoft. Điều này sẽ cho phép bạn làm một số việc lập kế hoạch lại trong trường hợp thay đổi yêu cầu hay nếu khách hàng quyết định thay đổi một số chức năng vào phút chót. Bạn cũng có thể dùng công cụ “Bugzilla” để theo dõi và ghi lại các lỗi, những nâng cao, hay thay đổi yêu cầu. Bugzilla là chọn lựa phổ biến trong những người phát triển phần mềm và nó dễ dùng với đào tạo tối thiểu.

Anh ta đồng ý: “Em có thể kiếm những công cụ này và để cho người phát triển của em bắt đầu dùng chúng.”

Tôi tiếp tục: “Làm việc tổ trong Agile là rất quan trọng. Trước khi bắt đầu dự án bạn phải đào tạo mọi người phát triển về cách làm việc trong tổ. Cho dù họ có thể đã biết "Scrum," bạn vẫn cần nhắc nhở họ về cách tiếp cận này và vai trò của họ để chắc không có hiểu lầm nào. Với dự án Scrum, có ba vai trò: Người chủ sản phẩm là người chịu trách nhiệm cho khía cạnh doanh nghiệp của dự án và ra quyết định về sản phẩm. Thầy Scrum người quản lí qui trình phát triển, giúp người phát triển làm việc cùng nhau, tạo điều kiện họp và theo dõi

tiến độ và vấn đề. Tổ phát triển người xây dựng ra sản phẩm bằng làm việc cùng nhau để đạt tới mục đích của dự án. Vì dự án của bạn là nhỏ, bạn nên giả định giữ cả vai trò người chủ sản phẩm và thầy Scrum."

Anh ta gật đầu: "Vâng, em không thể đảm đương được việc có nhiều người chừng nào em còn chưa thể phát triển công ti lên. Gợi ý của thầy rất hợp lí."

Tôi tiếp tục: "Điều quan trọng là thiết lập việc tách bạch môi trường phát triển, môi trường kiểm thử, và môi trường sản xuất sớm để tránh việc trộn lẫn công việc. Đây là khái niệm đơn giản nhưng tôi đã thấy nhiều công ti nhỏ phạm sai lầm và trộn lẫn công việc của họ với những phiên bản sai, phần mềm đã kiểm thử để cùng phần mềm chưa kiểm thử, cho nên tôi nghĩa đáng nhắc bạn. Bạn phải thiết lập hệ thống kiểm soát nguồn để tạo điều kiện cho qui trình phát triển và lưu trữ mọi công việc đầy đủ. Quản lí cấu hình phần mềm là quan trọng bởi vì là công ti nhỏ, bạn có thể không có qui trình kiểm soát mạnh và những người được chỉ định để thực hiện vai trò quản lí cấu hình. Mọi người phát triển phải hiểu cơ sở về làm sao "Kiểm vào" công việc của họ và "Kiểm ra" khi chúng được thực hiện để tránh dư thừa, khi nào họ làm thay đổi cho phần mềm. Trong khi có một số công cụ sẵn có, bạn vẫn có thể dùng phương pháp thủ công bằng việc để những người phát triển gõ vào tình trạng số hiệu nhiệm vụ như "Mở," "Đóng" và "Đã phân công" để cho mọi người có thể theo dõi thay đổi một cách dễ dàng. Để tiết kiệm tiền, bạn có thể xem xét dùng công cụ "nguồn mở" như CVS cho quản lí cấu hình. Là công ti nhỏ, bạn không cần phải mua những công cụ đắt tiền nhưng ĐỪNG BAO GIỜ thử tiết kiệm tiền bằng

việc giảm đào tạo. Để thành công bạn phải hội tụ vào việc cải tiến kĩ năng của những người phát triển của bạn bằng những đào tạo phụ vì công nghệ thường xuyên thay đổi."

Anh ta nói: "Em bao giờ cũng nhớ việc dạy của thầy từ nhiều năm trước, học liên tục là chìa khoá cho thành công trong khu vực này."

Tôi kết luận: "Tôi vui là bạn nhớ điều đó. Phần còn lại của qui trình phát triển là đơn giản. Với từng dự án, bạn sẽ có cuộc họp hàng ngày để phân công công việc cho người phát triển vì bạn xây dựng phần mềm trên cơ sở hàng ngày. Bạn sẽ có những người phát triển viết mã, thực hiện kiểm thử đơn vị, tạo ra trường hợp kiểm thử và các kiểm thử tự động để chạy mọi đêm sau khi phần mềm được dựng ban ngày và kiểm tra các lỗi. Là thầy Scrum, bạn phải giám sát tiến độ trên cơ sở hàng ngày, kiểm tra các lỗi và để chúng được sửa và phải chắc mọi người đang làm nhiệm vụ được phân công của họ một cách tương ứng. Nếu bạn kiểm soát qui trình phát triển đơn giản này tốt thì bạn có thể mong đợi sản phẩm chất lượng cao. Bạn KHÔNG cần mua nhiều công cụ, bạn KHÔNG cần chi tiền vào bất kì cái gì phụ thêm, yếu tố quan trọng là có qui trình đơn giản mà mọi người hiểu và cam kết tuân theo nó cho công việc của họ. Một qui trình có chất lượng KHÔNG phải phức tạp, hay tinh vi mà nó phải được mọi người phát triển hiểu. Chìa khoá để thành công trong cách tiếp cận Agile KHÔNG phải là kĩ thuật mà là làm việc tổ và trao đổi. Bạn phải động viên mọi người thảo luận vấn đề, hỗ trợ lẫn nhau và sẵn lòng giải quyết vấn đề khi chúng tới. Tổ phải có mục đích chung, biết vai trò và trách nhiệm của họ và tuân theo

qui trình chất lượng và đạo đức để hỗ trợ cho công ti đạt tới việc thoả mãn khách hàng.

Để thành công cải tiến qui trình phần mềm

"Thầy có thể cho lời khuyên về làm sao để thành công trong cải tiến qui trình phần mềm?"

Đáp: Bước đầu tiên trong cải tiến qui trình hiệu quả là thay đổi hành vi của người quản lí và người phát triển. Là người kĩ sư phần mềm, bạn có thể hỗ trợ cho thay đổi nhưng thay đổi thực sự chỉ xảy ra khi người quản lí chấp nhận thái độ mới đối với cải tiến.

Vấn đề là làm sao để người phát triển phần mềm làm những điều không liên quan trực tiếp tới việc chuyển giao sản phẩm phần mềm? Đây là vấn đề khó bởi nhiều lí do. Thứ nhất, người phát triển bao giờ cũng bận rộn. Thứ hai, họ có thể không hiểu điều bạn muốn họ làm hay tại sao phải làm. Và thứ ba, họ có thể không tin rằng điều bạn gợi ý sẽ giúp họ trong việc của họ.

Do đó, điều quan trọng nhất là làm cho cấp lãnh đạo hành xử khác đi. Đây là lí do tại sao chúng ta cần giải thích cho cấp lãnh đạo về rủi ro của việc không cải tiến để cho họ có thể nhận ra sự khẩn thiết. Nếu họ sẵn lòng sống với các hậu quả của quá trình hỗn độn và rủi ro, chẳng cái gì sẽ xảy ra.

Đây là vài gợi ý mà bạn có thể thấy có ích:

1. Phải chắc rằng cấp quản lí thừa nhận việc cải tiến qui trình là trách nhiệm của họ. Nếu người phát triển không tham gia vào hay hỗ trợ tích

cực cho việc cải tiến qui trình, nó sẽ không xảy ra.

2. Có được thoả thuận từ cấp quản lí về vài hành động mấu chốt cần thực hiện trước nhất. Tôi gợi ý mỗi lúc làm một thay đổi nhỏ thôi. Đừng cố làm cái gì đó mơ hồ và vô nghĩa như "Lấy CMMI mức 5." Để tạo ra tiến bộ, bạn cần tập trung vào cái gì đó có ích, thực tế, như giám định phần mềm để loại bỏ lỗi. Hành động này là đo được nếu đầu tiên bạn đặt ra tuyến sở sở. Bạn phải thu thập một số lỗi của việc đưa ra trước đó và số lỗi được tìm thấy trong giám định sản phẩm.
3. Bạn cần chắc rằng mọi người đều biết trách nhiệm của mình. Một ý hay là viết tất cả những điều đó ra.
4. Thiết lập kế hoạch vận hành, giữ nó đơn giản và có các điểm kiểm (trạng thái tuần) và nhận diện tài nguyên rõ ràng.
5. Biểu thị những điều tham gia vào việc làm cho một thay đổi được hoàn thành. Bạn có thể làm điều này qua báo cáo tiến độ cho cấp quản lí và phải chắc đưa vào lịch biểu và cam kết tài nguyên. Những điều này sẽ giúp mọi người nhận ra cái gì được tham dự vào và nó chiếm bao lâu.
6. Làm vì thành công. Đều đặn nhận diện những thành tựu thực, tuyên dương những người có trách nhiệm và công bố thành tựu của họ. Tuy nhiên bạn phải khiêm tốn và vẫn ở ngoài quá

trình thừa nhận này, việc của bạn là điều phối cải tiến và làm cho nó xảy ra. Điều này sẽ tạo ra nhiệt tình trong những người phát triển và biểu lộ tiến bộ. Một khi đà này diễn ra, khó mà dừng được nó và bạn sẽ đi vững chắc trên con đường của mình.

7. Chìa khoá cho cải tiến qui trình là làm nhiều thay đổi qui trình nhỏ và đơn giản. Ích lợi chính từ phần lớn những thay đổi tới từ vài hành động. Đừng làm thay đổi lớn, bạn sẽ không thành công đâu. Nhớ câu hỏi "Làm sao ăn được voi?" Đáp: "Bằng nhiều miếng nhỏ."

8. Nhớ cung cấp đủ thông tin để cho mọi người biết điều cần làm và khi nào làm. Thử điều đó trong dự án nhỏ, lấy phản hồi từ người trong dự án, người dùng qui trình cải tiến rồi tinh lọc nó dựa trên các kết quả. Bạn có thể phải huấn luyện mọi người và chắc chắn phải giúp họ được bắt đầu bởi vì cải tiến là quá trình học hỏi. Bạn sẽ học nhiều từ việc thực hiện hơn là từ lập kế hoạch hay nói về nó. Học từ thực tế chứ đừng từ ý kiến, cho nên trước khi làm cho người khác thay đổi, bạn phải tự thay đổi mình trước.

9. Nếu cấp quản lí không phân việc cho người phát triển để làm việc với những nhiệm vụ cải tiến, hãy đi tới quản lí cấp cao và nêu rõ ràng rằng nếu thiếu sự tham dự của người phát triển, việc cải tiến qui trình là phí thời gian và tiền bạc. Với những điều kiện này, hoặc quản lí cấp

cao phải tiến bước và giúp đỡ, hoặc bạn sẽ phải tìm việc ở đâu đó khác.

10. Nếu quản lí cấp cao từ chối giải quyết vấn đề, họ không thực sự được thuyết phục về nhu cầu cải tiến. Bất kể điều họ nói, bạn cần dừng lại hay ngắt nỗ lực cho tới khi bạn thu được sự chú ý của họ. Đừng phí thời gian của bạn. Ý tưởng là để làm cho người ta thành công nhanh chóng, với dữ liệu cải tiến vững chắc thì phần còn lại sẽ vào cuộc. Hãy nhớ, thay đổi cần thời gian và bạn phải kiên nhẫn.

Thay đổi qui trình

Bạn tôi, người quản lí một công ti lớn bảo tôi rằng anh ấy đã đem một sản phẩm phần mềm mới ra cải tiến về năng suất và hiệu quả nhưng anh ấy gặp thời gian khó khăn khi để nó làm việc trong công ti của anh ấy.

Tôi bảo anh ấy: "Nhiều người mua phần mềm và giả định rằng "cứ cài đặt nó đi và nó sẽ làm việc" thế rồi phát hiện ra rằng họ phải làm nhiều hơn bởi vì mua phần mềm thì dễ, nhưng thay đổi cách mọi người làm việc lại đòi hỏi nhiều nỗ lực hơn. Tôi đã thấy nhiều công ti đầu tư nhiều tiền vào phần mềm nhưng đã không nhận được ích lợi bởi vì họ không hiểu "thay đổi qui trình."

Với bất kì công nghệ mới nào đều có một tập các bước cần được thực hiện tại chỗ trước khi nó có thể là hữu dụng. Điều quan trọng nhất là con người. Phải có trao đổi và đào tạo về công nghệ mới bởi vì mọi người phải hiểu giá trị của công nghệ mới và nhận đào tạo về

cách dùng nó. Họ cần biết cách nó ảnh hưởng tới công việc của họ và cách nó làm lợi cho họ như làm cho công việc được thực hiện nhanh hơn, tốt hơn, trước khi họ sẽ chấp nhận nó và dùng nó. Theo ý kiến của tôi, Con người là trung tâm của mọi thứ do đó trao đổi là bản chất cho thay đổi xảy ra.

Bước thứ hai là qui trình hay cách con người làm việc. Để dùng công nghệ mới thì mọi người cần tham gia vào qui trình nào đó. Chẳng hạn, người quản lí quen dựa vào người kế toán, cho họ con số và để họ tính ra kết quả mà có thể mất vài giờ. Với phần mềm, người quản lí có thể bấm vào biểu tượng, đưa vào con số và để cho máy tính hiển thị kết quả ngay lập tức. Qui trình cũ dựa vào người kế toán để làm tính toán là cách mọi người đã hiểu rõ rồi. Qui trình dùng máy tính thay vì con người là mới và mọi người có thể cảm thấy không thoải mái. Điều quan trọng là phải giải thích cho họ về ích lợi (nhanh hơn và chính xác hơn) và đào tạo họ dùng phần mềm mới. Khi mọi người hiểu cách mọi sự được thực hiện ngày nay và cách mọi sự sẽ được thực hiện sau khi cài đặt phần mềm mới và ích lợi họ thu được thì họ sẽ thay đổi.

Thay đổi qui trình yêu cầu rằng chúng ta phải bắt đầu từ con người, giúp họ hiểu qui trình mới bằng việc cung cấp đào tạo, và kết thúc với phần mềm được sử dụng. Không có thay đổi này về qui trình, mọi người sẽ trải nghiệm lẫn lộn và đôi khi chống lại thay đổi.

Cùng điều này có thể được áp dụng cho chương trình giáo dục mới. Đừng trông đợi "Cứ đem nó vào đi và sinh viên sẽ đăng kí học." Đại học phải trao đổi rõ ràng với sinh viên về khác biệt giữa chương trình đào tạo

mới và cũ. Các giáo sư cần giải thích giá trị, ích lợi và cách nó sẽ giúp cho sinh viên học tri thức, kĩ năng mới để cho họ có thể có việc làm tốt hơn, nghề nghiệp tốt hơn. Hơn bao giờ hết, trao đổi là bản chất trong việc thay đổi qui trình này. Chừng nào sinh viên và gia đình của họ còn chưa hiểu rõ nó và chấp nhận nó như chương trình tốt hơn, chương trình giáo dục mới có thể không thành công.

Cải tiến qui trình- Việc của cấp quản lí

Tuần trước tôi nhận được một email từ một kĩ sư phần mềm. Anh ấy viết: "Tôi gặp khó khăn để làm cho công ti của tôi cải tiến cách chúng tôi phát triển phần mềm. Tôi đã đọc blog SEGVN của thầy về CMMI và tin rằng nó có thể giúp cho công ti của tôi. Vấn đề là làm cho mọi người bắt đầu cải tiến. Người chủ công ti nói: "Việc đó tốn kém." Người quản lí nói: "Chúng ta phải thuê tư vấn và điều đó là tốn kém." Người quản lí dự án nói: "Điều đó yêu cầu nhiều đào tạo và chúng tôi không có thời gian." Nhiều người phát triển cân nhắc về nó: "Sao lại bận tâm? điều đó là phí thời gian." Xin thầy cho tôi lời khuyên về cách thay đổi những thái độ này. Rất cám ơn thầy."

Đáp: Chừng nào cấp quản lí của bạn chưa muốn làm điều đó, chẳng cái gì sẽ thay đổi. Để cải tiến qui trình phần mềm, bạn phải có quyết tâm của cấp quản lí và có đào tạo, bằng không nó sẽ không xảy ra. Nếu người chủ nghĩ cải tiến qui trình là tốn kém, bạn có thể thu thập dữ liệu về số dự án đáp ứng lịch biểu, chi phí và chất lượng và số dự án không đáp ứng. Bằng việc biết

con số thực tế, người chủ có thể đổi ý. Nếu ông ấy biết rằng ông ấy có thể tăng lợi nhuận một cách có ý nghĩa khi công ti của ông ấy trưởng thành và phát triển tốt hơn, nhanh hơn và chi phí thấp hơn thì ông ấy có thể đồng ý với bạn.

Ngày nay các công ti sợ cải tiến qui trình bởi vì giá của đào tạo và đánh giá CMMI. Nếu ích lợi của cải tiến là lớn hơn nhiều so với chi phí của cải tiến thì sự việc sẽ thay đổi. Nếu người chủ quyết định cải tiến thì mọi người quản lí sẽ phải tuân theo. Họ có thể không thích điều đó nhưng họ sẽ không bất đồng với người chủ. Tuy nhiên, người quản lí cần hiểu CMMI cũng như có kế hoạch để quản lí hoạt động cải tiến để cho nó sẽ thành công. Đây là chỗ nhà tư vấn tốt và có hiểu biết có thể giúp để chắc mọi người quản lí có quyết tâm đầy đủ để cải tiến qui trình phần mềm của họ. Đây là yếu tố then chốt xác định thành công hay thất bại. Nhà tư vấn "không tốt lắm" sẽ vội vàng trong việc cung cấp đào tạo và tiến hành đánh giá (Đó là chỗ họ được trả nhiều tiền) nhưng nhà tư vấn tốt sẽ không làm điều đó, họ biết rằng họ phải hội tụ vào quyết tâm của cấp quản lí để làm điều đó một cách đúng đắn. Đây là tình huống:

Giả sử công ti bắt đầu cải tiến qui trình. Những người phát triển phàn nàn rằng họ phải tham gia đào tạo CMMI điều lấy đi thời gian của họ làm việc trên dự án phần mềm. Điều gì sẽ xảy ra khi một số dự án bị muộn? Điều gì sẽ xảy ra khi khách hàng không hài lòng bởi vì người phát triển không làm việc trên dự án của họ mà dành thời gian theo lớp học? Điều gì sẽ xảy ra khi người phát triển phải tuân theo cái gì đó mà họ không quen thuộc? Điều gì sẽ xảy ra mà người phát triển phải thu

thập độ đo, cái gì đó họ chưa bao giờ làm trước đây? Điều gì sẽ xảy ra nếu người phát triển được bảo làm cái gì đó khác với điều họ vẫn thường làm? Điều gì sẽ xảy ra khi người phát triển tiếp tục phàn nàn? Chừng nào người chủ và mọi người quản lí còn chưa quyết tâm đầy đủ, sự việc có thể thay đổi. Công ti có thể quyết định bỏ cải tiến và coi nó như sai lầm. Trong trường hợp đó, chẳng cái gì sẽ xảy ra lần nữa. Trừ phi người chủ và mọi người quản lí quyết định duy trì quyết tâm của họ thì cải tiến mới có thể xảy ra.

Điều cơ sở nhất của cải tiến là cải tiến cách người quản lí quản lí dự án phần mềm. Thay vì đồng ý với khách hàng về lịch biểu dựa trên yêu cầu được xác định nghèo nàn, người quản lí dự án sẽ nhấn mạnh vào việc có qui trình quản lí yêu cầu để quản lí mọi thay đổi. Họ sẽ phải chắc rằng kế hoạch dự án được xác định tương ứng theo qui trình (ước lượng, thương lượng, thoả thuận) trước khi họ bắt đầu dự án. Mọi dự án sẽ phải thiết lập các vai trò, trách nhiệm được xác định rõ ràng. Bằng việc để cho mọi dự án tuân theo thực hành quản lí dự án, thay đổi sẽ bắt đầu xảy ra trong công ti.

Người phát triển sẽ bắt đầu đòi hỏi nhiều đào tạo hơn, đặc biệt trong khu vực họ cần thực hiện công việc của họ. Người quản lí bắt đầu dùng độ đo để ra quyết định thay vì phỏng đoán. Dự án sẽ có ước lượng tốt hơn về lịch biểu và ít lỗi hơn. Tất nhiên, một số người sẽ hài lòng và một số có thể không hài lòng vì họ quen làm mọi sự theo cách riêng của họ. Tuân theo qui trình chuẩn không phải là điều họ muốn. Những người không thích sẽ bỏ đi và sẽ được thay thế bởi người phát triển mới. Người phát triển mới có thể không biết điều đã xảy ra

cho nên họ vẫn theo qui trình được xác định. Qua thời gian, khi hiệu năng dự án tốt dần lên, người chủ sẽ nhận ra rằng lợi nhuận của công ti ông ấy đang cải thiện và nhiều dự án đáp ứng lịch biểu, chi phí và chất lượng. Nhiều khách hàng bắt đầu tin vào cải tiến và yêu cầu nhiều kinh doanh hơn thì bạn đạt tới "thay đổi chính" trong cách công ti của bạn phát triển phần mềm.

Điều quan trọng phải nhận ra là với mọi thay đổi, cấp quản lí là yếu tố then chốt. Chẳng cái gì sẽ được hoàn thành nếu họ không tin tưởng và quyết tâm. Quyết tâm bắt đầu từ hiểu biết và hỗ trợ và nếu họ không hiểu CMMI, nó cần gì, nó cần bao lâu, họ phải quản lí hoạt động nào thì họ có thể không hỗ trợ cho nó. Không có hỗ trợ của họ, cải tiến sẽ không xảy ra.

Do đó, tôi nghĩ rằng chìa khoá thực để cải tiến qui trình thành công là quyết tâm của cấp quản lí. Dù bạn thích hay không, người phát triển không thể làm cho sự việc xảy ra được nếu không có quyết tâm từ cấp quản lí. Đây là chỗ nhà tư vấn tốt và có tri thức sẽ tới để giúp. Khó tìm được nhà tư vấn tốt và họ có thể là đắt giá, nhưng họ thực sự xứng đáng điều đó.

Cải tiến qui trình- Câu hỏi cần hỏi

Tôi đã nhận được nhiều emails liên quan tới bài báo tôi viết về cải tiến qui trình dùng CMMI. Dường như vẫn có các ý kiến khác nhau về việc thực hiện nó. Các nhà tư vấn biện hộ rằng điều đầu tiên cần làm là có đánh giá để xác định mức độ trưởng thành của công ti. Bằng việc biết mức độ của công ti, nhà tư vấn có thể giúp phát

triển kế hoạch cải tiến cho nên công ti có thể chuyển sang mức tiếp. Các công ti đào tạo không đồng ý và nói rằng điều đầu tiên phải được thực hiện là có đào tạo về CMMI cho người phát triển để cho họ có thể hiểu và thực hiện hoạt động cải tiến. Mặc dầu cả hai là những hoạt động cần thiết nhưng chúng phải không được thực hiện trước. Tôi tin rằng thu được cam kết từ người chủ công ti phải là điều đầu tiên và là quan trọng nhất. Bất kì cải tiến nào mà không có cam kết từ người quản lí cấp cao sẽ không bao giờ có tác dụng.

Lí do các nhà tư vấn nhấn mạnh vào việc có đánh giá trước hết là vì đó là cách họ làm tiền. Đánh giá điển hình tốn nhiều tiền ($20,000 USD tới $ 40,000 USD) để cho công ti biết nó đang ở đâu trên các mức CMMI nhưng đó chỉ là hoạt động đầu tiên. Lập kế hoạch cải tiến thêm yêu cầu phí phụ thêm. Có nhiều hoạt động tư vấn hơn cần phải đi theo nơi công ti phải tiếp tục trả tiền trước khi nó có thể cải tiến. Một số công ti đào tạo chủ trương đào tạo CMMI trước bởi vì họ làm tiền theo cách đó nữa. Một lớp CMMI căn bản có thể tốn $10,000 USD hay hơn nhưng nó chỉ là chi phí đầu tiên. Có nhiều đào tạo hơn được cần tới về sau. Bằng việc đi theo những lời khuyên đó, công ti phải chi nhiều tiền trước khi họ có thể cải tiến được cái gì. Do đó, điều quan trọng là người chủ công ti hiểu mọi bước cần thiết về cải tiến qui trình.

Trước khi bắt đầu cải tiến qui trình người chủ công ti cần trả lời những câu hỏi sau: “Tại sao bạn quan tâm tới việc dùng CMMI? Tại sao bạn muốn cải tiến qui trình? Lí do gì bạn muốn thay đổi cách công ti của bạn phát triển phần mềm? Bạn có sẵn lòng đầu tư vào con người riêng của bạn không? Bạn có cam kết làm cho mọi

sự xảy ra trong công ti của bạn không? Bạn có biết rằng cải tiến qui trình là cuộc hành trình lâu dài không? Bạn có sẵn lòng quyết tâm theo đuổi trong nhiều năm các hoạt động cải tiến để đạt tới kết quả mong muốn không?" Chừng nào bạn chưa thể trả lời được những câu hỏi này một cách trung thực, bạn KHÔNG nên bắt đầu cải tiến qui trình. Tất nhiên, nếu mọi điều bạn muốn là mảnh giấy "chứng nhận CMMI" thì tôi không có bình luận thêm gì nữa. Không cần thảo luận thêm nữa.

Người chủ công ti phải hiểu rằng "cải tiến thực" là cuộc hành trình dài. Nó có nhiều chướng ngại, nhiều vấn đề, và không có quyết tâm mạnh để đi theo mọi cách, nó sẽ không thành công. Nếu dự án của bạn có nhiều lỗi, nếu dự án của bạn không đáp ứng lịch biểu, nếu công ti của bạn có nhiều việc thay người phát triển, nếu công ti của bạn mất tiền và mất khách hàng thì cải tiến qui trình là lí do tốt. Nếu bạn muốn mở rộng doanh nghiệp của bạn từ địa phương ra toàn cầu, muốn xây dựng danh tiếng tốt là công ti có chất lượng, thiết lập "tên hiệu" cho công ti của bạn thì cải tiến qui trình cũng là lí do tốt. Nếu bạn muốn công ti của bạn có những người phát triển giỏi với kết quả dự án tốt thì cải tiến qui trình là lí do tốt. Nếu bạn muốn có lợi nhuận, làm nhiều tiền hơn trước đây thì cải tiến qui trình cũng là lí do tốt. Tuy nhiên, bạn phải kiên nhẫn và quyết tâm làm cho nó xảy ra, không phải trong một năm, không phải trong hai năm, mà liên tục bởi vì cải tiến không bao giờ dừng lại.

Nền tảng của cải tiến dùng CMMI là mối tương hỗ giữa qui trình, sản phẩm và kết quả doanh nghiệp. Qui trình tốt phải mang tới sản phẩm tốt và sản phẩm tốt phải mang tới kết quả doanh nghiệp tốt. Tuy nhiên, nhiều

người chỉ hội tụ vào việc có "qui trình" mà không tính tới sản phẩm và kết quả doanh nghiệp. Điều làm tôi ngạc nhiên là người chủ công ti để cho điều đó xảy ra theo cách đó. Tại sao mọi người chỉ muốn có chứng chỉ mà không có kết quả thực nào? Tại sao mọi người muốn "quảng cáo" cái gì đó mà họ thậm chí không biết? Họ có biết rằng "quảng cáo giả" có thể đem tới kết quả thảm hoạ không? Khi một công ti đạt tới chứng nhận CMMI Mức 5, người chủ có thấy thay đổi gì so với năm trước không? Bao nhiêu dự án đã đạt tới chất lượng cao, đáp ứng mục tiêu chi phí và thời gian? Lỗi trung bình trong công ti là gì? Phần trăm dự án đáp ứng lịch biểu là gì? Doanh nghiệp có tăng lên không? Công ti có nhiều khách hàng hôm nay hơn vài năm trước không? Thu nhập có tăng lên đáng kể không so với vài năm trước?

Tôi tin rằng đánh giá CMMI chính thức là tiêu chí cho tính hợp thức, chính xác, cộng tác, nhất quán, và đủ là cách rất tốt để hiểu những điểm mạnh và điểm yếu của công ti của bạn. Tôi cũng tin đào tạo CMMI là có giá trị để giúp người phát triển biết nhiều hơn về qui trình phần mềm. Tuy nhiên, không có quyết tâm mạnh của người chủ công ti tôi tự hỏi điều gì sẽ xảy ra? Cái gì sẽ xảy ra cho người phát triển, người tin rằng công ti sẽ cải tiến rồi chẳng thấy gì xảy ra? Cái gì sẽ xảy ra khi mọi người thấy nhiều lần cải tiến bị thất bại? Họ mất niềm tin của họ vào cấp quản lí thì cái gì xảy ra cũng chẳng thành vấn đề, công ti sẽ không bao giờ có khả năng cải tiến cái gì.

Cải tiến qui trình với CMMI-1

Tôi đã nhận được vài email liên quan tới việc thực hiện CMMI và việc đạt tới mức CMMI nào đó. Có nhiều "cường điệu" về CMMI và tôi đã viết hàng trăm bài báo về chúng trên website này. Về căn bản CMMI là khuôn khổ cho cải tiến qui trình mà công ti có thể dùng để cải tiến chất lượng sản phẩm của nó. CMMI bao gồm năm mức từ 1 tới 5 với 1 là thấp nhất không có cải tiến nào và chất lượng thấp. Mức 2 là về cải tiến ở mức dự án vì dự án là cơ sở của công việc trong bất kì công ti phần mềm nào. Nếu bạn không thể cải tiến được dự án, bạn không thể cải tiến được cái gì. Mức 3 hội tụ vào chuẩn hoá các hoạt động trong công ti để đạt tới sự nhất quán trong mọi công việc. Khi mọi dự án được quản lí tương ứng, chất lượng có thể được cải tiến. Mức 4 hội tụ vào việc đo cả sản phẩm và qui trình. Mục đích của nó là quản lí mọi công việc bằng các độ đo và kiểm soát thống kê để giảm biến thiên. Đạt tới điều này nghĩa là công ti có kiểm soát toàn bộ về mọi công việc cả về chất lượng và hiệu năng. Mức 5 hội tụ vào ngăn ngừa khiếm khuyết và thay đổi qui trình và mục đích của nó là để làm tối ưu cách phần mềm được phát triển để đạt tới chất lượng cao nhất có thể.

Bước quan trọng nhất trong cải tiến là xem xét toàn thể việc phát triển phần mềm như một qui trình mà có thể được đo, được kiểm soát và được cải tiến. Công ti phải chọn mục đích rõ ràng cho cải tiến và có khả năng kiểm tra tiến bộ khi cải tiến xảy ra qua thời gian vì bạn không thể làm mọi thứ ngay một lúc. Thuật ngữ “trưởng thành” ngụ ý thời gian điều có nghĩa là sẽ phải mất vài năm để có được kết quả mong muốn. Một công ti không

thể có được mức cao nhất ngay lập tức. Cũng như đứa bé phải bò trước khi đi và đi trước khi chạy, cải tiến phần mềm phải bắt đầu từ dự án tới tổ chức và rồi tới toàn thể công ti.

Không may, một số người chủ công ti quá bận rộn và không để thời gian để hiểu các nguyên lí này. Thay vì đặt mục đích cải tiến họ lẫn lộn các mức CMMI như mục đích. Tôi đã thấy công ti đặt mục đích đạt tới mức mà thậm chí không hiểu mức 3 là gì. Đặt mức CMMI như mục đích là thuần tuý dốt nát và chứng tỏ thiếu hiểu biết về khuôn khổ này. Mục đích cải tiến qui trình nên được dựa trên kinh doanh của công ti như giảm chi phí, tăng lợi nhuận, cải tiến chất lượng, và đạt tới thoả mãn của khách hàng. Do đó cải tiến qui trình phải là một phần của chiến lược công ti. Nó KHÔNG phải là cái gì đó bạn làm khi bạn có thời gian. Tưởng tượng một doanh nghiệp đặt mục đích có lợi nhuận nhất nếu có thể. Điều đó có nghĩa là công ti sẽ thường xuyên mất tiền không? Cải tiến KHÔNG phải là cái gì đó bạn làm để tìm ra xem công ti ở mức CMMI nào. Tưởng tượng một doanh nghiệp chi nhiều tiền chỉ để biết liệu họ có là doanh nghiệp hay không. Nhưng điều này không có nghĩa gì cả. Cải tiến qui trình KHÔNG phải là cái gì đó để làm chỉ một lần rồi dừng lại MÀ là hoạt động liên tục không bao giờ chấm dứt. Ngay cả khi bạn đạt tới mức cao nhất như CMMI mức 5, nếu bạn dừng lại thì công ti có thể rơi trở lại mức 1 nhanh hơn bạn có thể tưởng.

Việc thiếu hiểu biết về CMMI đã tạo ra nhiều vấn đề, làm nảy sinh phí tiền, mang tiếng xấu và làm tổn hại tới cộng đồng phần mềm. Vài năm trước đây, chính phủ Trung Quốc đặt mục đích thưởng cho công ti phần mềm

đạt tới mức CMMI nào đó. Chỉ trong không đầy một năm, hàng trăm công ti công bố thành tựu của họ về các mức CMMI cao. Vài công ti đã được đánh giá bởi những người đánh giá có đủ tư cách nhưng nhiều công ti đã tự tuyên bố hay được tuyên bố bởi nhà tư vấn không đủ tư cách. Phần lớn nhận được phần thưởng nhưng khi công ti nước ngoài tiến hành đánh giá riêng của họ trước khi đầu tư vào những công ti này, kết quả là khác nhiều với những lời tuyên bố thành đạt vì đa số trong họ đều ở CMMI mức 1. Điều này đã tạo ra danh tiếng xấu cho công nghiệp phần mềm Trung Quốc về chất lượng thấp, không trung thực, lừa dối và đã ngăn cản đầu tư vào khu vực này. Thậm chí ngày nay công nghiệp phần mềm Trung Quốc vẫn bị nhiều doanh nghiệp coi là "Không đáng tin cậy" với chất lượng thấp và hiệu năng kém.

Để cải tiến, mục đích phải được gióng thẳng với nhu cầu và chiều hướng của quản lí của công ti. Người chủ công ti phải chỉ định một người quản lí cấp cao chuyên trách để duy trì việc gióng thẳng này và quản lí mọi hoạt động cải tiến. Người quản lí này phải kiểm điểm và chấp thuận kế hoạch cải tiến, nhận tình trạng về hành động cải tiến, và cung cấp hướng dẫn về công việc sắp tới. Mọi hoạt động đều phải được đo và báo cáo cho người chủ công ti. Người quản lí cải tiến phải xây dựng kế hoạch cải tiến nói rõ phương hướng và mục đích chính, điều hội tụ vào nhu cầu của công ti như "giảm chi phí 10% đến cuối năm," hay tăng số bán lên 10% đến cuối năm. Kế hoạch cải tiến phải nhận diện vấn đề mà cấp quản lí muốn giải quyết và các hành động được thiết kế để làm điều đó và chuyển công ti hướng tới mục đích của nó. Với từng khu vực cải tiến chính, bản kế hoạch liệt kê mọi nhiệm vụ mà mọi người làm, kể cả cung cấp

đào tạo cho nhân viên. Cải tiến phải dựa trên việc đo. Chẳng hạn, dễ nói rằng chất lượng sản phẩm đã cải tiến khi có một cách đo về chất lượng xảy ra sau. Việc đo cải tiến không phải làm cấp tập mà bất kì cách đo nào trong chi phí, chất lượng, và thời gian cũng sẽ đủ. Chẳng hạn số lỗi trong pha dự án, độ chính xác lịch biểu (khác biệt giữa ngày tới hạn và ngày chuyển giao thực tế) v.v.

Khi kế hoạch này đã được chấp thuận, người quản lí phải theo dõi nó để đảm bảo rằng tiến bộ được thực hiện. Có hai biến cố theo dõi điển hình, phiên họp hàng tuần của tổ cải tiến, và phiên họp trạng thái hàng tháng với người quản lí cấp cao. Mục đích của phiên họp tuần là đảm bảo rằng tiến bộ được thực hiện về nhiệm vụ cải tiến. Tình trạng hàng tháng là để thông tin cho người chủ công ti và người điều hành cấp cao về các hoạt động. Chỉ bằng việc đối xử với cải tiến như một dự án doanh nghiệp thực, việc cải tiến mới có thể xảy ra và làm lợi cho công ti.

Cải tiến qui trình với CMMI-2

Hôm qua, một người chủ công ti phần mềm gửi cho tôi một email hỏi: “Làm sao chúng tôi bắt đầu cải tiến qui trình dùng CMMI? Thầy có gợi ý mức CMMI nào để chúng tôi bắt đầu nếu chúng tôi không muốn trả tiền cho việc đánh giá? Phải mất bao lâu để chuyển lên một mức CMMI? Xin thầy lời khuyên."

Đáp: Để bắt đầu cải tiến qui trình dùng CMMI, bạn cần xác định bạn đang ở đâu. Bạn có thể tự đánh giá bằng việc hỏi người của bạn một số câu hỏi theo mô hình

CMMI. Chẳng hạn: “Bạn có kế hoạch dự án cho từng dự án không? Bạn có ước lượng lịch biểu dự án của bạn dựa trên dữ liệu lịch sử không? Bạn có qui trình chuẩn của tổ chức được xác định cho mọi dự án trong công ti của bạn không?" Dựa trên câu trả lời bạn có thể quyết định vấn đề gì bạn cần cải tiến. Tất nhiên, là người chủ công ti, bạn có thể bắt đầu cải tiến qui trình của bạn mà không biết về bất kì mức nào. Bạn có thể đặt ra ưu tiên riêng của bạn dựa trên các vấn đề trong công ti của bạn và phát triển các bản kế hoạch hành động để giải quyết các vấn đề đó. Chẳng hạn, bạn có thể hỏi người phát triển của bạn những điều chính nào tác động tới dự án của họ? Nguyên nhân của việc trượt lịch biểu là gì? Tại sao dự án phần mềm bị trễ? Tại sao phần mềm vẫn có nhiều lỗi sau khi kiểm thử? Bằng việc biết những vấn đề này, bạn có thể bắt đầu sửa chúng.

Dựa trên kinh nghiệm riêng của tôi trong thực hiện CMMI, sẽ phải mất từ 2 tới 4 năm cho một tổ chức để chuyển từ mức nọ lên mức kia. Dữ liệu tại SEI chỉ ra rằng trung bình, phải mất 28 tháng để chuyển từ mức này lên mức kia. Tất nhiên, môi trường làm việc của công ti có thể tác động tới việc nó có thể cải tiến nhanh hay chậm thế nào. Nhiều vấn đề như làm việc tổ, thiếu trách nhiệm hay xung đột cá nhân có thể là vấn đề chính, không chỉ vấn đề kĩ thuật cho nên cải tiến nên lớn hơn việc chỉ tuân theo mô hình kiểu như CMMI. Bởi vì bạn là người chủ, bạn có thể thay đổi môi trường làm việc và làm cho nó tốt hơn, năng động hơn, cộng tác nhiều hơn giữa các công nhân. Nếu bạn quyết tâm cải tiến, bạn có thể làm nhiều điều. Khuyến cáo của tôi là bắt đầu cải tiến qui trình từ từ thôi, chọn điều bạn muốn cải tiến một cách cẩn thận, và giải quyết một vấn đề mỗi lúc. (Lưu ý:

Xin xem lại các bài viết và hỏi & đáp về CMMI trong website này.)

Xin đừng chú ý quá nhiều tới các mức CMMI. Điều quan trọng là để cải tiến, không phải là để đạt mức CMMI. Cải tiến thực phải đem lại kết quả doanh nghiệp thực. Cải tiến thực phải đem lại chất lượng đo được có ý nghĩa lớn với dữ liệu. Cải tiến thực phải đem tới sự thoả mãn nhiều hơn của khách hàng. Cải tiến thực phải đem lại nhiều hài lòng hơn cho người phát triển. Cài tiến thực phải giảm lỗi và cải tiến chất lượng. Tất nhiên, nó phải đem tới nhiều lợi nhuận hơn cho công ti của bạn và cho cả bạn nữa. Nếu những điều này không xảy ra, công ti của bạn không cải tiến đâu. Đừng chi tiền vào cái gì đó chỉ cho bạn "mức vô nghĩa" hay "mảnh giấy." Nhiều người đã làm điều đó và họ hối tiếc vì điều đó không giúp cho doanh nghiệp của họ hay lợi nhuận của họ. Năm ngoái, khi tôi ở Trung Quốc, một quan chức điều hành nói với tôi là công ti ông ta đã được đánh giá ở CMMI Mức 4 nhưng tôi cũng để ý rằng để giảm chi tiêu, công ti đó cũng đã xoá bỏ nhóm kiểm thử và sa thải một số người kiểm thử vì họ đã không "viết mã." Người quản lí khác bảo tôi rằng ông ta làm việc cho công ti CMMI mức 5, khi hỏi về chất lượng phần mềm ông ta nói: “Chúng tôi vẫn có nhiều lỗi trong phần mềm của chúng tôi.” Tôi ngạc nhiên: “Công ti CMMI mức 5 mà vẫn có nhiều lỗi sao? Điều đó là không thật.” Theo kinh nghiệm của tôi với CMMI mức 5, tổ chức phải có chất lượng ít nhất bằng hay tốt hơn 6 sigma. Điều đó nghĩa là 99.9996% sản phẩm phần mềm phải không có lỗi hay không quá 3 lỗi trên một triệu dòng mã. Những tuyên bố này của các công ti được đánh giá ở CMMI mức cao đã chẳng có nghĩa gì với tôi cả.

Trong nhiều năm, tôi đã để ý tới số các công ti tuyên bố rằng họ đã được đánh giá ở CMMI mức 5 đã tăng lên khá nhiều. Dường như là có cạnh tranh trên khắp thế giới về các mức CMMI. Khái niệm rằng "Nếu công ti bạn có chứng chỉ CMMI mức 5 thì công ti tôi cũng muốn có nó nữa" chỉ giúp cho các nhà tư vấn, đặc biệt các nhà tư vấn "không đạo đức mấy" làm tiền. Xin lưu ý rằng Viện kĩ nghệ phần mềm - Software Engineering Institute (SEI) tại Carnegie Mellon KHÔNG "xác nhận" bất kì công ti nào và KHÔNG có "chứng chỉ CMMI" nào dù là bất kì cái gì. "Chứng chỉ" được ban hành bởi nhà tư vấn và nó chẳng liên quan gì tới SEI.

Vì bạn là người chủ của công ti, tôi chắc bạn quan tâm tới doanh nghiệp của bạn và muốn cải tiến nó. Tuy nhiên, có nhiều điều bạn có thể làm và điều đó không cần tới nhiều tiền. Lời khuyên của tôi là: "Nếu bạn muốn đầu tư, xin đầu tư vào người riêng của bạn. Cung cấp đào tạo tốt hơn, giúp cho họ cải tiến kĩ năng của họ, tạo ra môi trường làm việc tốt hơn và dùng độ đo để xác định căn nguyên của vấn đề và sửa chúng. Xin nhớ cho là CMMI chỉ là mô hình, chính bạn mới làm cho mô hình đó hoạt động bằng quyết tâm làm cải tiến thực xảy ra."

Cải tiến thực

Sau khi đăng bài "Cải tiến qui trình với CMMI," tôi nhận được nhiều emails hỏi về "Cải tiến thực" và làm sao họ biết rằng công ti của họ thực sự được cải tiến? Cho nên sau đây là cách nhìn của tôi về "Cải tiến thực."

Cải tiến thực có thể được đo bằng nhiều thứ như tăng năng suất và chất lượng, hài lòng của khách hàng tốt hơn, ít lỗi hay làm lại việc, và lợi nhuận và thị phần cao hơn. Nếu công ti của bạn có "Cải tiến thực," bạn sẽ để ý năng suất tăng lên, chính là số lượng cái ra của sản phẩm làm việc tính theo kích cỡ đối với số công nhân đã cho và thời hạn dự án. Chẳng hạn trước bất kì cải tiến nào, một tổ mười người có thể viết 10 000 dòng mã một tháng. Sau cải tiến, cùng tổ đó có thể viết 15 000 dòng mã một tháng vậy năng suất của bạn đã tăng lên 50%. Cải tiến thực không phải là cái gì đó trừu tượng mà có thể là điều đơn giản như ít lỗi hơn trong sản phẩm phần mềm. Bạn có thể giảm số lỗi bằng việc dành nhiều thời gian hơn cho pha yêu cầu bằng làm việc chặt chẽ với khách hàng để hiểu nhu cầu của họ; làm tài liệu yêu cầu bằng việc dùng kịch bản use case và thẩm tra lại với người dùng để có yêu cầu chính xác hơn; dựng bản mẫu để làm sáng tỏ thiết kế và KHÔNG vội vàng nhảy vào viết mã. Bạn càng có nhiều phiên kiểm điểm với tổ, bạn càng có ít lỗi hơn. Đến cuối, bạn đạt tới chất lượng sản phẩm tốt hơn.

Vì nhiều dự án thường bỏ lỡ các cột mốc và lịch biểu, bạn có thể cải tiến nó bằng việc có cấu trúc phân việc tốt hơn trong khi chia yêu cầu thành các nhiệm vụ chi tiết nhỏ hơn, đủ nhỏ để bạn có thể lập kế hoạch và theo dõi chúng trên cơ sở hàng tuần. Tôi bao giờ cũng khuyên sinh viên chia các yêu cầu lớn thành các nhiệm vụ nhỏ hơn để họ có thể hoàn thành trong quãng 32 giờ (một người-tuần). Nếu cái gì xảy ra gây ra chậm trễ, sẽ dễ sửa chữa một nhiệm vụ nhỏ hơn là nhiệm vụ lớn. Bạn cũng có thể cải tiến lịch biểu bằng việc có ước lượng tốt hơn (nỗ lực, thời gian, chi phí) và cân xứng tốt hơn giữa

kĩ năng và công việc. Đừng chỉ phân công bất kì người phát triển nào cho bất kì nhiệm vụ nào một cách ngẫu nhiên. Lúc bắt đầu của mọi dự án, bạn phải biết kĩ năng và kinh nghiệm của từng người phát triển để cho bạn có thể phân công họ vào nhiệm vụ tương ứng. Điều cũng rất quan trọng là tiến tục cung cấp đào tạo để giữ cho kĩ năng của họ được cập nhật. Để duy trì phần mềm chất lượng cao (tính bảo trì được) bạn nên chia thiết kế thành các mô đun nhỏ hơn để dễ sửa và bảo trì. Bạn phải chắc những mô đun này có nhiều chú thích và tài liệu tốt, đặc biệt làm tài liệu cho bất kì thay đổi nào để sánh đúng với mã.

Có nhiều điều mà bạn có thể đạt tới "Cải tiến thực." Điều tốt nhất là cải tiến kĩ năng của người phát triển để tuân theo qui trình được xác định tốt cho mọi dự án. Vấn đề số một với phát triển phần mềm hiện thời là người phát triển vội vàng lao vào mã và không dành đủ thời gian trong pha yêu cầu và thiết kế. Vấn đề khác là nhiều người quản lí dự án không được đào tạo tốt. Nhiều người được đề bạt tới chức vụ đó mà không có đào tạo chính thức. Theo ý kiến tôi, đầu tư tốt nhất một công ti có thể làm là đầu tư vào nhân viên riêng của họ bằng việc có đào tạo thêm trên cơ sở đều đặn. Ngày nay, phần lớn các công ti phần mềm đang cạnh tranh trong hai khu vực. Một khu vực về kinh doanh (phát triển sản phẩm và dịch vụ) và một khu vực về tài năng được cần để thực hiện những việc đó. Thành công của công ti trong kinh doanh được xác định bởi thành công của nó trong khu vực tài năng. Có nhân viên có kĩ năng nhất sẽ đem lại sản phẩm và dịch vụ chất lượng cao hơn. Thay vì hội tụ vào cái gì đó "trừu tượng" như trả tiền cho nhà tư vấn để cho bạn "chứng chỉ" nói rằng công ti của bạn là "rất tốt,"

bạn có thể hội tụ vào cái gì đó "đơn giản" như nhân viên riêng của bạn. Bởi vì có thiếu hụt trầm trọng người có kĩ năng trên khắp thế giới đồng thời công nghệ đang thay đổi nhanh chóng, bạn phải hội tụ vào phát triển nhân viên riêng của bạn. Vì tri thức được cần để dựng sản phẩm và cung cấp dịch vụ tăng lên cùng công nghệ mới, việc thu nhận và giữ nhân viên có kinh nghiệm trở thành mấu chốt để cải tiến thị phần và lợi nhuận. Người chủ công ti giỏi phải biết rằng năng lực của họ để cạnh tranh trong công nghiệp có liên quan trực tiếp tới năng lực của họ để hấp dẫn, phát triển, thúc đẩy, tổ chức và duy trì người có kĩ năng cao.

Có nhân viên có kĩ năng là quan trọng hơn chỉ là thuê và giữ họ. Bạn phải phát triển họ bằng việc đảm bảo rằng họ có đào tạo tốt nhất để xây dựng kĩ năng mới mà thị trường cần. Ngày nay phần lớn các đại học đều có chương trình đào tạo tương tự cho nên phần lớn các sinh viên tốt nghiệp có kĩ năng tương tự. Điều tuỳ thuộc ở đào tạo của công ti là tạo ra khác biệt. Chương trình đào tạo tốt có thể phát triển sinh viên mới tốt nghiệp trung bình thành nhân viên có kĩ năng cao. Trong thị trường cạnh tranh cao này, nhiều công ti đang xô vào làm nhiều điều khác nhau, nhảy từ ý tưởng "trừu tượng" này sang ý tưởng "phức tạp" khác mà không biết rằng cải tiến thực là cái gì đó đơn giản và theo nghĩa thông thường.

Cải tiến qui trình-1

Năm ngoái, tôi đã tiến hành một khoá đào tạo cải tiến qui trình ở Bắc Kinh cho vài người chủ công ti phần mềm. Sau đây là tóm tắt về khoá đào tạo đó:

Phần lớn những người phát triển không muốn thay đổi cách họ làm việc. Một khi họ học được cái gì đó, họ bám lấy nó. Nếu họ được đào tạo về lập trình, họ bám lấy lập trình. Đó là lí do tại sao nhiều người phát triển thường bắt đầu bằng viết mã và bỏ qua yêu cầu và thiết kế. Cho dù họ biết về vòng đời phát triển phần mềm nhưng họ vẫn nhảy vào viết mã vì phần lớn họ được dạy suốt ba năm lập trình trong đại học nhưng chỉ vài tháng về vòng đời phát triển. Do đó lập trình trở thành thói quen và kĩ năng của họ.

Phần lớn người quản lí không muốn thay đổi cách họ làm việc. Một khi họ học cái gì đó, họ bám lấy nó. Nếu họ được đào tạo trong quản lí, họ bám lấy quản lí. Đó là lí do tại sao những người quản lí bao giờ cũng thường bắt đầu bằng lịch biểu và bỏ quên qui trình, ước lượng, chất lượng và nỗ lực. Cho dù họ biết về vòng đời phát triển phần mềm nhưng họ vẫn hội tụ vào lịch biểu bởi vì phần lớn họ được dạy trong chương trình MBA rằng đáp ứng lịch biểu là quan trọng. Trường quản lí kinh doanh chỉ dạy về tài chính, tiếp thị nhưng hiếm khi nhắc tới chất lượng. Do đó chuyển giao sản phẩm theo lịch biểu để được trả tiền trở thành thói quen và kĩ năng của họ.

Thay đổi thói quen của những người "viết mã trước, hỏi câu hỏi sau" thành ai đó tuân theo qui trình là rất khó. Thay đổi thói quen của những người ưa thích "đáp ứng lịch biểu trước, hội tụ vào chất lượng khi bạn có thời gian" thành ai đó biết cách lập kế hoạch, cách ước lượng, cách thương lượng, tuân theo qui trình, là rất khó. Đó là lí do tại sao cải tiến phần mềm hiếm khi xảy ra trong công ti phần mềm.

Thách thức lớn nhất cho bất kì cải tiến qui trình nào là vượt qua những thói quen xấu đó. Không chỉ người quản lí phải quyết tâm làm cho cải tiến xảy ra mà họ phải sẵn lòng thay đổi cách nghĩ riêng của họ và học về qui trình cải tiến. Nếu người quản lí không thay đổi trước thì không cái gì sẽ thay đổi. Đây là lí do tại sao nhiều công ti thế đã thất bại vì người quản lí tin người phát triển phải thay đổi nhưng họ không phải làm gì cả. Họ sẵn lòng trả tiền cho nhà tư vấn để dạy cách làm tài liệu cho nhiều qui trình và hi vọng rằng cải tiến sẽ xảy ra vì họ không phải làm gì mấy bên cạnh việc ra lệnh cho người phát triển làm nhiều việc hơn.

Không có cấp quản lì nhìn vào trong và thay đổi cách nhìn của họ, cải tiến sẽ KHÔNG xảy ra. Dễ thấy cách người khác phải thay đổi, không dễ thế để cho bản thân họ thay đổi là vấn đề lớn nhất trong hầu hết các công ti phần mềm. Trong nhiều năm, tôi đã quan sát nhiều nỗ lực cải tiến phần mềm thất bại vì những người quản lí không hiểu việc "tự thay đổi" cơ bản này. Làm sao người phát triển có thể cải tiến được khi toàn thể qui trình phát triển dựa trên “lịch biểu hi vọng” do người quản lí ra lệnh. Làm sao cải tiến có thể xảy ra khi thái độ chung là: "Để cho dự án xong đã, đáp ứng lịch biểu trước rồi làm cải tiến sau."

Mặc dầu cải tiến phần mềm có thể là quan trọng, nó là không cấp bách, không phải là ưu tiên cao nhất, cho nên nó thường bị bỏ qua. Nhiều lần, công ti chỉ muốn có chứng chỉ mức CMMI. Họ trả tiền cho tư vấn tới, cung cấp các khoá đào tạo, tiến hành đánh giá, rồi cấp cho một mảnh giấy như “Được chứng nhận CMMI mức 3.” Sau đó mọi sự sẽ trở lại bình thường như không

cái gì xảy ra. Khi mà người chủ công ti còn tin rằng đó là tất cả mọi điều họ cần thì cải tiến chỉ là "trò chơi tốn tiền" chẳng có kết quả gì.

Cải tiến thực phải tới với kết quả đo được. Nếu chất lượng là quan trọng, kết quả phải được đo theo chất lượng. Nếu lịch biểu là quan trọng thì kết quả phải là ở chỗ mọi dự án phải đáp ứng lịch biểu. Nếu lợi nhuận là quan trọng thì kết quả phải được đo bằng lợi nhuận tốt hơn cho người chủ. Cải tiến là về thay đổi cách toàn thể công ti tiến hành kinh doanh để đạt tới mục đích nào đó mà người chủ công ti ước ao. Nếu người chủ ước muốn có mảnh giấy chứng nhận rằng công ti được đánh giá ở "CMMI mức 3" thì công ti sẽ nhận được nó sau khi trả nhiều tiền cho "nhà tư vấn vô đạo đức." Không may, tình huống xấu này đã xảy ra trên khắp thế giới.

Năm ngoái, khi tôi gặp một số người chủ công ti phần mềm, họ bảo tôi rằng họ đã đọc cuốn sách của tôi về cải tiến phần mềm (nhiều sách của tôi đã được dịch sang tiếng Trung Quốc) và đồng ý với cách nhìn của tôi về cải tiến thực. Nhiều người bảo tôi rằng họ đã không trả tiền cho nhà tư vấn để cho họ "chứng chỉ" với hi vọng rằng họ sẽ nhận được nhiều kinh doanh hơn từ Mĩ và châu Âu. Cuối cùng nhiều công ti phương tây đã tới công ti của họ để tìm hiểu cơ hội kinh doanh. Tuy nhiên, tất cả các công ti này đều tiến hành kiểm điểm riêng của mình để thẩm tra "kết quả đã được chứng nhận." Sau đó họ không bao giờ nghe nói gì về các công ti đó. Về sau, họ tìm ra rằng những công ti này đã quyết định làm kinh doanh với các công ti khác ở nước khác.

Khi những người chủ này thấy rằng tôi đã dạy ở Đại học Thanh Hoa, họ yêu cầu cuộc gặp gỡ với tôi vì

tôi là một trong các tác giả của CMMI. Tôi đồng ý giúp họ bằng việc làm buổi tập huấn tại Thanh Hoa nơi họ phải tuân theo một số chỉ dẫn để học về "cải tiến qui trình thực."

Đầu tiên tôi yêu cầu họ viết ra viễn kiến về công ti của họ và điều họ muốn đạt tới trong năm tới chục năm tới. Điều này không dễ, nhiều người chủ có ý tưởng nào đó trong đầu nhưng chưa bao giờ thực tế viết chúng ra trên giấy. Tôi giải thích rằng viễn kiến là phát biểu chính xác xác định ra cái gì, tại sao và ai từ quan điểm doanh nghiệp. Nó thiết lập đích và mục tiêu doanh nghiệp có liên quan tới cách công ti vận hành. (Công ti làm kinh doanh gì? Tại sao bạn muốn cải tiến? Ai sẽ chịu trách nhiệm cho công ti? Ai là khách hàng của bạn? Công ti của bạn làm cái gì cho khách hàng? Lí do nào mà khách hàng muốn làm kinh doanh với công ti của bạn? Trạng thái môi trường vận hành sẽ là gì? Làm sao công ti của bạn được phân biệt trong thị trường toàn cầu? Mục đích của công ti là gì? Làm sao bạn đo được mục đích của bạn? V.v.).

Nhiều người ngạc nhiên rằng điều đó chẳng liên quan gì tới CMMI hay bất kì khía cạnh kĩ thuật nào mà họ mong đợi tôi khuyên họ. Tôi bảo họ rằng không có viễn kiến và mục đích rõ ràng, mọi người sẽ bị lẫn lộn và không biết lí do tại sao họ cần cải tiến qui trình. Trong quá khứ, nhiều người phát triển được bảo phải tuân theo chỉ dẫn CMMI nào đó như làm tài liệu qui trình nhưng họ không biết tại sao. Họ đã làm điều họ được bảo và sau khi hoàn thành tài liệu qui trình dài dòng, họ cất nó lên giá để trưng bày nhưng thói quen của họ không bao giờ thay đổi. Họ tiếp tục bám lấy điều họ biết rõ nhất – viết

mã và không cái gì thay đổi trong công ti cho nên cải tiến thất bại.

Cải tiến qui trình-2

Để cải tiến, công ti phải có viễn kiến rõ ràng để trao đổi với mọi nhân viên về chiều hướng mà người chủ muốn đi. Không có điều đó, mọi người có thể bị lẫn lộn viễn kiến và có thể quyết định đi vào các hướng khác nhau. Sau khi người chủ đã hoàn chỉnh viễn kiến của họ, tôi yêu cầu họ viết ra mục đích doanh nghiệp của họ. Phần lớn mọi người viết ra về giảm lỗi, tăng lợi nhuận, làm ra nhiều tiền hơn, có nhiều khách hàng hơn, mở rộng doanh nghiệp toàn cầu v.v. Rồi tôi yêu cầu họ về điều sẽ cho phép họ đạt tới những mục đích này. Tất nhiên, câu trả lời điển hình là có sản phẩm và dịch vụ có chất lượng vì chất lượng là điều họ có trong đầu. Câu hỏi tiếp của tôi là họ cần gì để xây dựng sản phẩm chất lượng?

Sau vài thảo luận giữa các nhóm, kết luận cuối cùng là "Tri thức và kĩ năng." Tôi hỏi họ: "Họ thu nhận những điều này ở đâu? Sinh viên đại học có những kĩ năng này không? Các đại học có dạy về kĩ năng cải tiến không?" Nếu sinh viên tốt nghiệp đã có những kĩ năng này thì tại sao công ti không có sản phẩm có chất lượng và phải cải tiến? Cho nên chúng ta đi xuống lí do chính của thói quen xấu như người phát triển không tuân theo qui trình được xác định. Thói quen xấu của việc bỏ qua các pha và nhảy vào viết mã. Thói quen xấu của người quản lí thiết lập lịch biểu mà không ước lượng v.v. Tôi giải thích cho họ rằng khi dự án không có qui trình mà nó sẽ dùng, người phát triển sẽ làm bất kì cái gì họ

muốn. Đến cuối, họ sẽ dành nhiều thời gian vào việc sửa lỗi và không có thời gian để cải tiến phần mềm. Khi dự án dường như trượt lịch, mọi người đều hoảng hốt. Người phát triển sẽ hội tụ vào công việc cá nhân của riêng mình để chắc rằng nếu cái gì đó xảy ra, đó không phải là lỗi của họ. Khi các thành viên tổ không làm việc cùng nhau và rút lui khỏi tương tác với nhau, việc điều phối dự án sẽ thất bại. Khi sự việc thành tồi tệ, mọi người sẽ bỏ qua những điều như kiểm thử, tích hợp v.v. làm nảy sinh sản phẩm chất lượng thấp. Khi khách hàng phàn nàn về lỗi, công ti phải sửa chúng. Sẽ tốn kém nhiều để sửa lỗi sau khi đưa ra cho khách hàng. Hậu quả là chi phí cao hơn, thời gian dài hơn, khách hàng giận dữ, người phát triển thất vọng, và người chủ mất tiền.

Có bằng chứng rằng các công ti hội tụ vào qui trình đã đạt tới sự hài lòng của khách hàng, đáp ứng lịch biểu, chất lượng cao hơn và lợi nhuận nhiều hơn. Nếu người phát triển chỉ nhận được đào tạo kĩ thuật trong đại học thì họ sẽ cần đào tạo thêm về kỉ luật tuân theo qui trình. Họ phải hiểu ích lợi của qui trình cho nên đào tạo qui trình là quan trọng. Tuy nhiên, để thay đổi thói quen, đào tạo phải bắt đầu từ cấp quản lí trước. Nếu người quản lí thay đổi thói quen, mọi thứ sẽ thay đổi bởi vì người phát triển sẽ tuân theo chỉ đạo của họ. Tất nhiên, đổi thói quen mọi người, những người ưa thích "đáp ứng lịch biểu trước, hội tụ vào chất lượng khi họ có thời gian" thành ai đó biết cách lập kế hoạch, cách ước lượng, cách thương lượng là khó nhưng mọi đào tạo đều phải bắt đầu với cấp quản lí.

Điều này dường như là điều ngạc nhiên lớn với nhiều người chủ. Nhiều nhà tư vấn thường hội tụ đào đào

tạo người phát triển về cách làm tài liệu qui trình cho nên có nhiều câu hỏi và thảo luận về cách tiếp cận này. Tôi giải thích cho họ rằng có qui trình được làm tài liệu sẽ chỉ giúp cho công đi qua được đánh giá CMMI nhưng không thay đổi thói quen xấu của nhân viên. Vì hoạt động làm tài liệu tốn thời gian, thường vài tháng hay đôi khi vài năm, trong trường hợp đó nhà tư vấn có thể làm được nhiều tiền hơn. Để làm cải tiến thực xảy ra, cấp quản lí phải ra quyết định đúng bằng việc để bản thân họ là "tấm gương" cho những người phát triển đi theo. Nếu họ không thay đổi thói quen xấu của họ, không cái gì sẽ xảy ra và trong tình huống này, điều này sẽ làm ra khác biệt giữa thành công và thất bại.

Lí do là trong hoạt động cải tiến, cấp quản lí phải lập kỉ luật cho những người phá luật hay những người từ chối tuân theo qui trình. Điều quan trọng cho công ti là đặt ra qui tắc mới cho cải tiến qui trình và điều đó phải bắt đầu từ cấp quản lí. Tất nhiên, lập kỉ luật đúng là dễ hơn được thực hiện, và đó là vấn đề nhạy cảm. Nếu người quản lí không tuân theo qui trình, họ có thể có thời gian khó khăn khi lập kỉ luật cho ai đó làm cùng điều đó. Chẳng hạn, nếu một người phát triển bỏ qua thiết kế và nhảy vào viết mã, người quản lí phải ra quyết định về cách xử trí với tình huống này. Điều gì xảy ra nếu người quản lí cũng đặt lịch biểu dựa trên trực giác thay vì tuân theo qui trình ước lượng? Người phát triển có thể cãi rằng vì lịch biểu không được ước lượng tốt và quá ngắn, người đó không có thời gian để tuân theo vòng đời phát triển. Người đó phải bỏ qua một số pha chỉ để làm cho công việc của mình được thực hiện xong. Trong nhiều công ti phần mềm, người phát triển thường nghĩ rằng họ có thể phá luật và làm bất kì điều gì người đó muốn làm

chừng nào họ vẫn làm cho công việc của họ được hoàn thành. Đây là thói quen xấu phải được thay đổi nếu công ti muốn cải tiến. Nếu người quản lí không tuân theo qui trình, những thành viên tổ khác sẽ chú ý tới điều đó, và họ sẽ trở nên không bằng lòng. Nếu họ cảm thấy rằng người quản lí không thay đổi, họ sẽ nghĩ họ cũng có thể làm cùng điều đó. Điều này có thể làm yếu đi năng lực của công ti để cải tiến.

Để bắt đầu cải tiến qui trình, người chủ phải đặt chiều hướng rõ ràng. Không ai được ở trên qui tắc, ngay cả người quản lí cũng không được. Nếu họ làm sai, họ phải chịu cùng kỉ luật được giáng cho người phát triển. Điều này tạo ra môi trường công bằng nơi mọi người cảm thấy bình đẳng. Tất nhiên, cách người chủ lập kỉ luật cho nhân viên là quan trọng. Họ không muốn quá khắt khe nếu điều đó là không cần thiết. Nếu họ làm điều này, nhân viên sẽ sợ họ. Cải tiến không nên là cái gì đó mà họ sợ nhưng là cái gì đó họ cần.

Theo ý kiến riêng của tôi, điều quan trọng nhất trong cải tiến qui trình là khuyến khích hay thưởng để đạt tới mục đích. Với khuyến khích đủ, mọi nhân viên sẽ làm việc chăm chỉ để làm cho sự việc xảy ra. Họ sẽ dành thời gian của họ, nỗ lực của họ để giúp người quản lí thành công. Đây là chỗ tôi quay lại đặt mục đích doanh nghiệp về cải tiến. Mục đích kém nhất là đạt tới mức CMMI vì nó không ngụ ý gì cho bất kì ai. Mục đích thực phải là cái gì đó đo được và "thực" như giảm lỗi 20% mỗi năm, cải tiến số các dự án đáp ứng lịch biểu lên 50% trong hai năm v.v. Cái gì đó đơn giản mà mọi người có thể hiểu được. Chẳng hạn, hôm nay lỗi trung bình là 20 lỗi trên một nghìn dòng mã lệnh và nếu tỉ lệ lỗi trung

bình giảm đi trong năm nay đến 20%, mọi nhân viên có thể có điểm thưởng vào cuối năm. Điểm thưởng có thể là bằng tiền hay vài ngày nghỉ phụ thêm, cái gì đó nhân viên muốn. Với khuyến khích đúng, mọi sự sẽ thay đổi. Sẽ có sức ép trong các nhân viên để giảm số lỗi, người phát triển sẽ giám sát thành viên tổ để chắc rằng mọi người sẽ tuân theo qui trình và kiểm thử mã của họ một cách cẩn thận. Nếu người quản lí yêu cầu kiểm điểm nhiều hơn để nhận diện và sửa lỗi, người phát triển sẽ kiểm điểm cẩn thận điều họ làm bởi vì không muốn là "kẻ làm hỏng" người ngăn cản mọi người được nhận khuyến khích. Đây là chỗ thay đổi thói quen xấu xảy ra vì người phát triển muốn quan sát lẫn nhau và nhắc nhở họ về việc tuân theo qui trình.

Với người chủ người thực sự muốn cải tiến doanh nghiệp của họ, họ cần cân nhắc cải tiến như cái gì đó có ưu tiên cao bởi vì đó là quyết định doanh nghiệp chứ không phải quyết định kĩ thuật. Để cải tiến, mọi người trong công ti phải đảm nhiệm để hoàn thành nhiệm vụ của họ. Người chủ công ti phải mạnh trong các thông điệp rằng nhiệm vụ cải tiến là quan trọng và phải không được để trễ. Chỉ với tầm quan trọng và chỉ đạo khẩn thiết từ cấp cao nhất của công ti và với khuyến khích đúng, mọi người sẽ làm việc cần cù để thay đổi thói quen xấu của họ. Khi những điều này xảy ra, cải tiến sẽ xuất hiện. Để làm cải tiến thực xảy ra, người chủ và người quản lí phải thừa nhận rằng nhiệm vụ đầu tiên của họ là thay đổi bản thân họ khỏi thói quen xấu. Rồi điều đó có thể đưa tới thay đổi lớn lao. Bằng không việc cải tiến phần mềm sẽ không được thực hiện.