Quản lý dự án phần mềm trong thực tiễn (Software Project Management in Practice)

Các quy trình được dùng ở Infosys, một công ty đạt CMM mức 5, để quản lý các dự án phần mềm

Tác giả: Pankaj Jalote

Nhà xuất bản: Addison-Wesley

Năm xuất bản: 2002

Biên dịch sang tiếng Việt: Nguyễn Công Danh

Sửa bản dịch: Trần Cao Đệ

Bộ môn: Công nghệ phần mềm

Khoa CNTT&TT, ĐH Cần Thơ

Năm biên dịch: 2013

(Dùng nội bộ làm tài liệu tham khảo bổ sung cho sinh viên)

Mục lục

Chương 1: Quản lý dự án phần mềm 2

Phần 1: Lập kế hoạch cho dự án

Chương 2: Cơ sở hạ tầng để lập kế hoạch cho dự án 25

Chương 3: Lập kế hoạch cho quy trình 44

Chương 4: Ước lượng nỗ lực và thời gian biểu 62

Chương 5: Lập kế hoạch cho chất lượng 98

Chương 6: Quản lý rủi ro 119

Chương 7: Đo lường và kế hoạch theo dõi 143

Chương 8: Kế hoạch quản lý dự án 164

Chương 9: Quản lý cấu hình 200

Phần 2: Thực hiện và kết thúc dự án

Chương 10: Xem xét lại 227

Chương 11. Giám sát và kiểm soát dự án 257

Chương 12. Kết thúc dự án 305

Chương 1. Quản lý dự án phần mềm

Trên thế giới, khoảng nửa triệu người quản lý dự án thực hiện khoảng một triệu dự án phần mềm mỗi năm, sản xuất phần mềm trị giá 600 tỷ USD. Nhiều dự án trong số này có chất lượng không như kỳ vọng của khách hàng hoặc không cung cấp các phần mềm trong phạm vi ngân sách và thời gian hoàn thành. Một phân tích cho thấy khoảng một phần ba các dự án có chi phí và thời gian hoàn thành (cost and schedule) vượt hơn 125% [1].

Tại sao quá nhiều dự án phần mềm thất bại? Mặc dù có rất nhiều lý do, một trong những lý do quan trọng nhất là quản lý dự án không phù hợp. Ví dụ, các lý do chính làm cho dự án chệch ra khỏi tầm kiểm soát là mục tiêu không rõ ràng, lập kế hoạch tồi, công nghệ mới, thiếu một phương pháp quản lý dự án, và không đủ nhân sự [2]. Ít nhất ba trong năm lý do này rõ ràng liên quan đến quản lý dự án. Hai lý do còn lại - không đủ nhân sự và công nghệ mới - có thể được coi như những rủi ro mà để quản lý chúng cũng là một phần của quản lý dự án.

Rõ ràng, bằng cách sử dụng các kỹ thuật quản lý dự án có hiệu quả, một người quản lý dự án có thể cải thiện các cơ hội để thành công. Tuy nhiên, hiệu quả của những kỹ thuật này là gì?

Hãy xem xét một tương tự. Giả sử bạn muốn phát triển cơ thể, với cơ bắp săn chắc. Để đạt được mục tiêu, bạn bắt đầu nhìn vào chương trình tập thể dục được mô tả trong tạp chí. Một bài viết mô tả làm thế nào để phát triển sức mạnh cánh tay, đưa ra một danh sách gồm 10 bài tập – không quá nhiều nếu so sánh với bất kỳ chuẩn nào. Nhưng sau đó, một bài viết khác hướng dẫn để phát triển sức mạnh đùi, cũng cung cấp 10 bài tập, và người huấn luyện cho bụng phẳng cũng cảm thấy rằng làm 10 bài tập là không quá nhiều. Nếu bạn muốn phát triển cơ thể tổng thể của bạn bằng cách làm theo của các chương trình tập thể dục riêng lẻ, bạn sẽ thấy rằng bạn có một bộ 50 đến 100 bài tập để làm - rõ ràng là không thể đối với hầu hết mọi người, hãy để một mình một người quản lý dự án bận rộn. Để đạt được mục tiêu của bạn, bạn cần một chương trình đào tạo toàn diện, thiết thực và hiệu quả.

Tương tự như vậy, bạn sẽ tìm thấy một sự phong phú về các đề xuất để thực hiện các khía cạnh khác nhau của quản lý dự án, bao gồm ước lượng/dự toán nỗ lực (effort estimation), quản lý rủi ro, giám sát dự án, quản lý cấu hình, v.v. Mặc dù mỗi kỹ thuật

được đề xuất để giải quyết vấn đề riêng biệt, nhưng vẫn có sự chưa rõ ràng trong việc làm thế nào để kết hợp những kỹ thuật này lại với nhau thành một quy trình khả thi trong thực tế. Để quản lý dự án hiệu quả, cần phải trải qua thời gian thực tế quản lý, thực hiện các "chương trình tập luyện" được quản lý để tạo ra kết quả. Nói cách khác, những gì cần thiết là một quy trình cân bằng bao gồm việc quản lý toàn bộ dự án từ khi bắt đầu đến khi hoàn thành. Thật không may, có một số lượng ít các phương pháp đã được công bố chỉ dẫn làm thế nào để tích hợp các kỹ thuật lại với nhau theo cách này.

Cuốn sách này sẽ lấp đầy khoảng trống này bằng cách mô tả các quy trình đã được sử dụng trong một công ty tầm cỡ thế giới để quản lý hiệu quả các dự án phần mềm. Công ty này là Infosys, một công ty phát triển phần mềm có một thành tích tuyệt vời trong việc thực hiện dự án; chỉ tính riêng trong năm 2000, những người quản lý dự án ở Infosys đã sử dụng các quy trình được mô tả ở đây để thực hiện thành công khoảng 500 dự án cho khách hàng. Cuốn sách này thảo luận về tất cả các khía cạnh khác nhau của quản lý dự án ở Infosys - lập kế hoạch, thực hiện, và kết thúc. Bạn sẽ học làm thế nào những người quản lý dự án ở Infosys ước lượng/dự toán, lập kế hoạch quản lý rủi ro, thu thập các số đo (metrics), thiết lập các mục tiêu chất lượng, sử dụng các phép đo để giám sát một dự án, v.v. Một khía cạnh thú vị của các quy trình này, sẽ hấp dẫn cho những người quản lý dự án bận rộn, là chúng không phức tạp, cũng không cồng kềnh, sử dụng các số đo đơn giản.

Infosys đã được đánh giá ở mức 5 (mức cao nhất) của Mô Hình Trưởng Thành Năng Lực (CMM - Capability Maturity Model). Bằng cách trích ra các quy trình quản lý dự án từ các tập hợp các quy trình ở Infosys, cuốn sách này cũng minh họa làm thế nào các dự án được quản lý trong một công ty trưởng thành mức cao. Thông qua minh hoạ này, tôi hy vọng sẽ giới thiệu những lợi ích của CMM cho những người quản lý dự án - những người đã không nghiên cứu nó vì thiếu thời gian, hoặc bởi vì họ đã nhận thấy khó khăn khi dùng CMM vào thực hành quản lý dự án.

Chương này giới thiệu hai chủ đề tạo nên bối cảnh cho cuốn sách: CMM và Infosys. Bởi vì trọng tâm của cuốn sách là quản lý dự án và không phải là CMM, các cuộc thảo luận trong sách tập trung vào các khía cạnh quản lý dự án của CMM. Chương này cũng cung cấp một cái nhìn tổng quan về quy trình quản lý dự án và trường hợp nghiên cứu (case study) chính; các chi tiết về những vấn đề này sẽ được thảo luận trong phần còn lại của cuốn

sách. Trước tiên, chúng ta hãy thảo luận ngắn gọn về vai trò của quy trình trong quản lý dự án.

1.1 QUY TRÌNH VÀ QUẢN LÝ DỰ ÁN

Một dự án phần mềm có hai nhóm hoạt động chính: phát triển và quản lý dự án. Nhóm dự án quan tâm đến các vấn đề về thiết kế, kiểm thử, cài đặt mã, v.v. Nhóm quản lý dự án quan tâm đến hoạch định và quản lý các hoạt động để đạt được mục tiêu của dự án: chi phí, thời gian hoàn thành, chất lượng.

Đối với dự án nhỏ: một nhóm một hoặc hai người làm thì có thể dùng các kỹ thuật không hình thức. Kế hoạch dự án có thể là một e-mail xác định ngày giao hàng và có lẽ một vài cột mốc (milestones) bên trong khoảng thời gian đó. Yêu cầu (requirements) có thể được truyền đạt trong một bức thư ngắn hoặc thậm chí bằng lời nói, và các sản phẩm trung gian, chẳng hạn như tài liệu thiết kế, có thể là mảnh giấy ghi vội vàng được lưu trong tập giấy cá nhân.

Tuy nhiên, những kỹ thuật không hình thức này không dùng được cho các dự án lớn: nhiều người làm việc trong nhiều tháng, như hầu hết các dự án phần mềm thương mại. Trong những dự án như thế, việc phát triển phải được làm cẩn thận theo các phương pháp đã được thử nghiệm, và kết quả của các công việc này phải được lập tài liệu rõ ràng để những người khác có thể xem xét lại (review) chúng. Các công việc trong dự án phải được hoạch định và phân công cho các thành viên và sau đó được theo dõi khi dự án bắt đầu. Nói một cách khác, để dự án lớn thành công thì phải gia tăng tính hình thức và cứng nhắc trong quản lý và thực hiện.

Những kỹ thuật theo hình thức cần các quy trình được định nghĩa rõ ràng để thực hiện các công việc khác nhau để mà kết quả trở nên phụ thuộc nhiều hơn vào khả năng của các quy trình. Những kỹ thuật hình thức tiếp tục được mở rộng hơn nữa nếu các phương pháp định lượng được sử dụng trong các quy trình thông qua việc sử dụng các số đo phù hợp.

Quy trình là gì? Về mặt kỹ thuật, một quy trình cho một công việc bao gồm một chuỗi các bước cần được theo để thực hiện công việc đó. Tuy nhiên, đối với một công ty, các quy trình được khuyến cáo để được sử dụng bởi các kỹ sư và những người quản lý dự án nhiều hơn một chuỗi các bước; chúng đóng gói những gì các kỹ sư và những người quản

lý dự án đã học được từ các dự án đã được thực hiện thành công. Thông qua các quy trình, lợi ích của kinh nghiệm được trao cho tất cả mọi người, kể cả người mới gia nhập công ty. Những quy trình này giúp những người quản lý và các kỹ sư đạt được thành công và tránh những vấn đề dẫn đến thất bại.

Đối với một dự án, các quy trình xác định làm thế nào để thực hiện các hoạt động phát triển như đặc tả yêu cầu, thiết kế, thử nghiệm, v.v. Các quy trình quản lý dự án, mặt khác, xác định làm thế nào để thiết lập các cột mốc (milestones), tổ chức nhân sự, quản lý rủi ro, giám sát tiến độ, v.v. Cuốn sách này tập trung vào quy trình quản lý dự án.

Tuy nhiên, khi bạn xem xét các quy trình quản lý dự án, bạn phải đặt câu hỏi: liệu người quản lý dự án có muốn sử dụng chúng không? Tôi thường nghe những người thiết kế quy trình than phiền rằng những người quản lý dự án không thực hiện theo các quy trình và rằng họ không thích những thay đổi. Kinh nghiệm của tôi với những người quản lý dự án tại Infosys và các công ty khác là họ thực sự muốn sử dụng các quy trình nhưng chỉ khi chúng hợp lý và có thể giúp những người quản lý dự án thực hiện các dự án của họ tốt hơn. Tuy nhiên, nhiều người quản lý dự án cảm thấy phẫn nộ khi nghĩ các quy trình dường như không cần thiết và đóng góp ít giá trị cho công việc của họ. Trong trường hợp này, cần phải có những quy trình trọng lượng nhẹ (lightweight processes) – có thể giúp những người quản lý dự án lập kế hoạch và kiểm soát các dự án của họ tốt hơn và cung cấp cho họ sự linh hoạt để xử lý các tình huống khác nhau.

Để trả lời cho câu hỏi “Tại sao người quản lý dự án cần theo quy trình?”, S.D. Shibulal – người thành lập và giám đốc Infosys cho rằng:

- Quy trình mô tả kiến thức chung. Sử dụng chúng làm tăng cơ hội thành công của bạn.

- Một quy trình có thể có các bước dư thừa, nhưng bạn sẽ không phải luôn luôn biết trước cái nào là không cần thiết, và do đó bạn sẽ có nhiều rủi ro nếu chọn một cách nhanh chóng.

- Nếu không có quy trình, bạn không thể dự đoán về kết quả của dự án của bạn.

- Bạn và công ty không thể học một cách hiệu quả mà không có quy trình được được định nghĩa. Học tập và cải tiến có tính bắt buộc trong thế giới dựa trên tri thức ngày nay.

- Quy trình làm giảm mức độ lo lắng của bạn. Các bản liệt kê những mục cần kiểm tra (checklist) chắc chắn bao gồm 80% những gì cần phải được thực hiện. Do đó, nhiệm vụ của bạn là làm giảm rủi ro trong 20% còn lại.

1.2 QUẢN LÝ DỰ ÁN VÀ CMM

Khi đã chấp nhận sử dụng các quy trình hiệu quả có thể giúp thực hiện dự án thành công, một câu hỏi là: Những đặc trưng được được mong muốn của các quy trình này là gì? CMM cho phần mềm là một khung làm việc (framework) để cố gắng trả lời câu hỏi này.

CMM cho phần mềm là một khung làm việc được phát triển bởi Viện Công Nghệ Phần Mềm (SEI) tại Đại học Carnegie Mellon bằng cách quan sát các thực hành tốt nhất trong phần mềm và các công ty khác. Do đó, CMM phản ánh kinh nghiệm về quy trình chung và mong đợi của nhiều công ty. Nó xác định các các đặc điểm được mong muốn của các quy trình mà không mô tả các quy trình cụ thể. Vì thế, các quy tình khác nhau có thể thực hiện đầy đủ các yêu cầu của CMM. Nó có thể được sử dụng để đánh giá quy trình phần mềm của một công ty và xác định thiếu sót.

CMM là một trong những khung làm việc phổ biến nhất để cải tiến quy trình phần mềm (các khung làm việc khác cũng thường được sử dụng là ISO 9001 [3][4][5]). Các cơ sở của CMM đã được quy định trong *Managing the Software Process* [6] của Watts Humphrey, và khung làm việc này được mô tả đầy đủ trong *The Capability Maturity Model: Guidelines for Improving the Software Process* của Viện Công Nghệ Phần Mềm [7]. Một "phiên bản mới" của CMM, được gọi là CMM-I, đã được phát hành. Nhưng bởi vì trọng tâm của cuốn sách này không phải là về các mô hình và bởi vì vẫn còn ít kinh nghiệm về CMM-I, tôi chỉ thảo luận CMM cho phần mềm và các khía cạnh quản lý dự án, mặc dù CMM này cũng bao gồm các vấn đề công ty và quản lý quy trình. Tôi không thảo luận về các thủ tục đánh giá, mô tả ngắn gọn về những vấn đề này được đưa ra trong cuốn sách *CMM in Practice [8]*, và mô tả chi tiết được đưa ra trong *CMM Based Appraisal for Internal Process Improvement* của S. Masters [9].

1.2.1 Tổng quan về CMM

Một mục tiêu của CMM là để phân biệt các quy trình trưởng thành với các quy trình chưa trưởng thành, hoặc không theo thể thức (ad hoc). Quy trình phần mềm chưa trưởng thành

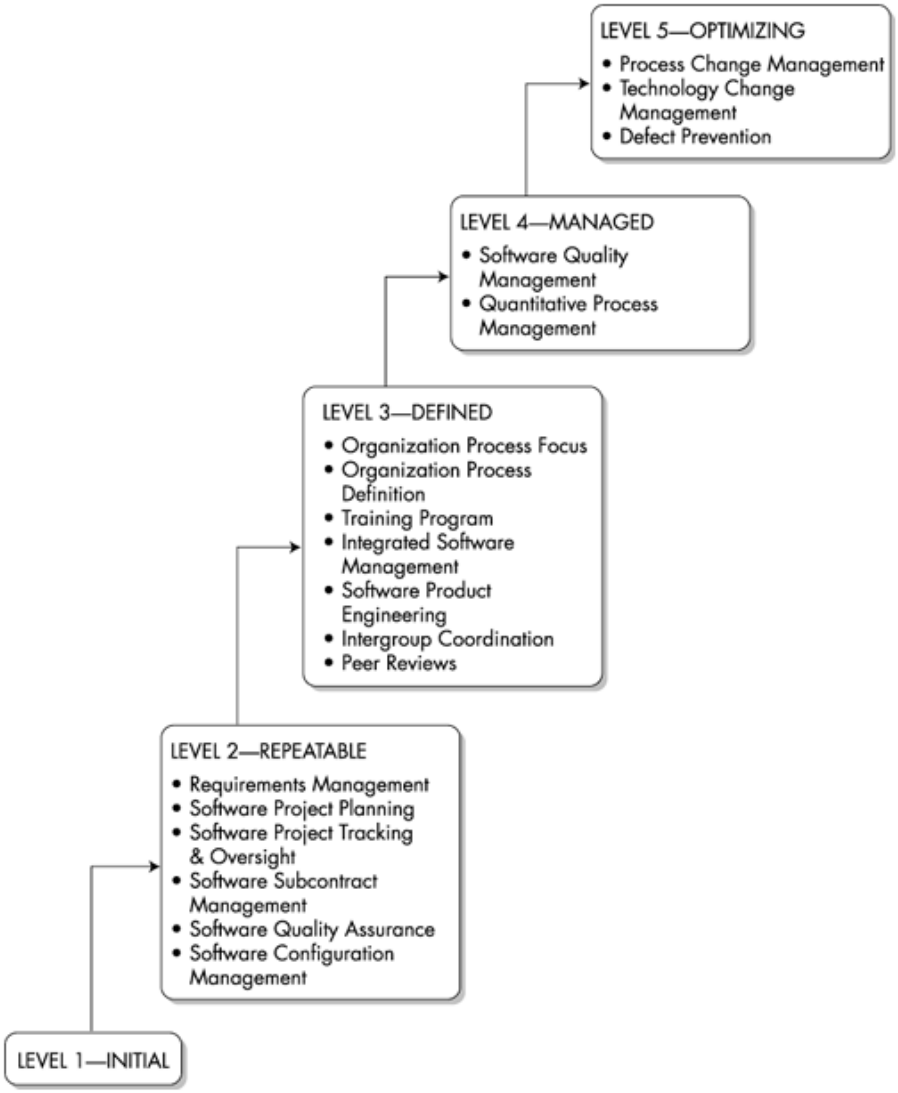
ngụ ý rằng các dự án được thực hiện mà không có nhiều hướng dẫn, và kết quả của một dự án phụ thuộc phần lớn vào khả năng của nhóm dự án và lãnh đạo dự án. Ngược lại, với quy trình trưởng thành, một dự án được thực hiện bằng cách theo các quy trình được định nghĩa. Trong trường hợp này, kết quả của dự án là ít phụ thuộc vào con người và phụ thuộc nhiều hơn vào quy trình. Khi đó, các quy trình càng trưởng thành, kết quả của dự án càng được dự đoán và được kiểm soát tốt hơn.

Phạm vi của các kết quả có thể được dự đoán trong một dự án khi nó được thực hiện bằng cách sử dụng một quy trình là *khả năng quy trình (process capability)* của nó. Kết quả thực tế đạt được trong một dự án được thực hiện bằng cách sử dụng quy trình là *hiệu suất quy trình (process performance)* của nó. Rõ ràng, hiệu suất quy trình phụ thuộc vào khả năng quy trình. Để luôn cải thiện hiệu suất quy trình trong các dự án, bạn phải tăng cường khả năng quy trình, quy trình tự đó càng phải trưởng thành hơn.

Con đường dẫn đến mức trưởng thành cao hơn bao gồm một số trạng thái ổn định đã được xác định rõ ràng được gọi là các *mức trưởng thành (maturity levels)* của CMM. Mỗi mức trưởng thành quy định cụ thể một số đặc điểm cho quy trình, mức trưởng thành cao hơn có đặc điểm cao cấp hơn được tìm thấy trong quy trình phần mềm trưởng thành hơn. Do đó, khung làm việc CMM mô tả các yếu tố chính của quy trình phần mềm ở các mức khác nhau của sự trưởng thành. Bởi vậy, nó cũng xác định con đường mà một quy trình phần mềm đi theo để chuyển đổi từ các quy trình chưa trưởng thành đến các quy trình trưởng thành cao hơn. Con đường này bao gồm năm mức trưởng thành, như thể hiện trong Hình 1.1 [7].

Ở mức 1, mức *khởi tạo (initial level),* dự án được thực hiện theo cách thức để mà nhóm dự án và người quản lý dự án thấy phù hợp. Mức *lặp lại (repeatable level)* (mức 2) áp dụng khi thực hành quản lý dự án được sử dụng, mặc dù quy trình chưa áp dụng cho toàn bộ công ty. Ở mức *định nghĩa (defined level)* (mức 3), quy trình cho toàn bộ công ty đã được định nghĩa và thường xuyên được thực hiện theo. Ở mức *quản lý (managed level)* (mức 4), định lượng sự hiểu biết về khả năng quy trình làm cho nó có thể dự đoán theo định lượng và kiểm soát hiệu suất của quy trình thực hiện dự án. Ở mức *tối ưu hóa (optimizing level)* (mức 5), khả năng quy trình được cải tiến một cách có kiểm soát và cải tiến này là dựa trên định lượng.

Hình 1.1.7 Các mức trưởng thành của CMM



Mỗi mức trưởng thành (trừ mức 1) được đặc trưng bởi *các lĩnh vực quy trình trọng điểm (KPAs – key process areas)*, xác định các lĩnh vực mà tại đó các công ty nên tập trung để nâng cao quy trình của họ đến mức trưởng thành đó. Hình 1.1 cũng cho thấy KPAs cho các mức khác nhau. Đối với một công ty để đạt được một mức trưởng thành, nó phải đáp ứng tất cả các KPAs ở mức trưởng thành đó cũng như các KPAs tại tất cả các mức trưởng thành thấp hơn.

Duy trì các quy trình ở các mức cao của sự trưởng thành là một nhiệm vụ đầy thách thức đòi hỏi phải cam kết từ công ty và văn hóa làm việc thích hợp. Trong 900 đánh giá được tiến hành từ năm 1996 đến tháng 6 năm 2000, kết quả đánh giá đã được cung cấp cho

SEI, chỉ 3% của các công ty ở mức 5, và 5% ở mức 4 [10]. Phần còn lại ở mức 3 hoặc thấp hơn, với 38% ở mức 2 và 18% ở mức 3.

1.2.2 KPAs cho quản lý dự án

Mỗi KPA quy định cụ thể các mục tiêu mà các quy trình của công ty phải đáp ứng để thỏa mãn KPA đó. Ngoài ra, mỗi KPA xác định một nhóm các hoạt động, được gọi là các *thực hành chủ chốt* (*key practices*), để đáp ứng các mục tiêu của KPA đó. Các mục tiêu cho mỗi KPA là nắm bắt bản chất của nó. Chúng xác định các mục tiêu mà CMM đã đặt ra cho các quy trình liên quan đến KPA. Để minh họa KPAs kết hợp với quản lý dự án như thế nào, chúng tôi thảo luận ngắn gọn về các mục tiêu của những KPAs này. Những mục tiêu này được lấy từ CMM [7], với một số thay đổi nhỏ trong cách diễn đạt của một số mục tiêu.

Bảng 1.1 liệt kê tất cả các mục tiêu cho KPAs ở mức 2, cho thấy rõ ràng rằng trọng tâm mức 2 là gần như dành riêng cho quản lý dự án. Theo các mục tiêu này, bạn tạo một tài liệu kế hoạch dự án, đánh giá hiệu quả dự án đang diễn ra so với kế hoạch và hành động khi hiệu suất thực tế sai lệch đáng kể so với kế hoạch. Các yêu cầu (requirements) được lập tài liệu một cách đúng đắn, và các thay đổi yêu cầu đều được quản lý. Tất cả các sản phẩm làm ra được kiểm soát, và thay đổi các thành phần đều được quản lý thông qua một kế hoạch quản lý cấu hình. Các xem xét lại (reviews) và kiểm tra (audits) được thực hiện để đảm bảo rằng các quy trình, kế hoạch và các tiêu chuẩn đang được tuân thủ. Nếu một số phần của dự án là hợp đồng phụ với các nhà cung cấp khác thì việc gia công hợp đồng phụ này cũng được theo dõi đúng cách.

Bảng 1.2 trình bày chi tiết các mục tiêu của ba trong bảy KPAs ở mức 3. Các KPAs khác tập trung vào các vấn đề tổ chức và quản lý quy trình. Một dự án trong một công ty mức 3 sử dụng một phiên bản được điều chỉnh phù hợp của quy trình chuẩn và tái sử dụng các thành phần (asset), dữ liệu, và kinh nghiệm từ các dự án trong quá khứ cho việc lập kế hoạch. Các nhóm khác nhau đóng góp vào dự án hợp tác một cách suôn sẻ thông qua giao diện và cơ chế được xác định rõ ràng. Xem xét lại (review) được thực hiện để xác định các lỗi trong các sản phẩm được tạo ra, và hỗ trợ đầy đủ để thực hiện việc xem xét lại (review) và theo dõi các hoạt động được cung cấp.

Bảng 1.1. Các mục tiêu cho KPAs ở mức 2 (Repeatable)

KPA (lĩnh vực quy trình trọng điểm)	**Các mục tiêu (Goals)**
Quản lý yêu cầu *(RM - Requirements Management)*	- Yêu cầu phần mềm được kiểm soát để thiết lập một baseline cho công nghệ phần mềm và các hoạt động quản lý. - Các kế hoạch, sản phẩm, và các hoạt động được lưu giữ nhất quán với yêu cầu.
Hoạch định dự án phần mềm *(SPP - Software Project Planning)*	- Các ước lượng được lập tài liệu để sử dụng trong việc lập kế hoạch và theo dõi dự án. - Các hoạt động của dự án và các cam kết đã được lên kế hoạch và được lập tài liệu. - Nhóm bị ảnh hưởng và cá nhân đồng ý các cam kết của họ liên quan đến dự án.
Theo dõi và giám sát dự án phần mềm *(SPTO - Software Project Tracking and Oversight)*	- Kết quả thực tế và hiệu quả được theo dõi để so sánh với kế hoạch phần mềm. - Hoạt động khắc phục lỗi được thực hiện và quản lý khi kết quả thực tế và hiệu suất sai chệch đáng so với các kế hoạch phần mềm. - Các thay đổi được đồng ý bởi các nhóm và cá nhân bị ảnh hưởng.
Quản lý hợp đồng phụ *(SSM - Software Subcontract Management)*	- Các nhà thầu chính và nhà thầu phụ đồng ý cam kết của họ. - Các nhà thầu chính theo dõi kết quả thực tế của nhà thầu phụ và đối chiếu với các cam kết của họ. - Các nhà thầu chính và nhà thầu phụ duy trì liên lạc liên tục. - Các nhà thầu chính theo dõi hiệu suất thực tế của nhà thầu phụ và đối chiếu với các cam kết của họ.
Đảm bảo chất lượng phần mềm *(SQA - Software Quality Assurance)*	- Các hoạt động đảm bảo chất lượng phần mềm đã được lên kế hoạch. - Tuân thủ của các sản phẩm phần mềm và các hoạt động theo các tiêu chuẩn, thủ tục, và các yêu cầu được áp dụng được xác minh một cách khách quan. - Các nhóm và cá nhân bị ảnh hưởng được thông báo về các hoạt động đảm bảo chất lượng phần mềm và kết quả.

	- Các vấn đề về không tuân thủ mà không thể được giải quyết trong phạm vi dự án sẽ được giải quyết bởi người quản lý cấp cao.
Quản lý cấu hình phần mềm *(SCM - Software Configuration Management)*	- Hoạt động quản lý cấu hình phần mềm được lên kế hoạch. - Các sản phẩm công việc (work products) được chọn ra kiểm soát. - Các thay đổi đến các sản phẩm công việc được kiểm soát. - Các nhóm và cá nhân bị ảnh hưởng được thông báo về tình trạng và nội dung của các baselines phần mềm.

Bảng 1.2. Các mục tiêu của ba KPAs ở mức 3 (Defined)

KPA (lĩnh vực quy trình trọng điểm)	**Các mục tiêu (Goals)**
Quản lý phần mềm tích hợp *(ISM - Integrated Software Management)*	- Quy trình phần mềm của dự án đã được định nghĩa là một phiên bản được thay đổi cho phù hợp của quy trình phần mềm tiêu chuẩn của công ty. - Dự án được lập kế hoạch và quản lý theo quy trình phần mềm của dự án đã được định nghĩa.
Phối hợp giữa các nhóm *(IC - Intergroup Coordination)*	- Tất cả các nhóm bị ảnh hưởng đồng ý với yêu cầu của khách hàng. - Tất cả các nhóm đồng ý với các cam kết giữa các nhóm khác nhau. - Các nhóm xác định, theo dõi, và giải quyết các vấn đề giữa các nhóm.
Xem xét lại *(PR - Peer Reviews)*	- Xem xét lại (review) lại các hoạt động đã được lên kế hoạch. - Lỗi trong các sản phẩm công việc (work products) được xác định và loại bỏ.

Bảng 1.3 cho thấy các mục tiêu cho hai KPAs ở mức 4. Ở mức 4, khả năng của quy trình của công ty được hiểu về mặt định lượng. Khả năng của quy trình được sử dụng để thiết lập các mục tiêu định lượng cho một dự án. Dữ liệu về thực hiện dự án được thu thập một cách liên tục và được so sánh với dữ liệu về hiệu suất trong quá khứ, nếu có sự chênh lệch đáng kể, các hành động khắc phục thích hợp được áp dụng để đưa các dự án trở lại

trong tầm kiểm soát. Một khía cạnh quan trọng của mức 4 là việc sử dụng các kỹ thuật kiểm soát quy trình bằng cách dùng các kỹ thuật kiểm soát quy trình dùng thống kê một cách liên tục để mà mỗi hoạt động có thể được đánh giá và và được khắc phục bởi các hành động cần thiết.

Ba KPAs ở mức 5 tập trung vào việc cải thiện khả năng của quy trình (capability of the process). Trong ba KPAs, KPA Phòng Ngừa Lỗi (Defect Prevention KPA) là cái ảnh hưởng trực tiếp nhất đến quản lý dự án. KPA này đòi hỏi các lỗi được phòng ngừa một cách chủ động bằng cách phân tích có hệ thống nguyên nhân của lỗi và sau đó loại bỏ những nguyên nhân đó. Nếu các lỗi có thể được ngăn chặn xâm nhập vào phần mềm, công sức bỏ ra trong việc loại bỏ chúng có thể được giảm, do đó nâng cao chất lượng và năng suất.

Bảng 1.3. Các mục tiêu cho KPAs ở mức 4 (Managed)

KPA (lĩnh vực quy trình trọng điểm)	**Các mục tiêu (Goals)**
Quản lý quy trình theo định lượng *(QPM - Quantitative Process Management)*	- Các hoạt động quản lý quy trình theo định lượng (quantitative process management) được lên kế hoạch. - Hiệu suất quy trình của quy trình phần mềm đã được định nghĩa cho dự án được kiểm soát dựa theo định lượng. - Khả năng của quy trình phần mềm theo tiêu chuẩn của công ty được biết theo định lượng.
Quản lý chất lượng phần mềm *(SQM - Software Quality Management)*	- Các hoạt động quản lý chất lượng phần mềm của dự án được lên kế hoạch. - Các mục tiêu đo lường được cho chất lượng sản phẩm phần mềm và độ ưu tiên của chúng được xác định. - Tiến độ thực tế để hướng tới đạt được các mục tiêu chất lượng cho các sản phẩm phần mềm được định lượng và quản lý.

1.3 QUẢN LÝ DỰ ÁN Ở INFOSYS

Infosys thực hiện hàng trăm dự án mỗi năm. Toàn bộ trách nhiệm để thực hiện một dự án thuộc về người quản lý dự án, những người phải chắc chắn rằng nhóm sẽ giao phần mềm có chất lượng cao cho khách hàng, đúng thời gian và trong giới hạn chi phí. Để giúp người quản lý dự án thực hiện trách nhiệm này, hỗ trợ từ công ty này là cần thiết. Phần này giới thiệu ngắn gọn về bối cảnh ở Infosys và hỗ trợ của nó cho việc quản lý dự án.

1.3.1 Bối cảnh: Infosys

Infosys là một công ty phần mềm có trụ sở tại Bangalore, Ấn Độ. Nhiệm vụ của nó như đã nêu là "là một công ty toàn cầu cung cấp các giải pháp phần mềm tốt nhất bởi những người giỏi nhất." Nó sử dụng các mô hình phân phối toàn cầu, trong đó khách hàng có thể ở bất cứ nơi nào trên thế giới. Trong mô hình này, khách hàng được tìm kiếm từ bất cứ nơi nào trên thế giới nơi để mà công ty đạt được lợi nhuận cao nhất. Để đáp ứng yêu cầu khách hàng, một sự kết hợp của các quy trình, công nghệ, và quản lý được sử dụng để tách riêng các công việc để mà giá trị đó có thể được thêm vào trong các vị trí tối ưu nhất và sau đó tích hợp lại để giao cho khách hàng.

Infosys hiện đang sử dụng khoảng 10.000 người, với khoảng 15 trung tâm phát triển tại bốn quốc gia và có văn phòng tại hơn một chục quốc gia. Công ty được thành lập năm 1981 bởi bảy chuyên gia phần mềm với nguồn vốn ban đầu chỉ có $ 300. Hôm nay, Infosys có nguồn vốn hơn 8 tỷ đô la (dựa trên giá thị trường vào tháng sáu năm 2001), và doanh thu của nó là hơn 400 triệu đô la trong năm 2000 (doanh thu năm 1994 là 9.5 triệu đô la). Khách hàng của nó ở khắp toàn cầu và bao gồm tập đoàn lớn - hơn 60 trong số họ là các công ty Fortune 1000 – tức là bao gồm các doanh nghiệp đa dạng như: ngân hàng, bán lẻ, sản xuất, viễn thông, dịch vụ tài chính, bảo hiểm, và giao thông vận tải.

Infosys là một công ty có uy tín cao đã được đánh giá là công ty quản lý tốt nhất và uy tín nhất ở Ấn Độ và là một trong các công ty công nghệ thông tin hàng đầu châu Á (IT). Nó đã đã thu được nhiều giải thưởng, bao gồm giải thưởng Ramakrishna Bajaj được hình thành sau giải thưởng Malcolm Balridge. Có thể nói một cách an toàn rằng Infosys là một trong những công ty dịch vụ phần mềm tốt nhất trên thế giới.

Infosys cung cấp một cơ sở hạ tầng hàng đầu để những người quản lý dự án có thể phục vụ tốt hơn nhu cầu của khách hàng trên toàn thế giới. Công ty đã cung cấp thiết bị cho hội nghị truyền âm thanh cho hầu hết các nhóm để những người quản lý dự án có thể tương tác dễ dàng với khách hàng và với các thành viên trong nhóm nằm ở các vị trí địa lý khác nhau. Tương tự, các phòng hội nghị qua video được sử dụng cho sự tương tác giữa các địa điểm khác nhau của công ty cũng như cho các cuộc họp ảo. Khuôn viên chính của nó ở Bangalore là một trong các cơ sở dịch vụ phần mềm lớn nhất trên thế giới, với các cơ sở liên quan đến công việc như thư viện, máy tính mạnh và các thiết bị cho kết nối mạng, các cơ sở đào tạo, phòng thảo luận, các cơ sở chiếu, v.v., cũng như các thiết bị giải trí như một phòng trưng bày nghệ thuật, câu lạc bộ sức khỏe, và cơ sở vật chất dành cho bóng rổ, tennis, v.v.

Làm việc theo quy trình và cải tiến là một phần của văn hóa làm việc của Infosys, và các quy trình được định nghĩa cho hầu hết các công việc thường xuyên được thực hiện. Để định nghĩa và cải tiến quy trình, trước hết Infosys đã áp dụng khung làm việc tiêu chuẩn ISO 9000 và đã nhận chứng nhận ISO vào năm 1993. Để cải tiến hơn nữa quy trình phần mềm, Infosys sau đó đã áp dụng khung làm việc CMM. Lần đầu tiên nó được đánh giá ở mức 4 trong tháng 12 năm 1997, và sau đó ở mức 5 trong tháng 12 năm 1999. Trong việc theo đuổi các cải tiến liên tục, Infosys hiện đang sử dụng khung làm việc Malcolm Balridge cho tất cả các vấn đề liên quan đến cải tiến và xây dựng đội ngũ lãnh đạo xuất sắc trong tất cả các lĩnh vực hoạt động.

1.3.2 Hỗ trợ của nhóm quy trình công nghệ phần mềm cho các dự án

Bộ phận đảm bảo chất lượng tại Infosys chứa nhóm quy trình công nghệ phần mềm (SEPG - software engineering process group). SEPG chịu trách nhiệm phối hợp tất cả các hoạt động về quy trình, bao gồm cả định nghĩa quy trình, cải tiến quy trình và triển khai quy trình. Nó cũng quản lý tất cả các thông tin và dữ liệu liên quan đến việc sử dụng các quy trình (chẳng hạn như cơ sở dữ liệu về quy trình (process database) và baseline về khả năng quy trình (process capability baseline), được thảo luận thêm trong Chương 2).

Mặc dù trách nhiệm đối với tất cả các khía cạnh của sản phẩm được giao, bao gồm chất lượng, thuộc về nhóm dự án, nhưng SEPG sẽ tạo điều kiện cho nhóm dự án thực hiện dự án theo đúng các quy trình. SEPG cũng tạo một kênh độc lập để giám sát và báo cáo cho

người quản lý cấp cao về các vấn đề của quy trình và chất lượng. Bởi vì "các quy trình sẽ không tự bối rối," [6] SEPG giúp đảm bảo rằng các quy trình đã được định nghĩa được thực hiện theo và trở thành thực hành chuẩn.

Để kết thúc điều này, ngoài việc cung cấp các khóa đào tạo về quy trình, SEPG cung cấp một thành viên có liên quan đến dự án như là một *cố vấn chất lượng phần mềm (software quality adviser).* Cố vấn chất lượng hỗ trợ trong việc định nghĩa và thực hiện theo các quy trình, đảm bảo rằng các quy trình được thực hiện theo, hỗ trợ trong việc phân tích dữ liệu, và cung cấp bất kỳ khóa đào tạo cần thiết nào về quy trình. Bởi vì cố vấn thạo về các quy trình, các hướng dẫn, v.v., sự giúp đỡ chính của cố vấn sẽ diễn ra trong quá trình lập kế hoạch dự án. Cố vấn cũng xem xét các kế hoạch dự án để đảm bảo rằng nó có chứa tất cả các yếu tố chính.

Ngoài việc cung cấp tư vấn và giúp đỡ về các quy trình và các số đo (metrics), SEPG ở Infosys ước lượng thời gian hoàn thành và quản lý việc kiểm toán một cách độc lập và thường xuyên (xem Chương 11) để đảm bảo rằng các quy trình đã được định nghĩa và tiêu chuẩn đang được tuân thủ.

1.3.3 Sự liên quan của người quản lý cấp cao trong các dự án

Infosys tự hào trong việc cung cấp giá trị cho khách hàng của mình thông qua việc giao các sản phẩm xuất sắc. Tất cả mọi thứ tại Infosys, bao gồm cả cơ cấu tổ chức của nó, được thúc đẩy bởi mục tiêu phục vụ khách hàng một cách hiệu quả và khai thác nhanh chóng các cơ hội kinh doanh mới.

Để cung cấp các dịch vụ khách hàng, Infosys có nhiều đơn vị kinh doanh. Trong mỗi đơn vị kinh doanh*, một nhóm (team)*, đứng đầu là một *người quản lý dự án (project manager),* phát triển một dự án. Người quản lý dự án chịu trách nhiệm cho tất cả các khía cạnh trong thực hiện dự án, từ việc xác định các yêu cầu để đến cài đặt sau cùng của phần mềm. Người quản lý dự án báo cáo cho *người quản lý kinh doanh (business manager*) để người này tiếp tục báo cáo cho *người đứng đầu đơn vị kinh doanh (business unit head).*

Để xử lý các tình huống không thể được giải quyết bởi người quản lý dự án, người quản lý cấp cao (senior management) liên quan vào các dự án là cần thiết. Ở Infosys, người quản lý kinh doanh thường xuyên tương tác với người quản lý dự án và giám sát dự án thông qua các báo cáo tình trạng và báo cáo tại các cột mốc (milestones)(được thảo luận

trong Chương 11). Ngoài việc thường xuyên theo dõi, người quản lý kinh doanh cũng giúp giải quyết các vấn đề không thể được xử lý bởi nhóm dự án và phải cần đến cấp độ của mình (được thảo luận trong Chương 8). Người quản lý kinh doanh cũng tương tác với khách hàng để đảm bảo rằng họ hài lòng và bất kỳ vấn đề nào được đưa ra để giải quyết kịp thời.

Ngoài ra, những người cấp cao khác cũng xem xét lại (review) các dự án một cách định kỳ bằng cách thường xuyên tham gia vào cuộc kiểm toán nội bộ (được thảo luận trong Chương 11). Thông qua hai hệ thống – được gọi là PRISM (xem xét lại (review) dự án bởi người quản lý cấp cao) và IPM (quản lý dự án tích hợp/integrated project management) – các báo cáo ở các cột mốc (milestones) và kế hoạch dự án đã được chuẩn bị sẵn sàng cho người quản lý cấp cao xem xét lại. Tất cả người quản lý cấp cao dự kiến sẽ xem xét một số dự án định kỳ thông qua hệ thống này và để cung cấp thông tin phản hồi đến những người lãnh đạo dự án.

Nhìn chung, người quản lý cấp cao duy trì sự liên quan đến dự án chủ yếu bằng cách giám sát để đảm bảo rằng các mục tiêu của dự án được đáp ứng và khách hàng hoàn toàn hài lòng.

1.3.4 Đào tạo cho người quản lý dự án

Bởi vì những người quản lý dự án có trách nhiệm chính là thỏa yêu cầu khách hàng, họ cần phải nắm vững không chỉ thực hiện được các khía cạnh kỹ thuật của dự án mà còn tương tác với khách hàng, gợi ý để thu thập được yêu cầu khách hàng, quản lý nhóm dự án, v.v. Rõ ràng không ai có thể có được tất cả các kỹ năng cần thiết, vì vậy rất quan trọng để đào tạo con người phát triển các kỹ năng cần thiết. Infosys đã thực hiện một loạt các chương trình để giúp chuyển đổi từ kỹ sư thành những người lãnh đạo dự án.

Tất cả các kỹ sư mới (thí sinh) phải trải qua một chương trình đào tạo kéo dài từ ba đến bốn tháng. Ngoài đào tạo về kỹ thuật và công nghệ, chương trình này có một chương trình kéo dài một hoặc hai ngày về văn hoá kinh doanh, giao tiếp bằng văn bản, nói trước công chúng, ngôn ngữ cơ thể, v.v.

Sau đó, khi các kỹ sư đã sẵn sàng để trở thành những người lãnh đạo mô-đun (những người quản lý phát triển của một mô-đun hệ thống, đặc biệt là trong các dự án lớn hơn) hoặc quản lý dự án, họ tham gia một loạt các chương trình đào tạo kỹ thuật và kỹ năng

mềm. Bao gồm một khóa học quản quản lý dự án kéo dài năm ngày để tập trung vào tất cả các khía cạnh của quản lý dự án: lập kế hoạch, giám sát, kiểm soát, v.v. Một khóa học kéo dài hai tuần về đặc tả yêu cầu khách hàng và về quản lý để dạy làm thế nào để khơi gợi ra những yêu cầu, làm thế nào để lập tài liệu chúng, làm thế nào để xác minh (verify), v.v. Chương trình đào tạo kỹ năng mềm kéo dài năm ngày bao gồm các mô-đun về đánh giá và quản lý nhóm dự án, quan tâm khách hàng và quản lý khách hàng, lãnh đạo, nghi thức xã hội và văn hóa kinh doanh ở các quốc gia khác nhau, v.v.

Các chương trình được cung cấp khác thường xuyên tập trung vào các khía cạnh khác nhau của quản lý, những người lãnh đạo dự án tham dự các khóa học này khi họ có thời gian. Ngoài ra, các hội thảo về xây dựng nhóm cũng được tổ chức bởi các chuyên gia.

1.3.5 Quy trình quản lý dự án

Để một nhóm thực hiện một dự án thành công, hàng trăm công việc (tasks) phải được phải thực hiện, nhiều công việc trong số chúng phụ thuộc lẫn nhau. Quản lý một cách hiệu quả quy trình này là cực kỳ quan trọng để dẫn đến thành công. Ở Infosys, tập hợp các hoạt động được thực hiện bởi người quản lý dự án được xác định cụ thể trong quy trình quản lý dự án. Điều này khá chuẩn, có ba giai đoạn chính:

- Hoạch định dự án
- Thực hiện dự án
- Kết thúc dự án

Trong giai đoạn *lập kế hoạch cho dự án (project planning*), người quản lý dự án xem xét lại các cam kết trong hợp đồng và tạo ra một kế hoạch để đáp ứng chúng. Việc tạo một kế hoạch cho dự án liên quan đến việc định nghĩa một quy trình vòng đời (chu kỳ sống/life-cycle process) để được thực hiện theo, ước lượng nỗ lực (effort) và thời gian hoàn thành (schedule), chuẩn bị thời gian biểu chi tiết cho các công việc, v.v. Nó cũng bao gồm các kế hoạch về chất lượng và quản lý cấu hình cũng như quản lý rủi ro. Trong giai đoạn này, các hoạt động chính của người quản lý dự án như sau:

- Thực hiện các công việc khởi động và quản lý.
- Tạo một kế hoạch cho dự án và ước lượng thời gian hoàn thành (schedule).

- Xác định các mục tiêu của dự án.
- Xác định một quy trình chuẩn phù hợp để thực hiện dự án.
- Thay đổi quy trình chuẩn để đáp ứng được các yêu cầu (requirements) của dự án.
- Định nghĩa một quy trình để quản lý các thay đổi yêu cầu.
- Ước lượng các nỗ lực (effort).
- Lập kế hoạch cho nguồn nhân lực và tổ chức nhóm.
- Xác định các cột mốc (milestones) của dự án và ước lượng thời gian hoàn thành.
- Xác định các mục tiêu chất lượng và kế hoạch chất lượng để đạt được chúng.
- Lập kế hoạch để phòng ngừa lỗi.
- Xác định rủi ro và lập kế hoạch để giảm thiểu chúng.
- Xác định một kế hoạch đo lường đối với dự án.
- Xác định một kế hoạch đào tạo (training plan) cho dự án.
- Xác định các thủ tục theo dõi dự án.

- Thực hiện xem xét lại các kế hoạch và thời gian hoàn thành dự án.
- Có được ủy quyền từ người quản lý cấp cao.
- Xác định và xem xét lại kế hoạch quản lý cấu hình.
- Nhắc nhở nhóm dự án thực hiện theo kế hoạch quản lý dự án.

Ngoài người quản lý dự án, giai đoạn này liên quan đến các khách hàng, một đại diện của SEPG, và người quản lý kinh doanh của dự án. Tiêu chuẩn nhập là hợp đồng hoặc uỷ quyền dự án đã có. Tiêu chuẩn xuất là kế hoạch dự án đã được lập tài liệu và nhóm đã xem xét lại (xem Chương 10).

Giai đoạn thứ hai, *thực hiện dự án* (*project execution*), liên quan đến việc thực hiện kế hoạch dự án, theo dõi tình trạng của dự án, và làm các hiệu chỉnh bất cứ khi nào hiệu suất của dự án đi chệch với kế hoạch. Nói cách khác, nó liên quan đến việc theo dõi và kiểm soát việc thực hiện theo quy trình của dự án. Giai đoạn này là dài nhất trong quy trình quản lý dự án, kết hợp với các công việc định kỳ như giám sát tình trạng dự án và chất lượng và thực hiện bất cứ hiệu chỉnh nào nếu cần thiết. Trong giai đoạn này, người quản lý dự án thực hiện các hoạt động chính sau:

- Thực hiện dự án theo kế hoạch dự án.
- Theo dõi tình trạng dự án.
- Xem xét lại tình trạng dự án với người quản lý cấp cao.
- Giám sát xem có tuân theo quy trình dự án đã được xác định.
- Phân tích các lỗi và thực hiện các hoạt động ngăn chặn lỗi.
- Giám sát hiệu suất ở mức chương trình (program level).
- Tiến hành xem xét lại tại các cột mốc và hoạch định lại nếu cần thiết.

Các thành viên khác trong nhóm cũng tham gia trong giai đoạn này. Tiêu chuẩn nhập là kế hoạch dự án được hoàn thành và phê duyệt và tiêu chuẩn xuất là tất cả các sản phẩm công việc (work products) được chấp nhận bởi khách hàng.

Giai đoạn cuối cùng của quy trình quản lý dự án, *kết thúc dự án (project closure*), liên quan đến việc kết thúc dự án một cách có hệ thống sau khi khách hàng chấp nhận. Mục tiêu chính ở đây là để học hỏi từ kinh nghiệm để mà quy trình có thể được cải tiến. Phân tích dữ liệu của các dự án đã hoàn thành cấu thành các hoạt động chính; số đo được phân tích, các sản phẩm của quy trình (process assets) (nhiều tài liệu, như các bản mẫu (templates) và hướng dẫn (guidelines) được sử dụng để hỗ trợ trong việc quản lý quy trình) được thu thập để sử dụng trong tương lai, và các bài học được ghi nhận. Bởi vì học từ dự án là mục tiêu chính, đây là một nhóm các hoạt động có liên quan đến người quản lý dự án, SEPG, và các thành viên khác của nhóm. Tiêu chuẩn nhập là các khách hàng đã chấp nhận sản phẩm. Tiêu chuẩn xuất là một cuộc họp hậu dự án (postproject) đã được

tiến hành. Các kết quả đầu ra chính của giai đoạn này là báo cáo kết thúc dự án và các sản phẩm của quy trình (process assets) được thu thập.

Phần còn lại của cuốn sách này thảo luận về các yếu tố khác nhau của quy trình quản lý. Phần I bao gồm các chương riêng biệt dành cho các hoạt động lập kế hoạch quan trọng, chẳng hạn như định nghĩa quy trình và hiệu chỉnh, quản lý rủi ro, ước lượng nỗ lực và thời gian hoàn thành (schedule), lập kế hoạch chất lượng, và lập kế hoạch quản lý cấu hình. Các công việc khác trong giai đoạn lập kế hoạch (như hoạch định nguồn nhân lực, tổ chức dự án, các công cụ được sử dụng, thủ tục theo dõi dự án, v.v.) sẽ được thảo luận ngắn gọn trong các Chương 7 và 8. Phần II bao gồm các chương về giám sát và kiểm soát dự án và về kết thúc dự án.

1.4 TỔNG QUAN VỀ CASE STUDY ACIC

ACIC Corporation (tên đã được thay đổi để bảo vệ bí mật) là một viện tài chính trị giá nhiều tỷ đô la. Để theo kịp với thời đại, cách đây vài năm, nó bắt đầu từ từ dùng các ứng dụng Web, và nó muốn bắt đầu xây dựng một dịch vụ trực tuyến (on-line service) để mở và theo dõi tài khoản. Bởi vì Infosys đã xây dựng thành công một số dịch vụ điện tử (e-services) cho ACIC trước đó trong một dự án được gọi là Synergy (tên đã được thay đổi), ACIC thuê Infosys để phân tích vấn đề. Công việc này được thực hiện trong thời gian giới hạn và vật chất (T & M - time and material*),* tức là, khách hàng trả tiền cho nỗ lực (effort) đã được dùng bởi Infosys để làm việc phân tích. Dựa trên kết quả phân tích, Infosys đã đấu thầu thành công dự án Web, cung cấp một case study (trường hợp nghiên cứu) ACIC được dùng xuyên suốt cuốn sách này. Dự án phát hành dịch vụ mới thành công đúng thời gian, và phần mềm đã hoạt động mà không có bất kỳ vấn đề gì.

Dự án ACIC minh họa cho việc lập kế hoạch và giám sát các công việc được thực hiện trong một dự án tại Infosys. Nhiều kết quả đầu ra liên quan đến quản lý của dự án ACIC được đưa vào các chương trong sách. Chúng bao gồm những điều sau đây:

- Dữ liệu từ dự án Synergy, được sử dụng bởi người quản lý dự án ACIC trong suốt thời gian lập kế hoạch (Chương 2)

- Kế hoạch quy trình của dự án (Chương 3)

- Một phân tích về tác động của việc thay đổi yêu cầu (Chương 3)

- Ước lượng nỗ lực (effort estimation) và thời gian biểu (schedule) ở mức cao, cùng với một mô tả về cách thức thực hiện chúng (Chương 4)

- Kế hoạch chất lượng có chứa các mục tiêu chất lượng và các kế hoạch để đạt được chúng, bao gồm cả các kế hoạch để phòng ngừa lỗi và xem xét lại (review) (Chương 5)

- Kế hoạch quản lý rủi ro mô tả những rủi ro lớn, nguy cơ xảy ra và tác động của chúng, xếp độ ưu tiên, và các kế hoạch giảm thiểu rủi ro cho những rủi ro có độ ưu tiên cao (Chương 6)

- Các kế hoạch đo lường và theo dõi (Chương 7)

- Lập kế hoạch quản lý cho toàn dự án, bao gồm kế hoạch quản lý nhóm dự án và giao tiếp (communication) với khách hàng (Chương 8)

- Kế hoạch quản lý toàn bộ cấu hình (Chương 9)

- Các tài liệu theo dõi dự án, bao gồm cả nhật ký ghi lỗi (defect log), nhật ký ghi các vấn đề phát sinh (issues log), báo cáo tình trạng (status report), và báo cáo tại các cột mốc (milestone report) (Chương 11)

- Chi tiết về công tác phòng ngừa lỗi, bao gồm các kết quả phân tích lỗi và tác động lên dự án của kế hoạch ngăn chặn lỗi (Chương 11)

- Báo cáo kết thúc toàn bộ dự án, bao gồm các số đo về chất lượng, năng suất, chi phí cho chất lượng, hiệu quả loại bỏ lỗi, v.v. (Chương 12)

1.5 TÓM TẮT

Quản lý dự án phần mềm có lẽ là yếu tố quan trọng nhất trong các kết quả của một dự án. Nếu không có quản lý dự án phù hợp, một dự án gần như chắc chắn sẽ thất bại. Nhiều công ty đã phát triển các quy trình quản lý dự án hiệu quả. Cuốn sách này mô tả các quy trình này cho một trong những công ty như thế, Infosys, đã được đánh giá ở mức 5 của CMM và những người quản lý dự án đã thực hiện thành công hàng trăm dự án.

Dưới đây là những tóm tắt cho Chương này:

- Các quy trình cho các khía cạnh khác nhau của quản lý dự án không nên được xem xét trong sự cô lập. Trong một quy trình cân bằng, các thực hành tích hợp trơn tru.

- Các quy trình của một công ty nên đóng gói các thực hành tốt nhất của mình để áp dụng lại, giúp các dự án mới thành công và tránh thất bại.

- Ở mức cao nhất, quy trình quản lý dự án bao gồm ba giai đoạn: lập kế hoạch, thực hiện, và kết thúc.

- Để thực hiện hiệu quả các dự án, người quản lý dự án cần được hỗ trợ thông qua sự giúp đỡ của một SEPG trong quá trình thực hiện; người quản lý cấp cao giám sát và giải quyết vấn đề phát sinh; đào tạo tốt.

- Nhiều quy trình then chốt có mặt ở tất cả các mức trưởng thành của CMM tập trung trực tiếp vào quản lý dự án.

1.6 CÁC THAM KHẢO

1. L.H. Putnam and W. Myers. *Industrial Strength Software: Effective Management Using Measurement*. IEEE Computer Society Press, 1997.

2. R.L. Glass. *Software Runaways: Lessons Learned from Massive Software Project Failures*. Prentice Hall PTR, 1998.

3. International Standards Organization. *ISO 9001, Quality Systems—Model for Quality Assurance in Design/Development, Production, Installation, and Services*. 1987.

4. International Standards Organization. *ISO 9000-3, Guidelines for the Application of ISO9001 to the Development, Supply and Maintenance of Software*. 1991.

5. U.K. Dept. of Trade and Industry and British Computer Society. *TickIT: A Guide to Software Quality Management System Construction and Certification Using EN29001*. 1992.

6. W. Humphrey. *Managing the Software Process*. Addison-Wesley, 1989.

7. Carnegie Mellon University/Software Engineering Institute. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley, 1995.

8. P. Jalote. *CMM in Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley, 2000.

9. S. Masters. *CMM Based Appraisal for Internal Process Improvement (CBA-IPI): Method Description. Technical Report, Software Engineering Institute, CMU/SEI-96-TR-007*, 1996.

10. Software Engineering Institute. Maturity Profile Report, http://www.sei.cmu.edu/activities/sema/profile.html.

Phần I: Lập kế hoạch cho dự án

Chương 2. Cơ sở hạ tầng để lập kế hoạch cho dự án

Chương 3. Lập kế hoạch cho quy trình

Chương 4. Ước lượng nỗ lực và thời gian biểu

Chương 5. Lập kế hoạch cho chất lượng

Chương 6. Quản lý rủi ro

Chương 7. Đo lường và lập kế hoạch theo dõi

Chương 8. Kế hoạch quản lý dự án

Chương 9. Quản lý cấu hình

Chương 2. Cơ sở hạ tầng để lập kế hoạch cho dự án

Vào cuối cuộc họp xem xét lại kế hoạch (plan review meeting) cho dự án thứ 10 của mình, Dinesh, lãnh đạo giao hàng tại Infosys vào thời điểm đó, đã cảm thấy không vui. Trong khi xem xét lại kế hoạch của các dự án, Dinesh đã nhận thấy rằng mỗi người quản lý dự án đã làm việc trong một thế giới của riêng mình, cố gắng để tạo ra các quy trình tối ưu để thực hiện dự án của mình và làm các ước lượng để họ có thể đạt được mục tiêu đề ra – điều này xảy ra thậm chí Dinesh biết rằng nhiều dự án tương tự vừa được thực hiện bởi những nhóm dự án khác mà kinh nghiệm và dữ liệu có thể giúp làm dịu nỗi đau của người quản lý dự án. Những người quản lý dự án không chỉ đang đầu tư nỗ lực để lập kế hoạch cho dự án của họ, mà họ còn đang "lập kế hoạch" để tạo ra những sai lầm tương tự như những người quản lý dự án khác trước họ. Dinesh nhận ra rằng câu trả lời cho tình trạng khó khăn này nằm ở việc tạo ra một bộ nhớ của công ty mà mọi người quản lý dự án có thể truy cập vào.

Câu hỏi có giá trị triệu đô la là làm thế nào để xây dựng bộ nhớ công ty và sử dụng nó làm cơ sở hạ tầng để lập kế hoạch cho dự án. Các phẩn tử của nó là gì? Làm thế nào kinh nghiệm quá khứ có thể được ghi lại một cách có hệ thống và sẵn sàng để được tái sử dụng? Làm thế nào Infosys có thể giữ cho cơ sở hạ tầng luôn được cập nhật?

Chương này bàn về các phần tử chính của cơ sở hạ tầng để lập kế hoạch cho dự án tại Infosys: cơ sở dữ liệu quy trình, khả năng cơ sở của quy trình, và các thành phần của quy trình. Trong những chương sau, bạn sẽ thấy làm thế nào các phần tử này được sử dụng trong lập kế hoạch dự án.

Cơ sở dữ liệu quy trình (CSDLQT) lưu giữ dữ liệu hiệu suất (performance) của các dự án đã hoàn thành. *Baseline về khả năng quy trình (process capability baseline*) tóm tắt hiệu suất của nhiều dự án khác nhau. Do đó nó xác định cụ thể phạm vi của các kết quả được mong đợi nếu các quy trình tương tự được thực hiện theo. *Các đối tượng của quy trình (process assets)* là các tài liệu (documents) như bản liệt kê những mục cần kiểm tra (checklists), các bản mẫu (templates), phương pháp (methodologies), và các bài học – đây là những thứ giúp ghi nhận lại kinh nghiệm quá khứ và giúp những người quản lý dự án và các kỹ sư sử dụng hiệu quả các quy trình. Những phần tử này thường hiện diện

trong các công ty có mức trưởng thành (CMM) cao, mặc dù chúng vẫn có thể được sử dụng trong các công ty khác [1].

2.1 CƠ SỞ DỮ LIỆU QUY TRÌNH (CSDLQT)

Cơ sở dữ liệu quy trình là một kho lưu trữ lâu dài dữ liệu về hiệu suất quy trình của các dự án; nó có thể được sử dụng để lập kế hoạch, ước lượng, phân tích năng suất và chất lượng, và những mục đích khác. CSDLQT bao gồm dữ liệu từ các dự án đã hoàn thành, mỗi dự án cung cấp một mẫu tin (record) trong cơ sở dữ liệu. Như bạn có thể tưởng tượng, để được lưu trữ trong CSDLQT, dữ liệu phải được thu thập, phân tích, và sau đó tổ chức để nhập vào (entry). Ở đây chúng tôi tập trung vào cách trình bày dữ liệu trong CSDLQT tại Infosys; Chương 7 giải thích dữ liệu được thu thập như thế nào.

2.1.1 Nội dung của CSDLQT

Để sử dụng các thông tin trong CSDLQT trong suốt quá trình lập kế hoạch, những người quản lý dự án thường xuyên tìm kiếm thông tin về các dự án tương tự. Để cho phép kiểm tra sự tương tự, bạn nên nắm bắt các thông tin chung trong CSDLQT về dự án, chẳng hạn như ngôn ngữ được sử dụng, nền tảng (platforms), cơ sở dữ liệu được sử dụng, các công cụ được sử dụng, kích thước, và nỗ lực (effort). Với các loại thông tin này, một người quản lý dự án có thể tìm kiếm và tìm thấy thông tin trên tất cả các dự án, ví dụ, tập trung vào một lĩnh vực ứng dụng cụ thể, sử dụng một hệ quản trị cơ sở dữ liệu cụ thể (DBMS) hoặc ngôn ngữ, hoặc một nền tảng cụ thể.

Để giúp ích cho việc lập kế hoạch dự án, bạn nên thu thập dữ liệu về nỗ lực, lỗi, thời gian hoàn thành, rủi ro, v.v. Nếu nỗ lực tổng cộng được dùng cho một dự án được biết đến, cùng với kích thước và sự phân phối nỗ lực vào các giai đoạn phát triển khác nhau, dữ liệu này có thể được sử dụng để ước lượng nỗ lực cho dự án mới.

Vì thế, dữ liệu được thu thập trong CSDLQT ở Infosys có thể được phân loại như sau:

- Các đặc trưng (characteristics) của dự án
- Thời gian biểu của dự án
- Nỗ lực của dự án

- Kích thước
- Lỗi

Dữ liệu về các đặc trưng của dự án bao gồm tên dự án, tên của người quản lý dự án và các người lãnh đạo mô-đun (để họ có thể được liên lạc để biết thêm thông tin hoặc làm rõ), đơn vị kinh doanh (cho phép phân tích dựa trên đơn vị kinh doanh), quy trình được triển khai (để cho phép các phân tích riêng biệt các quá trình khác nhau), lĩnh vực ứng dụng (application domain), nền tảng phần cứng, các ngôn ngữ được sử dụng, DBMS được sử dụng, một mô tả ngắn gọn về các mục tiêu của dự án, thông tin về những rủi ro, thời gian để phát triển dự án, và kích thước nhóm dự án.

Dữ liệu về thời gian hoàn thành chủ yếu là ngày bắt đầu và ngày kết thúc (dự kiến và thực tế) của dự án. Các dữ liệu về nỗ lực của dự án bao gồm nỗ lực tổng cộng được ước lượng ban đầu và nỗ lực thực tế, và lượng phân phối của nỗ lực thực tế qua các giai đoạn phát triển khác nhau, chẳng hạn như lúc bắt đầu dự án, quản lý yêu cầu, thiết kế, cài đặt, kiểm thử đơn vị, và các giai đoạn khác. Chương 7 thảo luận cách thu thập những dữ liệu nỗ lực này như thế nào.

Kích thước của các phần mềm được phát triển có thể là số dòng mã (LOC – lines of code), số lượng các chương trình đơn giản, trung bình, hay phức tạp, hoặc một sự kết hợp của các số đo này. Thậm chí nếu số Điểm chức năng (FP - Function points) không được sử dụng để ước lượng, bạn vẫn có thể có được một số đo (metric) thống nhất cho năng suất bằng trình bày kích thước cuối cùng bằng số Điểm chức năng (FP), thường thu được bằng cách chuyển đổi kích thước được đo bằng LOC sang số Điểm chức năng (FP) bằng cách sử dụng bảng chuyển đổi [3].

Dữ liệu về lỗi bao gồm số lượng các lỗi được tìm thấy trong các hoạt động phát hiện lỗi khác nhau, và số lượng lỗi được tiêm vào (defects injected) ở các giai đoạn phát triển khác nhau. Do đó, bạn ghi lại số lượng lỗi được phát hiện từ các nguồn khác nhau: xem xét lại yêu cầu (requirements review), xem xét lại thiết kế (design review), xem xét lại mã (code review), kiểm thử đơn vị (unit testing), và các giai đoạn khác. Chương 7 giải thích làm thế nào các dự án ghi nhận lại các dữ liệu về lỗi.

Ngoài ra, nhiều ghi chú khác cũng được ghi nhận lại, bao gồm ghi chú về ước lượng (ví dụ, các tiêu chuẩn được sử dụng để phân loại một chương trình (program) là đơn giản,

trung bình, hay phức tạp) và ghi chú về quản lý rủi ro (ví dụ, làm thế nào nhận biết nguy cơ rủi ro trong suốt quá trình thực hiện dự án).

2.1.2 Ví dụ về một mục nhập vào cơ sở dữ liệu quy trình

Hãy nhìn vào một mục của cơ sở dữ liệu quy trình (CSDLQT) mẫu của một dự án, mà chúng tôi sẽ giới thiệu qua tên gọi Synergy. Trong dự án Synergy, một ứng dụng đã được xây dựng trước đây, dữ liệu trong CSDLQT của nó có thể được tham khảo đến bởi case study (dự án ACIC). Cụ thể là case study này sẽ tham khảo tới một mục của CSDLQT trong suốt thời gian lập kế hoạch.

Dữ liệu của bốn bảng chính được trình bày (ví dụ đã sử dụng các tên có ý nghĩa, nhưng các mã (codes) được sử dụng trong các cơ sở dữ liệu thực tế cho các giai đoạn khác nhau và các hoạt động về chất lượng). Trong ví dụ này, các dữ liệu thì khá đầy đủ; tuy nhiên, trong các trường hợp khác, các dữ liệu có thể không được đầy đủ như vậy. Những dữ liệu như thế có thể không phải lúc nào cũng bị vứt bỏ bởi vì các thông tin có thể vẫn có ích [4]. Do đó, những dữ liệu như thế cần được lưu trữ vào CSDLQT.

Bảng 2.1 cung cấp các thông tin chung về dự án, bao gồm ngày bắt đầu và ngày kết thúc (start and end dates) (ước lượng và thực tế), nỗ lực ước lượng (estimated effort) (nỗ lực thực tế không được lưu trong bảng này bởi vì nó có thể được tính toán từ bảng nỗ lực), kích thước nhóm dự án lúc cao điểm, thông tin về rủi ro, các công cụ được sử dụng. Ngoài ra, các thông tin khác - ví dụ, về khách hàng – cũng được lưu trữ trong bảng này.

Bảng 2.1. Dữ liệu chung về một dự án (General Data about a Project)

Các đặc trưng chung (General Characteristics)	
Tên trường (Field Name)	**Giá trị của Synergy (Value for Synergy)**
Loại quy trình (ProcessCategory)	Phát triển (Development)
Chu kỳ sống (LifeCycle)	Toàn bộ (Full)
Lĩnh vực nghiệp vụ (BusinessDomain)	Môi giới/Tài chính (Brokerage/Finance)
Ghi chú điều chỉnh quy trình (ProcessTailoringNotes)	Bổ sung nhóm xem xét lại cho các tài liệu bị ảnh hưởng nhiều. Chương chình đầu tiên của mỗi nhà phát triển được nhóm

	xem xét lại. (Added group review for high-impact documents. First program of each developer was group reviewed.)
Kích thước nhóm cao điểm (PeakTeamSize)	12
Công cụ được sử dụng (ToolsUsed)	VSS for document CM, VAJ for source code
Ngày bắt đầu dự kiến (EstimatedStart)	20 Jan 2000
Ngày kết thúc dự kiến (EstimatedFinish)	5 May 2000
Số giờ công ước lượng (EstimatedEffortHrs)	3,106
Ghi chú về ước lượng (EstimationNotes)	Dùng phương pháp Điểm trường hợp sử dụng (Use case point approach was one method used for estimation.)
Ngày bắt đầu thực tế (ActualStart)	20 Jan 2000
Ngày kết thúc thực tế (ActualFinish)	5 May 2000
Rủi ro đầu tiên (First Risk)	Sẽ làm việc thông qua liên kết đến cơ sở dữ liệu của khách hàng (Working through link on customer DB)
Rủi ro thứ hai (Second Risk)	Yêu cầu bổ sung (Additional requirements)
(Third Risk)	Mệt mỏi (Attrition)
Rủi ro thứ ba (RiskNotes)	Làm việc với nhiều thay đổi; đồng ý để thực hiện các mở rộng sau khi chấp nhận sản phẩm này; nhóm sẽ xây dựng các thực (Worked in shifts; agreed to take enhancements after acceptance of this product; team building exercises were done.)

Bảng 2.2 lưu giữ các thông tin về nỗ lực. Đối với các giai đoạn khác nhau trong quy trình này, nó bao gồm dữ liệu về những nỗ lực được dùng cho các hoạt động và nỗ lực dùng để làm lại công việc (rework) sau đó. Nỗ lực làm lại được thu thập bởi vì nó giúp ích trong việc tính toán và hiểu về chi phí của chất lượng. Bảng 2.2 cho thấy các dữ liệu về nỗ lực của dự án Synergy tính theo người-giờ (person-hours). Nỗ lực được ước lượng cho các giai đoạn cũng được ghi nhận. (Nỗ lực tổng cộng được dùng trong các giai đoạn chu kỳ sống là 2,950 người-giờ, và trong việc xem xét lại là 223 người-giờ, do đó nỗ lực thực tế được dùng tổng cộng là 3,173 người-giờ; trong khi nỗ lực tổng cộng được ước lượng là 3,012 người-giờ.)

Table 2.2. Dữ liệu về nỗ lực (Effort Data)

Nỗ lực của các giai đoạn (Effort by Stage)			
Giai đoạn (Stage)	**Nỗ lực của công việc (Task Effort)**	**Nỗ lực xem xét lại (Review Effort)**	**Được ước lượng (Estimated)**
Requirements analysis	0	0	0
Design	414	32	367
Coding	1147	76	1182
Independent unit testing	156	74	269
Integration testing	251	30	180
Acceptance testing and installation	183	0	175
Project management	237	8	357
Configuration management	30	3	38
Project-specific training	200	0	218
Others	332	0	226

Bảng 2.3 có chứa thông tin về lỗi. Có mong muốn để biết được không chỉ khi nào một lỗi được phát hiện mà còn khi nào nó được tiêm vào (injected). Do đó, bạn nên ghi nhận lại số lượng lỗi đã được tìm thấy cho mỗi kết hợp: *giai đoạn tiêm vào (injection stage) và giai đoạn phát hiện (detection stage)* (tức là, lỗi đã được tìm thấy trong tài liệu/sản phẩm của các giai đoạn phát triển nào và hoạt động đảm bảo chất lượng nào đã phát hiện ra được

lỗi đó). Các giai đoạn phát hiện lỗi (detection stages) bao gồm: xem xét lại (reviews) và kiểm thử (testing). Trong khi đó, các giai đoạn tiêm lỗi (injection stages) bao gồm: yêu cầu (requirements), thiết kế (design), và cài đặt mã (coding). Nếu bạn có thể tách riêng ra được các lỗi đã được phát hiện đúng theo từng giai đoạn đã tiêm chúng vào, khi đó bạn có thể tính toán được hiệu quả của việc loại bỏ của các giai đoạn phát hiện lỗi. Thông tin này có thể hữu ích cho việc xác định các khu vực cần được cải tiến thêm. Bảng 2.3 cho thấy dữ liệu về lỗi của dự án Synergy.

Bảng 2.3. Dữ liệu về lỗi của dự án Synergy

		Requirement Review	Design Review	Code Review	Unit Testing	System Testing	Acceptance Testing
Giai đoạn tiêm lỗi	Requirements	0	0	0	1	1	0
	Design		14	3	1	0	0
	Coding			21	48	17	6
		Giai đoạn phát hiện lỗi					

Bảng 2.4 chứa thông tin về kích thước của dự án. Các ngôn ngữ khác nhau có thể được sử dụng trong một dự án, do đó, bảng này có thể có nhiều mục. Nhiều đơn vị đo kích thước khác nhau có thể được sử dụng. Nói chung, nếu kích thước được đo bằng LOC, kích thước bằng Điểm chức năng (FP - Function points) cũng có thể được tính bằng cách sử dụng các bảng chuyển đổi khi cần thiết. Những thông tin này được sử dụng để tính toán năng suất (productivity) thông qua số Điểm chức năng (FP). Bởi vì kích thước là một yếu tố quan trọng trong việc xác định năng suất, các yếu tố khác, chẳng hạn như hệ điều hành và phần cứng được sử dụng, cũng được thu thập. Bảng 2.4 cho thấy các giá trị của dự án Synergy.

Bảng 2.4. Dữ liệu về kích thước của dự án Synergy

Kích thước (Size)					
Ngôn ngữ lập trình (LangCode)	**Hệ điều hành (OSCode)**	**Hệ quản trị cơ sở dữ liệu (DBMSCode)**	**Phần cứng (HWCode)**	**Đơn vị đo kích thước (MeasureCode)**	**Kích thước mã thực tế (ActualCode Size)**
Java	Windows	—	PC	LOC	8,082
Persistent Builder	Windows NT	DB2	Client MC	LOC	12,185

2.2 BASELINE VỀ KHẢ NĂNG CỦA QUY TRÌNH (THE PROCESS CAPABILITY BASELINE)

Cơ sở dữ liệu quy trình (CSDLQT) chứa dữ liệu của từng dự án,baseline về khả năng của quy trình cho biết một hình ảnh tức thời về khả năng (được định lượng) của quy trình tại một số điểm thời gian. *Khả năng của một quy trình* (capability of a process) về bản chất là phạm vi của các kết quả dự kiến của một dự án nếu quy trình này được áp dụng [5]. Khả năng của một quy trình ổn định có thể được xác định từ năng lực trong quá khứ của quy trình. Nếu các baseline được xây dựng thường xuyên, các khuynh hướng (trends) về khả năng của quy trình có thể được xác định dễ dàng – đây là lý do quan trọng để cần có một CSDLQT.

Vấn đề phát sinh đầu tiên phải được giải quyết là CSDLQT nên phải chứa cái gì – tức là nó nên chứa các các loại "kết quả" nào. CSDLQT ở Infosys chứa hiệu suất của quy trình (process performance): năng suất, chất lượng, thời gian hoàn thành (schedule), sự phân phối (distribution) nỗ lực và phân phối lỗi. Nó xác định các vấn đề sau đây:

- Chất lượng được giao (Delivered quality)
- Năng suất (Productivity)
- Thời gian biểu (Schedule)
- Sự phân phối nỗ lực (Effort distribution)
- Tỷ lệ tiêm lỗi (Defect injection rate)
- Hiệu quả của việc loại bỏ lỗi xuyên suốt chu kỳ sống (In-process defect removal efficiency)
- Chi phí cho chất lượng (Cost of quality)
- Sự phân phối lỗi (Defect distribution)

Những thông tin này có thể được sử dụng trong việc lập kế hoạch dự án. Ví dụ, một người quản lý dự án có thể sử dụng năng suất và kích thước được ước lượng để ước lượng nỗ lực cho dự án và có thể sử dụng sự phân phối nỗ lực để dự đoán nỗ lực cho các giai đoạn phát triển khác nhau và thực hiện phân công nhân sự. Tương tự như vậy, bạn

có thể sử dụng tỷ lệ tiêm lỗi để dự đoán tổng số lỗi và có thể sử dụng dữ liệu về sự phân phối lỗi để dự đoán mức lỗi (defect levels) của các hoạt động phát hiện lỗi khác nhau. Hiệu quả tổng thể của việc loại bỏ lỗi hoặc chất lượng có thể được sử dụng để dự báo số lỗi còn sót lại sau khi sản phẩm phần mềm được giao cho khách hàng và để lập kế hoạch cho việc bảo trì.

CSDLQT cũng có vai trò quan trọng trong quản lý quy trình tổng thể của công ty. Ví dụ, bạn có thể dễ dàng đo sự cải tiến quy trình bằng cách phân tích các khuynh hướng trong CSDLQT theo thời gian. Bạn cũng có thể cải tiến việc lập kế hoạch cho các sáng kiến cải tiến bằng cách sử dụng thông tin về sự phân phối nỗ lực và phân phối lỗi, tỷ lệ phát sinh lỗi, hiệu quả loại bỏ lỗi, và các số đo khác.

Bởi vì một baseline cho thấy khả năng của một quy trình, bạn phải tạo ra một baseline riêng biệt cho mỗi quy trình trong công ty. Ở Infosys, các quy trình riêng biệt được định nghĩa cho các dự án: bảo trì, tái cấu trúc (reengineering), và phát triển (development); do đó, một baseline riêng biệt được định nghĩa cho mỗi quy trình này. Tuy nhiên, các quy trình này thì quá rộng và chúng chỉ cung cấp hướng dẫn chung chung. Nếu một loại dự án cụ thể được thực hiện thường xuyên, một CSDLQT cho loại dự án đó sẽ được xây dựng. Khi đó, baseline sẽ trình bày một phạm vi (range) của kết quả dự kiến gần khít hơn (độ sai lệch thấp) cho các dự án có cùng loại.

CSDLQT được trình bày trong Bảng 2.5 có thể được sử dụng để áp dụng cho các dự án phát triển và các dự án sử dụng ngôn ngữ thuộc thế hệ thứ ba (3GL - Third-Generation Languages). Ví dụ, CSDLQT cho các dự án bảo trì sẽ khác đi không chỉ ở các con số thực tế trong CSDLQT mà còn ở thông tin được lưu trữ (để phù hợp với quy trình bảo trì).

Giải thích về CSDLQT này cũng không khó. CSDLQT đã nói rằng năng suất trung bình là khoảng 12 Điểm chức năng (FP) cho mỗi tháng-người (person-month), với dao động trong khoảng từ 4 đến 31 Điểm chức năng (FP). Chất lượng trung bình của các dự án này là 0.02 lỗi trên mỗi Điểm chức năng, với dao động trong khoảng từ 0.00 đến 0.094. CSDLQT cũng cho biết các giá trị trung bình và khoảng dao động của các thông số khác, như tỷ lệ tiêm lỗi (defect injection rate) và hiệu quả của việc loại bỏ lỗi tổng thể (total defect removal efficiency). Tỷ lệ tiêm lỗi (defect injection rate) thường được đưa ra tương ứng với kích thước cũng như nỗ lực, vì thế nỗ lực cũng như kích thước được ước lượng có thể được sử dụng để ước lượng số lỗi. Chi phí của chất lượng bao gồm chi phí của tất cả các hoạt

động có thể giúp phòng ngừa hoặc loại bỏ lỗi. Đối với nỗ lực, lỗi, các tỷ lệ tiêm lỗi, và sự phân phối của chúng qua các giai đoạn khác nhau cũng được ghi nhận.

Bảng 2.5. Cơ sở dữ liệu về khả năng của quy trình phát triển dự án

STT	Tham số (Parameter)	Các chú ý (Remarks)	Baseline tổng quát, các dự án phát triển (General Baseline, Development Projects)
1	Chất lượng được giao *(Delivered Quality)*	Số lỗi được giao/FP *(Delivered Defects /FP (Delivered Defects = Acceptance Defects + Warranty Defects))*	0.00–0.094 Lỗi được giao/FP (trung bình: 0.021) *(0.00–0.094 Delivered Defects/FP (avg: 0.021))*
		Chất lượng được diễn tả bằng nỗ lực *(Quality expressed in terms of effort)*	0.00–0.012 Lỗi được giao/Người-giờ (trung bình: 0.003) *(0.00–0.012 Delivered Defects/Person-hour (avg: 0.003))*
2	Năng suất *(Productivity)*	Đối với các ngôn ngữ thế hệ thứ 3 *(For Third-Generation Languages)*	4–31 FP/Người-tháng (trung bình: 12) *(4–31 FP/person-month (avg: 12))*
		Đối với các ngôn ngữ thế hệ thứ 3 *(For Fourth-Generation Languages)*	10–129 FP/Người-tháng (trung bình: 50) *(10–129 FP/person-month (avg: 50))*
3	Sự bám theo thời gian biểu *(Schedule Adherence)*		81% của dự án được giao trong khoảng ±10% của thời gian hoàn thành đã được đồng ý *(81% of projects delivered within ±10% of the agreed schedule)*

4	Nỗ lực *(Effort)*		
4.1	Nỗ lực xây dựng *(Build Effort)*	Nỗ lực xây dựng cho một dự án trung bình *(Build effort for a medium program)*	Nhỏ nhất-Trung bình-Lớn nhất 2-4-6 người-ngày *(Min–Mean–Max* *2–4–6 person days)*
4.2	Phân phối nỗ lực *(Effort Distribution)*		Nhỏ nhất-Trung bình-Lớn nhất *(Min–Mean–Max)*
		Req. Analysis + Design	1–15–29%
		Build (Code + Code review + UT)	22–41–60%
		Coding	14– 33– 52%
		Review	1–4–11%
		Unit Testing	1–5–14%
		Integration Testing + System Testing	1–9–20%
		Acceptance Testing & Warranty	1– 8–23%
		Project Mgmt + Config. Mgmt	1–10–20%
		Training	1–7–14%
		Others	1–10–26%
5	Lỗi (Defects)		
5.1	Tỷ lệ tiêm lỗi *(Defect Injection Rate)*	Tỷ lệ tiêm lỗi tổng thể trong toàn bộ chu kỳ sống tính theo kích thước *(Overall LC defect injection rate in terms of size)*	0.02–1.12 Lỗi/FP (trung bình: 0.33) *(0.02–1.12 Defects/FP (avg: 0.33))*

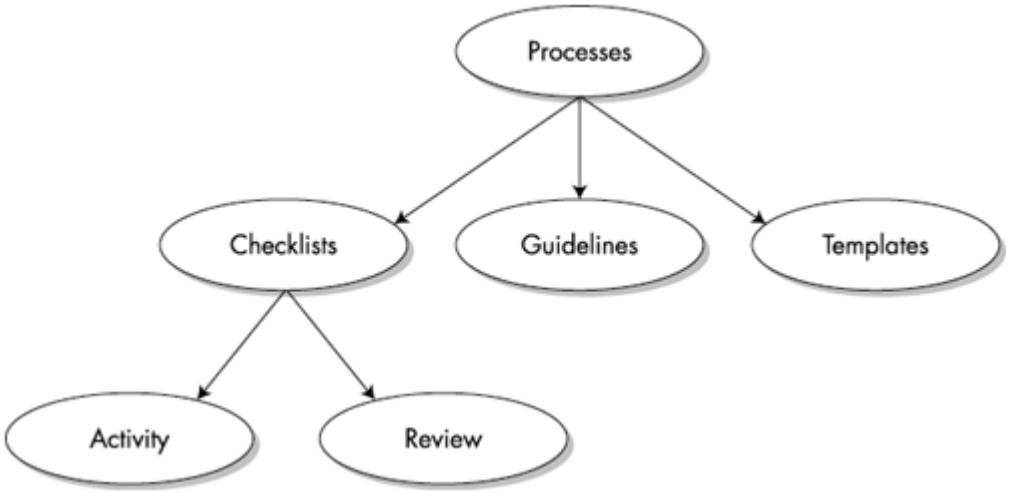
		Tỷ lệ tiêm lỗi tổng thể trong toàn bộ chu kỳ sống tính theo nỗ lực *(Overall LC defect injection rate in terms of size)*	0.00–0.1516 Lỗi/Người-giờ (trung bình: 0.052) *(0.00–0.1516 Defects/person-hour (avg: 0.052))*
		Tỷ lệ tiêm lỗi trong giai đoạn cài đặt mã *(Defect injection rate in the coding phase (in terms of effort))*	0.02–0.57 Lỗi/Người-giờ (trung bình: 0.155) *(0.02–0.57 Defects/person-hour (avg: 0.155))*
5.2	Phân phối của tiêm lỗi *(Defect injection distribution)*	Requirements and Design	Approx. 30%
		Coding	Approx. 70%
5.3	Hiệu quả loại bỏ lỗi trong suốt chu kỳ sống *(In-process Defect Removal Efficiency)*	Cho toàn bộ chu kỳ sống *(For the entire life cycle)*	78–100% (avg: 94%)
6	Chi phí của chất lượng *(Cost of Quality)*	(Review effort + rework effort + test effort + training effort) as a percentage of total effort	32%
7	Phân phối lỗi *(Defect Detection Distribution)*	% of Total Defects Min–Mean–Max	
		Req. Spec Review + HLD Review + Detail Design Review	2–13–20%

		Code review + Unit Testing	21–53–83%
		Integration Testing + System Testing	3–28–56%
		Acceptance Testing	1–6–17%

2.3 CÁC THÀNH PHẦN (ASSETS) CỦA MỘT QUY TRÌNH VÀ HỆ TRI THỨC (KNOWLEDGE SYSTEM)

Như đã thảo luận ở Chương 1, một quy trình đóng gói kinh nghiệm của một công ty dưới hình thức một "công thức nấu ăn" thành công. Tuy nhiên, mô tả quy trình thường chứa một trình tự các bước cần được thực hiện, xác định những người thực hiện chúng, nêu rõ tiêu chuẩn nhập và xuất cho các bước chính, v.v. Để tạo điều kiện thuận lợi cho việc sử dụng các quy trình, các hướng dẫn (guidelines), bản liệt kê những mục cần kiểm tra (checklists), và các bản mẫu (templates) thường cung cấp các hỗ trợ hữu ích. Tất cả các tài liệu này được gọi chung là *các thành phần của một quy trình* (*process assets*).

Hình 2.1. Các thành phần của một quy trình (Process assets)



Các hướng dẫn (guidelines) thường cung cấp các quy tắc (rules) và thủ tục (procedures) để thực hiện một bước. Ví dụ, một bước trong quá trình lập kế hoạch dự án "Ước lượng nỗ lực." Để thực hiện bước này, người quản lý dự án cần các hướng dẫn. *Bản liệt kê những mục cần kiểm tra (checklists)* thường có hai loại: *bản liệt kê những mục cần kiểm*

tra cho hoạt động (activity checklists) và *bản liệt kê những mục cần kiểm tra cho việc xem xét lại (review checklists).* Như tên gọi, một *bản liệt kê những mục cần kiểm tra cho hoạt động (activity checklist)* là một danh sách các hoạt động để cấu thành một bước của quy trình. Mục đích của *bản liệt kê những mục cần kiểm tra cho việc xem xét lại (review checklist)* là tập trung sự quan tâm của những người kiểm tra vào các lỗi thường gặp ở kết quả đầu ra (output) của quy trình. *Các bản mẫu (templates)* cung cấp các cấu trúc của các tài liệu mà tại đó kết quả của quy trình hoặc của một bước có thể đạt được. Hình 2.1 cho thấy mối quan hệ giữa quy trình và các thành phần này.

Bảng 2.6. Các hướng dẫn (guidelines), bản liệt kê những mục cần kiểm tra (checklists), và bản mẫu (templates) được dùng để quản lý dự án

Hướng dẫn (Guidelines)	**Bản liệt kê những mục cần kiểm tra (Checklists)**	**Bản mẫu (Templates/Forms)**
Hướng dẫn ước lượng nỗ lực và thời gian hoàn thành *(Effort and schedule estimation guidelines)*	Bản liệt kê những mục cần kiểm tra cho phân tích yêu cầu *(Requirements analysis checklist)*	Tài liệu đặc tả yêu cầu *(Requirements specification document)*
Thủ tục xem xét lại theo nhóm *(Group review procedure)*	Bản liệt kê những mục cần kiểm tra để lập kế hoạch kiểm thử đơn vị và kiểm thử hệ thống *(Unit test and system test plan checklists)*	Tài liệu kế hoạch kiểm thử đơn vị *(Unit test plan document)*
Hướng dẫn điều chỉnh quy trình *(Process tailoring guidelines)*	Bản liệt kê những mục cần kiểm tra cho quản lý cấu hình *(Configuration management checklist)*	Tài liệu kế hoạch kiểm thử chấp nhận *(Acceptance test plan document)*
Hướng dẫn ước lượng và giám sát lỗi *(Defect estimation and monitoring guidelines)*	Bản liệt kê những mục cần kiểm tra cho báo cáo tình trạng (trạng thái) *(Status report checklist)*	Kế hoạch quản lý dự án *(Project management plan)*
Các hướng dẫn đo và phân tích dữ liệu *(Guidelines for measurements and data*	Bản liệt kê những mục cần kiểm tra cho xem xét lại yêu cầu *(Requirement review checklist)*	Kế hoạch quản lý cấu hình *(Configuration management plan)*

analysis)		
Các hướng dẫn quản lý rủi ro *(Risk management guidelines)*	Bản liệt kê những mục cần kiểm tra cho xem xét lại thiết kế chức năng *(Functional design review checklist)*	Báo cáo phân tích số liệu được đo *(Metrics analysis report)*
Các hướng dẫn để dò theo yêu cầu *(Guidelines for requirement traceability)*	Bản liệt kê những mục cần kiểm tra cho xem xét lại kế hoạch dự án *(Project plan review checklist)*	Báo cáo tình trạng (trạng thái) tại cột mốc *(Milestone status report)*
Các hướng dẫn ngăn chặn lỗi *(Defect prevention guidelines)*	Bản liệt kê những mục cần kiểm tra cho xem xét lại mã C++ *(Code review checklist for C++)*	Báo cáo phân tích ngăn chặn lỗi *(Defect prevention analysis report)*

Mục đích chính của các thành phần này là để giúp sử dụng các quy trình dễ dàng hơn bằng cách tiết kiệm nỗ lực (chi phí), từ đó nâng cao năng suất. Ví dụ, sử dụng một bản mẫu (template) có sẵn sẽ giúp tạo ra một tài liệu sẽ dễ dàng và ít tốn thời gian hơn so với việc phải tạo ra nó từ đầu. Những thành phần này cũng giúp cải thiện chất lượng của dự án bằng cách cung cấp các hướng dẫn phù hợp; các bản liệt kê những mục cần kiểm tra cho hoạt động giúp giảm thiểu số lượng các lỗi được tiêm vào thêm; và việc hỗ trợ cho việc xem xét lại sẽ giúp sớm phát hiện lỗi.

Tóm lại, để đạt được nhiều lợi ích từ tiếp cận dùng quy trình để thực hiện dự án, những việc quan trọng cần phải làm là tạo và sử dụng các thành phần của quy trình. Ở Infosys, tất cả các hướng dẫn, bản liệt kê những mục cần kiểm tra, và các bản mẫu có sẵn trực tuyến và được cập nhật thường xuyên. Bảng 2.6 cho thấy một bản mẫu của các tài liệu liên quan đến quản lý dự án.

Ngoài những thành phần chung này (là một phần của hệ thống quản lý chất lượng ở Infosys), người quản lý dự án có thể muốn sử dụng lại một số kết quả đầu ra của một dự án tương tự (ở vài khía cạnh) trong quá khứ. Tái sử dụng các đối tượng (artifacts) đã có sẵn sẽ giúp tiết kiệm công sức và tăng năng suất. Để thúc đẩy mục tiêu này, các thành phần của quy trình của các dự án cần được thu thập khi dự án chấm dứt. Các thành phần

được thu thập và được được cung cấp cho việc tái sử dụng thông qua một hệ thống riêng biệt, bao gồm:

- Kế hoạch quản lý dự án
- Kế hoạch quản lý cấu hình
- Thời gian biểu (schedule)
- Các chuẩn, bản liệt kê những mục cần kiểm tra, hướng dẫn, bản mẫu, và các hỗ trợ khác
- Các công cụ được phát triển và các ghi chú liên quan
- Tài liệu tập huấn
- Các tài liệu khác có thể được tái sử dụng bởi các dự án trong tương lai

Mặc dù các thành phần của quy trình cố gắng đóng gói kinh nghiệm thông qua các bản liệt kê những mục cần kiểm tra, bản mẫu, v.v., nhưng chúng không thể luôn luôn ghi nhận được hết các hình thức đa dạng của kiến thức có thể đạt được trong việc thực hiện dự án. Việc ghi nhận và tái sử dụng các hình thức khác nhau của kiến thức đòi hỏi phải quản lý kiến thức đúng cách thức – đây là điều rất quan trọng trong các công ty dựa trên tri thức (knowledge-based organizations) như: các nhà cung cấp giải pháp (solution providers) và các công ty tư vấn (consulting companies). Nhiều công ty đã phát triển các hệ thống để dùng lại một cách hiệu quả kinh nghiệm và kiến thức nhân viên của họ. Ở Infosys, bên cạnh cơ sở dữ liệu quy trình và các thành phần của quy trình còn có một hệ thống được gọi là khung của kiến thức (BOK - body of knowledge) được sử dụng để đóng gói kinh nghiệm.

Hệ thống BOK dùng Web có tiện ích tìm kiếm theo khóa riêng hoặc theo tên tác giả. Kiến thức trong BOK, mà chủ yếu là trong các bài viết (articles), được tổ chức theo các chủ đề. Các chủ đề chính bao gồm:

- Máy tính và các dịch vụ truyền thông (Computer and communication services)
- Đặc tả yêu cầu (Requirements specification)
- Xây dựng (Build)

- Các công cụ (Tools)
- Các phương pháp và kỹ thuật (Methodologies and techniques)
- Giáo dục và nghiên cứu (Education and research)
- Thiết kế (Design)
- Xem xét lại, thanh tra, và kiểm thử (Reviews, inspection, and testing)
- Đảm bảo chất lượng và năng suất (Quality assurance and productivity)
- Quản lý dự án (Project management)

Hệ thống BOK đăng các bài viết liên quan đến những bài học kinh nghiệm và thực hành tốt nhất. Các mục thì khá tổng quát và không gần sát với một dự án cụ thể. Sử dụng một bản mẫu (template) đã được xây dựng cho mục đích này, bất kỳ thành viên nào của công ty cũng có thể gửi một mục để đưa vào BOK. Mỗi bày viết trước khi được đăng phải trải qua quá trình xem xét lại (review), trong đó tập trung vào tính hữu dụng, tính tổng quát, được thay đổi nếu cần thiết, và các đặc điểm khác. Việc kiểm soát quá trình biên tập luôn được duy trì để đảm bảo rằng các mục đáp ứng được các tiêu chuẩn chất lượng. Các hỗ trợ về tài chính đã được thực hiện để động viên các nhân viên gửi thông tin đến BOK, và các bộ phận quản lý BOK tích cực theo đuổi các bài viết mới.

2.4 TÓM TẮT

Người quản lý dự án có thể lập kế hoạch dự án tốt hơn nếu sử dụng lại kinh nghiệm từ các dự án trong quá khứ. Cơ sở hạ tầng để lập kế hoạch cho dự án có thể giúp thu thập dữ liệu và các bài học một cách hiệu quả, và cũng giúp những người quản lý dự án có thể sử dụng lại chúng. Các phần tử chính của cơ sở hạ tầng để lập kế hoạch tại Infosys là cơ sở dữ liệu quy trình, baseline về khả năng của quy trình (khả năng cơ sở của quy trình), và các thành phần của quy trình. Những phần tử này có các đặc điểm chính sau đây:

- Một CSDLQT chứa các dữ liệu hiệu suất của các dự án đã hoàn thành. Nó chứa dữ liệu về rủi ro, nỗ lực và sự phân phối nỗ lực, lỗi và sự phân phối lỗi, kích thước, và các đặc trưng khác của dự án.

- Baseline về khả năng của quy trình tóm tắt hiệu suất quy trình trong suốt thời gian thực hiện các dự án, từ đó xác định phạm vi của các kết quả dự kiến nếu các quy trình này được áp dụng. Nó bao gồm các số đo như chất lượng, năng suất, hiệu quả loại bỏ lỗi, và sự phân phối lỗi và nỗ lực.

- Các thành phần của quy trình là các tài liệu (documents) như các bản liệt kê những mục cần kiểm tra (checklists), các bản mẫu (templates), các phương pháp (methodologies), và các hướng dẫn (guidelines). Chúng giúp nâng cao năng suất bằng cách giảm nỗ lực cần thiết để thực hiện một số công việc, và chúng nâng cao chất lượng bằng cách sớm phát hiện và giảm lỗi.

Nếu những phần tử này chưa có trong công ty, những người quản lý dự án có thể xây dựng các hình thức giới hạn (limited forms) của một vài trong số các phần tử này. Vài người quản lý dự án đã cùng nhau thu thập kinh nghiệm của họ để có thể xây dựng một CSDLQT giới hạn (limited process database). Vấn đề phát sinh ở đây là thực hiện phân tích các dự án đã kết thúc (được thảo luận trong Chương 12). Khi bạn đã tạo được một CSDLQT giói hạn, bạn cũng có thể trích lọc ra một số khía cạnh của CSDLQT. Việc xây dựng một tập hợp các thành phần của quy trình thậm chí còn dễ dàng hơn; bạn chỉ đơn giản là thu thập các bản mẫu, bản liệt kê những mục cần kiểm tra, và các tài liệu tương tự khác được sử dụng trong dự án. Sau đó, bạn nên xem xét lại, tinh chỉnh lại và làm cho chúng sẵn sàng để được dùng.

Những phần tử về cơ sở hạ tầng này không trực tiếp đáp ứng các yêu cầu của các KPAs có liên quan đến quản lý dự án của CMM. Tuy nhiên, chúng cần thiết để đáp ứng nhiều KPAs có liên quan đến quản lý ở CMM mức 3 và 4. Các hệ thống được thảo luận trong chương này cũng đáp ứng các yêu cầu của một số KPAs khác của CMM. Ví dụ, sự tồn tại của một cơ sở dữ liệu quy trình, được cần cho CMM mức 3, để quản lý các thành phần của quy trình. Baseline về khả năng của quy trình (process capability baseline) thì cần thiết cho cả hai KPAs ở CMM mức 4.

2.5 CÁC THAM KHẢO

1. P. Jalote. *Use of Metrics in High Maturity Organizations*, SEPG2K Conference, Seattle, 2000.

2. W. Humphrey. *Managing the Software Process*. Addison-Wesley, 1989.

3. C. Jones. *Applied Software Measurement: Assuring Productivity and Quality*, second edition. McGraw Hill, 1996.

4. R. Grady and D. Caswell. *Software Metrics: Establishing a Company-wide Program*. Prentice Hall, 1987.

5. Carnegie Mellon/Software Engineering Institute. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley, 1995.

Chương 3. Lập kế hoạch cho quy trình

Ravi, giống như nhiều người quản lý dự án, đã nghiên cứu mô hình thác nước (waterdall model) của phát triển phần mềm như quy trình vòng đời phần mềm (software life-cycle process) chính. Ông đã làm tất cả các thiết lập cần thiết để sử dụng nó cho dự án sắp tới (nhiệm vụ đầu tiên được phân công cho ông ta). Tuy nhiên, Ravi thấy rằng mô hình thác nước không thể được sử dụng bởi vì khách hàng muốn phần mềm được gửi giao theo từng giai đoạn, điều này có nghĩa là hệ thống sẽ được chuyển giao theo từng phần chứ không phải giao toàn bộ.

Tình hình trong nhiều dự án khác cũng tương tự như vậy. Thế giới thực hiếm khi xuất hiện một vấn đề mà một quy trình chuẩn, hoặc một quy trình đã được sử dụng trong một dự án trước đó, có thể được sử dụng lại cho nó một cách tốt nhất. Để đạt được sự phù hợp nhất, quy trình hiện tại phải được thay đổi cho phù hợp với vấn đề mới.

Chương này mô tả quy trình phát triển (development process) ở Infosys và thay đổi quy trình (process tailoring). Một quy trình phát triển, ngay cả sau khi bị thay đổi, thường không thể quản lý các yêu cầu thay đổi (change requests). Để đáp ứng các yêu cầu thay đổi mà không làm mất kiểm soát dự án, bạn phải bổ sung vào quy trình phát triển một quy trình quản lý thay đổi yêu cầu (requirement change management process). Chương này cũng mô tả quy trình quản lý thay đổi yêu cầu tại Infosys.

3.1 QUY TRÌNH PHÁT TRIỂN Ở INFOSYS

Trong quá trình lập kế hoạch cho dự án, người quản lý dự án phải quyết định quy trình nào được sử dụng để phát triển phần mềm. Đây là vấn đề rất quan trọng bởi vì phần lớn các hoạt động phát triển sẽ bị chi phối bởi quyết định này. Nó giống như khi bạn đang xe trong một hành trình - các tuyến đường đã được hoạch định trước sẽ xác định hướng đi mà bạn sẽ tiếp tục lái xe đi theo.

Có vài mô hình quy trình để phát triển phần mềm đang tồn tại. Những mô hình phổ biến nhất bao gồm: mô hình thác nước (waterfall model) [1], tăng trưởng theo vòng lặp (iterative enhancement) [2], tạo mẫu (prototyping) [3], và xoắn ốc (spiral) [4]. Trong đó, mô hình thác nước được sử dụng rộng rãi nhất, nó tổ chức các giai đoạn theo một trình tự

tuyến tính, mặc dù hầu hết các cài đặt (implementations) của mô hình này thường được điều chỉnh để giảm thiểu các thiếu sót của nó.

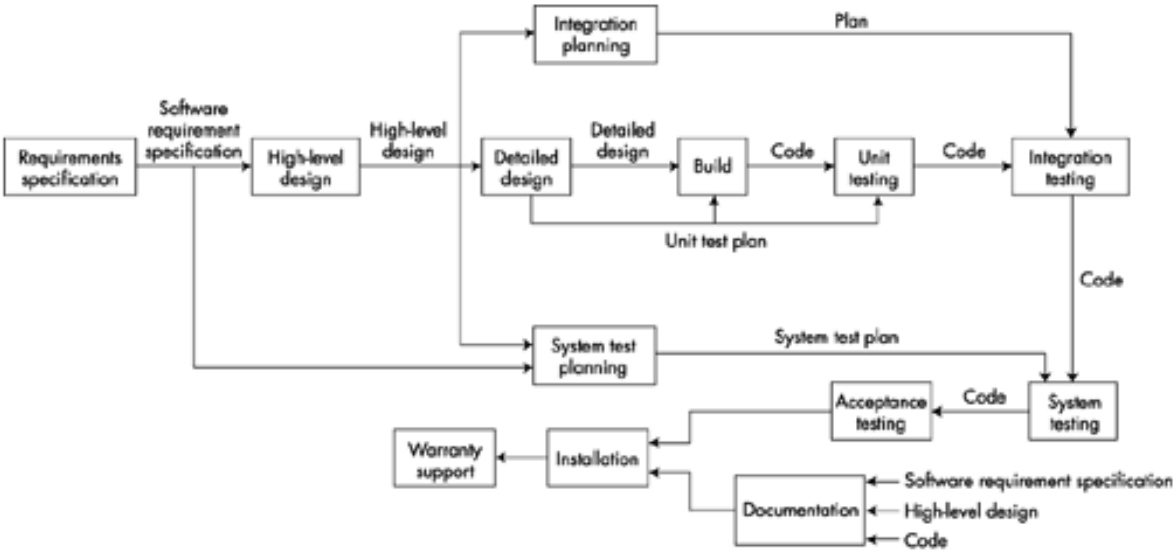
Ở mức độ vĩ mô, một quy trình chuẩn (standard process) có thể tổ chức một cách tối ưu các giai đoạn cho một lớp (loại) các dự án và làm điểm khởi đầu tốt để định nghĩa các quy trình mới. Tuy nhiên, một quy trình chuẩn không thể phù hợp cho tất cả các tình huống; quy trình tốt nhất có thể là một biến thể của một quy trình chuẩn. Do đó, để quyết định sử dụng quy trình nào, người quản lý dự án phải chọn một quy trình cơ sở (base process) và sau đó quyết định nó phải được thay đổi như thế nào cho phù hợp với một dự án cụ thể. Phần còn lại của phần này thảo luận về quy trình phát triển chuẩn đã được sử dụng tại Infosys và giải thích nó được thay đổi như thế nào bởi những người quản lý dự án.

3.1.1 Quy trình chuẩn (Standard process)

Quy trình phát triển chuẩn (standard development process) được sử dụng tại Infosys tương tự như mô hình thác nước, mặc dù các giai đoạn truyền thống đã bị phân chia thành các giai đoạn nhỏ hơn, để cho phép thực hiện song song ở vài giai đoạn. Ví dụ, lập kế hoạch để kiểm thử hệ thống (system testing) được thực hiện như một giai đoạn riêng biệt, đây là cách để cho phép các nhóm dự án có thể lập kế hoạch để thực hiện giai đoạn kiểm thử song song với giai đoạn cài đặt mã (coding), mặc dù theo truyền thống việc kiểm thử hệ thống chỉ diễn ra sau khi cài đặt mã xong.

Các giai đoạn trong quy trình bao gồm: phân tích yêu cầu (requirements analysis), thiết kế ở mức cao (high-level design), thiết kế chi tiết (detailed design), xây dựng (build), kiểm thử đơn vị (unit testing), lập kế hoạch tích hợp (integration planning), tích hợp (integration), lập kế hoạch kiểm thử hệ thống (system test planning), kiểm thử hệ thống (system testing), lập tài liệu (documentation), chấp nhận (acceptance) và cài đặt (installation), và hỗ trợ bảo hành (warranty support). Hình 3.1 mô tả các giai đoạn và sự phụ thuộc giữa chúng. Thông tin chi tiết về các giai đoạn không được mô tả ở đây (Mô tả chi tiết được cho trong [5]).

Hình 3.1. Quy trình phát triển ở Infosys



Mô tả hình thức (formal description) của quy trình này xác định cụ thể tiêu chuẩn nhập, xuất (entry and exit criteria), đầu vào và đầu ra (inputs and outputs), những người tham gia (participants), các hoạt động (activities), và các thông tin khác cho từng giai đoạn (stage). Các mô tả quy trình thì thường ngắn gọn, xác định danh sách các hoạt động được thực hiện trong giai đoạn đó.

Quy trình tổng thể vẫn giữ nguyên ngay cả khi áp dụng cho một dự án dùng tiếp cận hướng đối tượng, dù cho có một số các giai đoạn được thực hiện khác đi trong dự án như vậy. Sự khác biệt nằm chủ yếu trong giai đoạn phân tích và thiết kế, mặc dù các hướng dẫn và các chuẩn của các giai đoạn sau đó cũng khác đi.

Quy trình cơ sở này cũng được sử dụng cho các dự án phát triển theo vòng lặp (iterative development) hoặc tạo mẫu (prototyping) hoặc chỉ thực hiện một số giai đoạn của chu kỳ sống. Trong những trường hợp này, quy trình chuẩn này sẽ được điều chỉnh cho phù hợp với dự án.

3.1.2 Điều chỉnh quy trình (process tailoring)

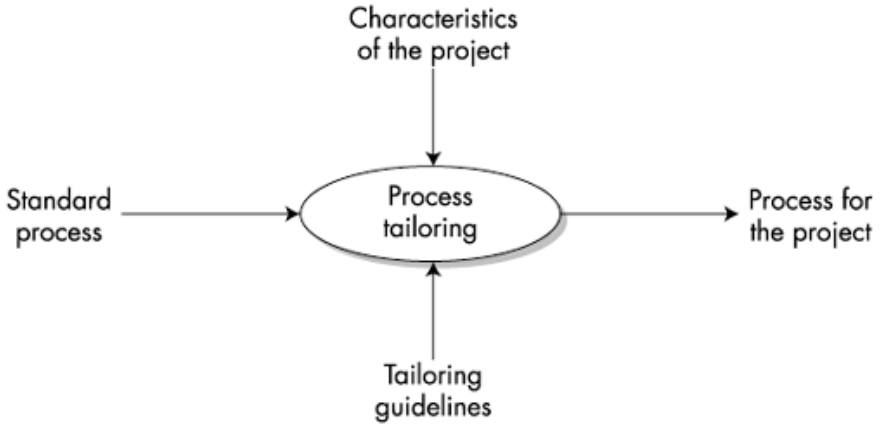
Không quy trình nào đã được định nghĩa trước - dù là một quy trình chuẩn hoặc quy trình đã được sử dụng trong một dự án trước đó - sẽ được áp dụng tốt cho tất cả các tình

huống và tất cả các dự án. Một quy trình có sẵn phải được điều chỉnh lại cho phù hợp với các nhu cầu của dự án hiện tại.

Việc điều chỉnh (tailoring) một quy trình (đã được định nghĩa trước đây) là để có được một quy trình mới phù hợp cho một công ty cụ thể hoặc các nhu cầu kỹ thuật riêng của một dự án [5][6]. Bạn có thể xem việc điều chỉnh như là thêm, xóa, hoặc sửa đổi các hoạt động của một quy trình để mà quy kết quả phù hợp hơn để đạt được các mục tiêu của dự án.

Để cho phép tái sử dụng (reuse) một cách hiệu quả các quy trình đã được định nghĩa trước đó, các hướng dẫn về cách điều chỉnh (tailoring guidelines) sẽ được cung cấp. Các hướng dẫn này xác định các điều kiện và các loại thay đổi phải được thực hiện lên quy trình chuẩn. Về bản chất, chúng xác định một tập hợp các *chênh lệch được phép* (*permitted deviations*) so với quy trình chuẩn với hy vọng rằng một quy trình tối ưu mới có thể được định nghĩa cho dự án. Hình 3.2 minh họa vai trò của các hướng dẫn về cách điều chỉnh.

Hình 3.2. Điều chỉnh quy trình



Để minh họa về nhu cầu điều chỉnh, chúng ta hãy lấy một hoạt động trong giai đoạn xây dựng của quy trình phát triển – *thực hiệu xem xét lại mã (Do code review).* Xem xét lại mã có thể mang lại nhiều giá trị trong nhiều trường hợp, nhưng đôi khi giá trị thu được từ nó lại không tương xứng với nỗ lực (chi phí) phải bỏ ra. Ngoài ra, xem xét lại có thể được thực hiện bởi một nhóm (thực hiện theo các thủ tục xem xét lại của nhóm) hoặc bởi một người duy nhất. Quy trình phát triển chuẩn không chỉ ra việc xem xét lại mã nên được thực hiện như thế nào. Các hướng dẫn về cách điều chỉnh có thể giúp người quản lý dự án bằng cách tư vấn rằng hoạt động *xem xét lại mã* chỉ được thực hiện cho một số loại

chương trình cụ thể (chẳng hạn như các chương trình phức tạp hoặc các giao diện bên ngoài/external interfaces) và bằng cách đề nghị một hình thức tối ưu cho hoạt động xem xét lại (nhóm xem xét lại hoặc chỉ một người xem xét lại).

Phương pháp điều chỉnh ở Infosys thì tương tự như phương pháp dùng bảng của Ginsberg và Quinn [7], ở đó người quản lý dự án xác định cụ thể các phần tử của quy trình, thuộc tính sẽ được điều chỉnh, các tùy chọn cho mỗi thuộc tính, và các lý do để lựa chọn một tùy chọn cụ thể. Một người quản lý dự án thực hiện điều chỉnh ở hai cấp độ: tóm tắt và chi tiết.

Điều chỉnh ở cấp độ tóm tắt (Summary-Level Tailoring)

Trong điều chỉnh ở cấp độ tóm tắt, tùy thuộc vào các đặc trưng của dự án (project characteristics), người quản lý dự án sẽ áp dụng các hướng dẫn để điều chỉnh quy trình chuẩn. Nghĩa là, nó cung cấp một số quy tắc chung liên quan đến một số loại hoạt động chi tiết. Để thực hiện bước này, người quản lý dự án đầu tiên xác định một số đặc trưng của dự án. Đối với các dự án phát triển (development projects), những đặc trưng sau đây sẽ được sử dụng để điều chỉnh:

- Mức độ kinh nghiệm và kỹ năng của nhóm và người quản lý dự án
- Kích thước nhóm lúc cao điểm (Peak team size)
- Sự rõ ràng của yêu cầu (requirements)
- Thời gian thực hiện dự án
- Mức độ quan trọng của ứng dụng (application criticality)

Mức độ kinh nghiệm của nhóm được xem là cao nếu phần lớn các thành viên trong nhóm có nhiều hơn hai năm kinh nghiệm làm việc với công nghệ đang được triển khai trong dự án mới; nếu không, nó được coi là thấp. Mức độ quan trọng của ứng dụng được xem là cao nếu tác động của ứng dụng lên kinh doanh của khách hàng hoặc kinh doanh của Infosys là lớn; nếu không, nó là thấp. Thời gian thực hiện dự án được xem là đặc biệt ngắn nếu dự án phải được giao trong vòng ba tháng.

Các hướng dẫn để thực hiện điều chỉnh ở cấp độ tóm tắt (summary tailoring guidelines) được cung cấp cho các giá trị khác nhau của những đặc trưng này. Các hướng dẫn tóm tắt thường có liên quan đến xem xét lại (review-related), liên quan đến nỗ lực (effort-related), liên quan đến thời gian biểu (schedule-related), liên quan đến nhân sự (resources-related), hoặc liên quan đến hình thức (formality-related). Các hướng dẫn liên quan đến xem xét lại xác định khi nào xem xét lại nên được thực hiện và kiểu xem xét lại nào nên được thực hiện. Tương tự như vậy, các hướng dẫn liên quan đến nỗ lực cho biết các bước thực hiện dự án có thể ảnh hưởng đến nỗ lực (chi phí). Những hướng dẫn chung này thiết lập bối cảnh để điều chỉnh quy trình một cách chi tiết và định nghĩa một quy trình mới thích hợp cho dự án.

Điều chỉnh ở cấp độ chi tiết (Detailed Tailoring)

Điều chỉnh ở cấp độ chi tiết bao gồm việc thực hiện các hoạt động, xem xét lại chúng, và lập tài liệu cho chúng. Các hướng dẫn cách điều chỉnh có thể xác định một hoạt động như tùy chọn, trong trường hợp người quản lý dự án có thể quyết định có cần thực hiện một hoạt động nào đó không. Tương tự như vậy, chuẩn bị một số tài liệu có thể là một tùy chọn, trong trường hợp người quản lý dự án quyết định xem dự án có cần được lập tài liệu không. Đối với việc xem xét lại, các tùy chọn thường gặp là: T*hực hiệu xem xét lại bởi nhóm (Do group review), Thực hiệu xem xét lại bởi một người (Do one-person review),* hoặc *Không xem xét lại (Do not review)*. Ngoài ra, người quản lý dự án có thể thêm một số hoạt động mới hoặc có thể lặp lại một số hoạt động.

Khi việc điều chỉnh chi tiết đã hoàn tất, một trình tự các hoạt động sẽ được thực hiện trong quy trình của dự án sẽ được định nghĩa. Những định nghĩa này sau đó được sử dụng để lập kế hoạch và xếp thời gian hoàn thành các hoạt động và hình thành cơ sở của thực hiện dự án. Việc điều chỉnh được nhấn mạnh trong kế hoạch dự án, vì vậy việc định nghĩa hoặc điều chỉnh quy trình cũng được xem xét lại khi xem xét lại kế hoạch.

3.1.3 Ví dụ: Điều chỉnh cho các dự án có thời gian thực hiện ngắn (short-duration projects)

Chúng tôi sẽ minh họa các khái niệm bằng cách trình bày cách điều chỉnh ở cấp độ tóm tắt cho các dự án có thời gian thực hiện ngắn. Infosys quan sát thấy rằng thời gian thực hiện các dự án phần mềm đang giảm trong những năm qua, và có nhu cầu gia tăng số

lượng các dự án có thời gian thực hiện rất ngắn. Rõ ràng, những dự án như thế cần các quy trình được điều chỉnh để tối đa hóa số công việc được thực hiện song song, cũng như giám sát và kiểm soát dự án một cách rất chặt chẽ. Việc điều chỉnh quy trình phụ thuộc vào sự rõ ràng của yêu cầu, mức độ kinh nghiệm của nhóm hoặc người lãnh đạo dự án, kích thước nhóm, v.v. Bảng 3.1 cho thấy các hướng dẫn cách điều chỉnh đã được chuẩn bị trước bởi những người quản lý dự án của các dự án như vậy.

Bởi vì thời gian hoàn thành ngắn là đặc trưng chính, các hướng dẫn liên quan đến thời gian biểu đã đề nghị sử dụng một kỹ thuật được gọi là *timeboxing (chia hộp thời gian)* và *mini-milestones (các sự kiện/cột mốc nhỏ).* Trong timeboxing, các chu kỳ thời gian kéo dài vài tuần được lên kế hoạch, và một hệ thống vận hành được (sản phẩm phần mềm) được giao cho khách hàng sau mỗi chu kỳ này. Để luôn luôn kiểm soát chặt chẽ các timebox, *nhiều cột mốc nhỏ (mini-milestones)* trong từng chu kỳ đã được thiết lập. Tại các cột mốc nhỏ (mini-milestones), nhóm và người quản lý dự án cùng nhau phân tích các vấn đề phát sinh quan trọng (xem Chương 11).

Người quản lý dự án – người phải thực hiện một dự án trong thời gian ngắn - có thể sử dụng các hướng dẫn để điều chỉnh quy trình phát triển. Ví dụ, người quản lý dự án có thể lập kế hoạch để lặp đi lặp lại việc giao sản phẩm sau các vòng lặp ngắn, thực hiện phân tích các vấn đề phát sinh tại các cột mốc (milestone) nhưng diễn ra thường xuyên hơn, xác định một quy trình để quản lý các thay đổi bằng cách trì hoãn các yêu cầu thay đổi nhiều nhất sang vòng lặp kế tiếp (tức là sẽ phát triển chúng ở vòng lặp kế tiếp), v.v. Ngoài ra, trong khi thực hiện điều chỉnh để tạo ra một quy trình mới, người quản lý dự án có thể giảm số tài liệu cho mỗi chu kỳ, ví dụ, giảm phạm vi (scope) của các bản mẫu (templates) - được sử dụng để báo cáo tình trạng.

Các hướng dẫn tóm tắt (summary guidelines) được trình bày trong Bảng 3.1 cho dự án có thời gian ít hơn ba tháng, kích thước nhóm lúc cao điểm có năm người, đang làm việc với công nghệ mới, và có một người lãnh đạo dự án có nhiều kinh nghiệm. Trong suốt quá trình điều chỉnh chi tiết, người quản lý dự án lựa chọn các tùy chọn dựa trên những hướng dẫn tóm tắt và phù hợp với dự án. Tuy nhiên, nếu các hướng dẫn này không đủ để cho phép bạn chọn lựa một quy trình thích hợp cho dự án, bạn có thể phải thay đổi quy trình vượt ra ngoài những gì đã được cho phép bởi các hướng dẫn này. Độ lệch như vậy có thể làm xuất hiện các rủi ro tiềm ẩn và do đó cần được đánh dấu (highlighted) trong kế hoạch quản lý dự án, sẽ được xem xét và phê duyệt.

Bảng 3.1. Các hướng dẫn để điều chỉnh cho các dự án có thời gian thực hiện ngắn		
Các giá trị (Values)	**Các hướng dẫn tóm tắt (Summary Guidelines)**	**Tại sao?**
Kích thước của nhóm >= 5, thời gian (duration) <3 tháng, yêu cầu không rõ ràng, công nghệ mới, người lãnh đạo dự án có nhiều kinh nghiệm	**Liên quan đến thời gian biểu (Scheduling-related)**	
	- Kế hoạch cho các cột mốc nhỏ (mini-milestones)	Xác định các vấn đề từ rất sớm và giúp kiểm soát dự án tốt hơn. Rủi ro liên quan đến thời gian biểu (Schedule-related risk) được giảm vì các sự việc có thể được hiển thị (visibility) trên thời gian biểu để được nhìn thấy thường xuyên hơn. Nỗ lực (chi phí) bị trượt có thể sớm được kiểm soát trong dự án.
	- Cố gắng để tạo các timebox cho các phần sản phẩm được giao (deliverables)	Thời gian biểu (schedule) có khả năng được rút ngắn vì nó đảm bảo hệ thống sẽ được chuyển giao cho khách hàng trong khoảng thời gian đã được ấn định và không khuyến khích bổ sung thêm yêu cầu chức năng cho sản phẩm. Rủi ro liên quan đến thời gian biểu được giảm.
	Liên quan đến nỗ lực (Effort-related)	
	- Áp dụng các yếu tố điều chỉnh (adjustment factors) thích hợp để ước lượng baseline nỗ lực (base effort) như đã được định nghĩa trong hướng dẫn ước lượng/dự toán	Tất cả các ràng buộc (constraints) của dự án được cung cấp cho việc ước lượng nỗ lực để kết quả ước lượng gần với thời gian thực tế của dự án.

	(estimation guidelines).	
	- Làm các ước lượng tính: lạc quan (optimistic), bi quan (pessimistic), và có thể xảy ra (probable).	Giúp giảm rủi ro liên quan đến ước lượng/dự toán không chính xác và cam kết "nhanh chóng" (cẩu thả).
	- Sửa đổi (revise) và đàm phán lại (renegotiate) ước lượng vào cuối của giai đoạn thiết (design stage).	
	Liên quan đến việc sử dụng nhân sự (Resource allocation-related)	
	- Phân công nhân sự có tay nghề cao (skilled) cho các nhiệm vụ. - Nhận trợ giúp của chuyên gia (expert) về các vấn đề liên quan đến hiệu suất (performance- related issues).	Năng suất sẽ được cải thiện. Rủi ro liên quan đến thời gian biểu giảm. Làm lại (rework) và chậm trễ (delays) do điều chỉnh hiệu suất được giảm thiểu.
	Liên quan đến hình thức (formality-related)	
	- Xác định tất cả các bên liên quan (stakeholders) của dự án, mục tiêu của họ (objectives), và xếp độ ưu tiên họ (ai là người quan trọng hơn).	Giúp tập trung vào các mục tiêu rõ ràng hơn. Giảm rủi ro liên quan đến thời gian biểu bởi vì các yếu tố thành công quan trọng đã được xác định, được đồng ý và được giải quyết.
	- Thường xuyên báo cáo tình trạng của các hoạt động giảm	Rủi ro liên quan đến thời gian biểu được giảm bởi vì của các hoạt động

	thiểu rủi ro (risk mitigation activities) đến người quản lý cấp cao (senior management).	giảm thiểu rủi ro được thực hiện một cách chủ động.
	- Xác định quy trình quản lý các thay đổi yêu cầu và đào tạo nhóm và nghiêm chỉnh thực hiện theo nó.	Giảm thiểu tác động của thay đổi yêu cầu lên tiến độ dự án.

3.2 QUẢN LÝ THAY ĐỔI YÊU CẦU

Những thay đổi yêu cầu có thể đến bất cứ lúc nào trong suốt chu kỳ sống của của một dự án (hoặc thậm chí sau đó). Việc thay đổi yêu cầu diễn ra trong suốt chu kỳ sống có thể tác động xấu đến dự án. Thay vì mong muốn rằng các thay đổi sẽ không xảy ra hoặc hy vọng rằng bằng cách nào đó các yêu cầu ban đầu sẽ là "rất tốt" để mà không có thay đổi nào sẽ được làm, giải pháp tốt hơn có lẻ là chuẩn bị để xử lý các thay đổi yêu cầu khi chúng xảy ra. Các thay đổi không kiểm soát được có thể có ảnh hưởng xấu đến chi phí (cost), tiến độ (schedule) và chất lượng (quality) của dự án. Thay đổi yêu cầu có thể chiếm tới 40% tổng số chi phí [8].

Ravi nhận ra sự cần thiết để có một quy trình quản lý thay đổi (change management process). Trong một nỗ lực để làm hài lòng khách hàng, ông sẵn sàng đồng ý để làm những thay đổi theo yêu cầu của khách hàng. Vì không có kiểm soát các thay đổi yêu cầu, dự án đã kéo dài thời gian thực hiện. Cuối cùng, sau nhiều tuần, các thành viên của nhóm phải làm việc 60-giờ/tuần, nhóm đã sử dụng số nỗ lực nhiều hơn 100% so với ước lượng ban đầu, Ravi đã hoàn thành dự án. Nhưng điều tồi tệ hơn là ở chỗ: ông thấy rằng khách hàng vẫn không hài lòng bởi vì trước đó ông đã nghĩ rằng những thay đổi sẽ làm cho sản phẩm tốt hơn.

Trong một dự án khác, Mary, một người quản lý dự án dày dạn kinh nghiệm, đã ghi lại từng thay đổi yêu cầu mà khách hàng của cô ta đã được thực hiện qua điện thoại hoặc e-mail và thông báo cho khách hàng về tác động của các thay đổi này đến nỗ lực (chi phí) và tiến độ. Không chỉ tần số yêu cầu thay đổi giảm theo thời gian, mà khách hàng cũng chuyển nhiều yêu cầu thay đổi sang lần giao sản phẩm tiếp theo (next release). Kết quả

cuối cùng? Dự án đã được hoàn thành trong thời gian dự kiến, và khách hàng hài lòng khi trả thêm tiền cho phần nỗ lực bổ sung/chi phí bổ sung (extra effort) để thực hiện (cài đặt) các phát sinh trong việc thực hiện (cài đặt) các yêu cầu thay đổi.

Phần này thảo luận về quy trình quản lý các thay đổi yêu cầu (requirement change management process) đã được sử dụng tại Infosys. Quy trình này xác định tập hợp các hoạt động sẽ được thực hiện khi có yêu cầu mới hoặc thay đổi lên yêu cầu hiện có (chúng tôi sẽ gọi chung là: những thay đổi trong yêu cầu). Với kinh nghiệm của những người quản lý dự án như Ravi và Mary, quy trình quản lý thay đổi thường được sử dụng bởi những người quản lý dự án và xác định nó như là một phần của kế hoạch quản lý dự án, được sử dụng chính yếu như là phương tiện để làm sáng tỏ các ảnh hưởng của thay đổi đến khách hàng.

3.2.1 Quy trình quản lý thay đổi (Change Management Process)

Trong quá trình lập kế hoạch cho dự án, người quản lý dự án sẽ quyết định quy trình nào sẽ được sử dụng để xử lý yêu cầu thay đổi. Quy trình được hoạch định sẽ được thảo luận với khách hàng để cho cả khách hàng và nhà cung cấp có một thỏa thuận về việc làm thế nào để quản lý các thay đổi. Nói chung, quy trình xác định các yêu cầu thay đổi sẽ được thực hiện như thế nào, khi nào chấp thuận chính thức xảy ra, v.v. Khi có một yêu cầu cho một sự thay đổi yêu cầu, quy trình quản lý các thay đổi yêu cầu phải được thực thi.

Bởi vì yêu cầu thay đổi có tác động đến chi phí, nên cần thiết để có một thỏa thuận rõ ràng về thanh toán. Thường xuyên, với sự chấp thuận của khách hàng, người quản lý dự án xây dựng một vùng đệm (buffer) cho phần nỗ lực/chi phí được ước lượng bổ sung để thực hiện các yêu cầu thay đổi (thường chiếm một tỷ lệ phần trăm (%) nhỏ của tổng nỗ lực (chi phí) của dự án). Một dự liệu ngân sách như vậy giúp làm đơn giản hóa các khía cạnh hành chính của việc thực hiện các yêu cầu thay đổi đã được phê duyệt.

Quy trình quản lý các thay đổi ở Infosys có các bước sau.

1. Ghi nhận các thay đổi (Log the changes)
2. Phân tích tác động lên các sản phẩm công việc (work products).
3. Ước lượng nỗ lực cần thiết cho các yêu cầu thay đổi.

4. Ước lượng lại thời gian giao hàng.

5. Thực hiện phân tích tác động lên chi phí tích lũy (cumulative cost impact analysis).

6. Xem xét lại các tác động với người quản lý cấp cao nếu các quá ngưỡng bị vượt qua.

7. Được khách hàng ký (sign-off).

8. Làm lại (rework) các sản phẩm công việc.

Bạn lưu giữ một nhật ký (bản ghi) để theo dõi các yêu cầu thay đổi. Mỗi mục trong nhật ký chứa một số yêu cầu thay đổi, mô tả ngắn gọn về sự thay đổi, ảnh hưởng của thay đổi, tình trạng của các yêu cầu thay đổi, và các ngày quan trọng. Bạn đánh giá ảnh hưởng của một yêu cầu thay đổi bằng cách thực hiện phân tích các tác động. Phân tích các tác động liên quan đến việc xác định các sản phẩm công việc cần phải được thay đổi và đánh giá khối lượng của thay đổi; đánh giá lại rủi ro của dự án bằng cách xem xét lại kế hoạch quản lý rủi ro; và đánh giá các tác động tổng thể của những thay đổi lên nỗ lực (chi phí) và ước lượng thời gian hoàn thành. Kết quả của phân tích được xem xét và được chấp thuận bởi khách hàng. Các yêu cầu thay đổi được kết hợp vào trong các tài liệu đặc tả yêu cầu, thường là các phụ lục. Đôi khi các phần liên quan của tài liệu cũng được sửa đổi để phản ánh các thay đổi. Giám sát các yêu cầu thay đổi đã được phê duyệt và đảm bảo quá trình thực hiện (cài đặt) chúng được quản lý theo đúng quy trình quản lý cấu hình (được thảo luận trong Chương 9).

Một thay đổi có thể được phân loại là nhỏ nếu tổng nỗ lực (chi phí) liên quan đến việc thực hiện (cài đặt) nó không vượt quá một giá trị đã được xác định trước – ví dụ, hai người-ngày (two person-days). Các thay đổi nhỏ thường trở thành một phần của nỗ lực của dự án, sử dụng vùng đệm (buffer) trong ước lượng/dự toán đã được làm trước đây. Các thay đổi lớn thường có một tác động lớn hơn đến nỗ lực và thời gian hoàn thành và phải được chính thức phê duyệt bởi khách hàng. Người quản lý cấp cao tăng khả năng nhìn thấy các thay đổi thông qua các báo cáo tình trạng và báo cáo tại các cột mốc (milestone) (được thảo luận trong Chương 11).

3.2.2 Các ví dụ

Để xác định các thay đổi và đầu ra của quy trình quản lý thay đổi, các dự án thường sử dụng một bản mẫu đơn giản (simple template). Mỗi thay đổi được gán một *số tham chiếu (reference number)* duy nhất, được xác định bằng trường *số của yêu cầu (request number)* trong bản mẫu (template). *Trường đặc tả các thay đổi (change specification field)* cung cấp một mô tả ngắn gọn về thay đổi được yêu cầu. Các loại thay đổi, ví dụ: thay đổi thiết kế (design change), thay đổi hợp đồng (contract change), thay đổi chức năng (functionality change), thay đổi hiệu suất (performance change), được xác định bằng trường *loại thay đổi (change category*). Tóm tắt của *phân tích tác động (impact analysis)* được ghi lại, cùng với thông tin ngắn gọn về các sản phẩm công việc (work products) sẽ bị ảnh hưởng, nỗ lực (chi phí) được cần, và các tác động đối với thời gian hoàn thành. Trạng thái của yêu cầu thay đổi - những gì đang được thực hiện cho nó - được ghi lại trong trường *tình trạng (status field)*. *Ngày (date)* của yêu cầu thay đổi có thể cũng được ghi lại, cùng với ngày thay đổi đã được phê duyệt, nếu sự phê duyệt cần thiết. Nói chung, người quản lý dự án có thể chỉnh lại bản mẫu này để phù hợp với nhu cầu của họ.

Hình 3.3. Phân tích tác động của yêu cầu thay đổi trong dự án ACIC

Request No: 3 **Date:** 11th August 2000

Change Specification

This change request is to allow the client screens to change the resolution automatically, depending on the monitor. This is required as the monitors of ACIC are of resolution 1600 * 1200, while the monitors of the business partner, who will also use the application, have a resolution of 800 * 600.

Impact Analysis

Change Category: Major enhancement as it affects all screens.

Solution: The Layout manager and setting the constraints for each of the components has to be changed. Because of Applet implementation, the code has to be changed so that the screens automatically adjust to the resolution of the monitor.

On Effort: Total number of screens is 40. Implementing the solution for a screen will take about 12 hours. Hence, total estimated effort to implement this change is 480 person-hours (about 53 person-days).

On Schedule: As we have an average team size of about 5.5, the impact on the overall schedule of implementing this change request will be about 10 days.

Status

Analysis approved by customer. Will be incorporated immediately. Project schedule will be changed; dates for milestones will also be changed, where needed.

Prepared by: xxxxxx **Reviewed by:** xxxxx

Hình 3.3 cho thấy một bản mẫu (template) đã được điền vào bởi một yêu cầu thay đổi cho dự án ACIC. Ví dụ này cho thấy một yêu cầu thay đổi có ảnh hưởng đến nỗ lực cũng như thời gian hoàn thành. Kết quả của việc phân tích tác động nói rõ tác động lên hai yếu tố này. Báo cáo phân tích cũng nói rằng khách hàng đã chấp nhận các tác động (bao gồm cả những thay đổi trong nỗ lực và thời gian hoàn thành).

Hình 3.4 đưa ra một ví dụ khác của một yêu cầu thay đổi. Trong ví dụ, nội dung chi tiết của phân tích tác động là không quan trọng cho các mục đích hiểu về quản lý thay đổi yêu cầu.

Hình 3.4. Ví dụ khác về yêu cầu thay đổi

Project XYZ

Request No: 11 **Date:** 23 Feb 1998

Change Spec. IS-41 Analyzer—IS-41 Analyzer support for CDMA

Impact Analysis

No particular change in configuration module and analyzers for CDMA.
The TDMA code can be reused as is. Scripts can also be reused. Netconfig and analyzer classes can be reused. The impacted modules are as follows: cgaapp module: Has to trigger analysis for IS-41 also, separately. cdmaroi module: (a) TRIS41ROI has to be copied as TRCDMAIS 41ROI; (b) There is a pure virtual method in TRCDMAROI for setting the ActualCallModelManager. It needs to be redefined. silver06guiapp++ module: IS-41 has to be added in the resource list.

On Schedule Nil

On Effort 5

Status Will be incorporated in the new CDMA package.

Lưu ý rằng mặc dù tác động của một yêu cầu thay đổi được quy định cụ thể, và được chấp thuận, bằng cách sử dụng các bản mẫu (template), theo dõi tiến độ thực tế để thực hiện (cài đặt) thay đổi được quản lý theo quy trình quản lý cấu hình, được thảo luận trong Chương 9.

Một nguy cơ của của các thay đổi yêu cầu là, mặc dù mỗi lần thay đổi không lớn, nhưng qua suốt chu kỳ sống của dự án, tác động tích lũy (cumulative impact) của những thay đổi này sẽ lớn. Vì thế, ngoài việc nghiên cứu tác động và theo dõi các thay đổi đơn lẻ, bạn phải giám sát tác động tích lũy của các thay đổi. Đối với các thay đổi tích lũy, một *nhật ký thay đổi (change log)* được sử dụng. Để tạo điều kiện thuận lợi cho phân tích này, nhật ký

(log) thường xuyên được cập nhật như một bảng tính (spreadsheet). Ví dụ trong Hình 3.5 minh họa làm thế nào các thay đổi tích lũy được ghi nhận. Đối với các chi tiết của mỗi yêu cầu, yêu cầu thay đổi có liên quan có thể được truy cập bằng cách sử dụng *số của yêu cầu thay đổi (change request number)* và *ngày (date) yêu cầu.*

Hình 3.5. Theo dõi các tác động tích lũy của các thay đổi

Chg. Req. No.	Date of Change Req.	Change Specs	Effort (person-days)	Status
1	18 Feb	Specify usage statistics	3	Closed, Feb 22
2	During demo	Blocking of users	2	Open
3	During demo	Force logout of users	2	Open
4	18 Feb	Archival of knowledge users	5	Closed, Feb 27
5	During demo	Cloning of window	1	Open
6	During demo	Saving off an expanded tree and retrieving the same on demand	10	Open
7	During demo	Ability to start from a specific node	2	Open
8	During demo	Listing of all nodes while deleting	1	Open
9	18 Feb	Annotations (creating/ deleting/approving/ modifying/and so on)	10	Open
10	23 Feb	PFNETCONFIG—Packed format netconfig support	10	Open
11	23 Feb	IS-41 Analyzer—IS-41 Analyzer support for CDMA	5	Closed, March 1
		TOTAL	51	

Từ một bảng tính loại này, bạn có thể thấy ngay lập tức tổng chi phí (effort/nỗ lực) của các thay đổi yêu cầu được thực hiện cho đến thời điểm hiện tại. Như đã đề cập trước đây, những người quản lý dự án ở Infosys đôi khi hoạch định trước các vùng đệm (buffer) để xử lý các yêu cầu thay đổi. Miễn là các nỗ lực/chi phí tích lũy (cumulative effort) của tất cả các yêu cầu thay đổi ít hơn vùng đệm này, không có phát sinh nào cần được giải quyết thêm. Tuy nhiên, nếu các nỗ lực tích lũy của tất cả các thay đổi vượt quá vùng đệm này, nếu tiếp tục thay đổi thêm nữa có thể sẽ ảnh hưởng xấu đến tổng chi phí và thời gian

hoàn thành. Trong tình huống này, người quản lý dự án phải sửa đổi lại ước lượng/dự toán và chờ sự chấp nhận từ phía khách hành.

3.3 Lập kế hoạch quy trình cho dự án ACIC

Vì dự án ACIC là một dự án phát triển (development project), quy trình phát triển theo tiêu chuẩn của Infosys áp dụng. Tuy nhiên, nó được thay đổi để thích ứng với các đòi hỏi của phương pháp Rational Unified Process (RUP) bởi vì sử dụng RUP đã được cam kết với khách hàng. RUP là một phương pháp phát triển vòng lặp. Các hướng dẫn điều chỉnh cho Infosys cho phép các giai đoạn khác nhau được thực hiện lặp đi lặp lại như đặc tả yêu cầu, thiết kế, cài đặt mã, và kiểm thử được thực hiện trong mỗi vòng lặp (iteration). Ngoài việc thực hiện phát triển theo vòng lặp, các sửa đổi quan trọng sau đây đã được thực hiện cho quy trình chuẩn RUP:

- Chỉ những trường hợp sử dụng (use cases) đã được chọn ra để phát triển trong một vòng lặp cụ thể sẽ được xây dựng chi tiết (elaboration) tại điểm đó.
- Các mô hình đối tượng lôgic (logical object model) và mô hình đối tượng vật lý (physical object model) sẽ được phát triển tăng dần (incrementally development) trong vòng lặp đầu tiên.
- Thiết kế cơ sở dữ liệu vật lý (physical database design) có thể được tinh chế (refine) sau các vòng.
- Một kế hoạch kiểm thử đơn vị (unit test plan) sẽ được phát triển trong mỗi vòng lặp.
- Lỗi sẽ được ghi nhận sau mỗi vòng lặp.

Ngoài ra, cơ chế của ma trận theo dõi dấu vết nguồn gốc chuẩn (standard traceability matrix mechanism) sẽ không được sử dụng để theo dõi dấu vết nguồn gốc yêu cầu (requirement traceability). Thay vào đó, các yêu cầu (requirements) sẽ được truy tìm bằng cách sử dụng các công cụ Requisite Pro, nó là một phần của môi trường phát triển.

Để quản lý thay đổi yêu cầu, quy trình chuẩn sẽ được sử dụng. Mặc dù việc phân tích tác động sẽ được thực hiện cho mỗi yêu cầu, nhưng việc ước lượng lại (reestimation) chi phí

(nỗ lực) sẽ được thực hiện lại nếu một yêu cầu thay đổi (change request) cần hơn 2% tổng số nỗ lực (total effort). Các kết quả đầu ra của việc lập kế hoạch quy trình sẽ được ghi nhận vào trong kế hoạch quản lý dự án, được thảo luận trong Chương 8.

3.4 TÓM TẮT

Vấn đề phát sinh chính của việc lập kế hoạch cho quy trình là thiết kế quy trình phát triển để xây dựng phần mềm thỏa yêu cầu của khách hàng. Quy trình này được hỗ trợ bởi một quy trình quản lý thay đổi yêu cầu.

Sau đây là những bài học quan trọng để lập kế hoạch cho quy trình theo phương pháp của Infosys:

- Khi bạn lập kế hoạch cho quy trình của dự án, hãy bắt đầu với một quy trình chuẩn (standard process). Mô hình thác nước, được chia thành các giai đoạn nhỏ hơn, có thể phục vụ như là một cơ sở (base) thích hợp.

- Để định nghĩa một quy trình tối ưu cho một dự án, hãy điều chỉnh quy trình chuẩn cho phù hợp với các ràng buộc của dự án. Trước tiên, hãy thiết lập bối cảnh (context) để điều chỉnh bằng cách sử dụng các đặc trưng chính (key project characteristics) của dự án. Sau đó thực hiện điều chỉnh chi tiết các hoạt động. Điều chỉnh các hướng dẫn để cung cấp các giúp đỡ.

- Có một quy trình quản lý thay đổi yêu cầu riêng biệt để có thể đánh giá tác động của mỗi yêu cầu thay đổi và cũng có thể theo dõi tác động tích lũy. Liên quan đến CMM, các phương pháp lập kế hoạch cho quy trình được mô tả ở đây đáp ứng một số yêu cầu của Lập Kế Hoạch Cho Dự Án Phần Mềm của CMM mức 2 (Software Project Planning KPA of level 2). Phương pháp điều chỉnh thỏa mãn một số yêu cầu của Quản Lý Dự Án Phần Mềm Tích Hợp của CMM mức 3 (Integrated Project Management KPA at level 3). Phương pháp quản lý thay đổi yêu câu đáp ứng một số yêu cầu của Quản Lý Yêu Cầu của CMM mức 2 (Requirement Management KPA of level 2).

3.5 CÁC THAM KHẢO

1. B.W. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

2. V.R. Basili and A. Turner. Iterative enhancement, a practical technique for software development. *IEEE Transactions on Software Engineering*, 1(4), 1975.

3. H. Gomma and D.B.H. Scott. Prototyping as a tool in the specification of user requirements. *Proceedings of the 5th International Conference on Software Engineering*, pp. 333–341, 1981.

4. B.W. Boehm. A spiral model of software development and enhancement. *IEEE Computer*, May 1988.

5. P. Jalote. *CMM in Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley, 2000.

6. W. Humphrey. *Managing the Software Process*. Addison-Wesley, 1989.

7. M.P. Ginsberg and L.H. Quinn. *Process Tailoring and the Software Capability Maturity Model*. Technical Report, Software Engineering Institute, CMU/ SEI-94-TR-024, 1995.

8. B.W. Boehm. Improving software productivity. *IEEE Computer*, Sept. 1987.

Chương 4. Ước lượng nỗ lực và thời gian biểu

Đầu năm 2000, báo chí và truyền hình ở Ấn Độ đã báo cáo với hân hoan về chuyến bay thử nghiệm thành công máy bay chiến đấu mới được xây dựng. Tuy nhiên, các báo cáo chi tiết cũng chỉ ra một mặt ảm đạm: Việc giao hàng mẫu đã trễ hơn năm năm, và dự án có chi phí gấp 10 lần chi phí được ước lượng lúc đầu. Nhiều dự án trên toàn thế giới phải chịu một số phận tương tự. Có vẻ như chi phí ước lượng và thời gian không bao giờ đủ để thực hiện một dự án. Ước lượng sai là nguyên nhân thất bại của quản lý dự án trong nhiều lĩnh vực công nghệ, và công nghệ phần mềm cũng như vậy.

Trong một công ty dịch vụ, ước lượng sai còn gây ra nhiều tổn thất hơn. Được yêu cầu để xác định một trong những điều ông ta muốn từ các dự án, Nandan Nilekeni, giám đốc quản lý của Infosys, trả lời rằng, "Không có bất ngờ". Không có bất ngờ ở mặt thỏa mãn yêu cầu khách hàng, và không có bất ngờ ở mặt doanh thu (revenue) và lợi nhuận (profit). Còn những bất ngờ thì thường đến muộn trong dự án ở hai mặt này là không ước lượng/dự toán đúng nỗ lực (effort) hoặc thời gian hoàn thành (schedule).

Không có giải pháp nhanh chóng và đã được làm sẵn nào cho các vấn đề ước lượng. Tuy nhiên, những người quản lý dự án có thể cải tiến ước lượng của họ bằng cách sử dụng các hướng dẫn đã được kiểm nghiệm (tested guidelines) dựa trên kinh nghiệm và dữ liệu quá khứ. Chương này bàn về các tiếp cận được sử dụng bởi những người quản lý dự án ở Infosys để ước lượng nỗ lực và thời gian hoàn thành. Nó bao gồm các ví dụ từ các dự án thực tế và các ước lượng cho case study (trường hợp nghiên cứu) ACIC.

4.1 CÁC KHÁI NIỆM VỀ ƯỚC LƯỢNG VÀ LẬP THỜI GIAN BIỂU

Trước khi chúng tôi thảo luận về các tiếp cận đã được thực hiện tại Infosys, phần này mô tả một số khái niệm liên quan đến ước lượng/dự toán và lập thời gian biểu. Ước lượng nỗ lực thường diễn ra trong giai đoạn đầu của một dự án, khi phần mềm sẽ được xây dựng đang được tìm hiểu. Nó có thể được làm lại khi có thêm thông tin chi tiết.

Ước lượng chính xác cao nói chung là không cần thiết. Ước lượng hợp lý trong một dự án phần mềm có khuynh hướng trở thành một lời tiên tri - mọi người làm việc để kịp thời gian biểu (được suy ra từ ước lượng). Thật vậy, trong các dự án phần mềm, người ta thậm chí

không thể trả lời chính xác câu hỏi, "ước lượng này có chính xác không?" bởi vì cách duy nhất để xác định độ chính xác của ước lượng là so sánh nó với những nỗ lực thực tế đã tiêu dùng. Do nguyên tắc chung về tâm lý con người đã được phản ánh trong câu châm ngôn "công việc được mở rộng để lấp đầy khoảng thời gian còn dư ra," không ai có thể nói ra điều đó chỉ bởi vì số nỗ lực thực tế đã được tiêu dùng bằng với số nỗ lực ước lượng, ước lượng là "chính xác." Do đó, mục tiêu cho người quản lý dự án là làm sao để có được các *ước lượng hợp lý (reasonable estimates)* để mà các mục tiêu được đáp ứng và các nhân viên của dự án không bị đốt cháy.

Nỗ lực có thể được ước lượng dựa vào linh cảm hoặc vào kinh nghiệm trước đây. Tuy nhiên, một mô hình ước lượng dùng tiếp cận khoa học sẽ được ưa thích nhiều hơn.

4.1.1 Các mô hình ước lượng nỗ lực (Effort Estimation Models)

Một *mô hình ước lượng phần mềm (*software *estimation model)* định nghĩa các đặc trưng (characteristics) của dự án mà giá trị của những đặc trưng này được dùng để tính nỗ lực (effort). Một mô hình ước lượng không - và không thể - hoạt động trong chân không, nó cần các yếu tố đầu vào để tạo ra nỗ lực ở đầu ra. Ở lúc bắt đầu dự án, khi các chi tiết của phần mềm chưa được biết đến, hy vọng là mô hình ước lượng sẽ đòi hỏi các giá trị của các đặc trưng có thể được đo ở giai đoạn này.

Kích thước (size) của phần mềm là yếu tố quan trọng nhất trong việc xác định bao nhiêu nỗ lực là cần thiết để xây dựng nó. Tuy nhiên, kích thước cuối cùng thì không được biết khi dự án vẫn còn đang được hình thành, và phần mềm chưa thực sự tồn tại. Do đó, nếu kích thước được sử dụng cho một mô hình ước lượng nỗ lực, nó phải được ước lượng ngay từ lúc đầu.

Một tiếp cận phổ biến là sử dụng một phương trình đơn giản để có được một ước lượng nỗ lực tổng thể từ ước lượng kích thước. Phương trình này có thể được xác định thông qua phân tích hồi quy (regression analysis) các dữ liệu quá khứ (past data) về nỗ lực (effort) và kích thước (size) [1][2]. Sau đó, một khi nỗ lực tổng thể của dự án được biết, nỗ lực phân bổ cho các giai đoạn hoặc các hoạt động khác nhau có thể được xác định bằng một tỷ lệ phần trăm của nỗ lực tổng thể.

Nhiều mô hình đã đề xuất sử dụng *tiếp cận từ trên xuống (top-down approach*) để ước lượng [1][3], ví dụ như mô hình nổi tiếng COCOMO [1][4]. Các mô hình sử dụng Điểm

chức năng (Function points) thay vì LOC làm đơn vị kích thước cũng đã được xây dựng [5][6]. Trong các mô hình này, bạn có thể xác định thêm các yếu tố khác (có ảnh hưởng đến nỗ lực) để điều chỉnh lại các ước lượng dựa trên các yếu tố này. Đây là tiếp cận được thực hiện trong mô hình COCOMO model [1]. Một tiếp cận khác là điều chỉnh kích thước của hệ thống dựa trên các thông số (parameters), như đã được thực hiện trong Điểm chức năng (Function points)[7].

Trong *tiếp cận từ dưới lên* (*bottom-up approach*), trước tiên, bạn thực hiện các ước lượng cho các phần của dự án và sau đó ước tính cho tổng thể dự án [1]. Tức là, ước lượng tổng thể của dự án được suy ra từ các ước lượng của các bộ phận của nó. Một tiếp cận từ dưới lên được gọi là *ước lượng dựa trên các hoạt động (activity-based estimation).* Trong chiến lược này, trước hết các hoạt động chính được liệt kê ra, và sau đó nỗ lực cho từng hoạt động được ước lượng. Từ những ước lượng này, bạn thu được nỗ lực cho toàn bộ dự án.

Tiếp cận tiếp cận từ dưới lên ước lượng trực tiếp (ước lượng ngay) nỗ lực; một khi dự án được phân chia thành các công việc/nhiệm vụ nhỏ hơn thì việc ước lượng trực tiếp nỗ lực cần thiết cho chúng là điều có thể. Mặc dù kích thước đóng một vai trò quan trọng trong việc xác định các nỗ lực cho các hoạt động trong một dự án, nhưng lợi thế của tiếp cận từ dưới lên là nó không cần ước lượng kích thước cho phần mềm. Thay vào đó, nó đòi hỏi một danh sách các công việc/nhiệm vụ trong dự án - có thể được chuẩn bị dễ dàng hơn. Một rủi ro của tiếp cận từ dưới lên là bạn có thể bỏ qua một số hoạt động quan trọng trong danh sách các công viêc/nhiệm vụ. Khi nỗ lực được ước lượng trực tiếp cho các công việc/nhiệm vụ, khó có thể ước lượng cho các công việc/nhiệm vụ về quản lý, chẳng hạn như quản lý dự án trong suốt chu kỳ sống, đây là các công việc không được xác định rõ ràng như cài đặt mã hoặc kiểm thử.

Cả hai tiếp cận từ trên xuống và từ dưới lên điều cần các thông tin về dự án: kích thước (đối với tiếp cận từ trên xuống) và một danh sách các công viêc/nhiệm vụ (đối với tiếp cận từ dưới lên). Bằng nhiều cách, những tiếp cận này có thể bổ sung cho nhau [1]. Cả hai loại ước lượng sẽ cho kết quả chính xác hơn nếu thông tin về dự án có được biết nhiều hơn. Ví dụ, ước lượng kích thước từ các yêu cầu ở mức cao (high level requirements) thì khó hơn nhiều ước lượng kích thước khi đã thiết kế (design) xong, và thậm chí sẽ dễ dàng hơn và chính xác hơn khi mã (code) đã được phát triển. Như vậy, độ chính xác của

ước lượng phụ thuộc vào thời điểm mà tại đó nỗ lực ước lượng, và độ chính xác sẽ tăng khi có thêm thông tin về dự án được biết.

4.1.2 Ước lượng thời gian biểu (Estimating Schedule)

Sau khi nỗ lực đã được biết hoặc đã được ấn định cố định (fixed) ngay từ đầu, các thời gian biểu khác nhau (tức là khoảng thời gian để hoàn dự án) có thể tính được tính, tùy thuộc vào số nhân sự (người) được đưa vào để thực hiện dự án. Ví dụ, đối với một dự án có nỗ lực ước lượng là 56 tháng-người (person-months), dự án có thể hoàn thành trong vòng 8 tháng với 7 người. Nó cũng có thể hoàn thành trong vòng 7 tháng với 8 người, và khoảng 9 tháng với 6 người.

Tuy nhiên, nhân lực (manpower) và số tháng (months) không hoàn toàn có thể được hoán đổi cho nhau trong một dự án phần mềm [8]. Ví dụ, trong ví dụ ở đây, một thời gian biểu 1 tháng với 56 người thì không thể mặc dù lượng nỗ lực “bằng” với lượng nỗ lượng được cần. Tương tự như vậy, không ai có thể thực hiện dự án trong 28 tháng với 2 người. Nói cách khác, một khi nỗ lực này đã được ấn định, bạn có thể đạt được sự linh hoạt trong việc thiết lập thời gian biểu thích hợp với nhân sự của dự án. Tuy nhiên, sự linh hoạt này không phải là không có giới hạn, thực tế này đã được chứng minh bằng dữ liệu, kết quả cho thấy rằng không có phương trình đơn giản nào phù hợp để diễn tả mối quan hệ giữ nỗ lực và thời gian biểu. Sự thật này đã được chứng minh bằng dữ liệu thực nghiệm, kết quả cho thấy rằng không có phương trình đơn giản nào có thể phù hợp (làm vừa khít) cho mối quan hệ nỗ lực và thời gian biểu [9].

"Kéo dài" thời gian biểu ra thì dễ, bạn chỉ cần sử dụng ít người hơn (mặc dù các dự án có thể không còn nhiều giá trị nếu phải được thực hiện trong một khoảng thời gian dài hơn). Tuy nhiên, nén thời gian biểu lại thì không dễ. Như một ví dụ rõ ràng đã được đưa ra trước đây: Bạn không thể nén thời hạn để hoàn thành của một dự án 56 người-tháng xuống còn 1 tháng bất kể nguồn nhân lực bạn sử dụng như thế nào. Và, nói chung, nén thời gian biểu lại vượt quá những gì là "bình thường" sẽ làm tăng số lượng nỗ lực, tức là sẽ cần nhiều ngày công hơn so với tối ưu cần thiết, bạn có thể lãng phí, làm lại (rework) nhiều hơn, v.v.

Có vài tiếp cận thảo luận về tác động của việc nén thời gian biểu lên nỗ lực tổng thể. Tuy nhiên, để đánh giá tác động này, trước tiên bạn phải xác định được một thời gian biểu bình thường cho dự án.

Một phương pháp để xác định *thời gian biểu bình thường (normal or nominal schedule)* là sử dụng một hàm thích hợp để xác định nó từ nỗ lực. Một trong những phương pháp để xác định hàm này là nghiên cứu các mẫu dữ liệu từ các dự án đã hoàn thành. Ví dụ, bạn có thể có được một biểu đồ (scatter plot) để phân tán các dự án đã hoàn thành theo nỗ lực và thời gian biểu của mỗi dự án và sau đó xác định một đường cong hồi quy (regression curve) vừa khít nhất cho biểu đồ phân tán này. Đường cong này thường là phi tuyến (nonlinear) vì thời gian biểu không phát triển tuyến tính (linearly) với nỗ lực. Sau đó, bạn có thể sử dụng phương trình của đường cong này để xác định thời gian biểu cho dự án mà nỗ lực của nó vừa được ước lượng. Có nhiều mô hình khác nhau thực hiện theo phương pháp này, và cuốn sách của Boehm đã tóm tắt các mô hình này [1].

4.2 ƯỚC LƯỢNG NỖ LỰC

Ở Infosys, ước lượng thường diễn ra sau giai đoạn phân tích (analysis). Tức là, khi người quản lý dự án ước lượng nỗ lực, các yêu cầu (requirements) đã được hiểu rõ ràng. Các quy trình nghiệp vụ (business processes) được tổ chức để hỗ trợ phương pháp này. Ví dụ, giai đoạn yêu cầu (requirement phase) đôi khi được thực hiện như một dự án riêng biệt với dự án phát triển phần mềm.

Ở Infosys, nhiều phương pháp ước lượng đã được đề xuất, vài trong số đó được thảo luận ở đây. Người quản lý dự án có thể lựa chọn bất kỳ phương pháp ước lượng nào miễn là nó thích hợp với tính chất công việc. Đôi khi, người quản lý dự án có thể thực hiện ước lượng bằng cách sử dụng nhiều phương pháp khác nhau, để xác nhận (validate) độ chính xác của một ước lượng được làm bởi một phương pháp chính nào đó hoặc để giảm rủi ro, đặc biệt là khi không có nhiều dữ liệu quá khứ của dự án tương tự.

4.2.1 Tiếp cận ước lượng từ dưới lên (Bottom-up estimation approach)

Bởi vì loại của các dự án thực hiện tại Infosys thì rất khác nhau, tiếp cận từ dưới lên được ưa thích và được khuyến khích. Công ty sử dụng phương pháp *đơn vị công việc (task unit approach),* mặc dù một số hạn chế của chiến lược này đã được khắc phục thông qua việc sử dụng các dữ liệu quá khứ và baseline về khả năng của quy trình (xem Chương 2).

Trong phương pháp *đơn vị công việc*, người quản lý dự án đầu tiên chia phần mềm sắp được phát triển ra thành các chương trình chính (major programs) hoặc đơn vị chương trình (program unit). Mỗi đơn vị chương trình (program unit) sau đó được phân loại là trung bình, đơn giản hoặc phức tạp dựa trên các tiêu chí nhất định. Đối với mỗi đơn vị phân loại (classification unit), người quản lý dự án xác định một *nỗ lực tiêu chuẩn (standard effort)* cần thiết để cài đặt mã (coding) và tự kiểm thử (self-testing) (cả hai công việc này được gọi chung là *nỗ lực xây dựng (build effort).* Nỗ lực xây dựng chuẩn này có thể được xác định từ dữ liệu quá khứ của một dự án tương tự, từ các hướng dẫn nội bộ có sẵn, hoặc kết hợp những khả năng này.

Một khi số lượng các đơn vị (units) trong ba loại phức tạp được biết và nỗ lực xây dựng đã được ước lượng cho mỗi chương trình được chọn, số nỗ lực tổng thể cho giai đoạn xây dựng (build phase) của dự án sẽ được biết. Từ nỗ lực xây dựng (build effort), nỗ lực được cần cho các giai đoạn và các hoạt động khác sẽ được xác định bằng một tỷ lệ phần trăm của nỗ lực cài đặt mã. Dựa vào baseline về khả năng của quy trình (process capability baseline) hoặc cơ sở dữ liệu quy trình (process database), phân phối nỗ lực trong dự án được biết. Nhài quản lý dự án sử dụng tỷ lệ phân phối này để xác định các nỗ lực cho giai đoạn và các hoạt động khác nhau. Từ những ước lượng này, nỗ lực tổng thể cho dự án được biết.

Phương pháp này đã sử dụng một hỗn hợp của kinh nghiệm và dữ liệu một cách hợp lý. Nếu không có sẵn dữ liệu phù hợp (ví dụ, nếu bạn đang bắt đầu một kiểu dự án mới), bạn có thể ước lượng nỗ lực xây dựng dựa theo kinh nghiệm sau khi bạn phân tích (analyze) dự án xong và khi bạn biết số lượng các đơn vị chương trình khác nhau. Với các ước lượng đã có sẵn, bạn có thể ước lượng cho các hoạt động khác nhau bằng cách sử dụng dữ liệu về tỷ lệ phân phối nỗ lực của các dự án trong quá khứ. Chiến lược này có khả năng tính đến các hoạt động mà chúng thường khó được liệt kê ra ở giai đoạn đầu của dự án; khi thực hiện phân phối nỗ lực cho một dự án, loại "khác" thường được sử dụng để thực hiện các công việc/nhiệm vụ linh tinh.

Quy trình (thủ tục) ước lượng có thể được tóm tắt theo trình tự các bước như sau:

1. Xác định các chương trình trong hệ thống và phân loại chúng là đơn giản, trung bình, hoặc phức tạp (S/M/C). Càng nhiều càng tốt, hãy sử dụng các định nghĩa chuẩn đã được cung cấp hoặc các định nghĩa của các dự án trong quá khứ.

2. Nếu một baseline riêng (project-specific baseline) của dự án tồn tại, hãy tính ra nỗ lực xây dựng trung bình cho các chương trình S/M/C từ baseline này.

3. Nếu baseline riêng của dự án chưa tồn tại, hãy sử dụng: loại dự án (project type), công nghệ (technology), ngôn ngữ (language), và các thuộc tính khác để tìm kiếm các dự án tương tự (similar projects) trong cơ sở dữ liệu quy trình. Hãy sử dụng dữ liệu từ các dự án này để xác định nỗ lực xây dựng cho các chương trình S/M/C.

4. Nếu không có dự án tương tự tồn tại trong cơ sở dữ liệu quy trình và cũng không có baseline riêng của dự án, hãy sử dụng nỗ lực xây dựng trung bình cho các chương trình S/M/C từ baseline chung về khả năng quy trình (general process capability baseline).

5. Sử dụng các yếu tố riêng của dự án (project-specific factors)[1] để tinh chỉnh các nỗ lực xây dựng cho chương trình S/M/C.

6. Tính ra tổng nỗ lực xây dựng bằng cách sử dụng nỗ lực xây dựng của các chương trình S/M/C và đếm chúng.

7. Sử dụng tỷ lệ phân phối nỗ lực được đưa ra trong baseline về khả năng hoặc trong các dự án tương tự trong cơ sở dữ liệu quy trình, ước lượng nỗ lực cho các công việc/nhiệm vụ khác và nỗ lực tổng.

8. Tinh chỉnh lại các ước lượng dựa trên các yếu tố riêng của dự án (project-specific factors)[1].

Thủ tục này sử dụng cơ sở dữ liệu quy trình (process database) và baseline về khả năng quy trình (process capability baseline), được thảo luận trong Chương 2. Như đã đề cập trước đây, nếu nhiều dự án có cùng một loại đang được thực hiện, bạn có thể xây dựng một baseline về khả năng của dự án (project-specific capability baseline). Một baseline như vậy là tương tự như các baseline chung (general baselines) nhưng chỉ sử dụng dữ

liệu từ vài dự án cụ thể. Các baseline này đã được nhận thấy là tốt nhất để ước lượng nỗ lực cho một dự án mới có cùng loại đó. Vì thế, chúng được ưa thích khi được sử dụng cho ước lượng.

Bởi vì nhiều yếu tố khác nhau có thể ảnh hưởng đến lượng nỗ lực cần thiết cho một dự án, điều quan trọng khi làm ước lượng là bạn phải xác định được các yếu tố riêng của dự án (project-specific factors). Thay vì phân loại các tham số ra thành các mức khác nhau và sau đó xác định mức độ ảnh hưởng lên lượng nỗ lực được cần, phương pháp được nêu ra ở đây cho phép người quản lý dự án xác định tác động của các yếu tố riêng của dự án lên ước lượng. Người quản lý dự án có thể thực hiện điều chỉnh bằng cách sử dụng kinh nghiệm của họ, kinh nghiệm của các thành viên trong nhóm, hoặc dữ liệu từ các dự án được tìm thấy trong cơ sở dữ liệu quy trình.

Lưu ý rằng phương pháp phân loại các chương trình ra thành vài loại và sử dụng số nỗ lực xây dựng trung bình cho mỗi loại được áp dụng để thực hiện ước lượng nỗ lực tổng thể. Tuy nhiên, khi lập kế hoạch chi tiết, người quản lý dự án phân công mỗi đơn vị kích thước cho một thành viên của nhóm để cài đặt mã, và thời gian cho các hoạt động - đặc trưng của một đơn vị kích thước sẽ được ghi nhận xem nó có cần thời gian nhiều hơn hoặc ít hơn so với trung bình.

4.2.2 Tiếp cận ước lượng từ trên xuống (The top-down estimation approach)

Giống như bất kỳ tiếp cận từ trên xuống, tiếp cận ở Infosys bắt đầu với một ước lượng kích thước của các phần mềm dùng các Điểm chức năng (Function points). Các Điểm chức năng có thể được đếm bằng cách sử dụng các quy tắc đếm Điểm chức năng chuẩn. Ngoài ra, nếu kích thước được ước lượng bằng LOC, nó có thể được chuyển đổi thành các Điểm chức năng.

Ngoài ước lượng kích thước, tiếp cận từ trên xuống đòi hỏi phải ước lượng năng suất (productivity). Các tiếp cận cơ bản là bắt đầu với các mức năng suất (productivity levels) của các dự án tương tự (dữ liệu đã có sẵn trong cơ sở dữ liệu quy trình) hoặc với số liệu năng suất chuẩn (standard productivity figures) (mà dữ liệu về nó đã có sẵn trong baseline về khả năng của quy trình), và sau đó phải điều chỉnh lại các mức này, nếu cần thiết, để phù hợp với dự án đang được ước lượng. Ước lượng năng suất sau đó được sử dụng để

tính ra nỗ lực tổng thể. Từ nỗ lực tổng thể, nỗ lực cho các giai đoạn khác nhau được ước lượng bằng cách sử dụng các bảng phân phối tỷ lệ phần trăm. Như trong tiếp cận từ dưới lên, những phân phối này có thể thu được từ cơ sở dữ liệu quy trình hoặc baseline về khả năng quy trình (capability baseline).

Để tóm tắt, tiếp cận tổng thể để thực hiện ước lượng từ trên xuống bao gồm các bước sau đây:

1. Ước lượng kích thước tổng cộng của phần mềm bằng các Điểm chức năng.
2. Sử dụng các dữ liệu về năng suất từ baseline về khả năng của dự án, từ baseline về khả năng của quy trình, hoặc từ các dự án tương tự, sửa chữa lại các mức năng suất cho phù hợp với dự án đang được ước lượng.
3. Có được nỗ lực tổng thể của dự án từ năng suất và kích thước.
4. Tinh chỉnh lại ước lượng, có xem xét đến mức độ ảnh hưởng của các yếu tố riêng của dự án.
5. Sử dụng dữ liệu về phân phối nỗ lực từ baseline về khả năng của quy trình hoặc từ các dự án tương tự để ước lượng nỗ lực cho các giai đoạn khác nhau.

Cũng giống như tiếp cận từ dưới lên, tiếp cận từ trên xuống cũng cho phép tinh chỉnh lại kết quả ước lượng bằng cách sử dụng các yếu tố riêng của dự án. Sự cho phép này (không thực sự định nghĩa các yếu tố nào) báo hiệu rằng mỗi dự án là duy nhất và có thể có vài yếu tố đặc trưng – có thể không có mặt trong các dự án khác. Không thể liệt kê ra những yếu tố đặc trưng này hoặc lập mô hình một cách hình thức về sự ảnh hưởng của chúng lên năng suất. Do đó, hãy để cho người quản lý dự án quyết định các yếu tố nào cần được xem xét và chúng sẽ ảnh hưởng lên dự án thế nào.

4.2.3 Phương pháp Điểm trường hợp sử dụng (The Use Case Points approach)

Phương pháp Điểm trường hợp sử dụng (use case points) được áp dụng tại Infosys được dựa trên phương pháp từ Rational và nó tương tự như phương pháp Điểm chức năng (Function points). Phương pháp này có thể được áp dụng nếu các trường hợp sử dụng (use case) được dùng để đặc tả yêu cầu (requirement specification).

Một số từ ngữ cần được hiểu trước khi thực hiện ước lượng:

- Phần mềm là những chương trình điều khiển các chức năng phần cứng và hướng dẫn phần cứng thực hiện các tác vụ của mình.
- Trường hợp sử dụng (use case) là một tập hợp các giao dịch giữa hệ thống phần mềm với các tác nhân bên ngoài hệ thống nhằm đạt được một mục tiêu sử dụng nào đó của tác nhân. Một trường hợp sử dụng mô tả một hoặc nhiều tình huống sử dụng xảy ra khi tác nhân tương tác với hệ thống phần mềm.
- Giao dịch (transaction) là một chuỗi các hành động có tính chất tương tác giữa tác nhân và hệ thống phần mềm. Khởi đầu của chuỗi hành động này là một hành động từ tác nhân tới hệ thống. Kết thúc của chuỗi hành động này là một hành động ngược trở lại của hệ thống lên tác nhân.
- Biểu đồ trường hợp sử dụng (use case diagram) dùng mô tả các tác nhân và kết nối giữa tác nhân với các trường hợp sử dụng nhằm miêu tả chức năng mà phần mềm cung cấp.
- Tác nhân (actor) là người hay hệ thống bên ngoài tương tác, trao đổi thông tin với phần mềm.

Các bước cơ bản của phương pháp này như sau:

1. **Tính giá trị điểm các tác nhân TAW:** Dùng Bảng 4.1.a để tính giá trị điểm các tác nhân.

Bảng 4.1a. Bảng tính toán điểm các tác nhân (actors) tương tác, trao đổi thông tin với phần mềm

TT	Loại Actor	Mô tả	Số tác nhân	Điểm của từng loại tác nhân
1	Đơn giản	Thuộc loại giao diện của chương trình	?	?
2	Trung bình	Giao diện tương tác hoặc phục vụ một giao thức hoạt động	?	?
3	Phức tạp	Giao diện đồ họa	?	?
	Cộng (1+2+3)	**TAW (giá trị điểm các tác nhân)**	**?**	**?**

***Ghi chú*:**

- Loại đơn giản: Một máy tính với giao diện lập trình ứng dụng API.
- Loại trung bình: Hoặc là giao diện người - máy qua "command line" hoặc thông qua một giao thức nào đó nhưng không có lập trình qua API.
- Loại phức tạp: giao diện người - máy qua GUI (giao diện đồ hoạ).

Điểm của từng loại tác nhân (đơn vị tính: điểm) được xác định theo công thức:

Điểm của từng loại tác nhân = Số tác nhân x Trọng số

Trong đó: Trọng số được qui định như sau:

TT	Loại Actor	Trọng số
1	Đơn giản	1
2	Trung bình	2
3	Phức tạp	3

2. **Tính giá trị điểm các trường hợp sử dụng TBF:** Phân loại từng trường hợp sử dụng là đơn giản, trung bình, hay phức tạp. Cơ sở của sự phân loại này là số lượng giao dịch (transaction) có trong một trường hợp sử dụng, bao gồm cả các kịch bản (scenarios) phụ. Một giao dịch được định nghĩa là một tập hợp nguyên tử (atomic) của các hoạt động được hoặc thực hiện hoàn toàn hoặc không gì được thực hiện. Bảng 4.1b trình bày cách phân loại độ phức tạp và trọng số của trường hợp sử dụng. Một trường hợp sử dụng đơn giản có 3 hoặc ít hơn 3 giao dịch, một trường hợp sử dụng trung bình có 4-7 giao dịch, và một trường hợp sử dụng phức tạp có hơn 7 giao dịch. Một trường hợp sử dụng đơn giản được gán trọng số là 5, một trường hợp sử dụng trung bình được gán trọng số là 10, và sử dụng một trường hợp phức tạp được một trọng số là 15.

Bảng 4.1b. Độ phức tạp của trường hợp sử dụng và trọng số

Loại trường hợp sử dụng (Use Case type)	Số giao dịch (No of transaction)	Trọng số
Đơn giản (Simple)	<= 3	5
Trung bình (Medium)	4–7	10
Phức tạp (Complex)	>7	15

Tổng (đã được gán trọng số) của tất cả trường hợp sử dụng trong ứng dụng (application).Tức là, đối với mỗi trong ba loại phức tạp, trước tiên bạn tính tích của số trường hợp sử dụng của một loại phức tạp với trọng số tương ứng với loại phức tạp đó. Tổng của ba tích này là Giá trị điểm các trường hợp sử dụng TBF cho ứng dụng. Dùng Bảng 4.1.c để tính điểm các trường hợp sử dụng.

Bảng 4.1c. Bảng tính toán điểm các trường hợp sử dụng (use-case)

STT	Loại	Số trường hợp sử dụng	Điểm của từng loại trường hợp sử dụng
1	Đơn giản	?	?
2	Trung bình	?	?
3	Phức tạp	?	?
Cộng (1+2+3)	**TBF (điểm các trường hợp sử dụng)**	**?**	**?**

Điểm của từng loại trường hợp sử dụng được tính theo công thức:

Điểm của từng loại trường hợp sử dụng = **Số trường hợp sử dụng** x **Trọng số**

3. **Tính giá trị Điểm trường hợp sử dụng trước khi hiệu chỉnh UUCP** (*unadjusted use case points*) như sau:

UUCP = TAW + TBF

4. **Tính Hệ số phức tạp về kỹ thuật - công nghệ TCF:** Hiệu chỉnh UUCP để phản ánh độ phức tạp của dự án và kinh nghiệm của con người đối với dự án. Để làm việc này, đầu tiên bạn tính toán các hệ số phức tạp kỹ thuật - công nghệ TCF (Technical complexity factor) bằng cách xem xét các hệ số và xếp hạng giá trị cho từng hệ số được đưa ra trong Bảng 4.2a.

Bảng 4.2a. Hệ số phức tạp về kỹ thuật - công nghệ

STT	Các hệ số	Trọng số	Giá trị xếp hạng	Kết quả
I	**Hệ số KT-CN (TFactor)**			**?**
1	Hệ thống phân tán (Distributed system)	2	?	?
2	Tính chất đáp ứng tức thời hoặc yêu cầu đảm bảo thông lượng (Response or throughput performance objectives)	1	?	?
3	Hiệu quả sử dụng trực tuyến (End-user efficiency - online)	1	?	?
4	Độ phức tạp của xử lý bên trong (Complex internal processing)	1	?	?
5	Mã nguồn phải tái sử dụng được (Code must be reusable)	1	?	?
6	Dễ cài đặt (Easy to install)	0.5	?	?
7	Dễ sử dụng (Easy to use)	0.5	?	?
8	Khả năng chuyển đổi (Portable)	2	?	?
9	Khả năng dễ thay đổi (Easy to change)	1	?	?
10	Sử dụng đồng thời (Concurrent)	1	?	?
11	Có các tính năng bảo mật đặc biệt (Includes special security features)	1	?	?
12	Cung cấp truy nhập trực tiếp tới các phần mềm của các hãng thứ ba (Provides direct access for third	1	?	?
13	Yêu cầu phương tiện đào tạo đặc biệt cho người sử dụng (Special user training facilities required)	1	?	?
II	**Hệ số phức tạp về KT-CN (TCF)**			**?**

Ý nghĩa của các hệ số thành phần như sau:

Bảng 4.2b. Ý nghĩa của các Hệ số phức tạp về kỹ thuật - công nghệ

STT	Tên hệ số	Mô tả
1	Hệ thống phân tán	Kiến trúc của hệ thống là tập trung hay phân tán? Hệ thống được thiết kế theo mô hình nhiều lớp hay không? Trọng số càng cao tương ứng với hệ thống càng phức tạp.

2	Tính chất đáp ứng tức thời hoặc yêu cầu đảm bảo thông lượng	Thời gian đáp ứng yêu cầu của người sử dụng là nhanh hay chậm? Ví dụ, máy tìm kiếm được đánh trọng số về thời gian đáp ứng yêu cầu cao hơn hệ thống cập nhật tin tức hàng ngày. Trọng số càng cao tương ứng với yêu cầu đáp ứng càng nhanh.
3	Hiệu quả sử dụng	Hệ thống có được thiết kế hướng tới tăng hiệu quả làm việc của người sử dụng hay không? Trọng số càng cao tương ứng với hệ thống đòi hỏi hiệu quả sử dụng càng cao.
4	Độ phức tạp của xử lý bên trong	Hệ thống có sử dụng những thuật toán phức tạp trong xử lý hay không? Hoặc hệ thống được thiết kế để hỗ trợ những quy trình nghiệp vụ phức tạp hay không? Trọng số càng cao tương ứng với hệ thống đòi hỏi các thuật toán xử lý càng phức tạp.
5	Khả năng tái sử dụng mã nguồn	Có yêu cầu phải thiết kế và cài đặt mã theo quy chuẩn để sau đó có thể tái sử dụng hay không? Sử dụng mã nguồn có thể tài sử dụng không những làm giảm thời gian triển khai một dự án còn làm tối ưu thời gian xác định lỗi của một phần mềm. Ví dụ, các chức năng sử dụng thư viện chia sẻ có thể tài sử dụng nhiều lần trong các dự án khác nhau. Trọng số càng cao tương ứng với mức độ yêu cầu về khả năng tái sử dụng mã nguồn càng cao.
6	Dễ cài đặt	Hệ thống có đòi hỏi những thủ tục cài đặt phức tạp hay không? Người sử dụng thông thường có thể tự cài đặt các thành phần của hệ thống phục vụ công việc hay không? Việc cập nhật các bản vá lỗi phần mềm có dễ dàng hay không? Trọng số càng cao tương ứng với mức độ yêu cầu về cài đặt càng dễ dàng.
7	Dễ sử dụng	Hệ thống có dễ sử dụng hay không? Người sử dụng có dễ dàng tiếp cận đối với các tính năng mà hệ thống cung cấp hay không? Tài liệu hướng dẫn sử dụng có dễ dàng tiếp cận hay không? Trọng số càng cao tương ứng với mức độ yêu cầu về sử dụng càng dễ dàng.
8	Khả năng chuyển đổi	Hệ thống có được thiết kế để có thể chạy trên nhiều nền tảng phần cứng hoặc hệ điều hành khác nhau hay không? Ví dụ các trình duyệt web thường được yêu cầu chạy trên nhiều thiết bị khác nhau, như máy tính cá nhân hay điện thoại, và nhiều hệ điều hành khác nhau, như Windows hay Linux. Trọng số càng cao tương ứng với càng nhiều nền tảng được yêu cầu hỗ trợ.
9	Khả năng dễ thay đổi	Hệ thống có được yêu cầu thiết kế có khả năng chỉnh sửa và thay đổi trong tương lai hay không? Trọng số càng cao tương ứng với càng nhiều yêu cầu về thay đổi/chỉnh sửa trong tương lai.
10	Sử dụng đồng thời	Hệ thống có được thiết kế để hỗ trợ nhiều người sử dụng tại cùng một thời điểm hay không? Trọng số càng cao tương ứng với mức độ yêu cầu sử dụng đồng thời càng cao.

11	Có tính năng bảo mật	Hệ thống có được thiết kế những tính năng bảo mật đặc biệt, sử dụng những phương thức bảo mật phức tạp hoặc tự phát triển đoạn mã phục vụ việc bảo mật hay không? Trọng số càng cao tương ứng với mức độ yêu cầu về tính năng bảo mật (cả về số lượng và chất lượng).
12	Cung cấp truy nhập trực tiếp tới phần mềm của các hãng thứ ba	Hệ thống có thể truy cập tới dịch vụ hoặc các giao diện lập trình ứng dụng của các ứng dụng do các nhà phát triển khác thực hiện hay không? Trọng số càng cao tương ứng với khối lượng mã nguồn sử dụng từ các nhà phát triển khác càng lớn (và yêu cầu về độ tin cậy đối với mã nguồn đó càng cao).
13	Đào tạo người sử dụng	Để triển khai hệ thống, có cần việc đào tạo người sử dụng hay không? Việc đào tạo người sử dụng có cần phải sử dụng các công cụ, phương tiện đặc biệt để đào tạo người sử dụng hay không? Trọng số càng cao tương ứng với mức độ yêu cầu đào tạo người sử dụng càng cao.

Các hệ số kỹ thuật – công nghệ có thể làm tăng thời gian dự án. Giá trị xếp hạng của mỗi Hệ số càng lớn, dự án cần nhiều nỗ lực hơn.

Cho mỗi hệ số, nhân giá trị xếp hạng với trọng số của nó từ bảng và cộng những con số này để có được giá trị cho Hệ số kỹ thuật – công nghệ TFactor.

Cụ thể TFactor tại cột Kết quả (đơn vị tính: giá trị) được xác định theo công thức:

$$\textbf{TFactor} = \sum_{i=1}^{13} Q_i^{xephang} \times TS_i$$

Trong đó:

- $Qi^{xephang}$: Giá trị xếp hạng của hệ số thứ i trong 13 hệ số thành phần. Giá trị xếp hạng được xác định trong khoảng từ 0 đến 5 với ý nghĩa:

  0 = Không quan trọng

  3 = Có vai trò tác động trung bình

  5 = Có vai trò tác động mạnh

- TSi: Trọng số tương ứng của hệ số thứ i trong 13 hệ số thành phần

Hệ số phức tạp kỹ thuật - công nghệ TCF được tính bằng cách sử dụng phương trình này:

TCF = 0.6 + (0.01 * TFactor)

Trong đó, 0.6 và 0.01 là trọng số đo chuẩn.

5. **Tương tự như vậy, tính toán các Hệ số phức tạp về môi trường EF** (Environment factor) thông qua Bảng 4.3.

Bảng 4.3. Hệ số phức tạp về môi trường

TT	Các hệ số tác động môi trường	Trọng số	Giá trị xếp hạng	Kết quả
I	**Hệ số tác động môi trường và nhóm làm việc (EFactor)**			**?**
	Đánh giá cho từng thành viên			
1	Có áp dụng quy trình phát triển phần mềm theo mẫu RUP và có hiểu biết về RUP hoặc quy trình phát triển phần mềm tương đương (Familiar with RUP)	1.5	?	?
2	Có kinh nghiệm về ứng dụng tương tự (Application domain experience)	0.5	?	?
3	Có kinh nghiệm về hướng đối tượng (Object-oriented experience)	1	?	?
4	Có khả năng lãnh đạo Nhóm (Lead analyst capability)	0.5	?	?
5	Tính chất năng động (Motivation)	1	?	?
	Đánh giá chung cho Dự án			
6	Độ ổn định của các yêu cầu (Stable requirements)	2	?	?
7	Sử dụng các nhân viên làm bán thời gian (Part-time workers)	-1	?	?
8	Dùng ngôn ngữ lập trình loại khó (Difficult programming language)	-1	?	?
II	**Hệ số phức tạp về môi trường (EF)**			**?**

- Ý nghĩa của các hệ số thành phần như sau:

Bảng 4.3b. Ý nghĩa của các Hệ số phức tạp về môi trường

STT	Tên hệ số	Mô tả
1	Có áp dụng quy trình phát triển theo mẫu RUP và có hiểu biết về RUP	Nhân viên phát triển có hiểu biết hoặc đã từng thực hiện công việc tại các công ty có áp dụng RUP hoặc các quy trình phát triển phần mềm tương đương hay không?
2	Có kinh nghiệm về ứng dụng tương tự	Nhà phát triển đã từng phát triển những ứng dụng tương tự, sử dụng công nghệ tương tự hay chưa?
3	Có kinh nghiệm về hướng đối tượng	Nhà phát triển có hiểu biết về công nghệ hướng đối tượng hay không? Hoặc có sử dụng thành thạo các công cụ phát triển hướng đối tượng hay không?
4	Có khả năng lãnh đạo nhóm	Người đứng đầu của nhóm phát triển có khả năng tổ chức, quản lý và triển khai nhiệm vụ trong nhóm phát triển tốt hay không? Người đứng đầu nhóm phát triển có kinh nghiệm lãnh đạo nhóm trong nhiều dự án hay chưa?
5	Tính chất năng động	Tốc độ giải quyết vấn đề từ lúc tiếp cận bài toán cần giải quyết là nhanh hay chậm?
6	Độ ổn định của các yêu cầu	Việc xác định yêu cầu phần mềm có thuận lợi hay không? Các yêu cầu là rõ ràng hay bất định? Có thường xuyên phải chỉnh sửa lại tài liệu đặc tả yêu cầu phần mềm hay không?
7	Sử dụng nhân viên làm bán thời gian	Nhóm phát triển có sử dụng nhân viên làm bán thời gian hoặc kiêm nhiệm hay không?
8	Dùng ngôn ngữ lập trình loại khó	Nhóm phát triển sử dụng công cụ phát triển đã quen thuộc hay hoàn toàn mới. Nhóm phát triển có cần phải tham gia các khóa học bổ sung để nâng cao kỹ năng sử dụng công cụ phát triển hay không?

Tất cả các Hệ số tác động môi trường và nhóm làm việc, ngoại trừ Hệ số 7 và 8, sẽ làm giảm nỗ lực (thời gian) của dự án. Nghĩa là, nếu các Hệ số từ 1 đến 6 càng lớn, nỗ lực (thời gian) của dự án càng ngắn, nhưng điều này sẽ ngược lại đối với Hệ số 7 và 8.

- Điểm đánh giá trong Bảng 4.3 tại mục I nằm trong khoảng từ 0 đến 5 (chấp nhận điểm đánh giá lẻ 1 chữ số thập phân sau dấu chấm thập phân) với các ý nghĩa như sau:

  0 = Trình độ yếu

  3 = Trung bình

  5 = Giỏi

- Kết quả đánh giá bằng cho điểm tại mục I là cơ sở cho việc xác định Giá trị xếp hạng tại mục II.
- Cho mỗi hệ số, nhân giá trị xếp hạng với trọng số của nó từ bảng và cộng những con số này lại với nhau để có được giá trị cho Hệ số tác động môi trường và nhóm làm việc (EFactor).

  Cụ thể, EFactor trong cột Kết quả (đơn vị tính: giá trị) được xác định theo công thức:

$$\textbf{EFactor} = \sum_{i=1}^{8} \mathbf{M_i}^{xephang} \quad \mathbf{x} \quad \mathbf{TS_i}$$

Trong đó:

+ $M_i^{xephang}$: Giá trị xếp hạng của hệ số thứ i trong 8 hệ số thành phần;

- TSi: Trọng số tương ứng của hệ số thứ i trong 8 hệ số thành phần;

+ Giá trị xếp hạng $Mi^{xephang}$ được đánh giá như bảng 4.3c:

Bảng 4.3c. Xếp hạng các Hệ số phức tạp về môi trường

Thứ tự các hệ số tác động môi trường (i)		Giá trị xếp hạng (Từ 0 đến 5)
Đánh giá cho từng thành viên		
1	Có áp dụng quy trình phát triển phần mềm theo mẫu RUP và có hiểu biết về RUP hoặc quy trình phát triển phần mềm tương đương	0 = Không có kinh nghiệm 3 = Trung bình 5 = Trình độ chuyên gia
2	Có kinh nghiệm về ứng dụng tương tự	0 = Không có kinh nghiệm 3 = Trung bình 5 = Trình độ chuyên gia
3	Có kinh nghiệm về hướng đối tượng	0 = Không có kinh nghiệm 3 = Trung bình 5 = Trình độ chuyên gia
4	Có khả năng lãnh đạo Nhóm	0 = Không có kinh nghiệm 3 = Trung bình 5 = Trình độ chuyên gia

5	Tính chất năng động	0 = Không năng động 3 = Trung bình 5 = Cao
Đánh giá chung cho Nhóm làm việc		
6	Độ ổn định của các yêu cầu	0 = Rất bất định 5 = Không hay thay đổi
7	Sử dụng các nhân viên làm bán thời gian	0 = Không có nhân viên làm bán thời gian 3 = Có nhân viên làm Part-time 5 = Tất cả đều làm Part-time
8	Dùng ngôn ngữ lập trình loại khó	0 = Ngôn ngữ lập trình dễ 3 = Trung bình 5 = Khó

Từ đó, Hệ số phức tạp về môi trường EF được tính từ phương trình sau đây:

EF = 1.4 + (–0.03 * EFactor)

Trong đó, 1.4 và -0.03 là trọng số đo chuẩn.

6. **Sử dụng hai Hệ số này, tính số Điểm trường hợp sử dụng sau hiệu chỉnh AUCP** như sau:

AUCP = UUCP * TCF * EF

Để ước lượng nỗ lực, ta cần biết số giờ cần thiết để thực hiện 1 trường hợp sử dụng (hours per use case point) là bao nhiêu. Gán trung bình, 20 người-giờ (person-hour) cho mỗi AUCP cho toàn bộ vòng đời (life cycle). Điều này sẽ tạo ra một ước lượng thô. Tinh chỉnh lại ước lượng này được làm như sau. Đếm xem có bao nhiêu Hệ số<3 và có bao nhiêu Hệ số>3. Nếu tổng số các Hệ số có giá trị < 3 ít, 20 người giờ mỗi AUCP là phù hợp. Nếu có rất nhiều, sử dụng 28 người giờ mỗi AUCP. Nói cách khác, khoảng 20 đến 28 người-giờ mỗi AUCP, và người quản lý dự án có thể quyết định giá trị sử dụng tùy thuộc vào các yếu tố khác nhau.

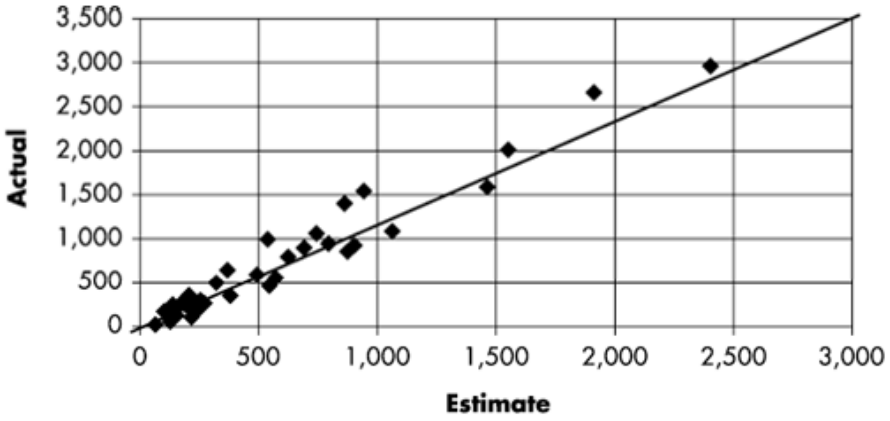
4.2.4 Hiệu quả của phương pháp ước tính

Cách phổ biến để phân tích tính hiệu quả của một phương pháp ước lượng là so sách nỗ lực ước lượng (estimated effort) với nỗ lực thực tế (actual effort). Như đã thảo luận trước đây, so sánh này chỉ đưa ra một ý niệm chung về mức độ chính xác của các ước lượng,

nó không chỉ ra được các ước lượng tối ưu đến mức nào. Để có được thông tin đó, bạn phải nghiên cứu những ảnh hưởng của ước lượng lên những lập trình viên (programmers) (ví dụ, xem họ có bị "kéo dài ra" hoặc bị "không đủ để sử dụng"). Tuy nhiên, so sánh nỗ lực thực tế đã sử dụng và nỗ lực ước lượng không cung cấp một ý niệm về mức độ hiệu quả của phương pháp ước lượng.

Đối với các dự án đã hoàn thành, như đã thảo luận trong Chương 3, cơ sở dữ liệu quy trình bao gồm các thông tin về nỗ lực ước lượng cũng như nỗ lực thực tế. Hình 4.1 là biểu đồ phân tán nỗ lực ước lượng và nỗ lực thực tế cho một số dự án đã hoàn thành.

Hình 4.1. Nỗ lực thực tế (actual) so với ước lượng (estimated)



Biểu đồ cho thấy, phương pháp ước lượng hoạt động khá tốt, hầu hết các điểm dữ liệu gần với dòng 45 độ trong biểu đồ (nếu tất cả các ước lượng chính xác bằng với nỗ lực thực tế, tất cả các điểm sẽ rơi vào đường 45-độ). Các số liệu cũng cho thấy rằng hơn 50% số dự án nằm trong vòng 25% của nỗ lực ước lượng. Tuy nhiên, các dữ liệu cho thấy rằng các ước lượng thường thấp hơn so với nỗ lực thực tế; lưu ý rằng hầu hết các điểm trên đường 45 độ chứ không phải là bên dưới nó. Tức là, con người có xu hướng ước lượng non (underestimate) thường xuyên hơn. Tính trung bình, các nỗ lực thực tế cao hơn 25% so với ước lượng (dự toán). Đánh giá chung: mặc dù cần được cải tiến thêm, nhưng phương pháp ước lượng có hiệu quả tạm được.

4.2.5 Ước lượng nỗ lực của dự án ACIC

Ở đây chúng tôi minh họa phương pháp ước lượng bằng cách trình bày ứng dụng của nó ở ACIC. Dự án ACIC sử dụng phương pháp Điểm trường hợp sử dụng (use case points). Do đó, các phân tích chủ yếu dùng các trường hợp sử dụng (use case) và không dùng các mô-đun. Để phân loại các trường hợp sử dụng, người quản lý dự án sử dụng các tiêu chí phân loại. ***Tuy nhiên, không hoàn toàn giống như phương pháp Điểm trường hợp sử dụng đã được giới thiệu ở phần trước, ACIC chỉ đếm điểm các trường hợp sử dụng (TBF) mà không đếm điểm các tác nhân (TAW).*** Bảng 4.4 liệt kê 26 trường hợp sử dụng và độ phức tạp của chúng.

Để ước lượng nỗ lực cho các loại trường hợp sử dụng khác nhau, người quản lý dự án ở ACIC sử dụng dữ liệu từ dự án Synergy, các mục trong cơ sở dữ liệu quy trình (process database) của dự án này đã được giới thiệu trong Chương 3. Dự án Synergy có 21 trường hợp sử dụng đơn giản, trung bình 11, và 8 phức tạp. Dữ liệu xây dựng chi tiết (detailed build data) của các trường hợp sử dụng khác nhau đã được sử dụng để ước lượng nỗ lực xây dựng trung bình (average build effort) đã dùng. (Nỗ lực xây dựng (thực tế) là khoảng 143 người-ngày. Với nỗ lực xây dựng trung bình là 1 ngày-người, 5 người-ngày, và 8 người-ngày tương ứng cho một trường hợp sử dụng đơn giản, trung bình, và phức tạp. Nếu tính theo mức trung bình này thì tổng sẽ là 140 người-ngày – một con số gần với giá trị thực (143)). Bảng 4.5 cho thấy nỗ lực xây dựng trung bình cho mỗi loại trường hợp sử dụng và tổng nỗ lực xây dựng.

Bảng 4.4. Các trường hợp sử dụng trong dự án ACIC

Số STT của trường hợp sử dụng	Mô tả	Loại phức tạp
1	Màn hình điều khiển *(Navigate Screen)*	Phức tạp
2	Cập nhật thông tin chi tiết cá nhân *(Update Personal Details)*	Trung bình
3	Thêm địa chỉ *(Add Address)*	Trung bình
4	Cập nhật địa chỉ *(Update Address)*	Phức tạp
5	Xóa địa chỉ *(Delete Address)*	Phức tạp
6	Thêm số điện thoại *(Add Telephone Number)*	Trung bình
7	Cập nhật số điện thoại *(Update Telephone Number)*	Phức tạp
8	Xóa số điện thoại *(Delete Telephone Number)*	Phức tạp

9	Thêm email *(Add E-mail)*	Trung bình
10	Cập nhật email *(Update E-mail)*	Trung bình
11	Xóa email *(Delete E-mail)*	Trung bình
12	Cập nhật thông tin công việc của đối tác *(Update Employment Details of a Party)*	Trung bình
13	Cập nhật thông tin tài chính của đối tác *(Update Financial Details of a Party)*	Trung bình
14	Cập nhật thông tin chi tiết về tài khoản *(Update Details of an Account)*	Trung bình
15	Các hoạt động bảo trì một tài khoản *(Maintain Activities of an Account)*	Phức tạp
16	Bảo trì sổ ghi nhớ của một tài khoản *(Maintain Memos of an Account)*	Đơn giản
17	Xem lịch sử chi tiết của đối tác *(View History of Party Details)*	Phức tạp
18	Xem lịch sử chi tiết của tài khoản *(View History of Account Details)*	Phức tạp
19	Xem lịch sử mức chọn lựa và các tùy chọn dịch vụ *(View History of Option Level and Service Options)*	Đơn giản
20	Xem lịch sử của các hoạt động và sổ ghi nhớ *(View History of Activities and Memos)*	Đơn giản
21	Xem lich sử của các vai trò *(View History of Roles)*	Phức tạp
22	Xem các chi tiết về tài khoản *(View Account Details)*	Đơn giản
23	Xem các cổ phần của một tài khoản *(View Holdings of an Account)*	Phức tạp
24	Xem các đơn đặt hàng chưa được xử lý của một tài khoản *(View Pending Orders of an Account)*	Phức tạp
25	Đóng/Kích hoạt một tài khoản *(Close/Reactivate Account)*	Đơn giản
26	Làm cập nhật thông minh cho các đối tác kinh doanh của ACIC *(Make Intelligent Update to Business Partners of ACIC)*	Phức tạp

Bảng 4.5. Nỗ lực xây dựng cho dự án ACIC

Loại trường hợp sử dụng	**Nỗ lực (người-ngày/1 trường hợp sử dụng)**	**Số lượng**	**Tổng nỗ lực xây dựng (người-ngày)**
Đơn giản	1	5	5
Trung bình	5	9	45
Phức tạp	8	12	96
Tổng			**146**

Để ước lượng phân phối nỗ lực cho các giai đoạn, người quản lý dự án đã sử dụng tỷ lệ phân phối của dự án Synergy. Bởi vì các dự án trước đó đã không có giai đoạn yêu cầu,

phân phối đã được sửa đổi. Bảng 4.6 trình bày nỗ lực ước lượng cho từng giai đoạn và tổng số.

Bảng 4.6. Nỗ lực ước lượng cho dự án ACIC

Hoạt động	**Nỗ lực được ước lượng (Estimated Effort)**	
	Người-ngày	**% của Tổng nỗ lực (% of Total Effort)**
Requirements	50	10
Design	60	12
Build	146	29
Integration testing	35	7
Regression testing	10	2
Acceptance testing	30	6
Project management	75	15
Configuration management	16	3
Training	50	10
Others	40	6
Nỗ lực được ước lượng (Estimated effort)	**501**	**100%**

Trong dự án này, ngoài việc ước lượng theo tiếp cận từ dưới lên (bottom-up), người quản lý dự án sử dụng phương pháp Điểm trường hợp sử dụng. Như mô tả ở trên, đầu tiên UUCPs được xác định từ các trường hợp sử dụng theo quy định: 5 điểm cho mỗi trường hợp sử dụng đơn giản, 10 điểm cho mỗi trường hợp sử dụng trung bình, và 15 cho mỗi trường hợp sử dụng phức tạp. Số lượng các trường hợp sử dụng đơn giản, trung bình, và phức tạp tương ứng là 5, 9, và 12, do đó UUCP là:

$$UUCP = 5 * 5 + 9 * 10 + 12 * 15 = 295$$

Để tính vào các hệ số khác, đầu tiên người quản lý dự án ACIC xếp hạng giá trị cho các hệ số phức tạp về kỹ thuật – công nghệ. Ông đã xếp hạng giá trị như sau (theo thứ tự được đưa ra trong Bảng 4.2): 4, 3, 5, 3, 4, 5, 5, 0, 4, 1, 2, 0, và 5, kết quả của TFactor là 40 (8 + 3 + 5 + 3 + 4 + 2.5 + 2.5 + 0 + 4 + 1 + 2 + 0 + 5) và TCF là 1.0. Tiếp theo, ông xếp hạng giá trị cho các hệ số phức tạp về môi trường như sau (theo thứ tự được đưa ra trong Bảng 4.3: 3, 1, 3, 4, 5, 5, 0, và 3; kết quả của EFactor là 22 (4.5 + 0.5 + 3 + 2 + 5 + 10 + 0 - 3) và EF là 0.74. Từ đó, ông tính toán tổng số Điểm trường hợp sử dụng.

AUCP = 295 * 1.0 * 0.74 = 218.3

Sử dụng nỗ lực chuẩn là 20 người-giờ cho mỗi AUCP, ông đã nhận được nỗ lực là:

218 * 20 = 4,360 người-giờ = 499 người-ngày (khi 8.75 giờ/ngày)

hoặc

513 người-ngày (khi 8.5 giờ/ngày)

Những ước lượng này đã gần với các ước lượng trước đó một cách đáng kinh ngạc, làm tăng sự tự tin cho người quản lý dự án trong ước lượng. Hơn nữa, ở tất cả các cột mốc (milestones), nỗ lực lớn hơn, khi so với kế hoạch, là rất nhỏ, như bạn sẽ thấy khi phân tích các cột mốc được đưa ra trong Chương 11.

Như đã đề cập trước đây, trong dự án này, quy trình lặp đi lặp lại của RUP được sử dụng. Bởi vì các giai đoạn thiết kế, phân tích và xây dựng (build) đã được trải ra nhiều vòng lặp (iterations), ước lượng nỗ lực theo-giai đoạn sẽ không cung cấp một đầu vào trực tiếp cho công tác lập kế hoạch. Để lập kế hoạch, người quản lý dự án phải ước lượng các nỗ lực cho các vòng lặp khác nhau. Để có được điều này, ông bắt đầu với ước lượng tổng thể như đã xác định trước đây. Ước lượng cho các yêu cầu (requirements) đã được chia thành hai giai đoạn: giai đoạn khởi xướng (project initiation) và giai đoạn bắt đầu (inception phase). Nỗ lực cho thiết kế (design), xây dựng (build) và kiểm thử đã được chia nhỏ ra thành giai đoạn phát thảo tỉ mỉ (elaboration phase) và xây dựng (construction phase), dựa trên các trường hợp sử dụng được lựa chọn trong các vòng lặp lại khác nhau và các hướng dẫn được đưa ra trong phương pháp RUP. Quản lý dự án, quản lý giá (CM), và các chi phí còn lại khác được làm tương tự. Bảng 4.7 cho thấy sự phân phối nỗ lực cho các vòng lặp khác nhau.

Bảng 4.7. Phân phối nỗ lực cho các vòng lặp trong dự án ACIC

Vòng lặp (Iteration)	**Nỗ lực ước lượng (Estimated Effort)**	
	Người-ngày (Person-days)	**% của nỗ lực tổng cộng (% of Total Effort)**
Project initiation	25	5
Inception phase	24	5
Elaboration phase: Iteration 1	45	9

Elaboration phase: Iteration 2	34	7
Construction phase: Iteration 1	27	5
Construction phase: Iteration 2	24	5
Construction phase: Iteration 3	21	4
Transition phase	110	22
Project closure	10	2
Project management	75	15
Configuration management	16	3
Training	50	10
Others	40	8
Tổng nỗ lực ước lượng (Total estimated effort)	**501 người-ngày (person-days)**	**100%**

4.3 LẬP THỜI GIAN BIỂU (SCHEDULING)

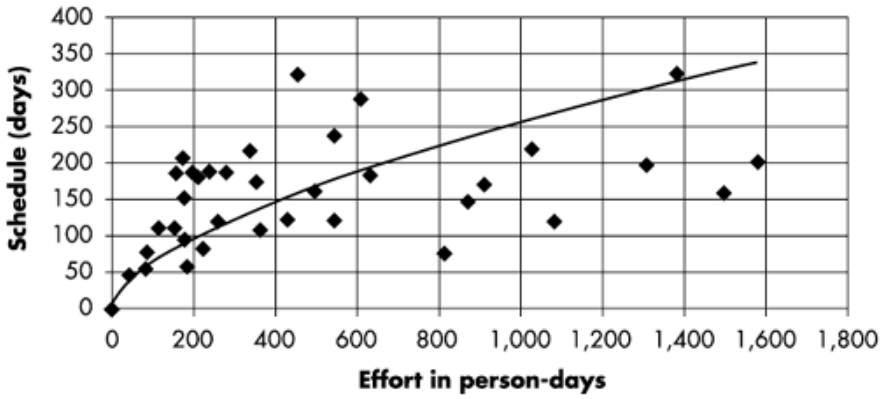
Lập thời gian biểu tại Infosys có thể được chia thành hai hoạt động nhỏ (subactivities): xác định thời gian biểu tổng thể (thời gian hoàn thành cả dự án) với các cột mốc lớn, và xây dựng thời gian biểu chi tiết cho các công việc/nhiệm vụ khác nhau.

4.3.1 Tổng quan về Lập thời gian biểu

Như đã thảo luận trong phần đầu chương này, bạn có thể đạt được nhiều linh hoạt trong việc xác định thời gian biểu bằng cách kiểm soát mức nhân sự (staffing), nhưng sự linh hoạt này có giới hạn. Vì khả năng linh hoạt này mà việc xây dựng các hướng dẫn nghiêm ngặt cho việc lập kế hoạch có thể không được ưa thích, các hướng dẫn nghiêm ngặt làm mất lợi thế của tính linh hoạt để được thông qua cho dự án hoặc khách hàng. Hơn nữa, thời gian biểu của dự án thường được xác định trong bối cảnh lớn hơn các kế hoạch kinh doanh (kế hoạch kinh doanh thường áp đặt một số yêu cầu tiến độ). Bất cứ khi nào có thể, bạn nên khai thác tính linh hoạt của thời gian biểu thời gian để đáp ứng các yêu cầu như vậy. Một phương pháp là sử dụng các hướng dẫn lập thời gian biểu để kiểm tra nhiều hơn nữa tính khả thi của thời gian biểu hơn là chỉ đi xác định thời gian biểu mà thôi.

Hình 4.2 cho thấy biểu đồ phân tán của thời gian biểu và nỗ lực cho một số dự án đã hoàn thành tại Infosys, cùng với một đường cong phi tuyến hồi quy vừa khít (thích hợp) cho biểu đồ phân tán.

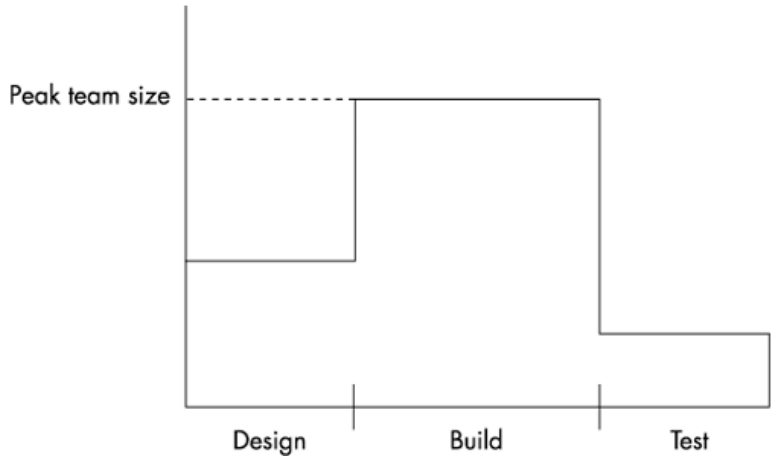
Hình 4.2. Thời gian biểu trình như là hàm của nỗ lực



Phương trình của đường cong trong Hình 4.3 là

$$\textbf{schedule} = 23.46\ (\textbf{effort})^{0.313}$$

Hình 4.3. Đường kết nối nhân (Manpower) trong một dự án điển hình



Sự phân bố của các điểm là bằng chứng cho thấy rằng thời gian hoàn thành không phải là một hàm duy nhất của nỗ lực. Tuy nhiên, thời gian hoàn thành đã được xác định có thể được sử dụng như là một hướng dẫn hoặc để kiểm tra tính hợp lý của thời gian hoàn thành được ước lượng – cái có thể được quyết định dựa trên các yếu tố khác. Tương tự như vậy, các dữ liệu về thời gian hoàn thành và nỗ lực từ các dự án tương tự có thể được sử dụng để kiểm tra tính hợp lý của bất kỳ thời gian hoàn thành được đề nghị nào.

Những người quản lý dự án thường sử dụng quy tắc thực nghiệm để tính (rule of thumb), được gọi là *kiểm tra căn bậc hai* (*square root check*), để kiểm tra thời gian hoàn thành của một dự án quy mô trung bình. Nguyên tắc là thời gian hoàn thành dự án nên được đề nghị gần bằng căn bậc hai của tổng nỗ lực (effort) được tính theo tháng-người; thời gian hoàn thành dự án có thể được đáp ứng nếu $\sqrt{\text{effort}}$ người được phân công cho dự án. Ví dụ, nếu nỗ lực ước lượng là 50 người-tháng, một thời gian biểu khoảng 7 đến 8 tháng sẽ phù hợp với khoảng 7 đến 8 người làm việc toàn thời gian (full-time).

Bởi vì mối quan hệ giữa thời gian hoàn thành dự án và số người, thời gian hoàn thành chỉ được chấp nhận nếu người đứng đầu của các đơn vị của công ty (mà dự án thuộc về) đồng ý để cung cấp đủ số người cần thiết. Nếu số người cần thiết chưa có sẵn, thời gian hoàn thành dự án phải được điều chỉnh. Các phụ thuộc (dependencies) của dự án cũng được kiểm tra trước khi thời gian hoàn thành được chấp nhận. Nếu việc thực hiện dự án phụ thuộc vào các yếu tố bên ngoài (chẳng hạn như phải hoàn thành một dự án khác hoặc chờ đợi sự có mặt của một phần mềm nào đó), thời gian hoàn thành phải được điều chỉnh để thích ứng với những yếu tố này.

Một khi thời gian tổng thể của dự án được ấn định cố định, thời gian cho các cột mốc chính phải được xác định. Để xác định các cột mốc, trước tiên bạn phải hiểu đường kết nối nhân lực thường diễn ra trong một dự án. Số lượng người trong một dự án phần mềm có xu hướng theo đường cong Rayleigh [9],[11]. Ở lúc bắt đầu và kết thúc, ít người làm việc trong dự án; nhóm đạt được kích thước tối đa (PTS - peak team size) ở một nơi nào đó gần giữa dự án. Cách thức hoạt động này xảy ra bởi vì chỉ có vài người là cần thiết trong giai đoạn đầu của phân tích yêu cầu và thiết kế. Yêu cầu nguồn nhân lực đạt mức tối đa trong suốt thời gian cài đặt mã và kiểm thử đơn vị (unit testing). Một lần nữa, trong suốt thời gian kiểm thử hệ thống và tích hợp hệ thống (system testing and integration), ít người hơn được cần. Trong nhiều trường hợp, mức phân công nhân sự không thay đổi thường xuyên, nhưng xấp xỉ của đường cong Rayleigh được sử dụng: phân công một vài người lúc bắt đầu, nhóm có đông người nhất trong suốt giai đoạn xây dựng (build phase), và sau đó để lại một vài người để làm tích hợp và kiểm thử hệ thống. Nếu bạn xem xét thiết kế, xây dựng (build) và kiểm thử nghiệm là ba giai đoạn chính để thực hiện các yêu cầu, đường kết nối nhân lực trong các dự án thường tương tự như hàm được hiển thị trong Hình 4.3.

Ở Infosys, tiếp cận này thường được áp dụng để phân công nhân sự. Ít người hơn được giao nhiệm vụ vào các giai đoạn bắt đầu và kết thúc, nhưng sẽ phân công số người tối đa vào suốt giai đoạn xây dựng. Trong suốt giai đoạn xây dựng, dự án thường đạt mức PTS.

Để dễ dàng lập kế hoạch, đặc biệt là đối với các dự án nhỏ hơn, tất cả mọi người thường được phân công ngay để cung làm việc với nhau từ lúc bắt đầu của dự án. Cách tiếp cận này có thể dẫn đến một số người sẽ rảnh ở đầu và cuối dự án. Thời gian rảnh này (slack time) thường được sử dụng cho đào tạo. Đào tạo ở cấp dự án nói chung cần thiết về các công nghệ sắp sửa được sử dụng và lĩnh vực nghiệp vụ (business domain) của dự án, và đào tạo này cần khá nhiều nỗ lực (thời gian), như có thể nhìn thấy được qua phân phối nỗ lực được cung cấp trong cơ sở dữ liệu về quy trình. Tương tự như vậy, thời gian rảnh xuất hiện ở giai đoạn cuối có thể được sử dụng để viết tài liệu và các nhiệm vụ kết thúc khác.

Phân phối thời gian hoàn thành dự án thì khác với phân phối nỗ lực. Đối với ba giai đoạn chính, tỷ lệ phần trăm của thời gian được dùng trong giai đoạn xây dựng thì nhỏ hơn so với tỷ lệ phần trăm của nỗ lực được dùng bởi giai đoạn này bởi vì nó liên quan đến nhiều người hơn. Tương tự như vậy, tỷ lệ phần trăm của thời gian được sử dụng trong giai đoạn thiết kế và kiểm thử thì lớn hơn tỷ lệ phần trăm nỗ lực của chúng. Thời gian chính xác được sử dụng phụ thuộc vào đường kết nối nhân lực đã được hoạch định. Khi biết nỗ lực ước lượng cho một giai đoạn, bạn có thể xác định thời gian của giai đoạn khi bạn biết đường kết nối nhân lực.

Nói chung, thiết kế cần khoảng 40% thời gian (20% cho thiết kế mức cao và 20% cho thiết kế chi tiết), xây dựng cần khoảng 40%, và tích hợp và kiểm thử hệ thống cần 20% còn lại. Thông thường thì đường kết nối nhân lực khoảng 01:02:01 cho thiết kế, xây dựng, tích hợp và thử nghiệm (được biết phân phối nỗ lực của các giai đoạn này là 01:04:01). Các hướng dẫn kiểu này giúp ích trong việc kiểm tra các cột mốc đã được thiết lập dựa trên các ràng buộc khác.

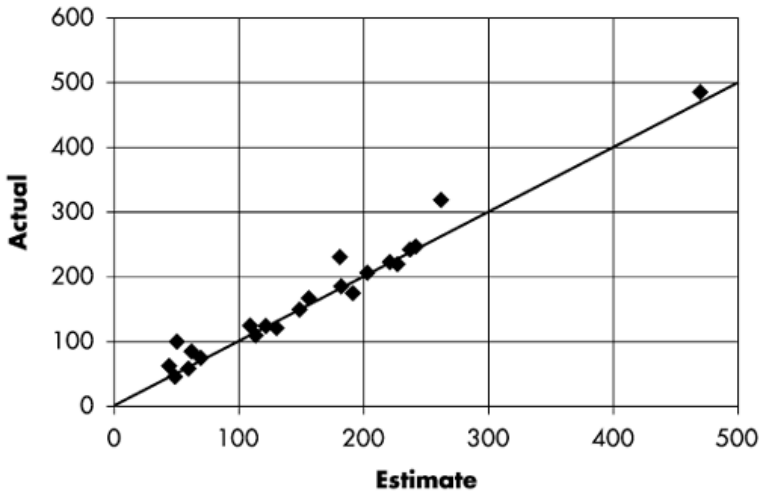
Điều quan trọng cần được biết là ngay cả khi một người được phân công toàn thời gian (full-time) cho một dự án, việc thực hiện các nhiệm vụ khác cũng tiêu tốn thời gian nhưng không đóng góp cho dự án. Những nhiệm vụ này bao gồm nghỉ phép, hoạt động đoàn thể, đào tạo nói chung (không phải cho dự án cụ thể), xem xét lại (review) các dự án khác, v.v.

4.3.2 Hiệu quả của phương pháp

Khi ước lượng nỗ lực, một cách để kiểm tra thời gian hoàn thành dự án là vẽ biểu đồ so sánh thời gian hoàn thành thực tế với thời gian ước lượng để xem các điểm có gần với đường 45-độ không. Nếu tất cả các điểm rơi rất gần với đường 45-độ, phương pháp ước lượng có thể được coi là có hiệu quả. Hình 4.4 cho thấy biểu đồ này cho các dự án phát triển đã hoàn thành trước đây.

Như bạn có thể thấy, kết quả của phương pháp ước lượng thời gian hoàn thành dự án là khá gần với thời gian thực tế. Tuy nhiên, hãy ghi nhớ rằng các yếu tố khác (được thảo luận trong phần 4.2) cũng có thể chỉ ra thời gian dự kiến này có khả thi không.

Hình 4.4. Thời gian biểu thực tế so với ước lượng



4.3.3 Lập kế hoạch Chi tiết

Một khi các cột mốc và số lượng nhân sự (người tham gia) đã được ấn định cố định, đây là lúc đi xây dựng kế hoạch chi tiết. Người quản lý dự án phân nhỏ các công việc/nhiệm vụ ra thành các công việc nhỏ (xếp lịch được) hơn theo thứ bậc kế thừa. Tại mỗi công việc chi tiết, người quản lý dự án ước lượng thời gian cần thiết để hoàn thành nó và phân công nhận sự thích hợp để có thể hoàn thành thời gian biểu tổng thể. Trong việc phân công nhân sự, người quản lý dự án xem xét đến nhiều yếu tố khác nhau như kế hoạch nghỉ phép của các thành viên trong nhóm, kế hoạch cá nhân của họ và con đường sự nghiệp, các kỹ năng của họ và kinh nghiệm được cần, yêu cầu được đào tạo và tư vấn,

tính chất quan trọng của công việc/nhiệm vụ, và giá trị của kinh nghiệm sẽ đạt được từ nhiệm vụ đó.

Ở mỗi cấp độ tinh chỉnh lại, người quản lý dự án kiểm tra số nỗ lực được cần cho toàn bộ các công việc trong kế hoạch chi tiết và so sánh với nỗ lực ước lượng. Nếu cần thiết, điều chỉnh lại các ước lượng chi tiết. Ví dụ, người quản lý dự án sẽ phân chia giai đoạn thiết kế chi tiết ra thành nhiều công việc/nhiệm vụ nhỏ khi đang phát triển các thiết kế chi tiết cho từng mô-đun, xem xét lại thiết kế chi tiết, sửa chữa các lỗi được tìm thấy, v.v., và người quản lý dự án có thể tiếp tục phân chia chi tiết hơn nữa. Sau đó, người quản lý sẽ lập thời gian biểu cho những công việc này và phân công nhân sự cho một vài khoảng thời gian.

Nếu kế hoạch chi tiết này không phù hợp với thời gian biểu và nỗ lực ước lượng tổng thể, người quản lý dự án phải thay đổi thời gian biểu chi tiết. Nếu người quản lý dự án nhận thấy rằng thời gian biểu chi tiết được xem là tốt nhất cũng không thể tương thích với nỗ lực và thời gian của các cột mốc, người quản lý dự án phải sửa đổi lại các ước lượng trước đó. Do đó, lập kế hoạch là một quá trình lặp đi lặp lại.

Nói chung, người quản lý dự án tinh chỉnh lại các công việc/nhiệm vụ đến một mức độ để mà các hoạt động ở mức thấp nhất có thể được lên kế hoạch để chiếm không nhiều hơn một vài ngày cho một người đơn lẻ. Người quản lý dự án cũng thêm vào các hoạt động chung, chẳng hạn như quản lý dự án, điều phối, quản lý cơ sở dữ liệu, và quản lý cấu hình. Những hoạt động này ít có tác dụng trực tiếp đến thời gian biểu bởi chúng là những công việc đang diễn ra chứ không phải là các hoạt động xếp lịch được. Tuy nhiên, tuy nhiên chúng tiêu tốn nhân sự và do đó phải được ghi nhận trong thời gian biểu của dự án.

Hiếm khi người quản lý dự án tạo kế hoạch chi tiết cho toàn bộ dự án với chỉ một lần. Sau khi thời gian biểu tổng thể được ấn định cố định, người quản lý dự án có thể chi tiết hóa mỗi giai đoạn chỉ vào lúc bắt đầu của giai đoạn đó.

Để lập kế hoạch chi tiết, người quản lý dự án thường xuyên sử dụng Microsoft Project (MSP) hoặc một bảng tính. Đối với mỗi hoạt động cấp thấp nhất, họ quy định nỗ lực, khoảng thời gian, ngày bắt đầu, ngày kết thúc, và các nhân sự. Đối với mỗi hoạt động, họ cũng xác định mã hoạt động (được thảo luận thêm trong Chương 7), mã chương trình, và mã mô-đun. Họ cũng có thể xác định sự phụ thuộc giữa các hoạt động (ví dụ, bạn chỉ có thể tiến hành một kế hoạch kiểm thử đơn vị cho một chương trình chỉ sau khi nó đã được

cài đặt mã) hoặc một sự phụ thuộc liên quan đến nhân sự (cùng một nhân sự được giao hai nhiệm vụ).

Một thời gian biểu chi tiết của dự án không bao giờ là tĩnh. Thay đổi có thể được cần vì tiến bộ thực tế của dự án có thể khác như những gì đã được lên kế hoạch, bởi vì các công việc mới được thêm vào để đáp ứng các thay đổi yêu cầu, hoặc do các tình huống bất khả kháng khác. Thay đổi nên chỉ được thực hiện khi cần thiết.

Thời gian biểu cuối cùng, như được ghi nhận trong Microsoft Project hoặc một số công cụ khác, là một tài liệu lưu kế hoạch dự án "sống" nhất. Trong suốt dự án, nếu kế hoạch này cần phải được thay đổi và các công việc bổ sung phải được thực hiện, sau khi quyết định được đưa ra, bất kỳ thay đổi nào đều phải được ghi nhận vào trong thời gian biểu chi tiết. Do đó, thời gian biểu chi tiết sẽ trở thành tài liệu chính để theo dõi các hoạt động và thời gian hoàn thành. Thời gian biểu chi tiết cũng là một đầu vào quan trọng cho việc giám sát dự án, được thảo luận trong Chương 11.

4.3.4 Thời gian biểu của dự án ACIC

Hãy xem xét ví dụ về dự án ACIC như đã thảo luận trước đây. Nỗ lực ước lượng cho dự án ACIC là 501 người-ngày, hoặc khoảng 24 người-tháng. Khách hàng đã cho khoảng 5.5 tháng để hoàn thành dự án (từ ngày 15/5 đến 3/11). Bởi vì giá trị này lớn hơn căn bậc hai của nỗ lực trong tháng-người, và bởi vì việc thu thập yêu cầu (requirements) phải được hoàn thành trước khi dự án bắt đầu, thời gian biểu trình này đã được chấp nhận. (Yêu cầu nhân sự cho thời gian biểu này cũng đã được ước lượng và đã được đưa vào trong kế hoạch quản lý dự án trong Chương 8.)

Những cột mốc đã được xác định bằng cách sử dụng nỗ lực ước lượng cho giai đoạn và một ước lượng số nhân sự cần thiết để thực hiện giai đoạn này. Trong dự án ACIC, người quản lý dự án liệt kê các hoạt động chủ yếu trong từng giai đoạn và phân công nhân sự cho chúng. Từ việc phân công này, ông đã xác định kế hoạch tổng thể và nỗ lực cho giai đoạn. Nếu nỗ lực tổng thể cho giai đoạn không phù hợp với nỗ lực đã được ước lượng, ông phải điều chỉnh lại sự phân công cho đến khi tổng nỗ lực phù hợp với nỗ lực đã được ước lượng. Khi đó thời gian biểu tổng thể, như được thu được từ sự phân công các hoạt động, được sử dụng làm thời gian biểu cho giai đoạn. (Các cột mốc đã được xác định trong kế hoạch quản lý dự án được trình bày trong Chương 8). Bảng 4.8 cho thấy thời

gian biểu mức cao của dự án ACIC. Thời gian biểu này sẽ tự động được thu được từ kế hoạch chi tiết cuối cùng của dự án.

Trong bảng, ID công việc là số thứ tự được gán trong Microsoft Project. Các ID công việc cho thấy rằng tổng số công việc trong thời gian biểu cuối cùng là hơn 330 và mỗi công việc trong số những công việc cấp cao đã có nhiều công việc ở thấp hơn được xếp lịch dưới nó. Công việc đầu tiên là tổng thể dự án, với thời gian hoàn thành khoảng 140 ngày và số nỗ lực là 560 người-ngày. (Thời gian biểu này là thời gian biểu sau cùng của dự án, trong đó đã đưa vào các thay đổi; hai công việc cuối cùng trong bảng này là hai thay đổi lớn)

Bảng 4.8. Thời gian biểu mức cao cho dự án ACIC

ID công việc	Công việc (Task)	Khoảng thời (Duration) tính theo ngày	Nỗ lực tính theo người-ngày (person-days)	Ngày bắt đầu (start date)	Ngày kết thúc (end date)
1	ACIC development schedule	139.56	559.93	4/3/00	11/3/00
2	Project initiation activities	33.78	24.2	5/4/00	6/23/00
29	Regular activities	87.11	35.13	6/5/00	10/16/00
74	Training	95.11	49.37	5/8/00	9/29/00
99	Organization activities	76.89	12.9	5/22/00	9/15/00
104	Knowledge sharing initiative	78.22	19.56	6/2/00	9/30/00
110	Inception phase activities	26.67	22.67	4/3/00	5/12/00
114	Elaboration Iteration 1	27.56	55.16	5/15/00	6/23/00
157	Elaboration Iteration 2	8.89	35.88	6/26/00	7/7/00
198	Construction Iteration 1	8.89	24.63	7/10/00	7/21/00
228	Construction Iteration 2	6.22	28.22	7/20/00	7/28/00
256	Construction Iteration 3	6.22	27.03	7/31/00	8/8/00
290	Transition phase activities	56	179.62	8/9/00	11/3/00
323	Window resized release of 2.0 code	26.67	39.11	8/14/00	9/22/00
331	Back-end mainframe work for 3.0	4.44	6.44	8/14/00	8/18/00

Thời gian biểu mức cao này không thích hợp cho việc phân bổ nhân sự và hoạch định chi tiết. Trong quá trình lập kế hoạch chi tiết, các công việc lớn được chia ra thành các hoạt động nhỏ hơn có thể xếp lịch được. Bằng cách này, thời gian biểu cũng trở thành một

danh sách liệt kê các mục cần kiểm tra (checklist) cho các công việc của dự án. Như đã đề cập trước đây, việc "phân rã" các hoạt động ở mức cao không được thực hiện hoàn toàn một lần ở lúc bắt đầu nhưng nó sẽ được thực hiện nhiều lần trong suốt thời gian thực hiện dự án.

Bảng 4.9 cho thấy một phần của thời gian biểu chi tiết của vòng lặp 1 của giai đoạn xây dựng (construction-iteration 1) của dự án ACIC. Cho mỗi hoạt động, bảng xác định cụ thể các mô-đun, chương trình, mã hoạt động, số nỗ lực, thời gian, v.v. Cột Mô-đun và cột Chương trình trình bày cho các module và chương trình mà công việc đang được thực hiện. Mã hoạt động trình bày hoạt động đang được thực hiện. (Các Mã hoạt động được đánh theo chuẩn công ty, sẽ được thảo luận thêm trong Chương 7.)

Bảng 4.9. Phần của thời gian biểu chi tiết cho dự án ACIC

Mô-đun (Module)	Chương trình (Program)	Mã hoạt động (Activity Code)	Công việc/nhiệm vụ Task	Thời gian hoàn thành (Duration: days)	Effort (days)	Ngày bắt đầu Start Date	Ngày kết thúc End Date	% Hoàn thành (Complete)	Nhân sự (Resource Initials
—	—	PRS	Requirements	8.89 days	1.33 days	7/10/00 8:00	7/21/00 17:00	100%	BB, BJ)
—	—	PDDRV	Design review	1 day	0.9 days	7/11/00 8:00	7/12/00 9:00	100%	BB, BJ, SB
—	—	PDDRW	Rework after design review	1 day	0.8 days	7/12/00 8:00	7/13/00 9:00	100%	BJ, SB
History	UC17	PCD	View history of party details, UC17	2.67 days	1.87 days	7/10/00 8:00	7/12/00 17:00	100%	HP
History	UC7	PCDRV	Code walkthrough, UC17	0.89 days	0.27 days	7/14/00 8:00	7/14/00 17:00	100%	BJ, DD
History	UC19	PCDRV	Code walkthrough, UC19	0.89 days	0.27 days	7/14/00 8:00	7/14/00 17:00	100%	BJ, DD
—	—	PCDRW	Rework after code walkthrough	0.89 days	2.49 days	7/17/00 8:00	7/17/00 17:00	100%	DD, SB, HP, BJ
—	—	PUTRW	Rework after testing	0.89 days	0.71 days	7/18/00 8:00	7/18/00 17:00	100%	BJ, SB, DD, HP
History	UC17	PUT	Test, UC 17	0.89 days	0.62 days	7/18/00 8:00	7/18/00 17:00	100%	SB
History	UC19	PUT	Test, UC 19	0.89 days	0.62 days	7/18/00 8:00	7/18/00 17:00	100%	HP
Configur ation	—	PCM	Reconciliation	0.89 days	2.49 days	7/19/00 8:00	7/19/00 17:00	100%	BJ, DD, SB, HP
Manage ment	—	PPMPT	Scheduling and tracking	7.11 days	2.13 days	7/10/00 8:00	7/19/00 17:00	100%	BB
Quality	—	PPMPT	Milestone analysis	0.89 days	0.62 days	7/19/00 8:00	7/19/00 17:00	100%	BB

Đôi khi, người tiền nhiệm (predecessors) của hoạt động cũng được chỉ định, mặc dù chúng được bỏ qua ở đây. Thông tin này giúp trong việc xác định đường giới hạn (critical path) và nhân sự giới hạn (critical resources).

Đối với mỗi công việc, bao nhiêu % đã được hoàn thành được trình bày trong cột % Hoàn thành. Những thông tin này được sử dụng để theo dõi hoạt động (activity tracking), sẽ tiếp

tục được thảo luận trong Chương 11. Thời gian biểu chi tiết cũng xác định các nhân sự mà công việc được giao.

Số của hoạt động (activity number) đã được bỏ qua khỏi Bảng 4.9. Như Bảng 4.8 cho thấy, đã có hơn 330 mục trong thời gian biểu sau cùng của dự án ACIC, các công việc mức thấp nhất sẽ được xếp lịch.

4.4 TÓM TẮT

Mục tiêu cơ bản của ước lượng nỗ lực (dự toán nỗ lực) là để tạo ra các ước lượng hợp lý tại hầu hết các thời điểm. Sau đây là những bài học quan trọng từ các tiếp cận và phương pháp ước lượng và lập kế hoạch được sử dụng tại Infosys:

- Sử dụng các dữ liệu quá khứ để ước lượng. Thích dữ liệu từ các dự án tương tự và sau đó dữ liệu về khả năng chung của quy trình. Sử dụng một mô hình để ước lượng, nhưng cho phép sự linh hoạt để điều chỉnh ước lượng cho phù hợp với các yếu tố cụ thể của dự án.

- Sử dụng các mô hình khác nhau trong các tình huống khác nhau. Ước lượng từ dưới lên thì hiệu quả khi chi tiết của dự án được biết. Sử dụng cách tiếp cận từ trên xuống nếu bạn có thể ước lượng kích thước và năng suất, và phương pháp Điểm trường hợp sử dụng khi dự án sử dụng một phương pháp phát triển phần mềm dựa trên trường hợp sử dụng.

- Đối với thời gian biểu tổng thể và các cột mốc mức cao, sử dụng sự linh hoạt hiện có để đáp ứng được những ngày đề nghị. Một khi kế hoạch tổng thể và các cột mốc được ấn cố định, hãy xác định yêu cầu về nhân sự cho từng giai đoạn dựa vào nỗ lực ước lượng của từng giai đoạn.

- Thời gian biểu chi tiết thì luôn động, hãy chú ý các vấn đề liên quan đến con người trong khi phân công nhiệm vụ. Không cần thiết để tinh chỉnh toàn bộ thời gian biểu ở lúc bắt đầu. Bạn có thể chi tiết hóa các công việc/nhiệm vụ trong kế hoạch tổng thể khi có yêu cầu.

- Thời gian biểu chi tiết bao gồm một danh sách các hoạt động được lập kế hoạch cho dự án. Ghi nhận tất cả các hoạt động được lập kế hoạch trong dự và sau đó sử dụng nó để theo dõi các hoạt động.

Từ quan điểm CMM, phương pháp ước lượng nỗ lực và thời gian biểu thích hợp là một yêu cầu cho KPA Lập Kế Hoạch Cho Dự Án Phần Mềm (Software Project Planning) của CMM mức 2. Ở CMM mức 4, việc sử dụng các dữ liệu quá khứ để ước lượng được mong đợi sẽ tăng, và các mục tiêu của KPA Quy Trình Quản Lý Định Lượng (Quantitative Process Management) không thể thỏa mãn trừ khi một thủ tục ước lượng tốt được sử dụng. KPA Quản Lý Phần Mềm Tổng Hợp (Integrated Software Management) ở mức 3 cũng giả định rằng các phương pháp ước lượng tốt được sử dụng để lập kế cho dự án. Các yêu cầu liên quan đến ương lượng trong những KPAs này đều được thỏa mãn khi sử dụng các phương pháp được thảo luận trong chương này.

4.5 CÁC THAM KHẢO

1. B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

2. S.D. Conte, H.E. Sunsmore, and V.Y. Shen. *Software Engineering Metrics and Models*. Benjamin/Cummings, 1986.

3. V.R. Basili. *Tutorial on Models and Metrics for Software Management and Engineering*. IEEE Press, 1980.

4. B. Boehm. Software engineering economics. *IEEE Transactions on Software Engineering*, 10(1), 1984.

5. C.F. Kemerer. An empirical validation of software cost estimation models. *Communications of the ACM*, 30(5), 1987.

6. J.E. Matson, B.E. Barrett, and J.M. Mellicham. Software development cost estimation using Function points. *IEEE Transactions on Software Engineering*, 20(4), 1994.

7. A.J. Albrecht and J.R. Gaffney. Software function, source lines of code, and development effort prediction: A software science validation. *IEEE Transactions on Software Engineering*, 9(6), 1983.

8. F. Brooks, Jr. *The Mythical Man Month, Anniversary Edition*. Addison-Wesley, 1995.

9. L.H. Putnam and W. Myers. *Industrial Strength Software: Effective Management Using Measurement*. IEEE Computer Society Press, 1997.

10. P. Jalote. *CMM in Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley, 2000.

11. L.H. Putnam. A general empirical solution to the macro software sizing and estimating problem. *IEEE Transactions on Software Engineering*, 4(4), 1978.

Chương 5. Lập kế hoạch cho chất lượng

Cho đến vài năm trước đây, ngành công nghệ phần mềm đã tạo ra các sản phẩm có chất lượng thấp như là các ngành sản xuất khác gặp phải cách đây rất lâu - chất lượng là một cái gì đó đã được thực hiện vào cuối của quy trình lắp ráp/quy trình phát triển, trước khi sản phẩm được chuyển giao.Thông thường để xem xét và nhận biết về chất lượng, người quản lý dự án phải lập kế hoạch kiểm thử hệ thống (system testing) sau khi phát triển (những người quản lý khác thậm chí còn không lập kế hoạch cho kiểm thử hệ thống!), nhưng họ đã không thực hiện bất kỳ công việc quan trọng nào để kiểm soát chất lượng trong suốt quá trình phát triển. Kết quả ra sao? Kiểm thử hệ thống một cách thường xuyên đã phát hiện ra nhiều lỗi hơn so với dự kiến. Những lỗi này đòi hỏi nhiều nỗ lực hơn so với ước lượng để sửa chữa chúng, kết quả cuối cùng là phần mềm có lỗi đã được giao chậm trễ.

Khi tình hình được cải thiện, những người quản lý dự án bắt đầu lập kế hoạch để xem xét lại (review) và kiểm thử đơn vị (unit testing). Nhưng họ không biết làm thế nào để đánh giá hiệu quả và tác động của các biện pháp này. Nói cách khác, các dự án vẫn thiếu các mục tiêu chất lượng rõ ràng, thiếu các kế hoạch để đạt được mục tiêu của chúng, và thiếu các cơ chế giám sát tính hiệu quả của hoạt động kiểm soát chất lượng, chẳng hạn như của kiểm thử đơn vị.

Với việc sử dụng các phép đo (measurements) thích hợp và dữ liệu quá khứ (past data), bạn có thể “chữa trị” vấn đề chất lượng theo cùng một cách thức mà bạn đã làm cho hai tham số quan trọng khác là nỗ lực và thời gian biểu. Nghĩa là, bạn có thể thiết lập các mục tiêu chất lượng dựa trên định lượng, cùng với các mục tiêu nhỏ (subgoals) – để giúp theo dõi xuyên suốt thời gian thực hiện dự án (xem nó có đang hướng tới các mục tiêu chất lượng không).

Chương này thảo luận làm thế nào những người quản lý dự án tại Infosys thiết lập các mục tiêu chất lượng cho các dự án của họ và làm thế nào họ phát triển một kế hoạch để đạt được những mục tiêu này bằng cách sử dụng các mục tiêu chất lượng trung gian (quality goals) để theo dõi xuyên suốt thời gian thực hiện dự án. Trước khi chúng tôi mô tả cách tiếp cận tại Infosys, chúng tôi thảo luận ngắn gọn về một số khái niệm chung trong quản lý chất lượng.

5.1 KHÁI NIỆM VỀ CHẤT LƯỢNG

Đảm bảo sản phẩm phần mềm cuối cùng có chất lượng cao là một trong những mối quan tâm chính của người quản lý dự án. Nhưng làm thế nào để chất lượng phần mềm (software quality) được xác định? Khái niệm về chất lượng phần mềm không dễ dàng được định nghĩa bởi vì phần mềm có thể có nhiều đặc trưng chất lượng (quality characteristics) khác nhau [1]. Tuy nhiên, trong thực tế, quản lý chất lượng thường xoay quanh các lỗi (defects). Do đó, chúng tôi sử dụng *mật độ lỗi được giao (delivered defect density)* – tức là số lượng lỗi trên mỗi đơn vị kích thước trong phần mềm được giao – làm định nghĩa của chất lượng. Định nghĩa này đang là một chuẩn công nghiệp trong thực tế (de facto industry standard) [2]. Việc sử dụng nó báo hiệu rằng mục tiêu của một dự án phần mềm là để cung cấp phần mềm có càng ít lỗi càng tốt.

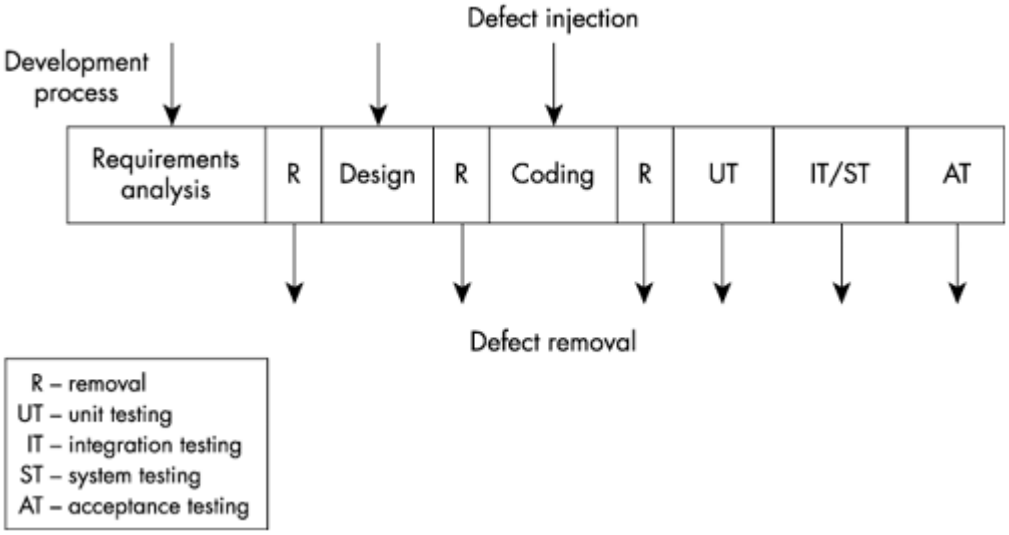
Lỗi (defect) là gì? Một lần nữa, không có định nghĩa chính xác về lỗi được áp dụng chung và rộng rãi (một phần mềm bị viết sai một từ có bị coi là có một lỗi?). Nói chung, chúng ta có thể nói *một lỗi* trong phần mềm là một cái gì đó làm cho phần mềm có cư xử theo một cách thức không nhất quán với các yêu cầu hoặc nhu cầu của khách hàng.

Trước khi xem xét các kỹ thuật để quản lý chất lượng, trước tiên bạn phải hiểu chu trình tiêm và loại bỏ lỗi (defect injection and removal cycle). Phát triển phần mềm là một hoạt động theo hướng con người (people-oriented activity) cao và do đó dễ bị lỗi. Lỗi có thể được tiêm vào phần mềm ở bất cứ giai đoạn nào trong quá phát tiến nó. Tức là, trong suốt quá trình chuyển đổi từ yêu cầu của khách hàng thành phần mềm để đáp ứng các yêu cầu đó, lỗi có thể được tiêm vào tại tất cả các hoạt động chuyển đổi được thực hiện. Các giai đoạn tiêm này chủ yếu là các đặc tả yêu cầu, thiết kế mức cao, thiết kế chi tiết, và cài đặt mã.

Để có phần mềm chất lượng cao, sản phẩm cuối cùng cần phải có ít lỗi nhất. Do đó, để cung cấp các phần mềm chất lượng cao, hoạt động loại bỏ lỗi là cần thiết; việc loại bỏ lỗi này diễn ra thông qua các hoạt động kiểm soát chất lượng là xem xét lại (review) và kiểm thử (testing). Bởi vì chi phí loại bỏ lỗi tăng lên khi lỗi được phát hiện muộn (khoảng thời gian nằm giữa lúc lỗi bị đưa vào và lúc lỗi được phát hiện) [3]. Mọi quy trình trưởng thành (CMM) sẽ bao gồm các hoạt động kiểm soát chất lượng sau mỗi giai đoạn, mà tại đó các lỗi có thể được tiêm vào. Các hoạt động để loại bỏ lỗi bao gồm xem xét lại yêu cầu, xem xét lại thiết kế, cài đặt mã, kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử hệ thống và kiểm

thử chấp nhận (chúng tôi không bao gồm việc xem xét lại các tài liệu của kế hoạch (plan documents), mặc dù xem xét lại này cũng giúp cải thiện chất lượng của phần mềm). Hình 5.1 cho thấy quá trình tiêm và loại bỏ lỗi.

Hình 5.1. Tiêm và loại bỏ lỗi



Nhiệm vụ của quản lý chất lượng là lên kế hoạch cho các hoạt động kiểm soát chất lượng một cách phù hợp rồi sau đó thực hiện và kiểm soát chúng để đạt được các mục tiêu chất lượng của dự án.

5.1.1 Tiếp cận theo thủ tục để quản lý chất lượng

Như đã đề cập trước đây, bạn phát hiện lỗi bằng cách thực hiện xem xét lại hoặc kiểm thử. Xem xét lại được cấu trúc, quy trình của nó theo hướng con người (human-oriented processes), trong khi đó, kiểm thử là quy trình thực thi (execute) phần mềm (hoặc các bộ phận của nó) để cố gắng xác định lỗi. Trong cách tiếp cận theo thủ tục, để quản lý chất lượng, các thủ tục và hướng dẫn cho các hoạt động xem xét lại và kiểm thử sẽ được xây dựng. Trong dự án, các hoạt động này sẽ được lập kế hoạch (có nghĩa là, cần xác định hoạt động và thời điểm thực hiện chúng); trong suốt quá trình thực hiện, chúng được thực hiện theo thủ tục đã được quy định. Tóm lại, tiếp cận theo thủ tục là thực hiện theo các quy trình xác định đã được đinh nghĩa trước để phát hiện lỗi.

Tiếp cận theo thủ tục không cho phép các đòi hỏi để được đáp ứng về tỷ lệ phần trăm của các lỗi được loại bỏ hoặc chất lượng của phần mềm sau khi hoàn thành thủ tục. Nói cách

khác, nó chỉ đơn thuần là thực hiện một tập hợp các thủ tục loại bỏ lỗi nhưng không cung cấp một cơ sở để đánh giá hiệu quả của chúng hoặc đánh giá chất lượng của mã (code) cuối cùng. Hơn nữa, một cách tiếp cận như vậy sẽ phụ thuộc nhiều vào chất lượng của các thủ tục và chất lượng thực hiện của nó. Ví dụ, nếu kế hoạch kiểm thử được thực hiện một cách cẩn thận vài kế hoạch đã được xem xét kỹ lưỡng, chất lượng của phần mềm sau khi thực hiện kiểm thử sẽ tốt hơn trường hợp kiểm thử được thực hiện với kế hoạch kiểm thử đã không được suy nghĩ cẩn thận và việc xem xét lại đã được thực hiện cầm chừng . Nhược điểm chính của cách tiếp cận theo thủ tục là thiếu phương tiện định lượng cho người quản lý dự án để đánh giá chất lượng của phần mềm được tạo ra; yếu tố duy nhất có thể nhìn thấy được cho những người quản lý dự án là xem các nhiệm vụ kiểm soát chất lượng có được thực hiện hay không.

5.1.2 Các tiếp cận định lượng để quản lý chất lượng

Để đánh giá tốt hơn hiệu quả của các quy trình phát hiện lỗi (defect detection processes), một tiếp cận phải vượt quá câu hỏi "Phương pháp đó có được thực hiện không?" và nhìn vào dữ liệu đo để đánh giá. Dựa trên phân tích các dữ liệu này, bạn có thể quyết định xem kiểm thử hoặc xem xét lại (review) có cần thiết không. Nếu các kiểm soát được áp dụng trong dự án dựa trên các dữ liệu định lượng để đạt được mục tiêu chất lượng, khi đó chúng ta nói rằng tiếp cận *quản lý chất lượng dựa trên định lượng (quantitative quality management approach)* đang được áp dụng.

Quản lý chất lượng theo định lượng có hai khía cạnh chính: (1) thiết lập một mục tiêu chất lượng theo định lượng và sau đó (2) quản lý quy trình phát triển phần mềm theo định lượng để mà mục tiêu chất lượng này được đáp ứng (với một độ tự tin cao).

Một cách tiếp cận quản lý chất lượng tốt nên cung cấp các dấu hiệu để cảnh báo sớm chứ không chỉ cho cảnh báo ở cuối dự án - khi mà không còn nhiều lựa chọn. Cảnh báo sớm cho phép can thiệp kịp thời. Để đạt được mục tiêu này, cần thiết phải dự đoán giá trị của một số thông số ở các giai đoạn khác nhau để mà có thể kiểm soát chúng trong suốt thời gian thực hiện dự án, đảm bảo rằng sản phẩm cuối cùng sẽ có chất lượng mong muốn. Nếu có thể thực hiện các dự đoán như vậy, bạn có thể sử dụng dữ liệu thực tế được thu thập để đánh giá xem quy trình đang được áp dụng có hiệu quả hay không. Với tiếp cận này, một quy trình phát hiện lỗi không kết thúc với việc báo cáo rằng quy vừa được thực hiện; thay vào đó, các dữ liệu được thu thập từ quy trình sẽ được sử dụng để đảm bảo

rằng quy trình này đã được thực hiện theo đúng cách thức để khai thác hết tiềm năng của nó.

Một tiếp cận theo định lượng để kiểm soát chất lượng phần mềm là làm việc với các *mô hình độ tin cậy phần mềm (software reliability models).* Hầu hết các mô hình như vậy sử dụng các dữ liệu về hỏng hóc được thu thập trong suốt các giai đoạn cuối cùng của kiểm thử để đánh giá độ tin cậy (reliability) của phần mềm. Những mô hình này có thể cho biết xem độ tin cậy như vậy là đủ để được chấp nhận hay cần kiểm thử thêm hay không. Thật không may, chúng không cung cấp các mục tiêu trung gian ở các giai đoạn đầu của dự án, và chúng cũng có những hạn chế khác. Nhìn chung, các mô hình như vậy là hữu ích trong việc đánh giá độ tin cậy của một sản phẩm phần mềm, nhưng chúng còn hạn chế cho việc quản lý chất lượng. (Xem thêm thông tin về các mô hình độ tin cậy của phần mềm ở [4][5][6].)

Một khái niệm về chất lượng nổi tiếng khác trong phần mềm là *hiệu quả loại bỏ lỗi (DRE-defect removal efficiency)*. Đối với hoạt động kiểm soát chất lượng (QC - quality control), chúng tôi xác định *hiệu quả loại bỏ lỗi* là tỷ lệ phần trăm của tổng số lỗi đang tồn tại đã được phát hiện bởi hoạt động kiểm soát chất lượng [5]. DRE cho toàn bộ vòng đời của dự án – tức là, cho tất cả các hoạt động thực hiện trước khi phần mềm được giao - đại diện cho hiệu quả "sống" đang xảy ra (in-process) của quy trình. Nếu toàn bộ tỷ lệ tiêm lỗi (defect injection rate) được biết cho dự án, khi đó DRE cho toàn bộ vòng đời cũng xác định chất lượng (mật độ lỗi được giao/delivered defect density) của phần mềm.

Mặc dù hiệu quả loại bỏ lỗi là một thước đo hữu ích để đánh giá một quy trình và xác định các khu vực cần cải tiến, nhưng nó không thích hợp cho việc quản lý chất lượng. Lý do chính là DRE cho một hoạt động kiểm soát chất lượng hoặc một quy trình tổng thể chỉ có thể được tính toán vào cuối của dự án, khi tất cả lỗi và nguồn gốc của chúng đã được biết. Do đó, nó không cung cấp một cách thức để trực tiếp để kiểm soát chất lượng trong quá trình thực hiện dự án.

Một tiếp cận khác để quản lý chất lượng theo định lượng là *dự đoán lỗi (defect prediction)*. Trong tiếp cận này, bạn thiết lập các mục tiêu chất lượng bằng cách sử dụng mật độ lỗi được giao (delivered defect density). Bạn có thể thiết lập các mục tiêu trung gian bằng cách ước lượng số lỗi có thể được xác định bởi các hoạt động phát hiện lỗi khác nhau; sau đó bạn so sánh số lượng lỗi thực tế (actual number of defects) với số lỗi ước lượng (estimated number of defects).

Tiếp cận này làm cho quản lý chất lượng giống như quản lý nỗ lực và thời gian biểu (hai tham số chính để đo thành công của một dự án). Mục tiêu chất lượng phải được thiết lập cho phần mềm được giao. Từ mục tiêu này, giá trị của các thông số đã được lựa chọn cho các giai đoạn khác nhau trong dự án được ước lượng; có nghĩa là, các cột mốc được xây dựng. Những cột mốc này được lựa chọn để mà, nếu các ước lượng được đáp ứng, chất lượng sau cùng của phần mềm sẽ đạt được mức độ như mong muốn. Trong suốt quá trình thực hiện dự án, giá trị thực tế của các tham số được đo lường và được so sánh với mức ước lượng để xác định xem dự án đang đi đúng đường hoặc để xác định xem những hoạt động nào cần phải được thực hiện thêm để đảm bảo rằng phần mềm cuối cùng có chất lượng như mong muốn.

Hiệu quả của tiếp cận này phụ thuộc vào mức độ chính xác của các dự đoán mức độ lỗi (defect levels) mà bạn thực hiện ở các giai đoạn khác nhau của dự án. Được biết rằng, tỷ lệ lỗi (defect rate) có mô hình tương tự như tỷ lệ nỗ lực (effort rate) - cả hai theo đường cong Rayleigh [5][7][8]. Nói cách khác, số lượng lỗi được tìm thấy vào lúc bắt đầu của dự án thì nhỏ, nhưng tiếp tục tăng cho đến khi nó đạt đến một giá trị cực đại (vào khoảng thời gian đang kiểm thử đơn vị) trước khi nó bắt đầu giảm trở lại. Bởi vì một quy trình đã định nghĩa các thời điểm để phát hiện lỗi, bạn cũng có thể xác định đường cong này bằng cách sử dụng tỷ lệ phần trăm của tổng số lỗi được phát hiện ở giai đoạn phát hiện khác nhau. Và từ ước lượng tỷ lệ tiêm lỗi (defect injection rate) và kích thước (size), bạn có thể ước lượng tổng số lỗi. Tiếp cận dự đoán mức độ lỗi (defect level prediction) này thì tương tự cả hai: (1) mô hình lỗi cở sở (base defect model) và (2) tiếp cận STEER của Hệ thống Liên bang của IBM (IBM's Federal Systems Division)[5].

Một tiếp cận khác là sử dụng *kiểm soát quy trình dùng thống kê (SPC - statistical process control)* để quản lý chất lượng (Chương 7 chứa một thảo luận ngắn về SPC). Trong tiếp cận này, bạn thiết lập các hiệu suất (khả năng/performance) được mong đợi cho các quy trình kiểm soát chất lượng khác nhau, chẳng hạn như kiểm thử và xem xét lại, bằng cách thiết lập các giới hạn kiểm soát (control limits). Nếu hiệu suất thực tế của công việc/nhiệm vụ kiểm soát chất lượng không nằm trong giới hạn này, bạn hãy phân tích tình hình và có hành động phù hợp. Các giới hạn kiểm soát thì tương tự như dự đoán mức độ lỗi dựa trên hiệu suất trong quá khứ, nhưng cũng có thể được sử dụng để theo dõi các hoạt động chất lượng ở một mức độ chính xác hơn (chi tiết hơn), chẳng hạn như việc xem xét lại hoặc kiểm thử đơn vị một mô-đun.

Khi bạn sử dụng tiếp cận dự đoán hiệu suất và số lượng lỗi thực tế ít hơn so với mục tiêu, tiếp cận có quá nhiều sự không chắc chắn, do đó bạn có thể nói rằng quy trình loại bỏ lỗi không được thực hiện đúng cách. Kết quả là, bạn phải nhìn vào các chỉ số khác để xác định nguyên nhân [5]. Nói cách khác, nếu dữ liệu thực tế nằm ngoài phạm vi, người quản lý dự án sẽ nhìn vào các chỉ số khác để quyết định xem tình trạng thực tế là gì và những hành động nào nên được thực hiện.

5.2 LẬP KẾ HOẠCH QUẢN LÝ CHẤT LƯỢNG THEO ĐỊNH LƯỢNG

Bây giờ chúng ta hãy xem xét làm thế nào những người quản lý dự án tại Infosys sử dụng tiếp cận dự đoán lỗi để quản lý dự án theo định lượng. (Như được thảo luận trong các Chương 10 và 11, các dự án cũng sử dụng SPC (kiểm soát quy trình dùng thống kê) ở mức công việc/nhiệm vụ). Như đã thảo luận trước đây, khi bạn lập kế hoạch để quản quản lý chất lượng theo định lượng cho một dự án, bạn phải đối mặt với hai vấn đề chính: thứ nhất, thiết lập các mục tiêu chất lượng, và thứ hai, dự đoán mức lỗi ở các cột mốc trung gian mà tại đó được dùng để giám sát tiến độ theo định lượng hướng về phía mục tiêu. Ngoài ra, nếu các mục tiêu của dự án vượt quá khả năng của các quy trình hiện tại, người quản lý dự án phải lập kế hoạch để cải tiến các quy trình này. Chúng ta hãy xem làm thế nào những người quản lý dự tại Infosys thực hiện ba nhiệm vụ này.

5.2.1 Thiết lập các mục tiêu chất lượng (Quality goal)

Những người quản lý dự án tại Infosys thiết lập các mục tiêu chất lượng trong suốt các giai đoạn lập kế hoạch. *Mục tiêu chất lượng (quality goal)* cho một dự án nói chung là số lượng lỗi dự kiến sẽ được tìm thấy trong kiểm thử chấp nhận (acceptance testing). Bạn có thể thiết lập các mục tiêu chất lượng theo những gì *được tính toán* bằng cách sử dụng dữ liệu quá khứ; trong trường hợp này, nó ngụ ý rằng bạn sẽ sử dụng quy trình chuẩn, và vì thế các kết quả chất lượng chuẩn sẽ được dự kiến (mong đợi). Hai nguồn chính có thể được sử dụng để thiết lập các mục tiêu chất lượng: (1) dữ liệu quá khứ từ các dự án tương tự và (2) dữ liệu từ các cở sở dữ liệu về khả năng của quy trình (PCB - process capability baseline).

Nếu bạn sử dụng dữ liệu từ các dự án tương tự, bạn có thể ước lượng số lượng lỗi sẽ được tìm thấy trong suốt quá trình kiểm thử chấp nhận của dự án hiện tại bằng cách tính tích của số lượng lỗi được tìm thấy trong kiểm thử chấp nhận của các dự án tương tự và

tỷ lệ của nỗ lực ước lượng (ratio of the estimated effort) của dự án hiện tại này và nỗ lực tổng cộng của các dự án tương tự.

Nếu bạn sử dụng dữ liệu từ cơ sở dữ liệu quy trình (PCB), bạn có thể sử dụng một số phương pháp để tính toán giá trị này. Nếu bạn thiết lập các mục tiêu chất lượng là số lượng lỗi trên mỗi Điểm chức năng (FP - Function points), bạn ước lượng kích thước Điểm chức năng (như đã nói trước đây), và số lượng lỗi dự kiến sẽ là tích của của hệ số chất lượng và và kích thước ước lượng. Trình tự các bước sau đây được sử dụng:

1. Thiết lập mục tiêu chất lượng là số lỗi trong mỗi FP.
2. Ước lượng mức năng suất dự kiến cho dự án.
3. Ước lượng kích thước dùng FP (năng suất dự kiến * nỗ lực ước lượng).
4. Ước lượng số lượng lỗi ở giai đoạn kiểm thử chấp nhận (AT- acceptance testing) (mục tiêu chất lượng * kích thước ước lượng).

Thay vì thiết lập các mục tiêu chất lượng bằng số lỗi trên mỗi Điểm chức năng, đôi khi sẽ hữu ích hơn nếu thiết lập mục tiêu về hiệu quả loại bỏ lỗi. Trong trường hợp này, bạn có thể xác định số lượng lỗi dự kiến trong suốt thời gian kiểm thử chấp nhận từ tỷ lệ tiêm lỗi, hiệu quả loại bỏ lỗi trong suốt tiến trình (target in-process removal efficiency), và kích thước ước lượng (estimated size). Trình tự các bước như sau:

1. Thiết lập các mục tiêu chất lượng bằng hiệu quả loại bỏ lỗi.
2. Ước lượng tổng số lỗi từ tỷ lệ tiêm lỗi và kích thước ước lượng, hoặc bằng tỷ lệ tiêm lỗi dựa trên nỗ lực và nỗ lực ước lượng.
3. Ước lượng số lượng lỗi trong AT từ tổng số lỗi và mục tiêu chất lượng.

5.2.2 Ước lượng lỗi cho các giai đoạn khác

Một khi mục tiêu chất lượng của dự án đã được thiết lập, bạn nên ước lượng mức lỗi cho các hoạt động kiểm soát chất lượng khác nhau để mà bạn có thể thể kiểm soát chất lượng theo định lượng. Tiếp cận để ước lượng mức lỗi cho các giai đoạn khác thì tương tự như các tiếp cận để ước lượng lỗi trong kiểm thử chấp nhận. Từ tổng số lỗi ước lượng, bạn dự đoán mức lỗi cho các giai đoạn kiểm thử khác nhau bằng cách sử dụng phân phối

tỷ lệ phần trăm lỗi như đã được đưa ra trong PCB (cơ sở dữ liệu về khả năng của quy trình). Ngoài ra, bạn có thể dự đoán lỗi cho các giai đoạn khác nhau dựa trên dữ liệu quá khứ từ các dự án tương tự.

Ở mức tối thiểu, bạn ước lượng số lỗi chưa được phát hiện trong tích hợp và kiểm thử hệ thống (system and integration testing). Kiểm thử hệ thống được tách riêng ra bởi vì nó là hoạt động kiểm thử chính và là hoạt động kiểm soát chất lượng cuối cùng được thực hiện trước khi phần mềm được giao đến khách hàng. Ước lượng số lỗi trong kiểm thử hệ thống và sau đó so sánh con số ước lượng này với số lượng lỗi thực tế vừa được tìm thấy sẽ giúp bạn xác định xem kiểm thử hệ thống đã được thực hiện đủ và phần mềm đã sẵn sàng để được giao chưa.

Đối với các hoạt động xem xét lại (reviews), thay vì làm một dự đoán trực tiếp mức lỗi, bạn có thể sử dụng các chỉ tiêu được đưa ra trong các baseline của việc xem xét lại để đánh giá hiệu quả của việc xem xét lại ngay lập tức sau khi nó được thực hiện xong. Những chỉ tiêu này được xác định dựa trên hoạt động SPC (kiểm soát quy trình dùng thống kê) và cho phép bạn đánh giá tính hiệu quả của mỗi lần xem xét lại chứ không phải là đánh giá tính hiệu quả của cả một giai đoạn. Chương 10 thảo luận chi tiết hơn về quản lý việc xem xét lại theo định lượng. Tương tự như vậy, chỉ tiêu cũng được cung cấp cho kiểm thử đơn vị. Giám sát kiểm thử đơn vị tiếp tục được thảo luận trong Chương 11.

5.2.3 Lập kế hoạch cho quy trình chất lượng (Quality process planning)

Bạn có thể thiết lập một mục tiêu chất lượng cao hơn (hoặc thấp hơn) mức chất lượng (quality level) của một dự án tương tự (a similar project), hoặc bạn có thể thiết lập các mức cần đạt được bởi quy trình chuẩn (standard process). Sau đó, bạn có thể xác định số lượng lỗi dự kiến cho các mục tiêu cao hơn bằng cách sử dụng mục tiêu chất lượng được thiết đặt cho dự án. Một cách làm khác là: sau khi xác định số lượng lỗi dự kiến của AT, bạn có thể thiết lập mục tiêu chất lượng bằng cách chọn một số lượng lỗi khác của AT làm mục tiêu.

Nếu mục tiêu chất lượng dựa trên dữ liệu từ các dự án tương tự và mục tiêu là cao hơn so với các dự án tương tự, sẽ có một bất hợp lý nếu mong đợi rằng việc sử dụng quy trình tương tự như cái đã được sử dụng trong các dự án trước đây sẽ đạt được mục tiêu chất

lượng cao hơn. Nếu quy trình tương tự được áp dụng, kỳ vọng hợp lý phải là đạt được các mức chất lượng tương tự. Do đó, nếu mong muốn có một mức chất lượng cao hơn, quy trình này phải được nâng cấp. Tương tự như vậy, nếu mục tiêu chất lượng được thiết lập cao hơn mức chất lượng được đưa ra trong PCB (cơ sở dữ liệu về khả năng của quy trình), sẽ có một bất hợp lý nếu mong đợi rằng việc sử dụng quy trình chuẩn sẽ dẫn đến mức chất lượng cao hơn. Do đó, một chiến lược mới sẽ là cần thiết - nói chung, một sự kết hợp của đào tạo, tạo mẫu (prototyping), kiểm thử, xem xét lại (reviews), và đặc biệt là trong công tác phòng ngừa lỗi (defect prevention). Chiến lược này phải được nói rõ trong kế hoạch chất lượng cho dự án. Ở đây chúng tôi thảo luận về kiểm thử và xem xét lại (reviews) - hai quy trình kiểm soát chất lượng chính. Lập kế hoạch phòng ngừa lỗi được thảo luận ở phần sau của Chương này.

Các mức kiểm thử (levels of testing) khác nhau được triển khai trong một dự án. Bạn có thể sửa đổi một kiểm thử bằng cách thêm hoặc xóa một số bước kiểm thử (những bước này sẽ tạo ra sự lệch (process deviations) của quy trình trong kế hoạch quản lý dự án). Ngoài ra, bạn có mở rộng tiếp cận kiểm thử bằng cách, ví dụ, thực hiện một xem xét lại nhóm (group review) cho kế hoạch và kết quả kiểm thử.

Việc lựa chọn sản phẩm công việc vừa được tạo ra (work products) nào để được xem xét lại thường được làm bởi người quản lý dự án. Tập hợp các tài liệu sẽ được xem xét, trên thực tế, có thể thay đổi từ dự án này sang dự án khác. Nó có thể được điều chỉnh theo mục tiêu chất lượng. Nếu một mức chất lượng cao hơn được thiết lập, một số lượng lớn hơn của các nhóm chương trình (programs group) sẽ được xem xét lại, bằng cách bao gồm nhóm xem xém lại các kế hoạch kiểm thử, bao gồm nhóm xem xét lại của các thiết kế chi tiết, v.v. Nếu phương pháp này được chọn, nó được đề cập như là chiến lược để đáp ứng các mục tiêu chất lượng. Để tiếp tục chi tiết hóa chiến lược này, bạn cần xác định trong kế hoạch dự án tất cả các tài liệu cần được xem xét lại và bản chất của những xem xét lại này.

Bạn có thể sử dụng dữ liệu baseline về khả năng quy trình (process capability baseline) để ước lượng tác động của những thay đổi quy trình được đề nghị đối với nỗ lực và thời gian biểu đã được hoạch định cho dự án. Thông thường, một khi quy trình bị thay đổi, tác động của chúng được dự đoán dựa trên kinh nghiệm quá khứ. Chiến thuật này thường là chấp nhận được vì những thay đổi nói chung là nhỏ.

5.3 LẬP KẾ HOẠCH PHÒNG NGỪA LỖI

Các hoạt động phòng ngừa lỗi (DP- Defect prevention) nhằm để cải thiện chất lượng và nâng cao năng suất. Nói chung một số lỗi hiện diện trong hệ thống nhưng chưa được phát hiện bởi các hoạt động kiểm soát chất lượng (QC - quality control), khác nhau thì chắc chắn sẽ tìm thấy chúng trong hệ thống cuối cùng. Hậu quả là, nếu số lượng lỗi hiện diện trong phần mềm trong suốt giai đoạn phát triển càng lớn, số lượng lỗi còn sót lại trong hệ thống cuối cùng khi giao sẽ càng nhiều.

Điểm này có thể được nói theo cách khác. Hiệu quả loại bỏ lỗi tổng thể của một quy trình (overall defect removal efficiency) là tỷ lệ phần trăm tổng số lỗi được loại bỏ bởi các hoạt động QC trước khi phần mềm được giao. Đối với một quy trình ổn định, hiệu quả loại bỏ lỗi cũng ổn định. Do đó, nếu tổng số lỗi trong hệ thống càng lớn, số lượng lỗi còn sót lại trong hệ thống được giao sẽ càng lớn. Nói cách khác, tỷ lệ tiêm lỗi (defect injection rate) càng cao, chất lượng sẽ càng nghèo nàn. Rõ ràng, đối với một quy trình mà hiệu quả loại bỏ lỗi của nó được biết trước, chất lượng của phần mềm được giao có thể được cải thiện nếu ít lỗi hơn xuất hiện trong khi phần mềm đang được xây dựng. Nhận ra được điều này đóng vai trò là động lực để nâng cao chất lượng bằng cách tăng cường các hoạt động phòng ngừa lỗi (DP).

Các hoạt động DP cũng có lợi ích giúp tăng năng suất. Như đã thảo luận trước đó, chu kỳ lỗi cơ bản trong suốt quá trình xây dựng phần mềm là các nhà phát triển tạo ra lỗi, rồi sau đó xác định và loại bỏ chúng. Nói cách khác, một cái gì đó được tạo ra và sau đó được loại bỏ. Rõ ràng, chu kỳ tiêm và loại bỏ lỗi này một sự lãng phí nỗ lực – vì nó không thêm giá trị nào cho phần mềm. Trong chu kỳ này, các nhà phát triển tạo ra một cái gì đó chỉ để sau này dùng thêm nỗ lực để loại bỏ nó (và họ hy vọng họ có thể loại bỏ hết tất cả các lỗi). Do đó, nếu lúc đầu không tạo ra lỗi; sau đó nỗ lực cần thiết để xác định và loại bỏ chúng sẽ không được cần đến. Nói cách khác, nếu tiêm vào ít lỗi hơn, ít lỗi hơn sẽ được gỡ bỏ, kết quả là nỗ lực cần thiết để loại bỏ lỗi sẽ được giảm, do đó làm tăng năng suất. Quan điểm này giúp tạo ra động lực khi trả chi phí cho DP.

DP được thực hiện như thế nào? Tiền đề của DP là có một số nguyên nhân đằng sau những lỗi tiêm vào. Nếu nguyên nhân có thể được hiểu, nhiều nỗ lực được làm để thực hiện loại bỏ hoặc để giảm thiểu tác động của chúng. Nói cách khác, DP nói chung đòi hỏi phải thu thập dữ liệu về các lỗi được tìm thấy trong quá khứ, phân tích dữ liệu này để tìm

ra nguyên nhân gốc của việc tiêm lỗi, và sau đó phát triển các giải pháp cho các vấn đề có nguyên nhân này.

Giống như bất kỳ công việc/nhiệm vụ lớn khác của dự án, các hoạt động DP phải được lên kế hoạch. Bạn thực sự đi phân tích các lỗi và tìm các giải pháp, tuy nhiên, điều này chỉ được thực hiện sau khi thu thập được nhiều dữ liệu về lỗi từ dự án. Để thực hiện DP, một người quản lý dự án có thể bắt đầu với một tập hợp các gợi ý (nên được thực hiện theo) ở mức công ty và sau đó xây dựng gợi ý cho từng loại dự án cụ thể bằng cách dựa trên phân tích dữ liệu về lỗi của dự án. Ở Infosys, các bước sau đây được thực hiện cho các hoạt động phòng ngừa lỗi ở mức dự án:

- Xác định một nhóm phòng ngừa lỗi (defect prevention team) trong phạm vi dự án.
- Có một cuộc họp đầu tiên (cho sự bắt đầu) (kick-off meeting) và xác định các giải pháp hiện có.
- Lập kế hoạch phòng ngừa lỗi.
    - Thiết lập mục tiêu phòng ngừa lỗi cho dự án.
    - Xét xem nhóm DP có cần để được huấn luyện về DP và phương pháp phân tích nguyên nhân, nếu cần thiết.
    - Xác định tần số mà tại đó các hoạt động phòng ngừa lỗi sẽ được thực hiện.
- Thực hiện phòng ngừa lỗi.
    - Tại các điểm đã xác định, thu thập dữ liệu về lỗi.
    - Xác định các loại lỗi phổ biến nhất bằng cách làm phân tích Pareto.
    - Thực hiện phân tích nguyên nhân và xếp độ ưu tiên các nguyên nhân gốc.
    - Xác định và phát triển các giải pháp cho các nguyên nhân gốc.
    - Thực hiện các giải pháp.
    - Xem lại các tình trạng (trạng thái) và lợi ích của DP tại các cột mốc của dự án.

- Rút ra bài học.
    - Trong báo cáo phân tích số liệu và BOK, rút ra bài học và lợi ích bạn đã thu được.

      (BOK (body of knowledge): là một khung của kiến thức được sử dụng để đóng gói kinh nghiệm.)
    - Gửi (submit) tất cả các kết quả đầu ra của DP để góp phần xây dựng các thành phần của quy trình (process assets).

Trong suốt quá trình lập kế hoạch chất lượng, các hoạt động duy nhất của bạn là các hoạt động lập kế hoạch. Các hoạt động bên trong của "Thực hiện phòng ngừa lỗi" hoặc "Rút ra bài học" được thực hiện trong khi dự án đang được thực hiện hoặc khi nó đã hoàn tất. Ở đây, chúng tôi tập trung chủ yếu vào các nhiệm vụ liên quan đến lập kế hoạch. Chương 11 mô tả các nhiệm vụ thuộc "Thực hiện phòng ngừa lỗi" trong một thảo luận về giám sát dự án.

Hầu hết các hoạt động liên quan đến lập kế hoạch DP thì có thể hiểu được (không cần giải thích) (self-explanatory). Người quản lý dự án xác định một nhóm sẽ thực hiện các phân tích DP (rõ ràng, tất cả mọi người trong dự án phải thực hiện các giải pháp thực tế để phòng ngừa lỗi). Một cuộc họp đầu tiên (kick-off meeting) để làm tăng nhận thức của các thành viên trong nhóm và xác định các giải pháp có thể có sẵn trong công ty. Nếu cần thiết, nhóm DP được đào tạo về DP và phương pháp phân tích nguyên nhân.

Thiết lập các mục tiêu cho DP là một hoạt động lập kế hoạch then chốt. Như đã đề cập trước đây, DP được xem như một chiến lược để đạt được chất lượng và năng suất cao hơn so với quy trình chuẩn của công ty có thể đạt được. Do đó, nếu một người quản lý dự án đã thiết lập các mục tiêu chất lượng và năng suất cao hơn so với mức của công ty, như là một phần của chiến lược của mình để đạt được những mục tiêu mà người quản lý dự án muốn có ở DP. Nói chung, một người quản lý dự án thiết đặt mục tiêu cho DP là về giảm tỷ lệ tiêm lỗi, thường khoảng 10% đến 20%. Với tỷ lệ này, tác động của nó đến chất lượng và năng suất có thể được ước lượng.

Một hoạt động lập kế hoạch quan trọng khác là để quyết định khi nào và bao lâu thì các công việc/nhiệm vụ của DP sẽ được thực hiện. Mặc dù một người quản lý dự án có thể thực hiện những quyết định này, các hướng dẫn chung tại Infosys là các hoạt động của

DP nên được thực hiện sau khi khoảng 20% của các chương trình đã được cài đặt mã, mã được xem xét lại (code reviewed), và đơn vị được kiểm thử (unit tested), và một lần nữa khi khoảng 50% của chúng đã được cài đặt mã, được xem xét lại, và được kiểm thử đơn vị. Tức là, trước tiên thực hiện DP khi dữ liệu về lỗi từ khoảng 20% các mô-đun đã có. Hy vọng rằng, các kết quả của DP này nên tạo ra kết quả là các hoạt động cần được thực hiện để giảm tỷ lệ tiêm lỗi cho phần còn lại của dự án. Sau đó, thực hiện DP khi dữ liệu về lỗi từ khoảng 50% các mô-đun đã có. Trong lần thực hiện này, người quản lý dự án có thể xác định xem các giải pháp nào đang đem lại hiệu quả tốt và xem các hành động tiếp theo nào nên được thực hiện. Tại thời điểm này, tất cả các loại lỗi khác nhau nên được nhìn thấy và nguyên nhân của chúng được hiểu rõ và các hành động thích hợp sẽ được thực hiện.

5.4 KẾ HOẠCH CHẤT LƯỢNG CỦA DỰ ÁN ACIC

Bây giờ chúng ta hãy thảo luận về việc lập kế hoạch chất lượng cho case study - dự án ACIC. Để thiết đặt các mục tiêu và ước lượng lỗi, người quản lý dự án ACIC sử dụng dữ liệu từ dự án Synergy - một dự án tương tự được thực hiện trước đó cho cùng một khách hàng. Tỷ lệ tiêm lỗi (defect injection rate) trong Synergy là 0.036 lỗi mỗi người-giờ (thu được bằng phép chia: tổng số lỗi cho tổng nỗ lực - cả hai tổng số này đã có sẵn trong cơ sở dữ liệu quy trình của Synergy). Người quản lý dự án ACIC muốn làm tốt hơn so với Synergy và dự kiến sẽ có giảm 10% tỷ lệ tiêm lỗi, tốt so nhiều so với các tiêu chuẩn của công ty. Tỷ lệ cho dự án mới này, số lượng lỗi tiêm đã được dự kiến là sẽ có khoảng 501 * 8.75 * 0.033 (tích của nỗ lực tính theo người-ngày, số giờ của một người-ngày, và tỷ lệ tiêm lỗi dự kiến). Tức là, ước lượng tổng số lỗi được tiêm vào trong suốt toàn bộ vòng đời của dự án là khoảng 145 lỗi.

Chất lượng được định nghĩa là mật độ lỗi (defect density) trong suốt kiểm thử chấp nhận (acceptance testing), hoặc hiệu quả loại bỏ lỗi tổng thể (overall defect removal efficiency) trước khi tiến hành kiểm thử chấp nhận. Synergy tìm thấy 5% lỗi trong quá trình kiểm thử chấp nhận. Dự án ACIC đã đặt mục tiêu để làm giảm nó xuống 3%, và nếu như vậy thì số lượng lỗi dự kiến sẽ được tìm thấy trong suốt quá trình kiểm thử chấp nhận là 145 * 0.03 = 5 (xấp xỉ).

Mục tiêu về năng suất đã được cải thiện nhẹ so với những gì đã đạt được trong Synergy. Mục tiêu về thời gian hoàn thành (thời gian biểu) là sẽ giao các sản phẩm đúng hạn, và

chi phí dự kiến của chất lượng là 32%, cái này thì giống như trong Synergy và baseline khả năng của công ty (organization capability baseline). Bảng 5.1 cho thấy tất cả các mục tiêu cho dự án ACIC.

Đối với mục đích theo dõi và kiểm soát dự án, người quản lý dự án tại ACIC muốn ước lượng số lượng lỗi sẽ được phát hiện trong các giai đoạn khác nhau. Khi đó, ông có thể so sánh những ước lượng này với số lỗi thực tế được tìm thấy và sử dụng chúng để theo dõi tiến độ của dự án. Từ ước lượng tổng số lỗi tiêm, ông có thể có được những ước lượng cho mỗi giai đoạn bằng cách sử dụng phân phối lỗi.

Bảng 5.1. Mục tiêu cho dự án ACIC

Mục tiêu (Goals)	**Giá trị (Value)**	**Cơ sở để thiết đặt mục tiêu (Basis for Setting Goals)**	**Tiêu chuẩn của công ty (Organization-wide Norms)**
Tổng số lỗi được tiêm vào *(Total number of defects injected)*	145	0.033 lỗi/người-giờ. Giá trị này lớn hơn 10% của dự án Synergy (dự án này chỉ là 0.036 lỗi/người-giờ) *(0.033 defects/person-hour.This is 10% better than Synergy, which was 0.036 defects/person-hour)*	0.052 lỗi/người-giờ *(0.052 defects/person-hour)*
Chất lượng (số lỗi chấp nhận được) *(Quality (acceptance defects))*	5	3% hoặc ít hơn 3% của tổng số lỗi được ước lượng *(3% or less of total estimated number of defects)*	6% của tổng số lỗi được ước lượng *(6% of estimated number of defects)*
Năng suất tính theo số Điểm chức năng/người-tháng *(Productivity in FP/person-month)*	57	Tăng thêm 3.4% so với năng suất của dự án Synergy *(3.4% productivity improvement over Synergy)*	50
Thời gian hoàn thành *(Schedule)*	Giao đúng hạn *(Delivery on time)*		10%
Chi phí của chất lượng *(Cost of quality)*	32%	31.5%	32%

Để có được phân phối lỗi, ông đã lựa chọn dùng dữ liệu baseline (capability baseline data) về khả năng hoặc dữ liệu của Synergy. Bởi vì Synergy đã không có giai đoạn yêu

cầu (requirements phase), ông đã điều chỉnh phân phối của mình cho phù hợp với chu kỳ cuộc sống hiện tại. Về cơ bản, ông giảm tỷ lệ lỗi được tìm thấy trong kiểm tra đơn vị (unit testing) từ 45% xuống còn 40%, giảm tỷ lệ phần trăm của lỗi trong kiểm thử chấp nhận (acceptance testing) xuống 3% (vì đó đã là mục tiêu chất lượng), và tăng tỷ lệ phần trăm của lỗi trong các hoạt động xem xét lại yêu cầu (requirements review) và xem xét lại thiết kế (requirements review) đến 20%. Những tỷ lệ này cũng phù hợp (nhất quán) với phân phối được đưa ra trong baseline về khả năng (capability baseline). Bảng 5.2 cho thấy ước lượng của lỗi sẽ được phát hiện ở các giai đoạn khác nhau.

Bảng 5.2. Các ước lượng số lỗi sẽ được phát hiện

Giai đoạn xem xét lại/Kiểm thử (Review/Testing Stage)	**Số lỗi ước lượng sẽ được phát hiện (Estimated Number of Defects to Be Detected)**	**% lỗi sẽ được phát hiện (% of Defects to Be Detected)**	**Cơ sở được dùng để ước lượng (Basis for Estimation)**
Xem xét lại yêu cầu và thiết kế *(Requirements and design review)*	29	20%	Tương tự như dự án Synergy và cơ sở dữ liệu về khả năng của quy trình. (Similar project Synergy and PCB)
Xem xét lại mã *(Code review)*	29	20%	Tương tự như dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Kiểm thử đơn vị *(Unit testing)*	57	40%	Tương tự như dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Kiểm thử hồi quy và tích hợp *(Integration and regression testing)*	25	17%	Tương tự như dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Kiểm thử chấp nhận *(Acceptance testing)*	5	3%	Tương tự như dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Tổng số lỗi ước lượng sẽ được phát hiện (Total estimated number of defects to be detected)	**143**	**100%**	

Bởi vì mục tiêu chất lượng đã được thiết lập là cao hơn chất lượng đã đạt được trong Synergy cũng như các tiêu chuẩn của công ty, và bởi vì mục tiêu về năng suất cũng cao hơn một chút, người quản lý dự án ACIC phải đưa ra chiến lược để đạt được những mục tiêu này vì thực hiện theo các quy trình tiêu chuẩn sẽ không giúp đạt được chúng. Chiến lược cơ bản gồm ba phần:

- Áp dụng phòng ngừa lỗi.
- Hướng dẫn nhóm xem xét lại (reviews) các đặc tả yêu cầu và chương trình đầu tiên được viết bởi các lập trình viên.
- Sử dụng phương pháp RUP.

Bảng 5.3. Chiến lược để đạt được các mục tiêu cao hơn của dự án ACIC

Chiến lược (Strategy)	**Lợi ích được mong đợi (Expected Benefits)**
Phòng ngừa lỗi bằng cách sử dụng các hướng dẫn phòng ngừa lỗi (defect prevention guidelines) và quy trình; sử dụng các chuẩn được phát triển trong Synergy để cài đặt mã (coding).	Giảm 10% -20% tỷ lệ tiêm lỗi và tăng khoảng 2% năng suất.
Nhóm xem xét lại (review) các đặc tả chương trình (program specs) ở những trường hợp sử dụng (use cases) đầu tiên và có lôgic phức tạp. Nhóm xét lại các tài liệu thiết kế (design docs) và những mã (code) đầu tiên được tạo ra.	Cải thiện chất lượng vì nâng cao hiệu quả loại bỏ lỗi tổng thể; một số lợi ích về năng suất bởi vì các lỗi sẽ được phát hiện sớm.
Giới thiệu phương pháp RUP và thực hiện dự án trong các vòng lặp (iterations). Tiến hành phân tích tại các cột mốc (milestone analysis) and và phân tích các thực hành phòng ngừa lỗi sau mỗi vòng lặp.	Giảm khoảng 5% tỷ lệ tiêm lỗi và tăng 1% năng suất tổng thể.

Tức là, khi được so với quy trình được sử dụng trong Synergy, quy trình của ACIC đã được thay đổi để đạt được các mục tiêu cao hơn.

Dựa trên dữ liệu từ các dự án khác, người quản lý dự án đã dự kiến thực hiện ngăn chặn lỗi để giảm từ 10% đến 20% lỗi. Điều này sẽ giúp làm giảm nỗ lực làm lại công việc (rework) sau khi kiểm thử nghiệm và xem xét lại, tạo ra một cải tiến xấp xỉ 2% của năng suất (dựa trên tỷ lệ phần trăm nỗ lực làm lại (rework) của Synergy). Ông mong rằng việc nhóm xem xét lại các đặc tả chương trình (program specifications) và mô-đun đầu tiên được cài đặt mã bởi các lập trình viên sẽ nâng cao hiệu quả loại bỏ lỗi tổng thể và cũng cung cấp một số lợi ích về năng suất trước mắt. Dựa trên các tài liệu (literature) và các giai thoại, ông mong đợi rằng sử dụng RUP đem lại lợi tích về chất lượng và năng suất. Bảng 5.3 cho thấy chiến lược và các lợi ích dự kiến. Lưu ý rằng mặc dù lợi ích được kỳ vọng của chiến lược đã được đề cập đến một cách riêng biệt, rất khó để theo dõi các tác dụng riêng lẻ như vậy.

Bảng 5.4. Các xem xét lại (reviews) trong dự án ACIC

Thời điểm xem xét lại (Review Point)	**Các sản phẩm công việc (work products) được xem xét lại (Review Item)**	**Kiểu xem xét lại (Type of Review)**
Kết thúc giai đoạn lập kế hoạch cho dự án *(End of project planning)*	• Kế hoạch dự án *(Project plan)* • Thiết lập hệ thống kiểm soát lỗi *(Defect control system set up)* • Thời gian biểu của dự án *(Project schedule)*	Xem xét lại bởi nhóm *(Group review)* Xem xét lại bởi cố vấn chất lượng *(Software quality adviser review)* Xem xét lại bởi cố vấn chất lượng *(Software quality adviser review)*
Kết thúc giai đoạn lập kế hoạch cho dự án	Kế hoạch quản lý cấu hình *(CM plan)*	Xem xét lại bởi nhóm *(Group review)*
Hoàn thành 90% yêu cầu (tức là nên ở cuối vòng lặp thứ 1 của giai đoạn Soạn thảo chi tiết - Elaboration) *(End of 90% of requirements (this should be at the end of the first elaboration iteration))*	• Tài liệu đặc tả yêu cầu và phân tích nghiệp vụ *(Business analysis and requirements specification document)* • Danh sách các trường hợp sử dụng *(Use case catalog)*	Xem xét lại bởi nhóm
Hoàn thành 90% thiết kế (tức là nên ở cuối vòng lặp thứ 2 của giai đoạn Soạn thảo chi tiết - Elaboration) *(End of 90% design (this should be at the end of the second elaboration iteration))*	• Tài liệu thiết kế *(Design document)* • Mô hình đối tượng *(Object model)*	Xem xét lại bởi nhóm
Bắt đầu mỗi vòng lặp	Các kế hoạch của vòng lặp	Xem xét lại bởi 1 người

(Beginning of each iteration)	*(Iteration plans)*	*(One-person review)*
Kết thúc giai đoạn thiết kế chi tiết *(End of detailed design)*	Các đặc tả của chương trình phức tạp và các chương trình đầu tiên được tạo ra bao gồm các ca kiểm thử, các biểu đồ tương tác *(Complex and first-time-generated program specs including test cases, interaction diagrams)*	Xem xét lại bởi nhóm
Sau khi cài đặt mã cho vài chương trình đầu tiên *(After coding of first few programs)*	Mã *(Code)*	Xem xét lại bởi nhóm
After self-testing of a process	Mã *(Code)*	Xem xét lại bởi 1 người
Kết thúc giai đoạn lập kế hoạch kiểm thử đơn vị *(End of unit test plan)*	Kế hoạch kiểm thử đơn vị *(Unit test plan)*	Xem xét lại bởi 1 người
Bắt đầu kiểm thử tích hợp *(Beginning of integration test)*	Kế hoạch kiểm thử tích hợp *(Integration test plan)*	Xem xét lại bởi 1 người

Bởi vì xem xét lại (reviews) có vài trò then chốt của quy trình chất lượng, chúng được đề cập một cách riêng biệt trong kế hoạch chất lượng. Kế hoạch này xác định các thời điểm trong vòng đời phát triển (development life cycle) để thực hiện xem xét lại, sản phẩm công việc (work product) nào cần được xem xét lại, và bản chất của việc xem xét lại. Bảng 5.4 cho thấy những xem xét lại này trong dự án ACIC.

5.5 TÓM TẮT

Việc đảm bảo phần mềm cuối cùng có ít lỗi là một trong những mối quan tâm chính của người quản lý dự án. Trong cách tiếp cận theo thủ tục để quản lý chất lượng, các thủ tục kiểm soát chất lượng được lập kế hoạch và sau đó thực hiện theo. Trong cách tiếp cận quản lý chất lượng theo định lượng, mục tiêu chất lượng theo định lượng được thiết lập cho dự án; để đạt được mục tiêu này, quá trình thực hiện quy trình được giám sát dựa vào định lượng.

Sau đây là những bài học quan trọng từ cách tiếp cận của Infosys để quản lý chất lượng theo định lượng thông qua dự đoán lỗi:

- Như đối với quản lý nỗ lực và thời gian biểu, bạn có thể quản lý chất lượng bằng cách sử dụng số lỗi làm thước đo chất lượng.

- Thiết lập các mục tiêu chất lượng cho một dự án bằng cách dùng số lượng lỗi trong suốt quá trình kiểm thử chấp nhận. Sử dụng dữ liệu quá khứ về khả năng của quy trình để thiết lập mục tiêu này.

- Sử dụng dữ liệu quá khứ, ước lượng các mức lỗi cho giai đoạn phát hiện lỗi khác nhau trong quy trình này. So sánh những ước lượng này với số lỗi thực tế được tìm thấy trong quá trình thực hiện dự án để xem dự án có đang tiến triển đúng như mong đợi hoặc những điều chỉnh nào nên được thực hiện thêm.

- Ngoài kiểm thử, lập kế hoạch xem xét lại (reviews), xác định rõ các thời điểm xem xét lại, các sản phẩm công việc (work products) cần được xem xét lại, và các kiểu xem xét lại.

- Nếu mục tiêu chất lượng của dự án là cao hơn so với năng lực trong quá khứ, nó không thể được đạt được bằng cách sử dụng quy trình tương tự như quy trình của các dự án trước đó. Để đạt được các mục tiêu cao hơn, bạn phải cải tiến quy trình.

- Sử dụng việc phòng ngừa lỗi như một chiến lược để đạt được các mục tiêu chất lượng và năng suất cao hơn. Để phòng ngừa lỗi, xác định nhóm (team) phòng ngừa lỗi, những thời điểm mà tại đó việc phân tích lỗi sẽ được thực hiện, và những lợi ích mong đợi.

Các phương pháp được mô tả trong chương này thỏa mãn các yêu cầu hoạch định chất lượng của KPA Công Nghệ Sản Xuất Phần Mềm (Software Product Engineering KPA) và các yêu cầu lập kế hoạch của KPA Xem Xét Lại (Peer Review KPA) của CMM mức 3. Chúng cũng thỏa mãn yêu cầu lập kế hoạch chất lượng theo định lượng của KPA Quản Lý Chất Lượng Phần Mềm (Software Quality Management KPA) của CMM mức 4. Việc lập kế hoạch phòng ngừa lỗi thỏa mãn một số yêu cầu của KPA Phòng Ngừa Lỗi (Defect Prevention KPA) của CMM mức 5.

5.6 CÁC THAM KHẢO

1. International Standards Organization. *Information Technology—Software Product Evaluation—Quality Characteristics and Guidelines for Their Use*. ISO/IEC IS 9126, Geneva, 1991.

2. N.E. Fenton and S.L. Pfleeger. *Software Metrics: A Rigorous and Practical Approach*, second edition. International Thomson Computer Press, 1996.

3. B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

4. A.L. Goel. Software reliability models: Assumptions, limitations and applicability. *IEEE Transactions on Software Engineering*, 11, 1985.

5. S.H. Kan. *Metrics and Models in Software Quality Engineering*. Addison-Wesley, 1995.

6. J.D. Musa, A. Iannino, and K. Okumoto. *Software Reliability: Measurement, Prediction, Application*. McGraw Hill, 1987.

7. L.H. Putnam and W. Myers. *Measures for Excellence: Reliable Software on Time, within Budget*. Yourdon Press, 1992.

8. L.H. Putnam and W. Myers. *Industrial Strength Software: Effective Management Using Measurement*. IEEE Computer Society Press, 1997.

9. P. Jalote. *CMM in Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley, 2000.

Chương 6. Quản lý rủi ro

Dự án phần mềm là một công việc phức tạp. Các sự kiện bất ngờ có thể có tác động xấu đến chi phí của dự án, tiến độ (thời gian biểu), hoặc chất lượng. *Quản lý rủi ro* (*Risk management*) là một nỗ lực để giảm thiểu nguy cơ thất bại do các sự kiện bất ngờ. Mục đích của quản lý rủi ro không phải là để tránh đi vào các dự án có rủi ro mà là để giảm thiểu tác động của rủi ro (impact of risks) trong các dự án được thực hiện. N.R. Narayana Murthy, CEO của Infosys, đã nói: "Bất cứ điều gì có giá trị đáng để làm điều có rủi ro, thách thức đối với một người lãnh đạo không phải là tránh chúng mà là quản lý chúng một cách hiệu quả thông qua chiến lược giảm thiểu rủi ro". Quản lý rủi ro không đúng cách - kết quả chủ yếu của căn bệnh lạc quan phổ biến - là nguyên nhân của nhiều dự án thất bại.

Vasu được phân công làm người quản lý dự án cho một dự án lớn được thực hiện bởi một công ty đa quốc gia có uy tín. Dự án này xây dựng các bộ phận của một hệ thống tích hợp quản lý nguồn nhân lực trên toàn thế giới. Đối với hệ thống cuối cùng, phần mềm của nhóm Vasu phát triển phải được tích hợp vào một hệ thống được phát triển bởi nhà cung cấp khác (another vendor). Để sử dụng cho dự án, khách hàng đã cung cấp các công cụ (tools) độc quyền mà phiên bản mới của chúng được phát hành trong khoảng thời gian ngắn. Nhóm 35 người của Vasu có hơn một năm để giao phần mềm. Mặc dù dự án sử dụng một nhóm giỏi và người quản lý dự án đã thực hiện các ước lượng một cách hợp lý, nhưng hệ thống vẫn bị giao muộn đến sáu tháng, và Infosys đã phải tiêu tốn một lượng nỗ lực (effort) tăng 50% so với ước lượng ban đầu.

Tại sao dự án đã thất bại? Có hai lý do rõ ràng. Trước tiên, các phần mềm được phát triển bởi các nhà cung cấp khác đã không được giao đúng hạn, và các giao diện (interfaces) được cung bởi nhóm của Vasu liên tục thay đổi. Thứ hai, một phiên bản mới của các công cụ của khách hàng đã được phát hành trong suốt quá trình phát triển, đòi hỏi phần mềm phải được chuyển đến các phiên bản mới này.

Cả hai sự kiện này là những ví dụ rõ ràng về các rủi ro không được quản lý đúng cách. Những rủi ro này được thấy rõ từ lúc bắt đầu, mặc dù - giống như bất kỳ rủi ro nào khác - chúng không chắc chắn sẽ xảy ra. Khi dự án bắt đầu, người quản lý dự án, quản lý kinh

doanh, và khách hàng hy vọng rằng các nhà cung cấp khác sẽ cung cấp phần mềm đúng hạn và rằng phiên bản mới của công cụ này sẽ không ảnh hưởng đến dự án.

Sau này, sau khi khi rủi ro đã thật sự xảy ra, Vasu đã nghĩ rằng nếu nhóm chỉ đạo - bao gồm những người quản lý dự án của hai dự án và đại diện một khách hàng - đã được thiết lập để đảm bảo phối hợp tốt giữa hai dự án và các giao hàng của họ, sự chậm trễ có thể được giảm thiểu. Đối với rủi ro thứ hai, ông nghĩ rằng phần mềm nên đã được phát triển trước tiên dùng phiên bản trước đó của các công cụ. Sau đó, nó nên đã được di chuyển đến các phiên bản mới hơn sau đó thông qua một dự án riêng. Giải pháp này trước nhất có ưu điểm là sẽ cần nỗ lực tối thiểu, và có các tác động nhỏ đến chi phí. Thứ hai là nó có tác động rõ ràng đến chi phí cho khách hàng. Có lẽ để tránh phật lòng khách hàng, rủi ro này đã không được đánh dấu và việc làm giảm thiểu tác động của nó đã không được lên kế hoạch.

Chương này thảo luận các khái niệm chung về rủi ro và quản lý rủi ro trước khi chuyển sang phương pháp của Infosys để đánh giá và kiểm soát rủi ro - hai bước chính trong quản lý rủi ro.

6.1 KHÁI NIỆM RỦI RO VÀ QUẢN LÝ RỦI RO

Các rủi ro là những sự kiện hoặc điều kiện *có thể* xảy ra, và sự xảy ra của nó gây thiệt hại hoặc có tác dụng tiêu cực lên dự án. Các rủi ro không nên bị nhầm lẫn với các sự kiện và các điều kiện cần được can thiệp quản lý hoặc hành động. Một người quản lý dự án phải tính đến (deal) và lập kế hoạch cho những tình huống có khả năng xảy ra nhưng tính chất chính xác thì không được biết trước; tuy nhiên, những tình huống như vậy không phải là những rủi ro. Ví dụ, gần như chắc chắn rằng các lỗi sẽ được tìm thấy trong quá trình kiểm thử phần mềm, do đó, một dự án hợp lý phải có kế hoạch để sửa chữa lỗi này khi chúng được tìm thấy. Tương tự như vậy, gần như chắc chắn rằng một số yêu cầu sẽ bị thay đổi, do đó, quản lý dự án phải được chuẩn bị cho những thay đổi và lập kế hoạch phù hợp để xử lý các sự kiện thông thường như vậy.

Ngược lại, một lỗi là một sự kiện xác suất (probabilistic event) – nó có thể hoặc không thể xảy ra. Vì lý do này mà chúng tôi thường xuyên có khuynh hướng lạc quan, không nhìn thấy được các rủi ro hoặc mong muốn rằng chúng sẽ không xảy ra. Các yếu tố xã hội và tổ chức cũng có thể góp phần làm tăng rủi ro và ngăn cản xác định rõ chúng [1]. Loại nguyên nhân này làm cho dự án gặp rắc rối nếu các sự kiện rủi ro xảy ra thực sự, nhất là

trong những dự án lớn. Do đó, không có gì đáng ngạc nhiên, khi quản lý rủi ro được xem là thực hành đầu tiên trong số những thực hành tốt nhất để quản lý các phần mềm lớn [2].

Quản lý rủi ro nhằm mục đích để xác định các rủi ro (identify the risks) và sau đó có các hành động (actions) để giảm thiểu tác động của chúng đến dự án. Quản lý rủi ro là một lĩnh vực tương đối mới trong quản lý phần mềm. Nó lần đầu tiên được đề cập đến bởi Boehm thông qua một hướng dẫn quản lý rủi ro [3]. Kể từ đó, nhiều sách đã xem quản lý rủi ro là mục tiêu của phần mềm [4][5].

Trước khi thảo luận về các rủi ro trong bối cảnh phần mềm, chúng ta hãy xem xét nhiều hơn về khái niệm này thông qua một ví dụ. Hãy xem xét một chương trình máy tính mà mục tiêu quan trọng của nó là cung cấp các dịch vụ máy tính không bị gián đoạn. Đối với mục tiêu này, một trong những rủi ro rõ ràng là hỏng hóc nguồn điện. Nguồn điện có thể hoặc không thể hỏng. Nếu hỏng, thậm chí chỉ trong một giây, các dịch vụ máy tính sẽ bị ảnh hưởng đáng kể (các máy sẽ phải khởi động lại, dữ liệu sẽ bị mất, v.v.). Nếu trường hợp này là không thể chấp nhận được (có nghĩa là, nếu chi phí của hỏng hóc cao), một nguồn cung cấp năng lượng phụ (UPS) có thể được triển khai để giảm thiểu hậu quả của nó. Nếu có nghi ngờ rằng nguồn điện có thể bị mất một khoảng thời gian dài, một máy phát điện dự phòng có thể được thiết lập để giảm thiểu các vấn đề. Với hệ thống quản lý rủi ro này, nếu điện bị mất, ngay cả trong một thời gian dài, mục tiêu liên quan sẽ không bị tổn hại.

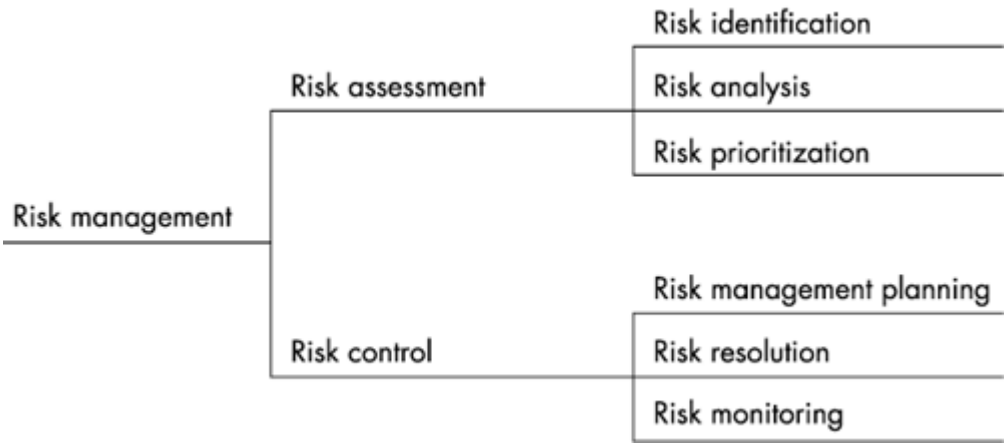
Vấn đề đầu tiên cần lưu ý từ ví dụ này là quản lý rủi ro đòi hỏi *chi phí bổ sung* (*additional cost*). Ở đây, chi phí cho UPS và máy phát điện được thêm vào bởi vì các thành phần này là không cần thiết nếu nguy cơ mất điện không tồn tại (ví dụ, nếu công ty cung cấp điện đảm bảo rằng nguồn điện luôn liên tục). Vì thế, quản lý rủi ro (risk management) có thể được coi là hiệu quả về chi phí (cost-effective) chỉ nếu chi phí quản lý rủi ro ít hơn đáng kể hơn so với các tổn thất khi rủi ro xảy ra thực sự [5]. (Thực tế, nói một cách ngắn gọn là chi phí quản lý rủi ro nên ít hơn so với *giá trị ước lượng (expected value)* của sự mất mát)) Ví dụ, nếu các tổn thất do mất điện là thấp, chi phí để đầu tư mua một UPS là không cần thiết.

Thứ hai, không phải dễ để đo lường giá trị của quản lý rủi ro, đặc biệt khi nhận thức muộn về nó. Nếu nguồn điện bị mất trong một giờ rưỡi, giá trị được cung cấp bởi UPS và máy phát điện có thể được tính như tiền "tiết kiệm" (savings) đạt được bằng cách có các máy tính chạy trong mất điện. Tuy nhiên, giả sử rằng nguồn cung cấp năng lượng không bị mất

thậm chí chỉ một giây và do đó UPS và máy phát điện không được sử dụng. Điều này có nghĩa là chi phí cho các thành phần này là một sự lãng phí? Không, bởi vì nguồn điện *có thể* mất. Nếu rủi ro không xảy ra thực sự, giá trị của việc sử dụng quản lý rủi ro không thể được đo trực tiếp bằng giá trị hoặc kết quả được tạo ra. Bởi vì các sự kiện rủi ro hầu như không xảy ra thường xuyên, nhiều khả năng là hệ thống quản lý rủi ro sẽ không được sử dụng trong quá trình thực hiện dự án. Đây là bản chất xác suất của các rủi ro và không thể luôn luôn nhận ra giá trị trực tiếp của nỗ lực làm giảm thiểu tác động của rủi ro – gây khó khăn cho việc quản lý rủi ro.

Từ ví dụ này, rõ ràng là bước đầu tiên trong quản lý rủi ro là đi xác định các rủi ro có thể xảy ra (mất điện trong ví dụ này) và đánh giá các hậu quả (mất uy tín hoặc khách hàng). Một khi bạn đã thực hiện đánh giá rủi ro (risk assessment), bạn có thể phát triển một kế hoạch quản lý rủi ro (ví dụ, cần có một UPS). Như vậy, nhìn chung, quản lý rủi ro có hai thành phần chính: đánh giá rủi ro (risk assessment) và kiểm soát rủi ro (risk control). Mỗi thành phần liên quan đến các nhiệm vụ khác nhau, như được trình bày trong Hình 6.1 [3].

Hình 6.1. Các hoạt động quản lý rủi ro



Mục đích của nhiệm vụ đánh giá rủi ro (risk assessment) là để xác định (identify) các rủi ro, phân tích chúng (analyze), và sau đó xếp độ ưu tiên chúng (prioritize). Trong việc xếp độ ưu tiên các rủi ro, bạn xác định những rủi ro cần được quản lý. Nói cách khác, độ ưu tiên xác định nỗ lực (chi phí) của việc quản lý rủi ro nên được dành cho chỗ nào để mang lại lợi ích nhiều nhất. Đối với nỗ lực (chi phí) này, có hai yếu tố quan trọng. Đầu tiên là khả năng để một rủi ro xảy ra; rủi ro nào có nhiều khả năng xảy ra sẽ là một ứng cử viên để đi quản lý nó. Thứ hai là tác động của rủi ro, rủi ro nào có tác động rất mạnh cũng là một ứng viên nặng ký.

Do đó, một cách để xếp độ ưu tiên cho các rủi ro là đi ước tính xác suất xảy ra (probability) và các hậu quả (consequences) của nó khi nó xảy ra. Tích của hai giá trị này, là giá trị ước lượng (dự kiến) (*expected value*) của sự mất mát do rủi ro (loss), có thể được sử dụng cho việc xếp độ ưu tiên. Giá trị này sẽ được gọi là *nguy cơ rủi ro* (*risk exposure*). Nếu Prob (R) là xác suất xảy ra của rủi ro R và Loss(R) là tổng mất mát nếu rủi ro R xảy ra, khi đó nguy cơ rủi ro, RE(R) được tính bởi phương trình sau:

$$RE(R) = Prob(R) \times Loss(R)$$

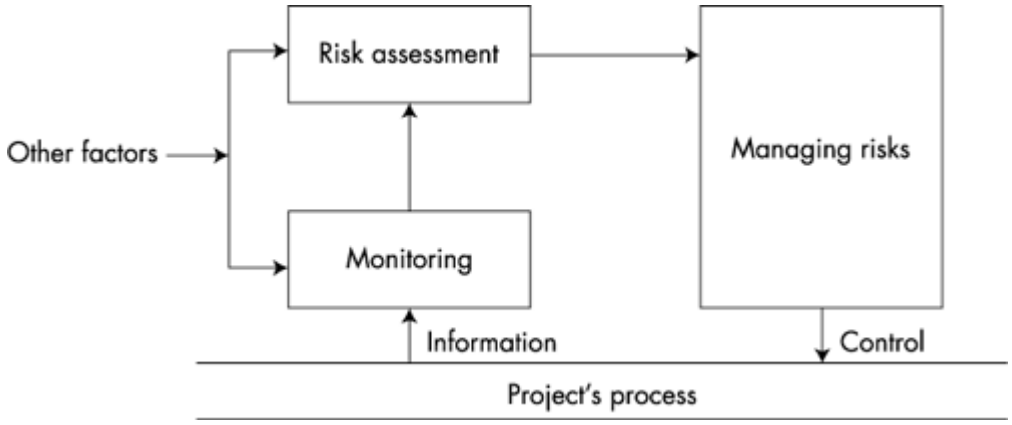
Sau khi những rủi ro đã được xếp độ ưu tiên, bạn phải quyết định xem làm gì đối với chúng và những rủi ro nào cần được quản lý bởi một quyết định quản lý. Có lẽ chỉ có vài rủi ro (nằm ở những dòng đầu) là cần được xử lý trong một dự án.

Một phương pháp để phòng ngừa hoặc tránh để các rủi ro không còn là một nguy cơ. Ví dụ, nếu phần cứng mới là một rủi ro (nguy cơ), nó có thể tránh được bằng cách thực hiện dự án bằng phần cứng đã được chứng minh. Tuy nhiên, hành động như vậy không phải lúc nào cũng khả thi, ví dụ, bị bắt buộc phải làm việc với phần cứng mới là một yêu cầu từ khách hàng. Trong những tình huống như vậy, những rủi ro của dự án phải được xử lý đúng cách.

Đối với mỗi các rủi ro được chọn để quản lý, bạn phải xây dựng và sau đó thực hiện một kế hoạch quản lý rủi ro (risk management plan). Bởi vì nhận thức (perception) về rủi ro thay đổi theo thời gian, bạn cũng phải giám sát (monitor) cả hai thứ: (1) rủi ro và (2) thực hiện (execution) các kế hoạch để giảm thiểu hậu quả của nó. Trong dự án, nhận thức về rủi ro có thể phát triển một cách tự nhiên, hoặc kế hoạch quản lý rủi ro ghi nhận các hành động có thể làm giảm tác động của rủi ro. Trong cả hai trường hợp, điều quan trọng là phải liên tục đánh giá tình trạng (status) của các rủi ro và kế hoạch quản lý chúng.

Quản lý rủi ro có thể được tích hợp vào quy trình phát triển, như được thực hiện trong phát triển phần mềm theo mô hình xoắn ốc (spiral model) [6]. Nếu bạn xem quản lý rủi ro như là một quy trình riêng biệt, bạn phải hiểu mối quan hệ của nó với quá trình thực hiện dự án (project's process), được miêu tả trong Hình 6.2. Như thể hiện trong hình, đánh giá rủi ro và giám sát thông tin từ quá trình thực hiện dự án, cùng với các yếu tố khác, để xác định các rủi ro cần được quản lý. Mặc khác, các hoạt động quản lý rủi ro sẽ ảnh hưởng đến quá trình giảm thiểu hậu quả của rủi ro.

Hình 6.2. Quản lý rủi ro và thực hiện dự án



Phần còn lại của chương này mô tả làm thế nào những người quản lý dự án tại Infosys quản lý rủi ro bằng cách sử dụng các kỹ thuật đơn giản, hiệu quả. Các hoạt động quản lý rủi ro được kết hợp lại thành hai nhiệm vụ: đánh giá rủi ro (risk assessment) và kiểm soát rủi ro (risk control). Chúng tôi sẽ thảo luận chúng một cách riêng rẽ.

6.2 ĐÁNH GIÁ RỦI RO (RISK ASSESSMENT)

Trong một dự án tại Infosys, đánh giá rủi ro bao gồm hai thành phần truyền thống: xác định (risk identification) và xếp độ ưu tiên rủi ro (risk prioritization). Các hoạt động xác định rủi ro tập trung vào liệt kê các rủi ro có thể xảy ra cho dự án. Các hoạt động cơ bản là để cố gắng hình dung ra tất cả các tình huống có thể làm cho dự án gặp rắc rối. Các hoạt động xếp độ ưu tiên rủi ro xem xét tất cả các khía cạnh của tất cả các rủi ro và sau đó xếp độ ưu tiên (cho các mục đích của quản lý rủi ro). Mặc dù là hai hoạt động riêng biệt, nhưng chúng thường được thực hiện cùng một lúc. Tức là, nhà người quản lý dự án có thể xác định và phân tích rủi ro chung với nhau.

6.2.1 Xác định rủi ro (Risk Identification)

Đối với một dự án, bất kỳ điều kiện (condition), tình hình (situation), hoặc sự kiện (event) có thể xảy ra và sẽ gây nguy hiểm cho sự thành công của dự án tạo nên rủi ro. Do đó, xác định rủi ro là một công việc hình dung những gì có thể làm cho dự án gặp rắc rối. Phương pháp giúp xác định rủi ro bao gồm các danh sách kiểm tra (checklists) của các rủi ro có thể xảy ra, các cuộc khảo sát (surveys), các cuộc họp (meetings), và xem xét lại (reviews) các kế hoạch, quy trình (process), và các sản phẩm công việc (work products) [5]. Danh

sách kiểm tra về các rủi ro thường xuyên xảy ra có lẽ là công cụ phổ biến nhất để xác định rủi ro. SEI cũng đã cung cấp một phân loại rủi ro để hỗ trợ việc xác định rủi ro này[7].

Ở Infosys, những rủi ro thường gặp đối với các dự án đã được biên soạn ra từ một cuộc khảo sát ở các dự án trước đó. Danh sách này sẽ là điểm khởi đầu để xác định các rủi ro cho dự án hiện tại. Thông thường thì các rủi ro trong dự án hiện tại sẽ xuất hiện trong danh sách.

Một người quản lý dự án cũng có thể sử dụng cơ sở dữ liệu quy trình (process database) để có được thông tin về rủi ro và quản lý rủi ro từ các dự án tương tự. Đánh giá và nghĩ về rủi ro gặp phải trước đây cũng giúp xác định các rủi ro khác có thể thích hợp cho dự án này, nhưng không xuất hiện trên danh sách.

Những người quản lý dự án cũng có thể sử dụng phán xét (judgment) và kinh nghiệm của họ để đánh giá tình hình để xác định các rủi ro tiềm năng. Một lựa chọn khác là sử dụng việc xem xét lại kế hoạch quản lý dự án và các cuộc họp thảo luận để gợi ra những quan điểm về rủi ro từ những người khác.

6.2.2 Xếp độ ưu tiên rủi ro (Risk Prioritization)

Những rủi ro đã được xác định cho một dự án chỉ ra các sự kiện có thể cản trở việc đạt được các mục tiêu của dự án. Tuy nhiên, các hậu quả của các rủi ro khác nhau có thể khác nhau. Trước khi bạn tiến hành quản lý rủi ro, bạn phải xếp độ ưu tiên cho chúng để quản lý sức lực có thể được tập trung vào những rủi ro cao nhất.

Bảng 6.1. Phân loại rủi ro

Xác suất (Probability)	**Phạm vi (Range)**
Thấp (Low)	0.0–0.3
Trung bình (Medium)	0.3–0.7
Cao (High)	0.7–1.0

Việc xếp độ ưu tiên đòi hỏi phải phân tích những tác động có thể có của từng sự kiện rủi ro trong trường hợp nó thực sự xảy ra. Tức là, nếu rủi ro xảy ra, dự án sẽ bị mất mát những gì? Sự mất mát có thể bao gồm một tổn thất trực tiếp, mất do mất cơ hội kinh doanh, hoặc kinh doanh trong tương lai, mất mát do tinh thần nhân viên giảm sút, v.v.

Dựa trên những hậu quả có thể xảy ra và xác suất để sự kiện rủi ro xảy ra, bạn có thể tính toán nguy cơ rủi ro (risk exposure), mà sau đó bạn có thể sử dụng để xếp độ ưu tiên rủi ro.

Cách tiếp cận này đòi hỏi phải có một đánh giá định lượng (quantitative assessment) xác suất và hậu quả rủi ro. Thông thường, ít dữ liệu lịch sử có sẵn để giúp bạn thực hiện một ước tính định lượng của các tham số. Bởi vì rủi ro là những sự kiện xác suất, chúng xảy ra không thường xuyên, và làm cho ta khó khăn để thu thập dữ liệu về chúng. Hơn nữa, bất kỳ dữ liệu nào như vậy phải được hiểu đúng bởi vì các hoạt động quản lý rủi ro ảnh hưởng đến chúng. Thực tế này ngụ ý rằng việc xếp độ ưu tiên rủi ro sẽ được làm dựa trên kinh nghiệm hơn là dựa trên dữ liệu cứng nhắc từ quá khứ. Trong tình huống này, phân loại cả hai xác suất và những hậu quả giúp ta tách ra các mục rủi ro có độ ưu tiên cao ra khỏi các mục rủi ro có độ ưu tiên thấp hơn [5]. Tại Infosys, xác suất để một rủi ro xảy ra được phân loại là thấp, trung bình, hoặc cao. Bảng 6.1 cung cấp cho phạm vi xác suất cho mỗi loại.

Để xếp hạng các tác động của một rủi ro trong một dự án, bạn phải chọn một đơn vị tác động (unit of impact). Để đơn giản hóa việc quản lý rủi ro, những người quản lý dự án tại Infosys xếp hạng tác động của rủi ro bằng cách dùng một thang điểm từ 0 đến 10. Trong phạm vi thang điểm này, tác động rủi ro có thể được xếp hạng là thấp, trung bình, cao, hoặc rất cao. Bảng 6.2 trình bày phạm vi điểm của mỗi xếp hạng.

Bảng 6.2. Phân loại tác động

Mức hậu quả (Level of Consequences)	**Phạm vi (Range)**
Thấp (Low)	0.0–3.0
Trung bình (Medium)	3.0–7.0
Cao (High)	7.0–9.0
Rất cao (Very high)	9.0–10.0

Với những xếp hạng và phạm vi điểm của chúng, phương pháp đơn giản sau đây sẽ được dùng để xếp độ ưu tiên rủi ro:

1. Đối với mỗi rủi ro, xếp hạng xác suất xảy ra của nó là thấp, trung bình, hoặc cao. Nếu cần thiết, gán phạm vi giá trị xác suất cho mỗi xếp hạng.

2. Đối với mỗi rủi ro, đánh giá tác động của nó lên dự án là thấp, trung bình, cao, hoặc rất cao. Nếu cần thiết, gán một trọng lượng trên thang điểm từ 0 đến 10.

3. Xếp hạng các rủi ro dựa trên xác suất và ảnh hưởng của chúng lên dự án; ví dụ, một rủi ro có xác suất cao, tác động cao sẽ có thứ hạng cao hơn so với một mục rủi ro có xác suất trung bình và tác động cao. Trong trường hợp xung đột, sử dụng phán xét của bạn (hoặc gán những con số để tính toán một giá trị số của nguy cơ rủi ro (risk exposure)).

4. Chọn các mục rủi ro hàng đầu (top few risk items) để giảm thiểu và theo dõi (mitigation and tracking).

Mục tiêu chính của quản lý rủi ro là xác định các mục rủi ro hang đầu và sau đó tập trung vào chúng. Với mục tiêu này, cần phải thực hiện tốt việc phân loại. Rõ ràng, một rủi ro có xác suất xảy ra cao và có những hậu quả cao sẽ được xếp độ ưu tiên cao cho việc quản lý rủi ro.

Khi bạn làm việc với phân loại, một vấn đề có thể phát sinh là tồn tại đồng thời hai rủi ro có xác suất xảy ra và ảnh hưởng lần lượt là (cao, trung bình) hoặc (trung bình, cao). Trong trường hợp này, ta không biết phải xếp rủi ro nào có độ ưu tiên cao hơn. Một tiếp cận dễ dàng để xử lý tình trạng này là đi giảm thiểu cả hai rủi ro này. Nếu cần thiết, bạn có thể phân biệt giữa các loại rủi ro bằng cách sử dụng các con số cụ thể.

Tiếp cận xếp độ ưu tiên cho những rủi giúp bạn chú ý tập trung vào rủi ro cao, nhưng nó không giúp bạn trong việc đưa ra một phân tích về hiệu quả chi phí (cost-benefit analysis) để chọn lựa các tùy chọn giảm thiểu rủi ro (risk mitigation options). Tức là, nó nêu rõ hậu quả của rủi ro bằng cách dùng tỷ lệ kích thước (scale) chứ không bằng giá trị tiền, phương pháp này không cho phép bạn để tính toán tổn thất dự kiến về tài chính. Do đó, bạn không thể phân tích xem một chiến lược giảm thiểu rủi ro cụ thể, sẽ tiêu tốn một chi phí là bao nhiêu tiền, có đáng để triển khai nó không. Tuy nhiên, một phân tích như thế nói chung là không cần thiết bởi vì trọng tâm của quản lý rủi ro thường là về quản lý rủi ro với chi phí thấp nhất, chứ không phải đi xem xét bản thân việc quản lý rủi ro có mang lại lợi ích hay không. Ngược lại, nếu bạn phải đưa ra quyết định về việc liệu một rủi ro có cần được quản lý hoặc không cần phải quản lý thì sẽ tiết kiệm tài chính hơn, bạn cần phải biết tác động lên tài chính của rủi ro.

6.3 KIỂM SOÁT RỦI RO (RISK CONTROL)

Một khi người quản lý dự án đã xác định và xếp độ ưu tiên cho rủi ro, câu hỏi đặt ra bây giờ là sẽ làm gì với chúng. Biết được những rủi ro chỉ có giá trị khi bạn có thể chuẩn bị một kế hoạch để mà hậu quả của chúng được giảm thiểu – đây chính là mục tiêu cơ bản của quản lý rủi ro. Bạn giảm thiểu các tác động của rủi ro ở bước thứ hai của quản lý rủi ro: kiểm soát rủi ro. Về cơ bản, bước này liên quan đến việc lập kế hoạch giảm thiểu rủi ro, sau đó là thực hiện theo kế hoạch này và giám sát các rủi ro.

6.3.1 Lập kế hoạch quản lý rủi ro

Một khi rủi ro được xác định và xếp độ ưu tiên, người quản lý dự án đã biết rõ rằng những rủi ro nào cần phải xử lý. Để quản lý những rủi ro này, lập kế hoạch thích hợp là cần thiết. Công việc chính là để xác định các hành động cần thiết để giảm thiểu những hậu quả rủi ro, thường được gọi là bước giảm thiểu rủi ro (*risk mitigation steps*). Như đối với hoạt động xác định rủi ro, bạn tham khảo một danh sách các bước giảm thiểu rủi ro thường được sử dụng rồi lựa chọn ra một bước giảm thiểu rủi ro (risk mitigation step) phù hợp. Danh sách được sử dụng tại Infosys được trình bày trong Bảng 6.3. Bảng này là một điểm khởi đầu không chỉ để xác định rủi ro xác định mà còn để lựa chọn các bước giảm thiểu rủi ro sau khi rủi ro đã được xếp độ ưu tiên. Như đối với xác định rủi ro, bạn sẽ không bị giới hạn là chỉ sử dụng các bước được đề cập trong Bảng 6.3. Bạn cũng có thể sử dụng cơ sở dữ liệu quy trình để xác định các rủi ro và các bước giảm thiểu rủi ro.

Hầu hết các rủi ro và các bước giảm thiểu trong Bảng 6.3 thì có thể hiểu được (không cần giải thích) (self-explanatory). Như bạn có thể thấy, những rủi ro hàng đầu có liên quan đến nhân lực và các yêu cầu. Nhiều mục rủi ro ở đây cũng tương tự như những mục rủi ro hàng đầu được đưa ra bởi Boehm [3] và Hall [5]. Chọn một bước giảm thiểu rủi ro không chỉ là một công việc trí óc. Bước giảm thiểu rủi ro phải được thực hiện (và được theo dõi). Để đảm bảo rằng các hành động cần thiết được thực hiện đúng, bạn phải kết hợp chúng vào kế hoạch của dự án. Nói cách khác, bạn phải cập nhật thời gian biểu của dự án (project schedule) – liệt kê các hoạt động khác nhau và xác định khi nào chúng xảy ra - để bao gồm các hành động liên quan đến các bước giảm thiểu rủi ro đã được chọn.

6.3.2 Giám sát và theo dõi rủi ro (Risk Monitoring and Tracking)

Xếp độ ưu tiên rủi ro và lập kế hoạch kết quả được dựa trên nhận thức (perception) về rủi ro tại thời điểm phân tích rủi ro được thực hiện. Phân tích rủi ro đầu tiên diễn ra trong quá trình lập kế hoạch dự án, và kế hoạch quản lý rủi ro ban đầu phản ánh quan điểm của tình hình tại thời điểm đó. Bởi vì rủi ro là những sự kiện xác suất, thường xuyên phụ thuộc vào các yếu tố bên ngoài*, mối đe dọa* do các rủi ro có thể thay đổi theo thời gian là một yếu tố thay đổi. Rõ ràng, khi đó, nhận thức về rủi ro cũng có thể thay đổi theo thời gian. Hơn nữa, các bước giảm thiểu rủi ro được thực hiện có thể ảnh hưởng đến nhận thức về rủi ro.

Tính động này (dynamism) cho thấy rằng rủi ro trong một dự án không nên được chữa trị một cách tĩnh và phải được đánh giá lại theo định kỳ. Do đó, ngoài việc theo dõi tiến độ của các bước giảm thiểu rủi ro đã được hoạch định, bạn phải định kỳ xem xét lại nhận thức về rủi ro cho toàn bộ dự án. Kết quả của xem xét lại được báo cáo trong mỗi báo cáo phân tích tại các cột mốc (milestone) (xem Chương 11), bạn báo cáo tình trạng của các bước giảm thiểu rủi ro cùng với nhận thức về rủi ro và chiến lược ở hiện tại. Để chuẩn bị báo cáo này, bạn thực hiện một phân tích rủi ro mới để xác định xem các ưu tiên có cần được thay đổi không.

Bảng 6.3. Tốp 10 rủi ro và các bước giảm thiểu rủi ro của chúng

STT	Loại rủi ro (Risk Category)	Các bước giảm thiểu rủi ro (Risk Mitigation Steps)
1	Thiếu nhân lực được huấn luyện về kỹ thuật *(Shortage of technically trained manpower)*	• Làm các ước lượng về chi phí dùng để học tập lúc bắt đầu *(Make estimates with a little allowance for initial learning time).* • Để dành bộ đệm cho nhân sự bổ *sung (Maintain buffers of extra resources).* • Xác định một chương trình đào tạo cho dự án *(Define a project-specific training program).* • Tiến hành các buổi đào tạo chéo *(Conduct cross-training sessions).*

2	Yêu cầu thay đổi quá nhiều *(Too many requirement changes)*	• Có được chữ ký của khách hàng cho các đặc tả yêu cầu ban đầu *(Obtain sign-off for the initial requirements specification from the client)*. • Thuyết phục khách hàng rằng những thay đổi trong các yêu cầu sẽ ảnh hưởng đến tiến độ *(Convince the client that changes in requirements will affect the schedule).* • Xác định một thủ tục để xử lý các thay đổi yêu cầu *(Define a procedure to handle requirement changes).* • Đàm phán để được thanh toán theo nỗ lực (chi phí) thực tế *(Negotiate payment on actual effort).*
3	Yêu cầu không rõ ràng *(Unclear requirements)*	• Sử dụng kinh nghiệm và lôgic để làm một số giả định và liên tục thông báo cho khách hàng biết; có được chữ ký của khách hàng *(Use experience and logic to make some assumptions and keep the client informed; obtain sign-off).* • Phát triển một bản (mẫu) thử nghiệm (prototype) và khách hàng thực hiện xem xét lại yêu cầu *(Develop a prototype and have the requirements reviewed by the client).*
4	Mất nhân sự (nhân viên có thể rời công ty) *(Manpower attrition)*	• Đảm bảo rằng các nhân sự được phân công vào các lĩnh vực chính của dự án *(Ensure that multiple resources are assigned on key project areas)*. • Có buổi họp để xây dựng đội nhóm *(Have team-building sessions)*. • Xoay công việc giữa các thành viên trong nhóm *(Rotate jobs among team members).* • Để dành nhân sự bổ sung cho dự án làm dự phòng *(Keep*

		extra resources in the project as backup). • Lưu giữ các tài liệu đúng đắn về công việc của mỗi cá nhân *(Maintain proper documentation of each individual's work).* • Thực hiện quy trình quản lý cấu hình và các hướng dẫn một cách nghiêm ngặt *(Follow the configuration management process and guidelines strictly).*
5	Nhiều quyết định từ bên ngoài gây ép buộc lên dự án *(Externally driven decisions forced on the project)*	• Liệt kê những bất lợi có kèm theo các sự kiện và dữ liệu và sau đó đàm phán với nhân viên chịu trách nhiệm về các quyết định ép buộc (Outline disadvantages with supporting facts and data and negotiate with the personnel responsible for forcing the decisions). • Nếu không thể tránh khỏi, xác định các rủi ro thực tế và thực hiện theo kế hoạch giảm thiểu nó *(If inevitable, identify the actual risk and follow its mitigation plan).*
6	Không đáp ứng được yêu cầu về hiệu suất *(Not meeting performance requirements)*	• Xác định các tiêu chuẩn về hiệu suất một cách rõ ràng và đưa cho khách hàn xem xét lại chúng *(Define the performance criteria clearly and have them reviewed by the client).* • Xác định các tiêu chuẩn để được thực hiện theo để đáp ứng các tiêu chuẩn về hiệu suất *(Define standards to be followed to meet the performance criteria).* • Chuẩn bị một thiết kế để đáp ứng các tiêu chuẩn về hiệu suất và xem xét lại nó *(Prepare the design to meet performance criteria and review it).* • Mô phỏng hiệu suất mẫu cho các giao dịch quan trọng *(Simulate or prototype performance of critical transactions).* • Kiểm thử với dữ liệu mẫu nếu có thể *(Test with a representative volume of data where possible).*

		• Thực hiện các kiểm thử về căng thẳng nếu có thể *(Conduct stress tests where possible).*
7	Thời gian biểu được xếp không đúng với thực tế *(Unrealistic schedules)*	• Đàm phán để có được một thời gian biểu tốt hơn (Negotiate for a better schedule). • Xác định các công việc song song (Identify parallel tasks). • Có nguồn nhân sự sẵn sàng sớm (Have resources ready early). • Xác định các khu vực có thể được tự động hóa (Identify areas that can be automated). • Nếu đường tới hạn không nằm trong phạm vi thời gian biểu, hãy đàm phán với khách hang *(If the critical path is not within the schedule, negotiate with the client).* • Đàm phán để được thanh toán theo nỗ lực (chi phí) thực tế *(Negotiate payment on actual effort).*
8	Làm việc với kỹ thuật mới (phần cứng và phần mềm) *(Working on new technology : hardware and software)*	• Hãy dùng cách giao sản phẩm (phần mềm) theo từng giai đoạn *(Consider a phased delivery).* • Bắt đầu giao các mô-đun quan trọng *(Begin with the delivery of critical modules).* • Đưa thời gian vào trong thời hạn hoàn thành để xây dựng đường cong được học *(Include time in the schedule for a learning curve).* • Cung cấp khóa đào tạo về công nghệ mới *(Provide training in the new technology).* • Phát triển một ứng dụng để chứng minh cho ý niệm *(Develop a proof-of-concept application).*

9	Thiếu kiến thức trong lĩnh vực ứng dụng *(Insufficient business knowledge)*	• Tăng cường tương tác với khách hàng và đảm bảo chuyển giao kiến thức đầy đủ *(Increase interaction with the client and ensure adequate knowledge transfer).* • Tổ chức tập huấn kiến thức chuyên môn trong lĩnh vực ứng dụng *(Organize domain knowledge training).* • Mô phỏng hoặc tạo bản mẫu (prototype) cho các giao dịch kinh doanh cho khách hàng và đạt được phê duyệt của khách hàng *(Simulate or prototype the business transaction for the client and get it approved).*
10	Lỗi liên kết hoặc hiệu suất chậm *(Link failure or slow performance)*	• Thiết lập các kỳ vọng hợp lý với khách hàng *(Set proper expectations with the client).* • Lập kế hoạch trước để nạp liên kết *(Plan ahead for the link load).* • Kế hoạch sử dụng liên kết một cách tối ưu *(Plan for optimal link usage).*

6.4 CÁC VÍ DỤ

Phần này bao gồm hai kế hoạch quản lý rủi ro: một cho dự án ACIC và một từ một dự án khác (ở đây gọi là XYZ). Như bạn sẽ thấy, các kế hoạch quản lý rủi ro có xu hướng nhỏ - thường là một bảng đặt vừa vào một trang. Các hoạt động được đề cập trong kế hoạch giảm thiểu trở thành một phần của các hoạt động của dự án và thậm chí có thể được xếp lịch rõ ràng.

6.4.1 Dự án ACIC

Người quản lý dự án ACIC đã chọn để làm việc thông qua những con số cho việc xếp độ ưu tiên và phân tích rủi ro. Như được trình bày trong Bảng 6.4, các mục rủi ro nằm ở tốp đầu có hạng ảnh hưởng trong khoảng từ 8 đến 3. Các bước giảm thiểu rủi ro cũng được chỉ ra cho mỗi rủi ro. Ví dụ, rủi ro thứ hai là làm việc với phương phát triển RUP. Đó là, dự

án phải gánh chịu một rủi ro bởi vì đây là một phương pháp mới đối với các thành viên trong dự án. Hơn nữa, có vẻ như là các dự án khác đã không sử dụng RUP. Một trong các kế hoạch giảm thiểu rủi ro rõ ràng nhất là: lập kế hoạch để đào tạo cho nhóm về phương pháp RUP. Ngoài ra, nó cũng gợi ý rằng các thành viên cần được tư vấn từ những người ở phòng thí nghiệm R&D bởi vì họ có kiến thức về RUP và các khái niệm liên quan, và giữ liên hệ thường xuyên với khách hàng trong suốt vòng lặp phát triển, và bất cứ vấn đề phát sinh nào cần được giải quyết nhanh chóng.

Bảng 6.4. Kế hoạch quản lý rủi ro cho dự án ACIC

STT	Các rủi ro (Risks)	Xác suất (Probability)	Ảnh hưởng (Impact)	Nguy cơ rủi ro (Risk Exposure)	Kế hoạch giảm thiểu (Mitigation Plan)
1	Chúng tôi sẽ cần sự hỗ trợ từ kiến trúc sư cơ sở dữ liệu và người quản trị cơ sở dữ liệu của khách hàng.	0.5	8	4	Lập kế hoạch cẩn thận về khoảng thời gian cần đến sự hỗ trợ từ mỗi nhóm này và cung cấp đủ thông báo trước. Có một điều phối viên để làm việc chặt chẽ với các nhóm này.
2	Bởi vì RUP sẽ được sử dụng lần đầu tiên, sự hiểu biết của nhóm về nó thì không đầy đủ.	0.9	3	2.7	Làm việc chặt chẽ với các chuyên gia ở phòng thí nghiệm R&D của Infosys. Giữ liên hệ thường xuyên với khách hàng trong vòng lặp phát triển suốt dự án, và giải quyết nhanh chóng bất cứ vấn đề phát sinh nào liên quan đến lịch và

					chi phí (nỗ lực). Lập kế hoạch để đào tạo cho nhóm về phương pháp RUP.
3	Mất nhân sự: Các thành viên trong nhóm có thể rời công ty trong thời gian ngắn sắp tới.	0.3	7	2.1	Phân công công việc/nhiệm vụ để mà có nhiều hơn một người có hiểu biết về các unit (đơn vị chương trình) và use case (trường hợp sử dụng) trong dự án.
4	Làm việc với máy tính lớn (mainframe) DB2 của khách hàng qua liên kết: Liên kết này có thể không được hiệu quả như dự kiến.	0.1	8	0.8	Thực hiện thêm các kỹ thuật tĩnh như: xem xét lại mã (code review), kiểm tra tại bàn (desk checking), v.v. để giảm thiểu sự phụ thuộc vào liên kết. Giải quyết ngay vấn đề leo thang (escalate) khi liên kết bị hõng (down).

Lưu ý rằng một khi các tùy chọn này được chấp nhận như là các bước giảm thiểu rủi ro, chúng sẽ ảnh hưởng đến kế hoạch chi tiết của dự án; thời gian biểu phải bao gồm thời gian đào tạo phù hợp và các hoạt động xây dựng để chứng minh giải pháp ý tưởng (proof-of-concept building activities). Nhu cầu này sẽ được cần đến ở nhiều bước giảm thiểu rủi ro. Bởi vì chúng trình bày các hoạt động và bởi vì thời gian biểu chi tiết của dự án trình bày hầu hết các hoạt động sẽ được thực hiện trong dự án, các bước giảm thiểu rủi ro sẽ thường xuyên làm thay đổi thời gian biểu chi tiết của dự án, cần bổ sung thêm chi phí (nỗ lực) cho dự án.

6.4.2 Dự án XYZ

Dự án này sử dụng hệ thống xếp hạng để quản lý rủi ro của nó. Bảng 6.5 cho các xếp hạng khác nhau và các bước giảm thiểu rủi ro. Kế hoạch quản lý rủi ro này là một phần của kế hoạch quản lý dự án cho dự án và đã được trích ra từ nó.

Phương pháp thực hiện phân tích rủi ro tại các một mốc quan trọng về cơ bản giống như mô tả trước đây, ngoại trừ rằng có sự chú ý nhiều hơn ở những rủi ro đã được liệt kê trong kế hoạch dự án (tức là, có nhấn mạnh nhiều hơn ở đầu ra của phân tích rủi ro trước đó của dự án). Suốt quá trình phân tích rủi ro này, những người quản lý dự án có thể xếp lại độ ưu tiên của rủi ro. Trong dự án XYZ, khi một phân tích được thực hiện tại một mốc quan trọng (milestone) khoảng ba tháng sau khi kế hoạch quản lý rủi ro ban đầu đã được thực hiện, nhận thức về rủi ro đã được thay đổi phần nào. Bảng 6.6 trình bày các rủi ro mà nguy cơ rủi ro (exposure) vừa được thay đổi.

Dựa trên kinh nghiệm thu được từ dự án đang được thực hiện đến thời điểm hiện tại, người quản lý dự án đã quyết định rằng hậu quả của các hòa giải thay đổi gây ra ít nghiêm trọng. Điều này đã xảy ra bởi vì, ví dụ, các vấn đề hòa giải gặp ít khó khăn hơn so với dự kiến. Tương tự như vậy, sự nhận thức về rủi ro mất nhân sự đã tăng lên – có lẽ là vì kinh nghiệm của các thành viên trong nhóm và có lẽ thực tế là người rời nhóm vào giữa dự án sẽ gây ra vấn đề lớn hơn người rời nhóm vào lúc bắt đầu dự án. Bất cứ khi nào rủi ro được phân tích lại, kế hoạch giảm thiểu rủi ro cũng có thể bị thay đổi, tùy thuộc vào thực tế hiện tại của dự án và tính chất của các rủi ro. Trong dự án này, không có thay đổi trong các kế hoạch giảm thiểu rủi ro.

Bảng 6.5. Kế hoạch quản lý rủi ro cho dự án XYZ

STT	Rủi ro (Risk)	Xác suất (Probability)	Hậu quả (Consequences)	Nguy cơ rủi ro (Risk Exposure)	Kế hoạch giảm thiểu (Mitigation Plan)
1	Không đáp ứng được hiệu suất cao	Cao *(High)*	Cao	Cao	• Chỉ ra cho khách hàng thấy được năng suất dự kiến

	(Failure to meet the high performance)				thông qua bản mẫu thử nghiệm *(Indicate expected performance to clients through requirements prototypes)* • Sử dụng lời khuyên từ cơ sở dữ liệu kiến thức để cải thiện hiệu suất *(Use tips from body of knowledge database to improve performance)* • Làm cho nhóm hiểu rõ được các yêu cầu *(Make team aware of the requirements)* • Cập nhật danh sách kiểm tra để xem xét lại, tìm kiếm chỗ nguy hiểm làm giảm hiệu suất *(Update the review checklist to look for performance pitfalls)* • Nghiên cứu và cải thiện hiệu suất liên tục *(Study and improve performance constantly)* • Thực hiện theo các

					hướng dẫn từ các nghiên cứu về hiệu suất trước đó *(Follow guidelines from earlier performance studies)* • Kiểm thử ứng dụng xuyên suốt giai đoạn tích hợp và kiểm thử hệ thống để xem nó có đáp ứng được hiệu suất kỳ vọng hay không *(Test application for meeting performance expectations during integration and system testing)*
2	Thiếu nhân sự có sẵn kỹ năng phù hợp *(Lack of availability of persons with the right skills)*	Trung bình *(Medium)*	Trung bình	Trung bình	• Đào tạo nhân sự *(Train resources)* • Cùng với khách hàng xem xét lại bản mẫu thử nghiệm *(Review prototype with customer)* • Phát triển các mã thực hành *(Develop coding practices)*
3	Phức tạp của yêu cầu của ứng dụng	Trung bình	Trung bình	Trung bình	• Đảm bảo chuyển giao kiến thức liên tục *(Ensure ongoing*

	(Complexity of application requirements)				*knowledge transfer)* • Triển khai người đã có kinh nghiệm về ứng dụng *(Deploy persons with prior experience with the application)*
4	Mất nhân sự *(Manpower attrition)*	Trung bình	Trung bình	Trung bình	• Đào tạo một nhóm nòng cốt có bốn người *(Train a core group of four people)* • Xoay vòng phân công công việc cho tất cả mọi người *(Rotate onsite assignments among people)* • Xác định nhân sự dự phòng cho các vai trò chủ chốt *(Identify backups for key roles)*
5	Yêu cầu không rõ ràng *(Unclear requirements)*	Trung bình	Trung bình	Trung bình	• Xem xét lại một bản mẫu thử nghiệm *(Review a prototype)* • Tiến hành xem xét lại ở giữa giai đoạn (Conduct a midstage review)

6	Khó khăn để xây dựng cơ chế hoà giải các thay đổi được thực hiện ở một phía trong suốt quá trình phát triển cách xa về mặt địa lý. *(Difficulty of reconciliation configuration of changes done in onsite maintenance during off-shore development)*	Trung bình	Thấp *(Low)*	Trung bình	• Tạo một kế hoạch quản lý và tuân thủ để tiếp cận hòa giải đã được định nghĩa rõ ràng *(Create a management plan and adhere to well-defined reconciliation approach).* • Hòa giải mỗi tháng một lần: ngày Thứ 3 đầu tiên hoặc vào ngày làm việc tiếp theo *(Reconcile once per month: first Tuesday or next working day)* • Đừng hòa giải các thay đổi đã được thực hiện sau ngày ngày khóa sổ/ ngày đáo hạn *(Do not reconcile changes done after a cut-off date)* .

Bảng 6.6. Sự tiến hóa rủi ro trong dự án XYZ

STT	Rủi ro (Risk)	Xác suất hiện tại (Current Probability)	Các hậu quả hiện tại (Current Consequences)	Nguy cơ rủi ro hiện tại (Current Risk Exposure)
2	Mất nhân sự *(Manpower attrition)*	Cao	Cao	Cao
3	Khó khăn trong việc hòa giải các thay đổi được tạo ra ở một phía trong quá trình phát triển cách xa về mặt địa lý. *(Difficulty of reconciliation of changes created in onsite maintenance during off-shore development)*	Thấp	Thấp/Trung bình	Thấp

6.5 TÓM TẮT

Một rủi ro của một dự án là một điều kiện không chắc chắn xảy ra nhưng có thể ảnh hưởng bất lợi đến dự án. Quản lý rủi ro đòi hỏi rằng những rủi ro phải được xác định và được xếp hạng ưu tiên, và đối với các rủi ro nằm ở tốp đầu, nhiều hoạt động phải được thực hiện để giảm thiểu tác động của chúng. Chi phí để giảm thiểu rủi ro có thể có vẻ lãng phí khi các rủi ro không xảy ra, nhưng chúng phải được đầu tư để giảm thiểu thiệt hại trong trường hợp rủi ro xảy ra.

Sau đây là một số trong những bài học quan trọng từ tiếp cận quản lý rủi ro ở Infosys:

- Để giúp bạn xác định các rủi ro, một danh sách các rủi ro thường xảy ra sẽ là một điểm khởi đầu tốt. Ngoài ra, nhìn về phía trước và cố gắng hình dung ra mọi thứ có thể làm cho dự án gặp rắc rối.

- Đối với việc xếp độ ưu tiên cho rủi ro, một cơ chế đơn giản và hiệu quả là đi phân loại xác suất rủi ro và tác động của chúng ra thành các loại như: thấp, trung bình và cao, và sau đó quản lý những rủi ro có xác suất và tác động cao.

- Đối với các rủi ro nằm ở tốp đầu, lập kế hoạch cho các bước giảm thiểu rủi ro, và đảm bảo rằng chúng sẽ được thực hiện trong quá trình thực hiện dự án.

- Giám sát và đánh giá lại các rủi ro định kỳ, có lẽ tại các cột mốc (milestones), để xem các bước giảm thiểu rủi ro đã được áp dụng có mang lại kết quả tốt không và để xem xét lại nhận thức về rủi ro.

Tương ứng CMM, KPA Lập Kế Hoạch Cho Dự Án (Project Planning KPA) ở CMM mức 2 yêu cầu rằng một dự án phải có một kế hoạch quản lý rủi ro. Các quy trình quản lý và giám sát rủi ro là một yêu cầu cho KPA Quản Phần Mềm Lý Tổng Hợp (Integrated Software Management KPA) ở CMM mức 3.

6.6 CÁC THAM KHẢO

1. R.N. Charette. Large-scale project management is risk management. *IEEE Software*, July 1996.
2. N. Brown. Industrial-strength management strategies. *IEEE Software*, July 1996.
3. B. Boehm. *Tutorial: Software Risk Management. IEEE Computer Society*, 1989.
4. R. Charette. *Software Engineering Risk Analysis and Management*. McGraw Hill, 1989.
5. E.M. Hall. *Managing Risk: Methods for Software Systems Development*. Addison-Wesley, 1998.
6. B. Boehm. A spiral model of software development and enhancement. *IEEE Computer*, May 1988.
7. M. Carr et al. *Taxonomy-based Risk Identification*. Technical Report, CMU/SEI-93-TR-006, 1993.

Chương 7. Đo lường và lập kế hoạch theo dõi

Một kế hoạch quản lý dự án đơn thuần là một tài liệu có thể được sử dụng để hướng dẫn thực hiện dự án. Trừ khi hiệu suất thực tế của sự thực hiện này được theo dõi theo sát với kế hoạch, kế hoạch quản lý dự án có một giá trị giới hạn. Và để theo dõi dự án, giá trị của một số thông số quan trọng phải được đo trong suốt quá trình thực hiện dự án. Theo dõi là một nhiệm vụ khó, và, giống như với các nhiệm vụ khác, nếu bạn muốn thực hiện nó đúng cách, bạn phải lập kế hoạch cho nó. Trong quá trình lập kế hoạch, bạn phải quyết định cho các vấn đề phát sinh như công việc/nhiệm vụ, nỗ lực (chi phí), và lỗi sẽ được theo dõi, những công cụ nào sẽ được sử dụng, những cấu trúc nào và tần suất của báo cáo sẽ được thực hiện như thế nào, v.v.

Chương này thảo luận về các phép đo được thực hiện trong các dự án tại Infosys. Nó cũng mô tả việc lập kế hoạch theo dõi và lựa chọn ngưỡng cho mức biến đổi hiệu suất - được sử dụng để kích hoạt các hành động quản lý. Việc theo dõi dự án thực tế được thảo luận trong các Chương 10 và 11.

7.1 CÁC KHÁI NIỆM VỀ ĐO LƯỜNG (CONCEPTS IN MEASUREMENT)

Mục đích cơ bản của các phép đo trong một dự án là để kiểm soát có hiệu quả dự án. Phần này thảo luận về một số khái niệm có liên quan đến các số đo (metrics), đo lường (measurement) và các số đo cơ bản (basic metrics) mà bạn cần đo lường để kiểm soát một dự án. Một tiếp cận để kiểm soát là Kiểm soát quy trình dùng thống kê (SPC - statistical process control). Phần này cũng thảo luận về một số khái niệm liên quan đến SPC và cách SPC có thể được sử dụng cho phần mềm.

7.1.1 Số đo và đo (Metrics and Measurements)

Số đo của phần mềm có thể được sử dụng để mô tả định lượng các khía cạnh khác nhau của quy trình phần mềm hoặc các sản phẩm phần mềm. Các *số đo của quy trình* (*process metrics*) xác định một cách định lượng các thuộc tính của quy trình phần mềm hoặc môi trường phát triển, trong khi đó *các số đo của sản phẩm* (*product metrics*) là đơn vị đo lường cho các sản phẩm phần mềm (software products) [1][2]. Các số đo của sản phẩm

lưu các số liệu độc lập của quy trình được sử dụng để tạo ra sản phẩm. Ví dụ về các số đo của quy trình bao gồm năng suất, chất lượng, số liệu về nhân công, tỷ lệ tiêm lỗi, và hiệu quả loại bỏ lỗi. Ví dụ về các số đo của sản phẩm bao gồm kích thước, độ tin cậy, chất lượng (chất lượng có thể vừa được xem là số đo của sản phẩm vừa được xem là số đo của quy trình), sự phức tạp của mã, và chức năng.

Để có được các số đo nhất thiết đòi hỏi rằng các phép đo phải được thực hiện để có được dữ liệu. Đối với bất kỳ chương trình đo nào, bạn phải hiểu rõ các mục tiêu của việc thu thập dữ liệu cũng như các mô hình được sử dụng để làm các phán quyết dựa trên các dữ liệu. Nói chung, số đo nào được sử dụng và các phép đo nào cần được thực hiện sẽ tùy thuộc vào dự án và các mục tiêu của công ty, bạn có thể sử dụng một khung làm việc (framework), chẳng hạn như phương pháp đo theo mục đích: Mục Đích-Câu Hỏi-Số Đo (goal-question-metric paradigm), để xác định các số đo cần được đo [3][4]. Tuy nhiên, trong thực tế, một vài số đo là đủ cho hầu hết các tình huống, và các số đo đặc biệt chỉ cần thiết cho các tình huống đặc biệt. Thời gian biểu (schedule), kích thước, nỗ lực (chi phí/effort), và lỗi (defect) là các số đo cơ bản cho các dự án và hình thành một tập hợp số đo bền vững [5][6].

Thời gian biểu (schedule) là một trong những số đo quan trọng nhất bởi vì hầu hết các dự án được điều khiển bởi thời gian biểu và thời hạn hoàn thành. Tuy nhiên, nó được đo dễ dàng nhất bởi vì thời gian lịch (calendar time) thường được sử dụng. Nỗ lực là nguồn nhân lực chủ yếu tiêu tốn trong một dự án phần mềm. Do đó, theo dõi nỗ lực là một hoạt động quan trọng trong suốt quá trình giám sát; nó là điều cần thiết để đánh giá xem liệu các dự án được thực hiện xong trong phạm vi ngân sách. Tức là, dữ liệu này sẽ được cần đến để lập các báo cáo như "Chi phí của dự án là có thể sẽ cao hơn khoảng 30% so với dự kiến trước đây" hoặc "Dự án này là có khả năng hoàn thành trong phạm vi ngân sách."

Bởi vì lỗi có mối quan hệ trực tiếp đến chất lượng phần mềm, hoạt động theo dõi lỗi là rất quan trọng để đảm bảo chất lượng. Một dự án phần mềm lớn có thể bao gồm hàng ngàn lỗi được tìm thấy bởi những người khác nhau ở các giai đoạn khác nhau. Thường thì người sửa chữa lỗi không phải là người đã tìm thấy hoặc báo cáo nó. Nói chung, người quản lý dự án sẽ muốn loại bỏ hầu hết hoặc tất cả các lỗi được tìm thấy trước khi phần mềm được giao. Trong trường hợp như thế, báo cáo lỗi (defect reporting) và đóng lỗi (closure) không thể được thực hiện mà không theo nghi thức (informally) nào đó. Việc sử dụng các cơ chế không theo nghi thức (informal mechanisms) có thể dẫn đến việc các lỗi

được tìm thấy nhưng sau đó bị bỏ quên sau đó, vì vậy sau cùng lỗi không được gỡ bỏ hoặc cần thêm nỗ lực (chi phí) để tìm kiếm các lỗi lại một lần nữa. Do đó, ít nhất, lỗi phải được ghi nhận lại (logged) và sự kết thúc nó (closure) phải được theo dõi. Để làm điều này, bạn cần thông tin, chẳng hạn như sự biểu hiện của lỗi, vị trí của nó, và tên của người tìm thấy nó và người đã đóng/kết thúc nó (closed it). Một khi lỗi được tìm thấy được ghi ghi nhận (và sau đó đóng/kết thúc nó), phân tích có thể tập trung vào có bao nhiêu lỗi đã được tìm thấy cho đến nay, bao nhiêu phần trăm lỗi vẫn còn, và các vấn đề phát sinh khác. Hoạt động dõi lỗi được xem là một trong những thực thành tốt nhất để quản lý dự án [7].

Nếu chỉ đơn thuần là ghi nhận lỗi và theo dõi chúng, sẽ không đủ để hỗ trợ cho các phân tích được mong muốn khác. Để biết được bao nhiêu phần trăm lỗi đã được phát hiện, bạn cũng cần ghi nhận lại thông tin về các giai đoạn mà lỗi được phát hiện ở đó. Để biết được hiệu quả của việc loại bỏ lỗi của của các hoạt động kiểm soát chất lượng khác nhau và do đó cải thiện hiệu suất của chúng, bạn phải biết được không chỉ nơi lỗi đã được phát hiện mà còn phải biết nơi mà nó đã được tiêm vào. Nói cách khác, đối với mỗi ghi nhận lỗi, bạn cũng nên cung cấp thông tin về giai đoạn đã gây ra (tạo ra) lỗi.

Kích thước là một số đo cơ bản khác bởi vì nhiều dữ liệu (ví dụ, mật độ lỗi được giao (delivered defect density)) được chuẩn hóa theo kích thước. Nếu không có dữ liệu về kích thước, bạn không thể dự đoán hiệu suất của việc sử dụng dữ liệu quá khứ. Ngoài ra, nếu không chuẩn hóa (normalization) theo số đo kích thước chuẩn (standard measure of size), bạn không thể xác định được khả năng chuẩn (benchmark performance) để dùng cho mục đích so sánh. Hai số đo kích thước phổ biến là số dòng mã (LOC - lines of code) và Điểm chức năng (function point). Nếu bạn sử dụng số dòng mã như một số đo, năng suất sẽ thay đổi trong các ngôn ngữ lập trình trình khác nhau. Ngược lại, Điểm chức năng cho kết quả không phụ thuộc vào ngôn ngữ lập trình.

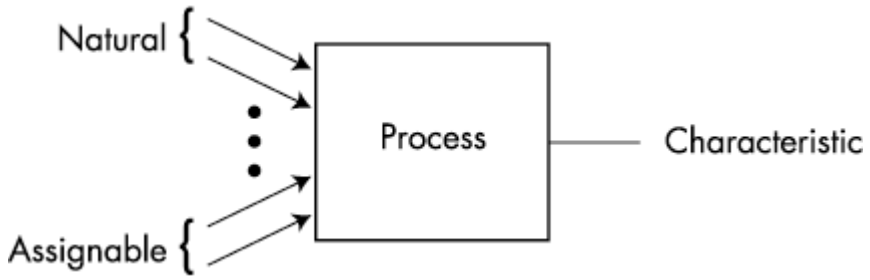
7.1.2 Giám sát quy trình bằng cách dùng Kiểm soát quy trình dùng thống kê (Process Monitoring through Statistical Process Control)

Kiểm soát quy trình dùng thống kê (SPC - statistical process control) đã được sử dụng rất thành công trong sản xuất (manufacturing), và việc sử dụng nó trong phần mềm cũng đang gia tăng [8]. Ở đây chúng tôi thảo luận ngắn gọn về một số khái niệm chung của SPC; để biết thêm thông tin, bạn có thể tham khảo bất kỳ cuốn sách nào về SPC [9][10].

Trong Chương 10 và 11, bạn sẽ thấy làm thế nào các ý tưởng cơ sở (concepts) của SPC được sử dụng để giám sát dự án.

Một *quy trình (process)* được sử dụng để tạo ra đầu ra (output), và chất lượng của đầu ra có thể được xác định bằng các đặc tính chất lượng cụ thể *(quality characteristics*). Một số yếu tố có thể làm biến đổi giá trị của những đặc tính này. Những yếu tố này có thể được phân thành hai loại: các nguyên nhân tự nhiên (hoặc cố hữu) (natural (or inherent) causes) gây ra biến đổi, và các nguyên nhân đặc biệt (special/assignable) không ngẫu nhiên. Các nguyên nhân tự nhiên là những nguyên nhân luôn luôn hiện diện và mỗi nguyên nhân này góp phần gây ra biến đổi. Ta không có cách nào để kiểm soát các nguyên nhân này, trừ khi quy trình tự thay đổi chính nó. Ngược lại, các nguyên nhân đặc biệt là các nguyên nhân ít khi xảy ra (chỉ xảy ra một lần trong một khoảng thời gian nào đó), nhưng lại có ảnh hưởng lớn hơn đến hiệu suất của quy trình, và có thể được kiểm soát. Hình 7.1 minh họa mối quan hệ giữa các nguyên nhân và các đặc tính chất lượng.

Hình 7.1. Nguyên nhân tự nhiên và đặc biệt



Một quy trình được nói là được kiểm soát dùng thống kê (*statistical control*) nếu những biến đổi trong các đặc tính chất lượng là chỉ do các nguyên nhân tự nhiên. Mục tiêu của Kiểm soát quy trình dùng thống kê (SPC) là để giữ cho quy trình sản xuất được kiểm soát dùng thống kê.

Các biểu đồ kiểm soát (control charts) là một công cụ được yêu thích để áp dụng cho SPC. Để xây dựng một biểu đồ kiểm soát, đầu ra của một quy trình được xem là một dòng các giá trị số trình bày các giá trị của các đặc tính (characteristics) được quan tâm. Các nhóm dữ liệu nhỏ (subgroups of data) được lấy từ dòng dữ liệu này, và giá trị trung bình của các nhóm dữ liệu nhỏ sẽ được vẽ, bằng một biểu đồ X-bar. *Một giới hạn kiểm soát dưới (LCL - lower control limit)* và *một giới hạn kiểm soát trên (UCL - upper control limit)* được thành lập. Nếu một điểm nằm ngoài các giới hạn kiểm soát, biến đổi lớn được xem là do các

nguyên nhân đặc biệt. Một biểu đồ khác, được gọi là biểu đồ R-chart, vẽ phạm vi (sự khác biệt giữa các giá trị tối thiểu và tối đa) của các phân nhóm được lựa chọn. Các giới hạn kiểm soát được thiết lập cho biểu đồ R-chart, và một điểm nằm ngoài các giới hạn kiểm soát này cũng được coi là do nguyên nhân đặc biệt.

Theo quy ước, LCL và UCL thường được đặt ở khoảng 3-sigma xung quanh giá trị trung bình, với sigma là độ lệch chuẩn (standard deviation) của dữ liệu với những biến đổi bình thường (normal variability) (có nghĩa là, biến đổi do nguyên nhân tự nhiên). Với những giới hạn này, xác suất của một *báo động sai (false alarm)* – tại đây một điểm có biến đổi tự nhiên rơi bên ngoài các giới hạn - chỉ là 0.27%.

Khi quy trình sản xuất không sinh ra cùng một mục một cách lặp đi lặp lại, như là trường hợp của các quy trình phần mềm, việc phân chia các nhóm nhỏ có thể là vô nghĩa; các giá trị đơn lẻ do dó sẽ được xem xét. Đối với quy trình như vậy, biểu đồ XMR [9][10] có thể được sử dụng. Trong một biểu đồ XMR-chart, một phạm vi di chuyển (moving range) của hai giá trị liên tiếp được dùng làm phạm vi cho các biểu đồ R-chart. Đối với biểu đồ X-bar, các giá trị đơn lẻ sẽ được vẽ; sau đó các giới hạn kiểm soát được xác định bằng cách sử dụng phạm vi di chuyển trung bình.

Lưu ý rằng các giới hạn kiểm soát khác với các giới hạn đặc tả (specification limits). *Các giới hạn đặc tả* xác định, dựa trên yêu cầu, hiệu suất được mong muốn từ quy trình. Ngược lại, các giới hạn kiểm soát dựa trên dữ liệu thực tế từ quy trình, xác định khả năng hiệu suất thực tế của quy trình – tức là, những gì quy trình thực sự có khả năng cung cấp. Rõ ràng, nếu các giới hạn kiểm soát nằm trong các giới hạn đặc tả (các giới hạn đặc tả thì rộng hơn các giới hạn kiểm soát), quy trình có khả năng tạo ra đầu ra đáp ứng được các đặc tả trong phần lớn thời gian. Ngược lại, nếu giới hạn đặc tả nằm trong phạm vi các giới hạn kiểm soát, xác suất để quy trình tạo ra một kết quả không thỏa mãn các yêu cầu sẽ gia tăng. Dựa trên mối quan hệ giữa giới hạn đặc tả và giới hạn kiểm soát, khả năng của một quy trình có thể được định nghĩa một cách hình thức [9][10].

Bạn sử dụng các biểu đồ kiểm soát (control charts) để liên tục giám sát hiệu suất của các quy trình và xác định ra một trạng thái ngoài tầm kiểm soát. Một cách riêng lẻ, bạn quyết định những hoạt động nào sẽ được thực hiện khi một điểm trình bày cho một đầu ra nằm ngoài giới hạn kiểm soát. Nói chung, có hai loại hành động được thực hiện:

- Làm lại đầu ra để mà nó có các đặc tính chấp nhận được – tức là, thực hiện hoạt động sửa chữa.

- Tiến hành phân tích sâu hơn để xác định ra các nguyên nhân đặc biệt và loại bỏ chúng khỏi quy trình – tức là, thực hiện các hoạt động phòng ngừa.

Để sử dụng biểu đồ kiểm soát cho các quy trình phần mềm, trước tiên bạn phải xác định các quy trình mà phương pháp Kiểm soát quy trình dùng thống kê (SPC) có thể được áp dụng. Một lựa chọn là quy trình tổng thể, mà đầu ra của quy trình này là sản phẩm phần mềm sẽ được giao. Các đặc tính có thể được nghiên cứu cho đầu ra của quy trình này bao gồm năng suất, mật độ lỗi được giao, và tỷ lệ tiêm lỗi. Bạn có thể có được các giá trị của hầu hết những đặc tính này cho đầu ra của toàn bộ quy trình chỉ sau khi dự án kết thúc, vì vậy Kiểm soát quy trình dùng thống kê (SPC) toàn bộ quy trình có giá trị giới hạn để theo dõi và kiểm soát dự án. Giá trị của nó nằm chủ yếu trong sự hiểu biết và cải tiến khả năng của quy trình.

Để kiểm soát một dự án, bạn có thể triển khai SPC cho "các quy trình nhỏ" (mini-processes) được thực hiện trong suốt tiến trình của dự án, chẳng hạn như quy trình xem xét lại hoặc quy trình kiểm thử. Theo SPC, ngay sau khi quy trình này được thực hiện, kết quả của nó có thể được phân tích. Nếu cần thiết, sau đó bạn có thể áp dụng kiểm soát trong các hình thức của các hoạt động sửa chữa hoặc phòng ngừa lỗi. Thông qua các hoạt động sửa chữa, kết quả đầu ra ngoài tầm kiểm soát được chấp nhận được; các hoạt động phòng ngừa giúp cải thiện sự thực hiện của phần còn lại của dự án. Chương 10 và 11 thảo luận về việc sử dụng các Công cụ kiểm soát quy trình phần mềm SPC để giám sát các dự án.

Được cho biết trước khả năng có sự thay đổi lớn về hiệu suất trong các quy trình phần mềm, không dễ gì để xác định ra được các điểm chỉ có biến đổi tự nhiên để mà bạn có thể xác định các giới hạn kiểm soát. Do đó, để tính toán các giới hạn kiểm soát từ dữ liệu hiệu suất trong quá khứ, bạn phải sử dụng phán xét của mình để xác định các điểm dữ liệu cần được loại ra. Hơn nữa, dữ liệu quá khứ không nên được sử dụng một cách mù quáng, mà cần phải thực hiện quản lý một cách sáng suốt để hỗ trợ cho việc sử dụng nó. Ví dụ, bạn không thể giả thiết rằng một quy trình đã thất bại chỉ bởi vì hiệu suất nằm ngoài phạm vi được tính tính từ dữ liệu quá khứ [2][11]. Một tiếp cận phù hợp hơn là sử dụng một khoảng (phạm vi) hiệu suất để xác định ra được sự chênh (deviation) và sau đó phân tích các nguyên nhân của sự chênh lệch này.

7.2 CÁC PHÉP ĐO (MEASUREMENTS)

Bất kỳ hoạt động kiểm soát dự asb dùng định lượng nào đều phụ thuộc rất nhiều vào các phép đo được thực hiện trong quá trình thực hiện dự án. Để thực hiện các phép đo trong quá trình thực hiện dự án, bạn phải lập kế hoạch cẩn thận về những gì để đo, khi nào đo, và làm thế nào để đo lường. Do đó, lập kế hoạch đo lường là một yếu tố chủ chốt trong việc lập kế hoạch cho dự án. Phần này bàn về phép đo lường chuẩn được thực hiện trong dự án tại Infosys. Những người quản lý dự án có thể thêm vào các phép đo này nếu dự án của họ cần đến chúng.

7.2.1 Thu thập dữ liệu về nỗ lực (Collecting Effort Data)

Để giúp một người quản lý dự án giám sát nỗ lực (effort/nhân công/chi phí), mỗi nhân viên ghi nhận vào trong *Hệ Thống Báo Cáo Hoạt Động Hàng Tuần (WAR - weekly activity report* system*)* công sức đã bỏ ra cho các công việc/nhiệm vụ khác nhau. Hệ thống trực tuyến này, do tự công ty phát triển, lưu trữ tất cả các báo cáo hoạt động hàng tuần (WARs) vào một cơ sở dữ liệu tập trung (centralized database). Mỗi người gửi nộp (submit) WAR của mình hàng tuần. Báo cáo sẽ được gửi đến người giám sát để được phê duyệt (approval). Một khi nó được chập nhận, bản nộp WAR đó đã là bản cuối cùng và không thể bị thay đổi thêm nữa. Mỗi người đều phải nộp một WAR, bao gồm cả giám đốc điều hành (CEO), và nếu một WAR không được nộp trong vòng một khoảng thời gian nhất định, nó sẽ bị bỏ qua.

Một mục nhập WAR (WAR entry) chứa một danh sách các mẫu tin (records), mỗi mẫu tin chứa thông tin cho một tuần. Mỗi mẫu tin là một danh sách các mục (items), mỗi mục chứa các trường (field/lĩnh vực) sau đây:

- Mã chương trình (Program code)
- Mã mô-đun (Module code)
- Mã hoạt động (Activity code)
- Mô tả hoạt động (Activity description)
- Giờ từ Thứ Hai đến Chủ Nhật (Hours for Monday through Sunday)

Mã hoạt động cho biết kiểu hoạt động. *Mã chương trình* và *mã mô-đun* cho phép tách dữ liệu về nỗ lực ra theo mô-đun hoặc chương trình - một cách thức quan trọng để cho phép giám sát ở mức thành phần (component-level monitoring). Để hỗ trợ phân tích và so sánh, công việc quan trọng cần làm là đi chuẩn hóa các hoạt động theo nỗ lực được báo cáo. Có một tập hợp các mã hoạt động đã được chuẩn hóa của để giúp bạn đạt được mục tiêu này. Bảng 7.1 cho thấy các mã hoạt động được sử dụng trong các dự án Infosys.

Trong các mã hoạt động, một mã riêng biệt dùng cho nỗ lực làm lại công việc ở nhiều giai đoạn. Phân loại này giúp ích trong việc tính toán chi phí của chất lượng (cost of quality). Với mức độ sàng lọc, bạn có thể thực hiện phân tích dữ liệu nỗ lực theo giai đoạn (phase-wise analysis) hay theo giai đoạn nhỏ (subphase-wise analysis). Mã chương trình và mã mô-đun, được quy định cụ thể bởi dự án, có thể được sử dụng để ghi lại dữ liệu nỗ lực cho các đơn vị chương trình (units) khác nhau trong dự án, do đó tạo điều kiện thuận lợi cho phân tích theo đơn vị (unit-wise analysis).

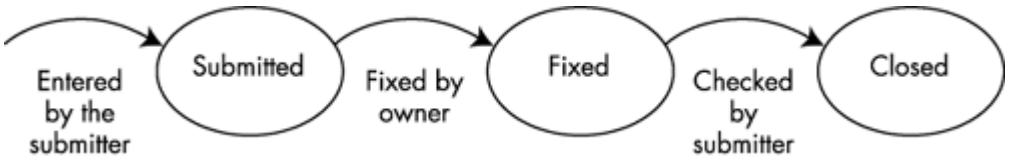
Để tạo dễ dàng cho hoạt động phân tích so sánh nỗ lực dự kiến với nỗ lực thực tế đã tiêu dùng ở mức dự án (project-level analysis), hệ thống WAR được kết nối với mô tả dự án dùng Microsoft Project (MSP). Nhân viên dự án có thể bắt đầu gửi nộp WARs cho dự án chỉ sau khi MSP cho dự án vừa đã được đệ trình lên (một khi MSP được gửi lên, hệ thống biết được những người nào được đề nghị để làm việc trong dự án). Các hoạt động đã được lên kế hoạch (planned activities) được định nghĩa bằng cách liệt kê chúng trong MSP và phân công nó cho một người có thẩm quyền trong dự án. Các hoạt động không có kế hoạch (unplanned activities) là tất cả các hoạt động khác của dự án.

Khi nhập vào WAR cho một tuần, người sử dụng dùng một màn hình được chia thành hai phần: phần cho các hoạt động đã được lập kế hoạch (planned activities) và phần cho các hoạt động không có kế hoạch (unplanned activities). Tất cả các hoạt động (được phân công trong MSP cho một người cụ thể cho tuần này sẽ được hiển thị trong phần: các hoạt động đã được lập hoạch của cô ta. Người dùng không thể thêm vào hoặc sửa đổi các hoạt động được hiển thị trong phần này. Cô ấy chỉ có thể nhập vào số giờ đã xài mỗi ngày cho các hoạt động khác nhau đã được cung cấp. Để ghi nhận thời gian đã xài cho các hoạt động không được liệt kê trong phần kế hoạch, người sử dụng có thể nhập mã, mô tả nó, và số giờ đã xài mỗi ngày cho các hoạt động này trong phần: các hoạt động không có kế hoạch của dự án.

7.2.2 Ghi nhận và theo dõi lỗi (Logging and Tracking Defects)

Trong một dự án của Infosys, việc phát hiện và loại bỏ lỗi được tiến hành như sau. Một lỗi được tìm thấy và được ghi lại bởi một người gửi (submitter). Sau đó, lỗi này sẽ có trạng thái (tình trạng) là “đã được gửi” (submitted). Kế tiếp, người quản lý dự án phân công công việc sửa chữa lỗi đó cho một ai đó, thường là tác giả của các tài liệu (document) hoặc mã (code) mà tại đó lỗi đã được tìm thấy. Người này thực hiện gỡ lỗi (debugging) và sửa chữa (fix) các lỗi đó, và sau đó lỗi này chuyển sang trạng thái “đã được sửa” (fixed). Một lỗi sau khi được sửa xong vẫn chưa được đóng lại (closed). Một người khác, thường là người gửi, xác nhận rằng lỗi đã được sửa chữa xong. Sau xác minh này, lỗi có thể được đánh dấu "đóng” (closed). Nói cách khác, vòng đời chung của một lỗi có ba trạng thái: đá được gửi, đã được sửa, và đóng (xem hình 7.2). Một lỗi chưa được đóng lại còn được gọi là mở (open).

Hình 7.2. Vòng đời của một lỗi



Bảng 7.1. Các mã hoạt động của nỗ lực (effort)

Mã hoạt động (Activity Code)	Mô tả (Description)
PAC	Chấp nhận (Acceptance)
PACRW	Làm lại công việc sau khi kiểm thử chấp nhận (Rework after acceptance testing)
PCAL	Tổng quan về dự án (Project catch-all)
PCD	Cài đặt mã và tự kiểm thử đơn vị (Coding and self unit testing)
PCDRV	Xem xét lại mã (Code walkthrough/review)
PCDRW	Làm lại công việc sau khi xem xét lại mã (Rework after code walkthrough)
PCM	Quản lý cấu hình (Configuration management)
PCOMM	Giao tiếp (Communication)
PCSPT	Các hoạt động hỗ trợ khách hàng (Customer support activities)
PDBA	Các hoạt động quản trị cơ sở dữ liệu (Database administration activities)
PDD	Thiết kế chi tiết (Detailed design)
PDDRV	Xem xét lại thiết kế chi tiết (Detailed design review)

PDDR	Làm lại công việc sau khi thiết kế chi tiết (Rework after detailed design review)
PDOC	Lập tài liệu (Documentation)
PERV	Xem xét lại các mô hình và các hình vẽ (Review of models and drawings)
PERW	Làm lại công việc cho các mô hình/hình vẽ (Rework of models and drawings)
PEXEC	Thực hiện xây dựng mô hình và thiết kế (Execution of modeling and drafting)
PHD	Thiết kế mức cao (High-level design)
PHDRV	Xem xét lại thiết kế mức cao (High-level design reviews)
PHDRW	Làm lại công việc sau khi xem xét lại thiết kế mức cao (Rework after high-level design review)
PIA	Phân tích tác động (Impact analysis)
PINS	Cài đặt và đào tạo cho khách hàng (Installation/customer training)
PIT	Kiểm thử tích hợp (Integration testing)
PITRW	Làm lại công việc sau khi kiểm thử tích hợp (Rework after integration testing)
PPI	Khởi tạo dự án (Project initiation)
PPMCL	Các hoạt động kết thúc dự án (Project closure activities)
PPMPT	Lập kế hoạch và theo dõi dự án (Project planning and tracking)
PRES	Nghiên cứu các vấn đề kỹ thuật (Research on technical problems)
PRS	Các hoạt động đặc tả yêu cầu (Requirement specification activities)
PRSRV	Xem xét lại đặc tả yêu cầu (Review of requirements specifications)
PRSRW	Làm lại công việc sau khi xem xét lại yêu cầu (Rework after requirements review)
PSP	Các hoạt động hoạch định chiến lược (Strategic planning activities)
PST	Kiểm thử hệ thống (System testing)
PSTRW	Làm lại công việc sau khi kiểm thử hệ thống (Rework after system testing)
PTRE	Các hoạt động huấn luyện nhân sự mới cho dự án (Project-specific trainee activities)
PUT	Kiểm thử đơn vị riêng lẻ (Independent unit testing)
PUTRW	Làm lại công việc sau khi kiểm thử đơn vị riêng lẻ (Rework after independent unit testing)
PWTR	Đợi chờ nhân sự (Waiting for resources)
PWY	Nỗ lực (chi phí) xài cho suốt giai đoạn bảo hành (Effort during warranty)

Một hệ thống kiểm soát lỗi (DCS- defect control system) được sử dụng trong các dự án để ghi nhận và theo dõi lỗi. Hệ thống này cho phép các kiểu phân tích khác nhau. Bảng 7.2 cho thấy các thông tin được ghi nhận cho mỗi lỗi của hệ thống.

Để xác định giai đoạn tiêm lỗi (defect injection stage) cần phải phân tích lỗi. Trong khi đó, các giai đoạn phát hiện lỗi bao gồm các hoạt động xem xét lại (review) và kiểm thử

(testing), các giai đoạn tiêm lỗi bao gồm các giai đoạn đã tạo ra sản phẩm công việc (work products), chẳng hạn như giai đoạn thiết kế (design) và giai đoạn cài đặt mã (coding). Dựa trên bản chất của lỗi vừa được tìm thấy, một số phán xét có thể được thực hiện. Không giống như các giai đoạn phát hiện lỗi - được biết đến một cách chắc chắn - giai đoạn tiêm lỗi thì mơ hồ hơn; nó được đánh giá từ bản chất của lỗi và các thông tin khác có liên quan. Sử dụng các thông tin về gian đoạn tiêm lỗi và giai đoạn phát hiện lỗi, bạn có thể tính toán hiệu suất loại bỏ lỗi, tỷ lệ phần trăm phân phối, và các số đo khác.

Đôi khi ta mong muốn hiểu được bản chất của các lỗi mà không cần tham chiếu đến các giai đoạn, thay vào đó ta tham chiếu đến các loại lỗi. Một sự phân loại như vậy có thể giúp bạn hiểu sự phân bố của các lỗi qua theo các loại lỗi khác nhau. Vì lý do này, các loại lỗi cũng được ghi nhận lại. Bảng 7.3 cho thấy các loại lỗi có thể có, cùng với một số ví dụ. Một dự án cũng có thể định nghĩa cách phân loại của riêng nó.

Bảng 7.2. Ghi nhận dữ liệu về lỗi

Dữ liệu (Data)	**Mô tả (Description)**	**Bắt buộc(M)/Tự chọn(O) (Mandatory/Optional)**
Mã dự án *(Project code)*	Mã dự án - nơi lỗi được ghi nhận *(Code of the project for which defects are captured)*	M
Mô tả *(Description)*	Mô tả lỗi *(Description of the defect)*	M
Mã mô-đun *(Module code)*	Mã mô-đun *(Module code)*	O
Tên chương trình *(Program name)*	Tên của chương trình mà tại đó lỗi được tìm thấy *(Name of program in which the defect was found)*	O
Giai đoạn phát hiện *(Stage detected)*	Giai đoạn mà tại đó lỗi được phát hiện *(Stage in which the defect was detected)*	M
Giai đoạn tiêm vào *(Stage injected)*	Giai đoạn mà tại đó lỗi được tiêm vào *(Stage at which the defect was injected/origin of defect)*	M

Loại *(Type)*	Phân loại lỗi *(Classification of the defect)*	M
Mức độ nghiêm trọng *(Severity)*	Mức độ nghiêm trọng của lỗi *(Severity of the defect)*	M
Kiểu xem xét lại *(Review type)*	Kiểu xem xét lại *(Type of review)*	O
Tình trạng (trạng thái) *(Status)*	Tình trạng hiện tại của lỗi *(Current status of the defect)*	M
Người gửi *(Submitter)*	Tên của người đã phát hiện ra lỗi *(Name of the person who detected the defect)*	M
Chủ nhân của lỗi *(Owner)*	Tên người chủ của lỗi *(Name of the person who owns the defect)*	M
Ngày gửi *(Submit date)*	Ngày lỗi được gửi đến chủ nhân *(Date on which the defect was submitted to the owner)*	M
Ngày đóng *(Close date)*	Ngày mà lỗi (đã được gửi) được đóng lại *(Date on which the submitted defect was closed)*	M

Cuối cùng, mức độ nghiêm trọng của lỗi được ghi nhận lại. Thông tin này là quan trọng cho người quản lý dự án. Ví dụ, nếu một lỗi là nghiêm trọng, bạn có thể sẽ xếp lịch để nó sớm được sửa chữa. Ngoài ra, bạn có thể quyết định rằng các lỗi nhỏ hoặc không quan trọng không cần phải được sửa chữa khi phải giao hàng khẩn cấp. Bảng 7.4 cho thấy cách phân loại được sử dụng tại Infosys.

Từ thông tin này, có thể có nhiều phân tích khác nhau được thực hiện. Ví dụ, bạn có thể phân chia các lỗi ra theo loại, mức độ nghiêm trọng, hoặc mô-đun; vẽ biểu đồ về khuynh hướng mở và đóng lỗi theo mô-đun, mức độ nghiêm trọng, hoặc lỗi tổng thể; xác định tỷ lệ lỗi tiêm vào hàng tuần; xác định hiệu quả loại bỏ lỗi; xác định tỷ lệ tiêm lỗi trong các giai đoạn khác nhau, v.v. Trong chương 11, bạn sẽ thấy sự sử dụng các dữ liệu này để theo dõi chất lượng và để phòng ngừa lỗi. Chương này cũng mô tả một ví dụ về dữ liệu về lỗi được nhập vào trong case study.

Bảng 7.3. Các loại lỗi

Loại lỗi (Defect Type)	**Ví dụ (Example)**
Lôgic *(Logic)*	Các lỗi không đầy đủ/không đúng trong giải thuật đã được sử dụng; các điều kiện, ca kiểm thử, hoặc tài liệu thiết kế sai *(Insufficient/incorrect errors in algorithms used; wrong conditions, test cases, or design documents)*
Các chuẩn *(Standards)*	Các vấn đề với các chuẩn cài đặt mã/các chuẩn tài liệu như định danh, định dạng theo hàng, cách trình bày, tạo mô-đun, chú thích, cài đặt mã cứng, lỗi chính tả *(Problems with coding/documentation standards such as indentation, alignment, layout, modularity, comments, hard-coding, and misspelling)*
Mã dư thừa *(Redundant code)*	Cùng một phần mã được dùng trong nhiều chương trình hoặc trong cùng một chương trình *(Same piece of code used in many programs or in the same program)*
Giao diện người dùng *(User interface)*	Các chức năng chính yếu không hoạt động; menu không thích hợp *(Specified function keys not working; improper menu navigation)*
Hiệu suất *(Performance)*	Tốc độ xử lý chậm; hệ thống bị hỏng bởi vì kích thước tập tin; các vấn đề về bộ nhớ *(Poor processing speed; system crash because of file size; memory problems)*
Khả năng tái sử dụng *(Reusability)*	Không có khả năng tái sử dụng lại mã *(Inability to reuse the code)*
Vấn đề về thiết kế *(Design issue)*	Các vấn đề liên quan đến thiết kế cụ thể *(Specific design-related matters)*
Lỗi về quản lý bộ nhớ *(Memory management defects)*	Các lỗi như hỏng lõi, tràn mảng, gọi hàm bất hợp lệ, treo hệ thống, hoặc tràn bộ nhớ *(Defects such as core dump, array overflow, illegal function call, system hangs, or memory overflow)*
Các lỗi về tài liệu *(Document defects)*	Các lỗi được tìm thấy trong khi xem xét lại các tài liệu như kế hoạch dự án, kế hoạch quản lý cấu hình, hoặc đặc tả *(Defects found while reviewing documents such as the project plan, configuration management plan, or specifications)*

Nhất quán *(Consistency)*	Lỗi khi cập nhật hoặc xóa các mẫu tin có cùng thứ tự ở khắp hệ thống *(Failure to updating or delete records in the same order throughout the system)*
Khả năng có thể theo dõi qua dấu vết (Traceability)	Thiếu khả năng theo dõi chương trình nguồn so với đặc tả *(Lack of traceability of program source to specifications)*
Khả năng dễ di chuyển *(Portability)*	Mã không độc lập với nền *(Code not independent of the platform)*

Bảng 7.4. Mức độ nghiêm trọng của lỗi

Loại nghiêm trọng (Severity Type)	**Giải thích về phân loại (Explanation for Categorization)**
Nghiêm trọng (Critical)	Lỗi này có thể rất nguy kịch vì ảnh hưởng đến thời gian biểu của dự án, hoặc nó có thể làm người dùng dừng ngay việc sử dụng hệ thống *(Defect may be very critical in terms of affecting the schedule, or it may be a showstopper—that is, it stops the user from using the system further.)*
Lớn (Major)	Cùng một loại lỗi xảy ra trong nhiều chương trình hoặc mô-đun.Ta cần phải sửa chữa lại tất cả. Ví dụ, các chuẩn cài đặt mã không được áp dụng trong bất kỳ chương trình nào. Lỗi làm người dùng dừng tiếp tục sử dụng hệ thống theo cách bình thường, nhưng có thể làm việc xung quanh nó. *(The same type of defect has occurred in many programs or modules. We need to correct everything. For example, coding standards are not followed in any program. Alternatively, the defect stops the user from proceeding in the normal way but a workaround exists.)*
Nhỏ (Minor)	Lỗi này thì biệt lập hoặc không làm người dùng dừng tiếp tục sử dụng hệ thống, nhưng nó gây ra bất tiện *(This defect is isolated or does not stop the user from proceeding, but it causes inconvenience.)*
Nhẹ bên ngoài (Cosmetic)	Lỗi không ảnh hưởng đến hiệu suất của sản phẩm phần mềm – ví dụ, các vấn đề về thẩm mỹ và lỗi ngữ pháp trong các thông báo. *(A defect that in no way affects the performance of the software product—for example, esthetic issues and grammatical errors in messages.)*

7.2.3 Đo thời gian biểu (Measuring Schedule)

Đo thời gian biểu (lịch) thì đơn giản vì bạn sử dụng thời gian lịch (calendar time). Các hoạt động và tiến độ chi tiết thường được lưu trong thời gian biểu của Microsoft Project (MSP), do đó, các ngày và khoảng thời gian của các công việc/nhiệm vụ (duration of tasks) ước lượng được đưa ra trong MSP. Biết các ngày thực tế, bạn có thể dễ dàng xác định được lượng thời gian thực tế (actual duration) đã dùng để thực hiện từng công việc/ nhiệm vụ (task).

7.2.4 Đo kích thước (Measuring Size)

Nếu sử dụng kỹ thuật ước lượng từ dưới lên (bottom-up estimation technique), kích thước được ước lượng bằng cách dùng số lượng chương trình ở các độ phức tạp khác nhau. Mặc dù số liệu này hữu ích cho việc lập dự toán (ước lượng), nhưng nó không thể xây dựng một định nghĩa chuẩn về năng suất (productivity) để có thể được dùng để so sánh năng suất giữa các dự án khác nhau. Vấn đề tương tự như vậy cũng xuất hiện, khi số dòng mã lệnh (LOC) được sử dụng để đo kích thước; năng suất sẽ thay đổi theo ngôn ngữ lập trình. Để bình thường hóa và sử dụng một số đo kích thước thống nhất cho mục đích tạo ra một baseline để so sánh hiệu suất, Điểm chức năng (function points) được sử dụng để đo kích thước.

Kích thước của phần mềm thường được đo bằng LOC thông qua việc thường xuyên sử dụng chương trình đếm số dòng lệnh. Việc đếm này được thực hiện khi dự án đã hoàn thành và sẵn sàng để giao cho khách hàng. Từ kích thước được đo bằng LOC, kích thước theo Điểm chức năng có thể được tính ra bằng cách sử dụng bảng chuyển đổi đã được công bố [12].

7.3 THEO DÕI DỰ ÁN (PROJECT TRACKING)

Mục tiêu chính của việc theo dõi đối với những người quản lý dự án là để có được khả năng nhìn thấy sự thực hiện của dự án để mà họ có thể xác định xem hành động nào cần phải được thực hiện để đảm bảo rằng các mục tiêu của dự án được đáp ứng. Bởi vì đáp ứng các mục tiêu đã được thiết lập là động lực cơ bản, cho nên tất cả các khía cạnh của việc thực hiện dự án – mà chúng có thể ảnh hưởng đến việc đạt được các mục tiêu - phải được giám sát, và việc giám sát này phải được lập kế hoạch. Ở Infosys, những người quản lý dự án thường lập kế hoạch cho các theo dõi sau đây:

- Theo dõi các hoạt động
- Theo dõi lỗi
- Theo dõi các vấn đề phát sinh

Theo dõi các hoạt động (activities tracking) sẽ nhìn vào các hoạt động đã được lập kế hoạch đầy đủ. Nếu mức độ chi tiết của các hoạt động (activity) là nhỏ, khi đó ở mức độ thấp nhất, bạn xem nó chỉ có một trong hai trạng thái: chưa được thực hiện (not done) hoặc đã được thực hiện xong (fully done). Đối với các công việc (tasks) ở mức cao hơn, bạn có thể tính toán tỷ lệ phần trăm hoàn thành bằng cách dựa vào tình trạng (trạng thái) của các công việc/nhiệm vụ ở mức thấp nhất và các ước lượng của chúng.

Theo dõi lỗi (defect tracking) được thực hiện kết hợp với việc ghi nhận lỗi, như đã thảo luận trước đây.

Theo dõi các vấn đề phát sinh (issues tracking) đảm bảo sẽ làm sáng tỏ các vấn đề tiềm ẩn có thể làm trì hoãn dự án, giúp dự án nằm trong tầm kiểm soát. Chương 11 sẽ giải thích làm thế nào các hoạt động theo dõi được thực hiện ở Infosys. Trong suốt quá trình lập kế hoạch, người quản lý dự án sẽ xác định kiểu theo dõi và các công cụ hoặc phương pháp nào họ muốn sử dụng.

Ngoài ra, để luôn theo dõi tình trạng (trạng thái) của dự án thông qua nỗ lực (chi phí), tiến độ (thời gian biểu), và chất lượng, những người quản lý dự án cũng lập kế hoạch cho những hoạt động sau đây:

- Giám sát ở mức hoạt động
- Báo cáo tình trạng
- Các báo cáo tại các cột mốc (milestone reports)

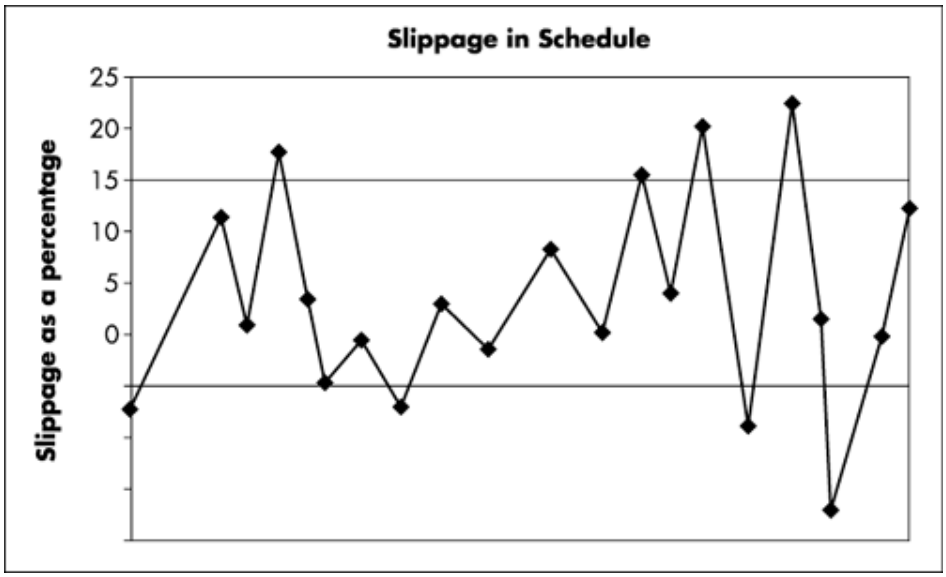
Giám sát ở mức hoạt động (activity-level monitoring) đảm bảo rằng mỗi hoạt động đã được thực hiện đúng. Bạn giám sát các hoạt động một cách định lượng thông qua việc sử dụng phương pháp Kiểm soát quy trình dùng thống kê (SPC - statistical process control). Dựa trên hiệu suất trong quá khứ, bạn thiết lập các giới hạn cho các thông số hiệu suất chủ chốt của các hoạt động/nhiệm vụ cụ thể. Sau đó, bạn so sánh hiệu suất thực tế của mỗi hoạt động/nhiệm vụ với các giới hạn đã được thiết lập trước đó. Nếu hiệu suất không

nằm trong phạm vi giới hạn chấp nhận được, bạn cần phải thực hiện các hành động thích hợp. Tại Infosys, xem xét lại (review) và kiểm thử đơn vị (unit testing), được thảo luận trong các Chương 10 và 11, là hai hoạt động chính sử dụng phương pháp này.

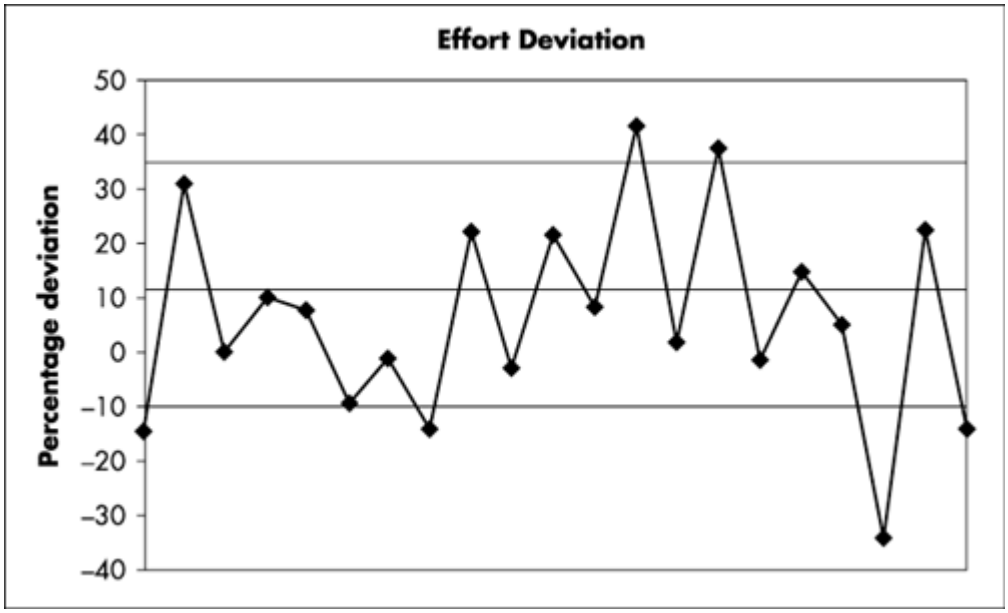
Các báo cáo tình trạng (status reports) thường được chuẩn bị hàng tuần để giúp bạn ghi nhận lại những gì đã xảy ra và những gì cần phải được thực hiện. Tại các cột mốc của dự án, bạn tiến hành kiểm tra một cách định lượng bằng cách so sánh các dữ liệu thực tế về nỗ lực (chi phí), tiến độ (thời gian biểu), và lỗi với các dữ liệu ước lượng đã được làm cho chúng. Ngoài ra, bạn cũng cần giám sát rủi ro, đào tạo, xem xét lại, khiếu nại của khách hàng, v.v. Việc phân tích tại các cột mốc đóng vai trò quan trọng trong việc kiểm soát dự án. Để đảm bảo rằng việc phân tích tại các cột mốc được thực hiện thường xuyên (để cho phép có những can thiệp kịp thời), nếu các cột mốc được cần bởi khách hàng quá xa nhau, bạn hãy lập kế hoạch để bổ sung thêm các cột mốc nội bộ (internal milestones) để mà việc phân tích tại các cột mốc được làm định kỳ mỗi 3-5 tuần.

Tại các cột mốc (milestones), bạn phân tích nỗ lực, tiến độ (thời gian biểu), và lỗi thực tế và so sánh chúng với các giá trị ước lượng. Nếu có sự chênh lệch lớn, một số hành động khắc phục phải được thực hiện. Để phân biệt được sự khác nhau giữa trường hợp bình thường và trường hợp có vấn đề, bạn cần phải xác định giới hạn chênh lệch chấp nhận được; để thiết lập những giới hạn này, bạn hãy sử dụng các ý tưởng của biểu đồ kiểm soát (control charts). Tại Infosys, các giới hạn kiểm soát (control limits) đã được xác định cho sự chênh lệch (giữa thực tế và ước lượng) của nỗ lực và tiến độ. Những giá trị này ban đầu đã được xác định bằng cách dựa vào óc phán đoán (judgment) và kinh nghiệm (experience), và bây giờ dựa vào dữ liệu quá khứ và được tính toán theo cách tương tự như các giới hạn kiểm soát khác. Các giới hạn trước đây là 35% cho chênh lệch nỗ lực và 15% cho chênh lệch tiến độ (thời gian biểu); với những cải tiến trong quy trình, những giới hạn này đã được giảm xuống còn 20% và 10%. Các biểu đồ kiểm soát ở hình 7.3 và 7.4 cho thấy các chênh lệch trong tiến độ (thời gian biểu) và nỗ lực (chi phí).

Hình 7.3. Biểu đồ kiểm soát về chênh lệch tiến độ



Hình 7.4. Biểu đồ kiểm soát về chênh lệch nỗ lực



Nếu chênh lệch (deviation) tại các cột mốc vượt quá những giới hạn này, nó hàm ý rằng dự án có thể đang rơi vào rắc rối và có thể không đáp ứng được các mục tiêu của nó; dưới áp lực thời gian, nhóm dự án (project team) có thể bắt đầu dùng các đường tắt (các biện pháp nhanh chóng trực tiếp hơn) không được mong muốn. Tình trạng này cần những người quản lý dự án phải hiểu được lý do của sai lệch và áp dụng các hành động khắc phục và phòng ngừa nếu cần thiết.

Các giới hạn này đã dựa trên dữ liệu và kinh nghiệm quá khứ. Bạn phải thiết lập các giới hạn riêng cho dự án của bạn - có thể là cao hơn so với các giới hạn của công ty - trong suốt quá trình lập kế hoạch.

7.4 ĐO LƯỜNG VÀ KẾ HOẠCH THEO DÕI DỰ ÁN ACIC

Trong dự án ACIC các số đo chuẩn về kích thước, nỗ lực, lỗi, và tiến độ đã được đo. Kế hoạch là sử dụng một chương trình để đếm kích thước theo số dòng mã lệnh (lines of code), và *Hệ Thống Báo Cáo Hoạt Động Hàng Tuần (WAR - weekly activity report* system*)* để thu thập dữ liệu về nỗ lực (effort), một hệ thống được gọi là BugsBunny để kiểm soát lỗi (defects), và Microsoft Project (MSP) để đo lịch (schedule).

Người quản lý dự án đã lên kế hoạch để sử dụng MSP và tổ chức các cuộc họp thường xuyên để theo dõi tình trạng của các hoạt động khác nhau. Các vấn đề phát sinh đã được chia ra làm các loại: tại chỗ (onsite), khách hàng (customer), quản lý kinh doanh (business manager), và các dịch vụ hỗ trợ (support services) và được theo dõi một cách riêng lẻ. Ý kiến phản hồi của khách hàng (customer feedback) - khiếu nại cũng như khen ngợi – cũng đã được ghi nhận.

Báo cáo tình trạng (trạng thái) được gửi hàng tuần cho người quản lý kinh doanh cũng như khách hàng. Các giới hạn chênh lệch (deviation limits) đã được thiết lập cho năm cột mốc đầu tiên (first five smilestone) là 10% cho nỗ lực và tiến độ và 20% cho lỗi. Đối với các cột mốc còn lại, các giới hạn được thiết lập là 5% cho nỗ lực và tiến độ và 20% cho lỗi. Các báo cáo tại các cột mốc cũng đã được gửi cho người quản lý kinh doanh và khách hàng.

Kết quả cuối cùng của kế hoạch theo dõi và đo lường được chứa trong trong kế hoạch quản lý dự án ACIC, được trình bày trong Chương 8.

7.5 TÓM TẮT

Trong suốt quá trình lập kế hoạch cho dự án, bạn phải quyết định làm thế nào để lập kế hoạch để theo dõi tiến độ của dự án. Giám sát tiến độ là hoạt động cần thiết để đảm bảo rằng dự án đang tiến triển hướng tới các mục tiêu và cho phép bạn có hành động khắc phục khi tình hình đã được xác định. Giám sát dự án thường đòi hỏi thực hiện các phép đo.

Sau đây là một số bài học được rút ra từ việc đo lường và lập kế hoạch theo dõi tại Infosys:

- Lập kế hoạch để đo kích thước, tiến độ (thời gian biểu), nỗ lực, và lỗi. Những số đo này là đầy đủ cho hầu hết các dự án phần mềm.

- Phân loại nỗ lực ra một vài loại, và thu thập dữ liệu về nỗ lực bằng cách sử dụng một hệ thống tự động với các mã số hoạt động cho mỗi loại. Để tránh sự không chính xác do quên, hãy ghi nhận lại dữ liệu về nỗ lực một cách thường xuyên.

- Ghi nhận lại lỗi và theo dõi chúng cho đến khi đóng. Đối với một lỗi, cũng ghi lại loại của nó, giai đoạn phát hiện, giai đoạn tiêm vào, và mức độ nghiêm trọng để hỗ trợ cho các phân tích như: hiệu quả loại bỏ lỗi, chất lượng được giao, và tỷ lệ tiêm lỗi.

- Để phân tích hiệu suất tại các cột mốc, hãy thiết lập các giới hạn chấp nhận được cho sự chênh lệch giữa giá trị thực tế và giá trị ước lượng đã được lên kế hoạch cho nỗ lực, tiến độ, và lỗi. Trong quá trình thực hiện dự án, nếu hiệu suất vượt quá những giới hạn này, bạn phải thực hiện các hoạt động can thiệp.

Mặc dù theo dõi dự án và đo lường được yêu cầu bởi KPA Theo Dõi và Giám Sát Dự Án Phần Mềm (Software Project Tracking and Oversight KPA) ở CMM mức 2, bởi KPA Quản Lý Dự Án Tích Hợp (Integrated Project Management KPA) ở CMM mức 3, và cả hai KPAs ở CMM mức 4, nhưng CMM không nêu rõ ràng về sự cần thiết phải lập kế hoạch cho các phép đo này. Biết rằng có một nguyên tắc cơ bản chung trong CMM là các hoạt động chính sẽ được thực hiện phải được lập kế hoạch, do đó nó cũng ngụ ý về sự cần thiết để lập kế hoạch để đo lường cho những KPAs này.

7.6 CÁC THAM KHẢO

1. S.D. Conte, H.E. Dunsmore, and V.Y. Shen. *Software Engineering Metrics and Models*. Benjamin/Cummings, 1986.

2. S.H. Kan. *Metrics and Models in Software Quality Engineering*. Addison-Wesley, 1995.

3. V.R. Basili and D.M. Weiss. A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, 10(6), 1984.

4. V.R. Basili, G. Caldiera, and H.D. Rombach. Goal question metric paradigm. In *Encyclopedia of Software Engineering, John J. Marciniak, editor*. John Wiley and Sons, 1994.

5. R. Grady and D. Caswell. *Software Metrics: Establishing a Company-wide Program*. Prentice Hall, 1987.

6. Carnegie Mellon University/Software Engineering Institute. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley, 1995.

7. N. Brown. Industrial-strength management strategies. *IEEE Software*, July 1996.

8. W.A. Florac and A.D. Carleton. *Measuring the Software Process: Statistical Process Control for Software Process Improvement*. Addison-Wesley, 1999.

9. D.C. Montgomery. *Introduction to Statistical Quality Control, third edition*. John Wiley and Sons, 1996.

10. D.J. Wheeler and D.S. Chambers. *Understanding Statistical Process Control*, second edition. SPS Press, 1992.

11. W. Humphrey. *Managing the Software Process*. Addison-Wesley, 1989.

12. C. Jones. *Applied Software Measurement: Assuring Productivity and Quality*, second edition. McGraw Hill, 1996.

Chương 8. Kế hoạch quản lý dự án

Tài liệu của kế hoạch quản lý dự án là điểm cao nhất của tất cả các hoạt động lập kế hoạch được thực hiện bởi những người quản lý dự án. Kết quả đầu ra của các hoạt động lập kế hoạch khác nhau xuất hiện trong tài liệu này - nó trở thành tài liệu baseline hướng dẫn thực hiện tổng thể của dự án. Nó không nên bị nhầm lẫn với thời gian biểu chi tiết của dự án, mà chỉ trình bày cho việc xếp lịch và phân công các hoạt động.

Lập tài liệu cho các kết quả đầu ra của việc lập kế hoạch cho phép kế hoạch dự án được xem xét lại để tìm lỗi. Ở Infosys, các kế hoạch dự án thường được xem xét lại bởi một nhóm bao gồm những người quản lý dự án, các thành viên của nhóm quy trình công nghệ phần mềm (SEPG - software engineering process group), và người quản lý cấp cao. Trong nhiều trường hợp, hoạt động xem xét lại kế hoạch dự án đã ra tìm ra những thiếu sót rõ ràng mà nếu không được sửa chữa, nó có thể gây ra vấn đề rắc rối cho dự án. Xem xét lại kỹ lưỡng kế hoạch quản lý là một trong những cách tốt nhất để phát hiện những vấn đề tiềm tàng – điều này có giá trị rất lớn những người quản lý dự án, đặc biệt là những người còn ít kinh nghiệm.

Tài liệu này cũng quan trọng cho mục đích giao tiếp. Nó cung cấp cho người quản lý cấp cao một cái nhìn tổng thể về các mục tiêu và các cam kết của dự án và mô tả dự án sẽ được quản lý như thế nào để đáp ứng chúng. Nó mang lại cho nhóm dự án một cái nhìn toàn diện về dự án và vai trò của từng thành viên trong nhóm.

Mặc dù chúng tôi đã thảo luận về hầu hết các hoạt động lập kế hoạch, nhưng cho đến thời điểm này chúng tôi vẫn chưa thảo luận về nhóm và giao tiếp trong nhóm – sẽ được trình bày ở chương này. Ở đây, chúng tôi cũng thảo luận về cấu trúc của bản mẫu (template) được sử dụng tại Infosys để lập tài liệu kế hoạch và kiểm tra kế hoạch quản lý dự án cho dự án ACIC.

8.1 QUẢN LÝ NHÓM (TEAM MANAGEMENT)

Phát triển phần mềm là một nỗ lực của nhóm. Chất lượng và năng suất cao được đạt được khi các thành viên trong nhóm đóng góp hiệu quả và duy trì động lực, và toàn bộ nhóm phải làm việc một cách trơn tru và hiệu quả. Quản lý nhóm vượt ra ngoài phạm vi kỹ thuật (engineering) và các vấn đề phát sinh trong quản lý dự án đi vào lĩnh vực liên quan

đến các vấn đề của con người. Nó là một khía cạnh quan trọng của dự án mà nếu bỏ qua nó, có thể gây ra những rắc rối cho dự án và người quản lý dự án.

8.1.1 Cấu trúc nhóm (Team Structure)

Ở Infosys, cấu trúc nhóm phân cấp (hierarchical team structure) thường được sử dụng; một nhóm được lãnh đạo bởi một người quản lý dự án – người sẽ báo cáo cho giám đốc kinh doanh hoặc cho người quản lý quan hệ khách hàng (hoặc cả hai). Ngoài ra, một nhóm điển hình gồm có các nhà phát triển (DVs - developers), người kiểm soát cấu hình (CC- configuration controller), và người quản trị cơ sở dữ liệu (database administrator); tất cả các thành viên này sẽ báo cáo cho người quản lý dự án. Một dự án lớn cũng có thể có những người lãnh đạo mô-đun (module leaders), mỗi người trong số họ báo cáo cho người quản lý dự án và có một số nhà phát triển bên dưới anh ta. Ngoài ra, một nhóm phòng ngừa lỗi sẽ được thành lập từ các thành viên hiện có của nhóm; nhóm này chịu trách nhiệm thực hiện các nhiệm vụ liên quan đến phòng ngừa lỗi.

Như đã thảo luận trước đây, một thành viên của nhóm quy trình công nghệ phần mềm (SEPG - software engineering process group) được biết đến như là cố vấn chất lượng phần mềm (SQA - software quality adviser) cũng có liên quan đến từng dự án. SQA tương tác rất nhiều với người quản lý dự án (và với người kiểm soát cấu hình) nhưng không báo cáo với người quản lý dự án. Thay vào đó, SQA có một kênh báo cáo độc lập.

Việc phân công vai trò (assignment of roles) là một công việc phức tạp. Mục tiêu của người quản lý dự án là để có được một nhóm cân bằng và tự lực để tiếp tục phát triển sự nghiệp và kỹ năng của mỗi thành viên trong nhóm. Do đó, trong việc xác định cấu trúc nhóm, người quản lý dự án phải xem xét nhu cầu phát triển cá nhân của từng thành viên trong nhóm cũng như nhu cầu của dự án. Sau đây là một số yếu tố mà một người quản lý dự án cần quan tâm:

- Kỹ năng, kiến thức nền, và kinh nghiệm của các thành viên trong nhóm
- Nguyện vọng cá nhân và con đường sự nghiệp của các thành viên
- Các nhu cầu phát triển con người

Cấu trúc nhóm ban đầu được ghi chép vào trong kế hoạch quản lý dự án (PMP - project management plan) cùng với vai trò và trách nhiệm của mỗi thành viên trong nhóm. Một

thành viên trong nhóm có thể có nhiều trách nhiệm. Trong suốt quá trình của dự án, đặc biệt là đối với một dự án dài hạn, cấu trúc nhóm có thể bị thay đổi dựa trên hiệu suất của các thành viên khi đảm nhiệm các vai trò khác nhau, nguyện vọng cá nhân, động lực, v.v.

8.1.2 Giao tiếp (Communication)

Một nhóm sẽ làm việc cùng nhau trong vài tháng để cùng hướng tới một mục tiêu chung và phải có giao tiếp bên trong nhóm tốt. Các giao tiếp trong nhóm có thể được phân chia thành hai loại: giao tiếp liên quan đến dự án và giao tiếp để giảm căng thẳng (stress). Các kế hoạch của người quản lý dự án là hướng đến cả hai loại này.

Một cách để giúp cho các thành viên trong nhóm luôn biết được về tiến độ và các vấn đề phát sinh trong dự án là cho phép họ truy cập được vào các báo cáo tình trạng của dự án, ý kiến của khách hàng, và ý kiến của người quản lý kinh doanh. Ngoài những báo cáo hình thức (formal reports), tùy thuộc vào kích thước nhóm và thời gian hoàn thành dự án, những người quản lý dự án tại Infosys sử dụng một trong các phương pháp sau đây để tăng cường giao tiếp nhóm:

- Bảng trình bày ngắn gọn tin tức cho dự án: thông báo, chú ý, báo cáo tình trạng, v.v.
- Danh sách những địa chỉ nhận thông tin thường xuyên của dự án
- Web site của dự án để công bố các tài liệu, các trang chủ của mỗi thành viên trong nhóm, các bài viết về kỹ thuật và các ghi chú có liên quan, và tài liệu huấn luyện dành cho tự học.
- Các cuộc họp nhanh để giải quyết vấn đề phát sinh
- Các thực hành tốt nhất và các trình bày của các thành viên trong nhóm về công việc của họ

Hơn nữa, bởi vì thời hạn để hoàn thành các công việc (deadlines) thường ngắn và tất cả mọi người bị áp lực về thời gian, căng thẳng có xu hướng gia tăng. Giao tiếp để giảm căng thẳng (stress) thì cực kỳ quan trọng để đảm bảo mọi người có động lực tiếp tục. Nhiều người quản lý dự án lập kế hoạch cho các sự kiện tạo ra giao tiếp "vui vẻ" (fun). Dưới đây là ví dụ về các phương pháp được sử dụng:

- Tiệc dự án (được hỗ trợ ngân sách từ công ty cho tất cả các nhóm)
- Tiệc sinh nhật
- Những sự kiện như đố vui và các trò chơi có thưởng

8.1.3 Phát triển nhóm (Team Development)

Nhóm dự án thường bao gồm nhiều người trẻ. Trách nhiệm của nhóm và người quản lý dự án để tăng cường phát triển cá nhân của các thành viên trong nhóm. Việc giúp phát triển các thành viên của nhóm cũng có lợi cho dự án và công ty. Khi các kỹ năng và khả năng của các thành viên trong nhóm được cải thiện, họ làm việc có năng suất cao hơn trong dự án và có thể đảm nhận nhiều trách nhiệm hơn. Và, tất nhiên, họ sẽ được trang bị tốt hơn để xử lý các nhiệm vụ của họ trong các dự án tương lai. Hướng tới mục tiêu này, những người quản lý dự án sử dụng các phương pháp như sau:

- Luân chuyển công việc (xoay vòng công việc)
- Tư vấn cho các thành viên mới bởi các thành viên nhiều kinh nghiệm hơn
- Nhận xét, đánh giá và phản hồi
- Thường xuyên công nhận các đóng góp ở mức dự án
- Huấn luyện, đào tạo, và luôn giúp đỡ những người gặp khó khăn

8.2 GIAO TIẾP VỚI KHÁCH HÀNG VÀ GIẢI QUYẾT VẤN ĐỀ PHÁT SINH (CUSTOMER COMMUNICATION AND ISSUE RESOLUTION)

Giao tiếp nhóm (team communication) hướng tới việc luôn giữ cho nhóm được thông báo thông tin và có động lực. Nhưng đối với khách hàng (customer) thì sao, ai là nhà tài trợ (sponsor) và các đối tượng liên quan đến việc thực hiện dự án (stakeholder)? Nhiều vấn đề có nguyên nhân là do sự hiểu lầm giữa khách hàng và các nhà phát triển. Giao tiếp thường xuyên giữa nhóm phát triển và khách hàng có thể giúp tránh những vấn đề này.

Các báo cáo tình trạng (trạng thái/status reports) (được thảo luận trong Chương 11), là một trong những phương tiện giao tiếp, có mục đích là để cung cấp cho khách hàng hiểu rõ ràng về tình trạng của dự án đang tiếp diễn. Tuy nhiên, những báo cáo này là chưa đủ.

Những người quản lý dự án cần lập kế hoạch cho các phương tiện giao tiếp khác, bao gồm thực hiện thảo luận từ xa (teleconferencing) hoặc thảo luận qua video (videoconferencing) hàng tuần và thường xuyên liên lạc qua e-mail. Trong một cuộc họp ảo hàng tuần, người lãnh đạo dự án xem qua bản báo cáo tình trạng với khách hàng và giải thích các ràng buộc của dự án. Một điểm then chốt của cuộc thảo luận là giải quyết các vấn đề chưa giải quyết. Ngược lại, khách hàng tìm cách làm rõ ràng và giải thích quan điểm của mình trong các cuộc họp này. Nhìn chung, thông qua giao tiếp thường xuyên (vượt ra ngoài khuôn khổ so viếc chỉ gửi báo cáo), cả khách hàng và nhóm phát triển sẽ giữ liên lạc đồng bộ. Điều này phòng ngừa các vấn đề tiềm ẩn bắt nguồn từ sự hiểu lầm.

Mặc dù việc sử dụng các kênh thông tin liên lạc thường xuyên, nhưng vẫn có các vấn đề phát sinh mà đại diện ở hai phía không thể giải quyết. Các vấn đề như vậy có thể có khả năng làm trì hoãn dự án và phải bị leo thang (escalated). Để tạo điều kiện thuận lợi cho việc giải quyết các vấn đề như vậy, một kế hoạch dự án cần xác định cụ thể các kênh leo thang (escalation channels) ở cả hai phía: khách hàng và Infosys. Kế hoạch cũng nêu rõ các chính sách có liên quan khi các kênh này sẽ được triển khai. Ngoài ra, cần cung cấp thêm một cơ chế để giải quyết các vấn đề phát sinh, đặc tả cho kênh leo thang này và các chính sách để tạo ra áp lực lên hai phía nhanh chóng giải quyết các phát sinh và khi cần thiết có thể thực hiện chúng nhiều hơn nữa.

8.3 CẤU TRÚC CỦA KẾ HOẠCH QUẢN LÝ DỰ ÁN (THE STRUCTURE OF THE PROJECT MANAGEMENT PLAN)

Kế hoạch quản lý dự án mẫu (template) được cung cấp tại Infosys có bốn phần chính. *Phần tóm tắt dự án (project summary section)* cung cấp một cái nhìn tổng quan ở mức cao về dự án. Nó bao gồm các thông tin về ngày bắt đầu và kết thúc, lãnh đạo dự án, địa chỉ liên hệ của khách hàng, mục tiêu của dự án, các cam kết chính được thực hiện cho khách hàng tại các cột mốc (milestones), các sản phẩm sẽ được giao, và các giả định được làm. Các giả định đã thực hiện được liệt kê rõ ràng bởi vì chúng thường xuyên được dùng như là một nguồn rủi ro (source of risks). Thông tin chi tiết về thanh toán cũng có thể được mô tả (nhờ vậy mà người quản lý kinh doanh có thể theo dõi chúng). Các mục tiêu của dự án - từ quan điểm của khách hàng đến quan điểm của Infosys - được đề cập đến để mà nó làm cho tất cả mọi người thấy rõ là tại sao dự án sẽ được thực hiện.

Phần lập kế hoạch dự án (*project planning section*) liệt kê các kết quả đầu ra (outputs) của việc thực hiện theo các thủ tục (procedure) trong kế hoạch dự án. Nó bao gồm quy trình phát triển đang được sử dụng, các ghi chú về điều chỉnh, quy trình quản lý thay đổi yêu cầu, các kế hoạch để theo dõi dấu vết nguồn gốc yêu cầu (requirement traceability plans), ước lượng nỗ lực và thời gian hoàn thành, và yêu cầu nhân sự theo kỹ năng, vai trò, và kinh nghiệm. Nó cũng quy định môi trường phát triển, các công cụ làm việc, và bất kỳ kế hoạch đào tạo cụ thể cho dự án. Kế hoạch chất lượng và kế hoạch quản lý rủi ro cũng được đưa ra trong phần này.

Phần theo dõi dự án (*project tracking section*) xác định các phép đo được thực hiện và các hệ thống được sử dụng để thu thập dữ liệu, theo dõi các hoạt động khác nhau của dự án được thực hiện, tần suất và bản chất của báo cáo tiến độ, và các thủ tục leo thang (escalation procedures).

Phần nhóm dự án (project team section) xác định nhóm dự án và cấu trúc của nó, cũng như vai trò và trách nhiệm của từng người tham gia.

8.4 KẾ HOẠCH CỦA DỰ ÁN ACIC (THE ACIC PROJECT PLAN)

Phần này trình bày kế hoạch dự án cho case study của chúng tôi, dự án ACIC. Phần đầu tiên của kế hoạch quản lý dự án (PMP) cung cấp những người tham gia chính, các cột mốc, và tổng quan về dự án. (Chương 4 giải thích làm thế nào thời gian biểu và các cột mốc đã được xác định.)

Phần 2 chứa thông tin chi tiết về kế hoạch. Đầu tiên, nó sẽ liệt kê đầu ra của kế hoạch quy trình (process planning). Kế hoạch này nói rằng "quy trình phát triển" (development process) sẽ được sử dụng và cung cấp điều chỉnh vừa được thực hiện cho dự án. Trong dự án này, Rational Rose Unified Process (RUP) sẽ được sử dụng bởi vì nó là một cam kết với khách hàng. Trong RUP, các giai đoạn chính là giai đoạn bắt đầu (inception phase), giai đoạn phát thảo tỉ mỉ (elaboration phase), giai đoạn xây dựng (construction phase), và giai đoạn chuyển giao (transition phase). Giai đoạn phát thảo tỉ mỉ và xây dựng thường được thực hiện trong các vòng lặp. Trong giai đoạn phát thảo tỉ mỉ (elaboration phase), phân tích (analysis) và thiết kế (design) là hai hoạt động chính, trong khi đó trong giai đoạn xây dựng (construction phase), cài đặt mã (coding) và kiểm thử đơn vị (unit testing) là các hoạt động chính. Dự án sẽ thực hiện hai vòng lặp để phát thảo tỉ mỉ và ba vòng lặp để xây dựng. Để thích ứng với phương pháp này đòi hỏi rằng quy trình phát triển chuẩn

(standard development process) của Infosys được điều chỉnh lại (tailored). Các ghi chú về việc điều chỉnh ghi rõ các điều chỉnh đã được làm lên quy trình chuẩn.

Để quản lý thay đổi yêu cầu (requirements change management), kế hoạch xác định cụ thể nơi các yêu cầu thay đổi sẽ được ghi nhận và quá trình xử lý chúng. Nó cũng xác định rằng nếu bất kỳ yêu cầu thay đổi nào cần hơn 2% tổng số nỗ lực (chi phí), nó sẽ được ước lượng lại. Để theo dõi dấu vết nguồn gốc yêu cầu (requirements traceability), một công cụ sẽ được sử dụng.

Phần nhỏ (subsection) về ước lượng nỗ lực (effort estimation) cho biết các nỗ lực ước lượng và cơ sở của ước lượng này. (Chương 4 thảo luận về ước lượng cho dự án ACIC.) Cùng với ước lượng nỗ lực, nó xác định thời gian biểu (thời hạn hoàn thành) của dự án (cái này giống với những cột mốc đã cam kết với khách hàng) và số nhân sự cần thiết để đáp ứng các cam kết về tiến độ. Yêu cầu về kỹ năng của nhân sự cũng được xác định. Kế hoạch đào tạo cho nhóm cũng được cung cấp, cùng với các tiêu chuẩn bỏ qua (không cần phải tham dự khóa đào tạo).

Phần nhỏ về kế hoạch cho chất lượng (quality plan) cho biết các mục tiêu chất lượng và các mục tiêu trung gian cho lỗi. (Chương 5 thảo luận về kế hoạch này.) Bởi vì dự án đã đặt ra các mục tiêu cao hơn so với tiêu của chuẩn công ty, chiến lược để đáp ứng những mục tiêu này cũng được đưa ra, cùng với những xem xét lại (reviews) theo những mục tiêu.

Phần lập kế hoạch cũng xác định các phần cứng và phần mềm cần thiết và kế hoạch quản lý rủi ro (được thảo luận trong Chương 6).

Phần theo dõi dự án (phần 3) đề cập rằng Microsoft Project (MSP) sẽ được sử dụng để lập lịch cho công việc, phân công, và giám sát tình trạng. BugsBunny, một công cụ do công ty tự phát triển, sẽ được sử dụng để theo dõi các vấn đề nội bộ của dự án. Người quản lý dự án chịu trách nhiệm rà soát, bảo đảm giải quyết những vấn đề này, ngoại trừ các vấn đề phát sinh liên quan đến người quản lý kinh doanh. BugsBunny cũng sẽ được sử dụng để theo dõi các khiếu nại và phản hồi của khách hàng. Báo cáo tình trạng (trạng thái) sẽ được gửi định kỳ mỗi hai tuần, và khách hàng sẽ nhận được một báo cáo tại mỗi cột mốc (milestone). Phân tích chi tiết được lên kế hoạch tại các cột mốc. Các kênh leo thang (escalation channels) ở cả hai phía (phát triển tại chỗ và ở nước ngoài) được thành lập.

Phần 4 xác định cấu trúc nhóm. Nhóm được đứng đầu bởi người lãnh đạo dự án (PL – project leader) (được đề cập ở những nơi khác trong cuốn sách này như người quản lý dự án), là người quản lý nhóm ngăn chặn lỗi (DP – defect prevention), người lãnh đạo mô-đun (module leader), và người kiểm soát cấu hình (CC - configuration controller). Nhóm DP chịu trách nhiệm cho việc phân tích lỗi và đề xuất các giải pháp để ngăn chặn chúng. (Chương 5 thảo luận về việc lập kế hoạch DP, và Chương 11 giải thích quá trình thực hiện các hoạt động DP. Vai trò của CC được thảo luận chi tiết trong Chương 9.) Cấu trúc nhóm xác định rằng một cố vấn chất lượng phần mềm sẽ kết hợp với nhóm nhưng không báo cáo cho PL. Kế hoạch này cũng quy định rõ vai trò và trách nhiệm của mỗi người trong nhóm.

Kế hoạch quản lý dự án ACIC (Project Management Plan for the ACIC Project)

1. TÓM TẮT DỰ ÁN (PROJECT SUMMARY)

1.1 Tổng quan về dự án (Project Overview)

ACIC là một công ty đầu tư. Ứng dụng này có 2 thành phần: 1) ứng dụng Mở Tài Khoản Môi Giới (Brokerage Account Opening) trên trang web của ACIC để cho phép bất kỳ người sử dụng Internet nào cũng có thể mở một tài khoản môi giới ở ACIC; 2) ứng dụng mở tài khoản và bảo trì, đây là phần chính giúp ACIC mở tài khoản cho các đơn xin (applications) mà công ty đã nhận được ở định dạng giấy. Đây là một ứng dụng Intranet (nội bộ). Ứng dụng sẽ cung cấp các tính năng như xem lịch sử tài khoản (account history) và số dư của tài khoản (account balance), tình trạng (status), và thông tin về các hoạt động (activity information). Điều này sẽ cho phép ACIC phát triển bằng cách mở rộng dần dần ứng dụng phục vụ khách hàng/tài khoản để bổ sung cho động cơ mở tài khoản (account opening engine). Đây là dự án để mở rộng ứng dụng hiện có; ứng dụng này trước đây cũng do Infosys phát triển.

Mã dự án (Project Code)	Tên dự án (Project Name)	Khách hàng (Customer)
xxxxxxxxx	Dự án ACIC	Công ty ACIC

Người lãnh đạo dự án (PL-Project Leader)	Người kiểm soát cấu hình (CC - Configuration Controller)	Người quản lý kinh doanh (BM-Business Manager)	PL dự phòng	CC dự phòng
BB	SB	HR	BJ	HP

Loại dự án (Project Type)	Nền (Platform)	Số giai đoạn (Number of Phases)
Phát triển (Development)	Java, Win NT, DB2	4

Ngày bắt đầu dự án (bao gồm phát triển tại chỗ và phát triển ở nước ngoài) (Project Start Date (including onsite, offshore)		Ngày kết thúc dự án (Project End Date)	Tổng lợi tức/tiền lời được ước lượng (Total Estimated Revenue)
Tại chỗ (Onsite)	**Ở nước ngoài (Offshore)**		
April 3, 2000	May 15, 2000	Nov. 3, 2000	US $ xxx,xxx

Dự án và địa chỉ liên hệ của khách hàng (Project and Customer Contact Personnel)			
Tên và chức vụ (Name and Designation)	Số điện thoại (Phone Number)	Số Fax (Fax Number)	E-mail ID

Phạm vi dự án (Project Scope)
• Để cung cấp một phương tiện hiệu quả cho các hoạt động bảo trì tài khoản • Để cho phép người dùng truy cập thông tin • Để cung cấp một bức tranh hoàn chỉnh cho khách hàng về tình trạng tài khoản, đánh giá, tình trạng của các đơn đặt hàng, và các hoạt động thương mại • Để nâng cao trí thông minh của quy trình cập nhật • Để cung cấp một giao diện có thể hiển thị lịch sử của tài khoản theo yêu cầu • Để cung cấp khả năng đóng và kích hoạt lại tài khoản

Giá trị của dự án đối với khách hàng (Project's Value-add to the Customer)
• Dự án này sẽ cho phép ACIC phát triển hiệu quả ứng dụng phục vụ tài khoản khách hàng (client account servicing application) để bổ sung cho động cơ mở tài khoản (account opening engine).

Các mục tiêu của Infosys (Infosys Objectives)
• Tăng cường hơn nữa mối quan hệ với ACIC bằng cách giao phần mềm chất lượng cao đúng hạn. • Trở thành nhà cung cấp ưa thích của ACIC bằng cách phát triển thành thạo về chuyên môn cho các sản phẩm và hệ thống của họ.

1.2 Các cam kết được làm với khách hàng (Commitments Made to the Customer)

STT	Ngày của cột mốc (Milestone Date)	Các cột mốc (Milestones)	Sản phẩm được giao (Deliverables)
1	26 May 2000	**Giai đoạn bắt đầu: Yêu cầu được ký** *(Inception: Requirements sign-off)*	Phân tích nghiệp vụ, đặc tả yêu cầu, danh sách các trường hợp sử dụng, màn hình, kế hoạch cho vòng lặp kế tiếp *(Business analysis and requirements specifications, use case catalog, screens, iteration plan)*
2	15 May–23 June 2000	**Giai đoạn phát thảo tỉ mỉ: Vòng lặp 1** *(Elaboration: Iteration 1)*	Biểu đồ tuần tự, biểu đồ lớp, mã nguồn, kế hoạch cho vòng lặp kế tiếp *(Sequence diagrams, class diagram, source code, plan for the next cycle)*
3	26 June–7 July 2000	**Giai đoạn phát thảo tỉ mỉ: Vòng lặp 2** *(Elaboration: Iteration 2)*	Các đặc tả bổ sung, biểu đồ tuần tự, biểu đồ lớp, tài liệu kiến trúc, mã nguồn, kế hoạch cho vòng lặp kế tiếp *(Supplementary specifications, sequence diagrams, class diagram, architecture document, source code, iteration plan for the next cycle)*

4	10 July–21 July 2000	**Giai đoạn xây dựng: Vòng lặp 1** *(Construction: Iteration 1)*	Mã nguồn, các báo cáo xem xét lại, các báo cáo kiểm thử, lập kế hoạch cho vòng lặp kế tiếp *(Source code, review reports, test reports, iteration plan for the next cycle)*
5	20 July– 28 July 2000	**Giai đoạn xây dựng: Vòng lặp 2** *(Construction: Iteration 2)*	Mã nguồn, các báo cáo xem xét lại, các báo cáo kiểm thử, lập kế hoạch cho vòng lặp kế tiếp *(Source code, review reports, test reports, iteration plan for the next cycle)*
6	31 July–8 Aug 2000	**Giai đoạn xây dựng: Vòng lặp 3** *(Construction: Iteration 3)*	Mã nguồn, các báo cáo xem xét lại, các báo cáo kiểm thử, lập kế hoạch cho vòng lặp kế tiếp *(Source code, review reports, test reports, iteration plan for the next cycle)*
7	9 Aug–1 Sep 2000	**Giai đoạn kiểm thử tích hợp** *(Integration testing phase)*	Các kế hoạch kiểm thử, các báo cáo kiểm thử *(Test plans, test reports)*
8	4 Sep–15 Sep 2000	**Giao mã và cài đặt lên máy khách hàng (tại công ty)** *(Onsite code delivery and setup)*	Mã *(Code)*
9	18 Sep–22 Sep 2000	**Kiểm thử chấp nhận và di chuyển sản xuất** (*Acceptance test and production migration)*	Các báo cáo kiểm thử *(Test reports)*
10	18 Sep–29 Sep 2000	**Hòa giải và kiểm thử chấp nhận tại phía khách hàng**	Mã *(Code)*

		(Onsite reconciliation and regression test)	
11	2 Oct–26 Oct 2000	**Kiểm thử chấp nhận** *(Acceptance test)*	Các kết quả kiểm thử *(Test results)*
12	27 Oct–3 Nov 2000	**Triển khai sản phẩm và hỗ trợ** *(Rollout and support)*	Ký nhận kết thúc dự án *(Project sign-off)*

Các cam kết khác (Other Commitments)	
STT	**Các cam kết (Commitments)**
1	Dự án này sẽ thực hiện theo Phương pháp Rational Unified (RUP).

1.3 Các giả định (Assumptions)

Các giả định được làm trong khi hoạch định (Assumptions Made while Planning)
• Việc chuyển sang Visual Age cho Java 3.0 sẽ không được thực hiện bởi nhóm này. • Cập nhật thông minh cho các đối tác kinh doanh (business partners) sẽ chỉ kết hợp vào trong phần bảo trì ứng dụng chứ không vào trong động cơ (engine) "mở tài khoản" (Account opening). • Những người có năng lực sẽ phê duyệt phương pháp Rational Unified Process (RUP) để thực hiện dự án này. • Các thay đổi trong các yêu cầu chức năng và kỹ thuật trong vòng đời của dự án có thể có ảnh hưởng đến thời gian biểu của dự án/lịch (schedule). Bất kỳ tác động nào đến chi phí và thời gian biểu do những thay đổi này gây ra sẽ được báo cho ACIC biết. • Những người xem xét lại (reviewer) của ACIC sẽ có 7 ngày để phê duyệt tài liệu của một cột mốc (milestone document). Nếu không nhận được ý kiến nào trong khoảng thời gian này, nó sẽ được coi là đã được phê duyệt.

2. LẬP KẾ HOẠCH DỰ ÁN (PROJECT PLANNING)

2.1 Các quy trình của dự án (Project Processes)

Quy trình chuẩn được theo (Standard Process Followed)

Quy trình phát triển chuẩn của Infosys sẽ được theo. Tuy nhiên, nó sẽ được mở rộng với phương pháp Rational Unified Process (RUP), vì đó là một cam kết với khách hàng. Quy trình phát triển sẽ được điều chỉnh để phù hợp với RUP.

Các ghi chú điều chỉnh (Tailoring Notes)

Sự lệch so với quy trình chuẩn (Deviations from Standard Process)	**Thêm/Sửa/Xóa (Added/Modified/Deleted)**	**Lý do lệch (Reasons for Deviations)**
Chỉ có những trường hợp sử dụng (use case) sẽ được thực hiện trong vòng lặp kế tiếp được thiết kế chi tiết ở thời điểm hiện hành. *(Only those use cases that are going to be taken up in a particular iteration will be elaborated at that point in time.)*	Sửa *(Modified)*	Phát triển theo vòng lặp được thực hiện *(Iteration-based development is being done.)*
Phát triển mô hình đối tượng lôgic sẽ được thực hiện tăng dần (từng bước) trong các vòng lặp đầu tiên. *(Development of logical object model will be done incrementally in the first few iterations.)*	Sửa *(Modified)*	Để phù hợp với phương pháp RUP *(Conformance to RUP methodology.)*
Phát triển mô hình đối tượng vật lý sẽ được thực hiện tăng dần (từng bước) trong các vòng lặp lại đầu tiên. *(Development of physical object model will be done incrementally in the first few iterations.)*	Sửa *(Modified)*	Để phù hợp với phương pháp RUP
Thiết kế của cơ sở dữ liệu vật lý có thể được tinh chỉnh lại trong các vòng lặp sau đó.	Sửa	Để phù hợp với phương pháp RUP

(Physical database design may be refined in later iterations.)	*(Modified)*	
Xây dựng kế hoạch kiểm thử đơn vị sẽ được thực hiện trong mỗi vòng lặp. *(Development of unit test plan will be done in each iteration.)*	Sửa *(Modified)*	Phương pháp vòng lặp sẽ được sử dụng *(Iterative approach is being used.)*
Ghi nhận lại lỗi trong từng vòng lặp. *(Logging of defects will be iteration-wise.)*	Sửa *(Modified)*	Phương pháp vòng lặp sẽ được sử dụng
Theo dõi dấu vết nguồn gốc yêu cầu sẽ được thực hiện bằng công cụ Requisite Pro *(Requirement traceability will be done through the Requisite Pro tool.)*	Sửa *(Modified)*	Để phù hợp với phương pháp RUP
Không có tài liệu trực quan và trường hợp nghiệp vụ bởi vì chúng tôi đã bắt đầu dự án từ tài liệu Phạm Vi (cái này được sử dụng cho cùng một mục đích). *(No vision document and business case as we started with the Scope document, which serves the same purpose.)*	Sửa *(Modified)*	Lệch với RUP *(Deviation from RUP.)*

Quy trình quản lý thay đổi yêu cầu (Requirements Change Management Process)

Theo dõi các yêu cầu thay đổi (Change Requests Tracking)
• Các thay đổi theo yêu cầu của khách hàng sẽ được ghi nhận vào trong tập tin ChangeRequest.xls và tác động của thay đổi này lên dự án sẽ được phân tích. Một biểu mẫu (form) về yêu cầu thay đổi sẽ được gửi cho khách hàng để phê duyệt. Các yêu cầu thay đổi sau khi được phê duyệt sẽ được gắn vào hợp đồng dự án như một phụ lục.

• Các thay đổi lớn thường tác động đến nỗ lực/thời gian giao sản phẩm. Khách hàng cần phê duyệt những điều này. • Bởi vì đây là một dự án có thời gian thực hiện ngắn, nếu có một hoặc nhiều yêu cầu thay đổi phải xài hơn 2% của nỗ lực ước lượng tổng thể của dự án, cần phải ước lượng lại tiến độ và nỗ lực cho cả dự án.

Theo dõi dấu vết nguồn gốc yêu cầu (Requirements Traceability)

Công cụ Requisite Pro sẽ được sử dụng.

2.2 Kích thước và nỗ lực ước lượng

Tiêu chuẩn ước lượng (Estimation Criteria)	
Chương trình/Chức năng (Trường hợp sử dụng) (Program/Function (Use Case))	**Tiêu chuẩn (Criteria)**
Trường hợp sử dụng đơn giản (Simple use case)	<=3 giao dịch
Trường hợp sử dụng trung bình (Medium use case)	4 - 7 giao dịch
Trường hợp sử dụng phức tạp (Complex use case)	> 7 giao dịch

Số STT của trường hợp sử dụng	**Mô tả**	**Loại phức tạp**
1	Màn hình điều khiển *(Navigate Screen)*	Phức tạp
2	Cập nhật thông tin cá nhân chi tiết *(Update Personal Details)*	Trung bình
3	Thêm địa chỉ *(Add Address)*	Trung bình
4	Cập nhật địa chỉ *(Update Address)*	Phức tạp
5	Xóa địa chỉ *(Delete Address)*	Phức tạp
6	Thêm số điện thoại *(Add Telephone Number)*	Trung bình
7	Cập nhật số điện thoại *(Update Telephone Number)*	Phức tạp
8	Xóa số điện thoại *(Delete Telephone Number)*	Phức tạp
9	Thêm email *(Add E-mail)*	Trung bình
10	Cập nhật email *(Update E-mail)*	Trung bình

11	Xóa email *(Delete E-mail)*	Trung bình
12	Cập nhật thông tin chi tiết về việc làm của đối tác *(Update Employment Details of a Party)*	Trung bình
13	Cập nhật thông tin tài chính của đối tác *(Update Financial Details of a Party)*	Trung bình
14	Cập nhật thông tin chi tiết của tài khoản *(Update Details of an Account)*	Trung bình
15	Các hoạt động bảo trì của một tài khoản *(Maintain Activities of an Account)*	Phức tạp
16	Bảo trì sổ ghi nhớ của một tài khoản *(Maintain Memos of an Account)*	Đơn giản
17	Xem lịch sử chi tiết của đối tác *(View History of Party Details)*	Phức tạp
18	Xem lịch sử chi tiết của tài khoản *(View History of Account Details)*	Phức tạp
19	Xem lịch sử mức tùy chọn và các tùy chọn dịch vụ *(View History of Option Level and Service Options)*	Đơn giản
20	Xem lịch sử của các hoạt động và sổ ghi nhớ *(View History of Activities and Memos)*	Đơn giản
21	Xem lich sử của các vai trò *(View History of Roles)*	Phức tạp
22	Xem các chi tiết về tài khoản *(View Account Details)*	Đơn giản
23	Xem các cổ phần của một tài khoản *(View Holdings of an Account)*	Phức tạp
24	Xem các đơn đặt hàng chưa được xử lý của một tài khoản *(View Pending Orders of an Account)*	Phức tạp
25	Đóng/Kích hoạt một tài khoản *(Close/Reactivate Account)*	Đơn giản
26	Làm cập nhật thông minh cho các đối tác kinh doanh của ACIC *(Make Intelligent Update to Business Partners of ACIC)*	Phức tạp

Nỗ lực xây dựng được ước lượng (Estimated Build Effort)			
Chương trình/Chức năng (Program/Function)	**Nỗ lực (Dựa trên dữ liệu từ dự án trước đây)**	**Số đơn vị**	**Tổng nỗ lực xây dựng (người-ngày)**
Trường hợp sử dụng đơn giản	1 Người-ngày	5	5
Trường hợp sử dụng trung bình	5 Người-ngày	9	45
Trường hợp sử dụng phức tạp	8 Người-ngày	12	96
Tổng			**146**

Ước lượng nỗ lực theo giai đoạn (Phase-wise Effort Estimate)		
Hoạt động/giai đoạn (Activity/Phase)	**Người-ngày (Person-days)**	**% của nỗ lực tổng (% of total effort)**
Yêu cầu *(Requirements)*	50	10
Thiết kế *(Design)*	60	12
Xây dựng *(Build)*	146	29
Kiểm thử tích hợp *(Integration testing)*	35	7
Kiểm thử hồi quy *(Regression testing)*	10	2
Kiểm thử chấp nhận *(Acceptance testing)*	30	6
Quản lý dự án *(Project management)*	75	15
Quản lý cấu hình *(Configuration management)*	16	3
Huấn luyện *(Training)*	50	10
Khác (Others)	40	6
Nỗ lực ước lượng *(Estimated effort)*	**501**	**100%**

Ước lượng nỗ lực theo các vòng lặp (Effort Estimate by Iterations)	**Người-ngày (Person-days)**	**% của nỗ lực tổng cộng (% of total effort)**
Khởi xướng dự án *(Project initiation)*	25	5
Giai đoạn bắt đầu *(Inception phase)*	24	5
Giai đoạn phát thảo tỉ mỉ: Vòng lặp 1 *(Elaboration phase: Iteration 1)*	45	9

Giai đoạn phát thảo tỉ mỉ: Vòng lặp 2 *(Elaboration phase: Iteration 2)*	34	7
Giai đoạn xây dựng: Vòng lặp 1 *(Construction phase: Iteration 1)*	27	5
Giai đoạn xây dựng: Vòng lặp 2 *(Construction phase: Iteration 2)*	24	5
Giai đoạn xây dựng: Vòng lặp 3 *(Construction phase: Iteration 3)*	21	4
Giai đoạn chuyển giao *(Transition phase)*	110	22
Kết thúc dự án *(Project closure)*	10	2
Quản lý dự án *(Project management)*	75	15
Quản lý (*Configuration management)*	16	3
Đào tạo *(Training)*	50	10
Khác *(Others)*	40	8
Tổng nỗ lực ước lượng (Total estimated effort)	**501 người-ngày (person-days)**	**100%**

2.3 Thời gian biểu (Schedule)

Được xác định như các cột mốc (milestones) trong phần *Các cam kết được làm với khách hàng (Commitments Made to the Customer).*

2.4 Con người (People)

Người theo vai trò (People by Role)		
Vai trò (Role)	**Số lượng cần (Required Number)**	**Ngày (Date)**
Người lãnh đạo dự án (PL)	1	4 May 2000
Điều phối viên tại chỗ (Onsite coordinator)	1 (50% thời gian)	4 May 2000
Người lãnh đạo mô-đun (Module leader)	1	15 May 2000
Nhà phát triển (Developers)	3	15 May 2000
Nhà phát triển (Developers)	1	17 July 2000

Nhà phát triển (Developers)	1	1 August 2000
Nhà phát triển (Developers)	1	14 August 2000
Tổng	**9 (thực sự 8.5)**	

Người theo kỹ năng và kinh nghiệm (People by Skill and Experience)			
Lĩnh vực	**Tổng số**	**0-12 tháng kinh nghiệm (0–12 months' experience)**	**>12 tháng kinh nghiệm**
Java	7	7	0
DB2	2	0	2
Total	9	7	2

Kế hoạch cần người (People Requirement Plan)			
Tháng	**Ở nước ngoài (Offshore)**	**Tại chỗ (Onsite)**	**Tổng**
May 2000	4	1 (50%)	5
June 2000	5	1	6
July 2000	5	1	6
Aug 2000	8	1	9
Sep 2000	7	2	9
Oct 2000	3	2	5

2.5 Môi trường phát triển (Development Environment)

Phần cứng (Hardware)	**Phần mềm (Software)**
NT Server	Win NT
MainFrame	DB2
Intel PC	VisualAge for Java, Java, Win NT

2.6 Yêu cầu tài nguyên phần cứng và phần mềm (Hardware and Software Resources Required)

Mô tả	Số lượng cần	Ngày
PCs with 128 RAM	6	1 May 2000
1GB space on server	1	1 May 2000
VisualAge for Java	6	4 May 2000
DB2	6	4 May 2000
Rational Rose	5	15 May 2000
Requisite Pro	1	15 May 2000

2.7 Các công cụ

Danh sách công cụ (Tools List)	
Các công cụ được phát triển trong dự án	Không
Các công cụ do công ty phát triển được dùng trong dự án	BugsBunny, WAR

2.8 Kế hoạch đào tạo (Training Plan)

Lĩnh vực đào tạo (Training Area)	Thời gian (Duration)	Tiểu chuẩn bỏ qua (Waiver Criteria)
Kỹ thuật (Technical)		
Ngôn ngữ Java	7 ngày	Nếu đã được đào tạo rồi (If already trained)
VisualAge cho Java	3 ngày	Đã được đào tạo rồi như là một phần của đào tạo ban đầu (Exposed as part of initial training)
Java Applets	4 giờ	Nếu đã được đào tạo rồi
Java Swing	4 giờ	Nếu đã được đào tạo rồi
Persistence Builder	4 giờ	Nếu đã được đào tạo rồi
Rational Rose và Requisite Pro	8 giờ	Bắt buộc (Mandatory)
OOAD	1 ngày	Nếu đã được đào tạo rồi

Lĩnh vực nghiệp vụ (Business Domain)		
Đánh giá hệ thống *(System appreciation)*	7 ngày	Nếu đã được đào tạo rồi
Liên quan đến quy trình (Process-Related)		
Hệ thống chất lượng *(Quality system)*	3 giờ	Nếu đã được đào tạo rồi
Quản lý cấu hình *(Configuration management)*	2 giờ	Nếu đã được đào tạo rồi cho người kiểm soát cấu hình (CC). Đối với những người khác, đào tạo các vấn đề liên quan đến công việc của họ. *(If already trained for CC. For others, on-the-job training)*
Xem xét lại bởi nhóm *(Group review)*	4 giờ	Nếu đã được đào tạo rồi
Phòng ngừa lỗi *(Defect prevention)*	4.5 giờ	Bắt buộc
Công cụ kiểm soát quy trình dùng thống kê (SPC tool)	4.5 giờ	Nếu đã được đào tạo rồi
Phương pháp RUP *(RUP methodology)*	2 giờ	Bắt buộc

2.9 Kế hoạch chất lượng (Quality Plan)

Các mục tiêu chất lượng (Quality Goals)

Các mục tiêu chất lượng của dự án (Project Quality Goals)			
Các mục tiêu (Goals)	**Giá trị (Value)**	**Cơ sở để thiết lập các mục tiêu (Basis for Setting Goals)**	**Tiêu chuẩn của công ty (Organization-wide Norms)**
Tổng số lỗi được tiêm vào *(Total number of defects injected)*	145	0.033 lỗi/người-giờ. Giá trị này lớn hơn 10% so với dự án Synergy (dự án Synergy chỉ là 0.036 lỗi/người-giờ) *(0.033 defects/person-hour.This is*	0.052 lỗi/người-giờ *(0.052 defects/person-hour)*

		10% better than Synergy, which was 0.036 defects/person-hour)	
Chất lượng (số lỗi chấp nhận được) *(Quality (acceptance defects))*	5	3% hoặc ít hơn 3% của tổng số lỗi được ước lượng *(3% or less of total estimated number of defects)*	6% của tổng số lỗi được ước lượng *(6% of estimated number of defects)*
Năng suất tính theo số Điểm chức năng/người-tháng *(Productivity in FP/person-month)*	57	Tăng thêm 3.4% so với năng suất của dự án Synergy *(3.4% productivity improvement over Synergy)*	50
Thời gian hoàn thành *(Schedule)*	Giao đúng hạn *(Delivery on time)*		10%
Chi phí của chất lượng *(Cost of quality)*	32%	31.5%	32%

Ước lượng số lỗi sẽ bị xóa (Estimates of Defects to Be Detected)

Giai đoạn xem xét lại/Kiểm thử (Review/Testing Stage)	**Số lỗi ước lượng sẽ được phát hiện (Estimated Number of Defects to Be Detected)**	**% lỗi sẽ được phát hiện (% of Defects to Be Detected)**	**Cơ sở được dùng để ước lượng (Basis for Estimation)**
Xem xét lại yêu cầu và thiết kế *(Requirements and design review)*	29	20%	Tương tự như các ước lượng đã được làm cho dự án Synergy và cơ sở dữ liệu về khả năng của quy trình. *(Similar project*

			Synergy and PCB)
Xem xét lại mã *(Code review)*	29	20%	Tương tự như các ước lượng đã được làm cho dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Kiểm thử đơn vị *(Unit testing)*	57	40%	Tương tự như các ước lượng đã được làm cho dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Tích hợp và kiểm thử hồi quy *(Integration and regression testing)*	25	17%	Tương tự như các ước lượng đã được làm cho dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Kiểm thử chấp nhận *(Acceptance testing)*	5	3%	Tương tự như các ước lượng đã được làm cho dự án Synergy và cơ sở dữ liệu về khả năng của quy trình.
Tổng số lỗi ước lượng sẽ được phát hiện ***(Total estimated number of defects to be detected)***	**143**	**100%**	

Chiến lược để đạt được các mục tiêu chất lượng (Strategy for Meeting Quality Goals)

Chiến lược (Strategy)	**Lợi ích được mong đợi (Expected Benefits)**
Phòng ngừa lỗi bằng cách sử dụng các hướng dẫn phòng ngừa lỗi (defect prevention guidelines) và quy trình; sử dụng các chuẩn được phát triển trong Synergy để cài đặt mã (coding).	Giảm 10% - 20% tỷ lệ tiêm lỗi và tăng khoảng 2% năng suất.
Nhóm xem xét lại (review) các đặc tả chương trình (program specs) ở những trường hợp sử dụng (use cases) đầu tiên và có lôgic phức tạp. Nhóm xét lại các tài liệu thiết kế (design docs) và những mã (code) đầu tiên được tạo ra.	Cải thiện chất lượng vì nâng cao hiệu quả loại bỏ lỗi tổng thể; một số lợi ích về năng suất bởi vì các lỗi sẽ được phát hiện sớm.
Giới thiệu phương pháp RUP và thực hiện dự án trong các vòng lặp (iterations). Tiến hành phân tích tại các cột mốc (milestone analysis) và phân tích các thực hành phòng ngừa lỗi sau mỗi vòng lặp.	Giảm khoảng 5% tỷ lệ tiêm lỗi và tăng 1% năng suất tổng thể.

Các xem xét lại (reviews)

Thời điểm xem xét lại (Review Point)	**Các sản phẩm công việc (work products) được xem xét lại (Review Item)**	**Kiểu xem xét lại (Type of Review)**
Kết thúc giai đoạn lập kế hoạch cho dự án *(End of project planning)*	• Kế hoạch dự án *(Project plan)* • Thiết lập hệ thống kiểm soát lỗi *(Defect control system set up)* • Thời gian biểu của dự án *(Project schedule)*	Xem xét lại bởi nhóm *(Group review)* Xem xét lại bởi cố vấn chất lượng *(Software quality adviser review)* Xem xét lại bởi cố vấn chất lượng *(Software quality adviser review)*

Kết thúc giai đoạn lập kế hoạch cho dự án	Kế hoạch quản lý cấu hình *(CM plan)*	Xem xét lại bởi nhóm *(Group review)*
Hoàn thành 90% yêu cầu (tức là nên ở cuối vòng lặp thứ 1 của giai đoạn Soạn thảo chi tiết *(End of 90% of requirements (this should be at the end of the first elaboration iteration))*	• Tài liệu đặc tả yêu cầu và phân tích nghiệp vụ *(Business analysis and requirements specification document)* • Danh sách các trường hợp sử dụng *(Use case catalog)*	Xem xét lại bởi nhóm
Hoàn thành 90% thiết kế (tức là nên ở cuối vòng lặp thứ 2 của giai đoạn Soạn thảo chi tiết (Elaboration) *(End of 90% design (this should be at the end of the second elaboration iteration))*	• Tài liệu thiết kế *(Design document)* • Mô hình đối tượng *(Object model)*	Xem xét lại bởi nhóm
Bắt đầu mỗi vòng lặp *(Beginning of each iteration)*	Các kế hoạch của vòng lặp *(Iteration plans)*	Xem xét lại bởi 1 người *(One-person review)*
Kết thúc giai đoạn thiết kế chi tiết *(End of detailed design)*	Các đặc tả của chương trình phức tạp hoặc của các chương trình đầu tiên được tạo ra, bao gồm các ca kiểm thử, các biểu đồ tương tác *(Complex and first-time-generated program specs including test cases, interaction diagrams)*	Xem xét lại bởi nhóm
Sau khi cài đặt mã cho vài chương trình đầu tiên *(After coding of first few programs)*	Mã *(Code)*	Xem xét lại bởi nhóm
After self-testing of a process	Mã *(Code)*	Xem xét lại bởi 1 người
Kết thúc giai đoạn lập kế hoạch kiểm thử đơn vị *(End of unit test plan)*	Kế hoạch kiểm thử đơn vị *(Unit test plan)*	Xem xét lại bởi 1 người
Bắt đầu kiểm thử tích hợp *(Beginning of integration test)*	Kế hoạch kiểm thử tích hợp *(Integration test plan)*	Xem xét lại bởi 1 người

2.10 Kế hoạch quản lý rủi ro (Risk Management Plan)

STT	Các rủi ro (Risks)	Xác suất (Probability)	Ảnh hưởng (Impact)	Nguy cơ rủi ro (Risk Exposure)	Kế hoạch giảm thiểu (Mitigation Plan)
1	Chúng tôi sẽ cần sự hỗ trợ từ kiến trúc sư cơ sở dữ liệu và người quản trị cơ sở dữ liệu của khách hàng.	0.5	8	4	Lập kế hoạch cẩn thận về khoảng thời gian cần đến sự hỗ trợ từ mỗi nhóm này và cung cấp đủ thông báo trước. Có một điều phối viên để làm việc chặt chẽ với các nhóm này.
2	Bởi vì RUP sẽ được sử dụng lần đầu tiên, sự hiểu biết của nhóm về nó thì không đầy đủ.	0.9	3	2.7	Làm việc chặt chẽ với các chuyên gia ở phòng thí nghiệm R&D của Infosys. Giữ liên hệ thường xuyên với khách hàng trong vòng lặp phát triển suốt dự án, và giải quyết nhanh chóng bất cứ vấn đề phát sinh nào liên quan đến lịch và chi phí (nỗ lực). Lập kế hoạch để đào tạo cho nhóm về phương pháp RUP.
3	Mất nhân sự: Các thành viên trong nhóm có thể rời công ty trong thời gian ngắn sắp tới.	0.3	7	2.1	Phân công công việc/nhiệm vụ để mà có nhiều hơn một người có hiểu biết về các unit (đơn vị chương trình) và use case (trường hợp sử dụng) trong dự án.
4	Làm việc với máy tính lớn (mainframe) DB2	0.1	8	0.8	Thực hiện thêm các kỹ thuật tĩnh như: xem xét lại

	của khách hàng qua liên kết: Liên kết này có thể không được hiệu quả như dự kiến.				mã (code review), kiểm tra tại bàn (desk checking), v.v. để giảm thiểu sự phụ thuộc vào liên kết. Giải quyết ngay vấn đề leo thang (escalate) khi liên kết bị hõng (down).

3. THEO DÕI DỰ ÁN (PROJECT TRACKING)

3.1 Kế hoạch đo lường (Measurement Plan)

Số đo sẽ được thu thập (Metric to Be Collected)	**Đơn vị đo lường (Unit of Measurement)**	**Công cụ được sử dụng (Tools Used)**
Kích thước (Size)	Đếm LOC, FP, S/M/C	Bộ đếm số dòng lệnh (Line counters)
Nỗ lực (Effort)	Người-ngày (Person-days)	Hệ Thống Báo Cáo Hoạt Động Hàng Tuần (WAR - *weekly activity report* system)
Lỗi (Defects)	Số lỗi (Number of defects)	BugsBunny
Thời gian hoàn thành (Schedule)	Thời gian đã trải qua (Elapsed time)	Microsoft Project (MSP)

3.2 Theo dõi công việc/nhiệm vụ (Task Tracking)

Hoạt động (Activity)	**Thủ tục (Procedure)**
Xếp lịch cho công việc (Task scheduling)	Lãnh đạo dự án xếp lịch cho các công việc dùng Microsoft Project (MSP). Tinh chỉnh và xếp lại lịch khi cần thiết.
Phân công công việc (Task assignment)	Lịch sau cùng được công bố cho các thành viên nhóm biết. Một khi lịch được nạp (upload) lên hệ thống WAR-MSP, các công việc sẽ trình bày các WARs tương ứng của chúng.
Theo dõi tình trạng (trạng thái) công việc	Theo dõi công việc được làm hàng ngày *(Task tracking is done daily.)*

(Task status tracking)	
Họp dự án *(Project meeting)*	Một lần mỗi tuần *(Once a week)*
Họp để phân tích nguyên nhân *(Causal analysis meeting)*	Sau mỗi vòng lặp *(After every iteration)*

3.3 Theo dõi các vấn đề phát sinh (Issues Tracking)

Kiểu phát sinh (Issue Types)	**Được ghi lại vào đâu? (Where Logged)**	**Ai ghi? (Who Can Log)**	**Ai xem xét lại, khi nào? (Who Reviews, When)**	**Bị leo thang khi nào? (When Escalated)**
Phát sinh tại chỗ *(Onsite issues)*	IssueTracker.xls	Bất kỳ thành viên nào của dự án	Người lãnh đạo dự án (PL), hàng ngày	2 ngày
Phát sinh ở phía khách hàng *(Customer issues)*	Issues Log.xls	Nhóm tại chỗ, người lãnh đạo dự án (Onsite team, PL)	Người lãnh đạo dự án (PL), hàng ngày	2 ngày
Phát sinh ở phía người quản lý kinh doanh *(Business manager issues)*	Báo cáo tình trạng (trạng thái) hàng tuần *(Weekly Status report)*	Người quản lý kinh doanh (BM)	Người quản lý kinh doanh (BM), người lãnh đạo dự án, hàng tuần (BM, PL weekly)	5 ngày
Các phát sinh ở phía các dịch vụ hỗ trợ *(Issues with support services)*	Request Tracker	Bất kỳ thành viên nào của dự án	Hỗ trợ các dịch vụ, hàng ngày	2 ngày

3.4 Phản hồi của khách hàng (Customer Feedback)

Mục (Item)	**Quy trình ghi nhận và theo dõi (Logging and Tracking Process)**
Phản hồi của khách hàng *(Customer feedback)*	Người quản lý quan hệ khoác hàng (AM) hoặc người lãnh đạo dự án (PL) nhận phản hồi của khách hàng. Người quản lý kinh doanh (BM) lưu nó vào tập tin (file).
Các than phiền của khách hàng *(Customer complaints)*	Các than phiền của khách hàng sẽ được nhập và theo dõi bằng cách dùng CustomerComplaints.xls

3.5 Theo dõi chất lượng (Quality Tracking)

Hoạt động chất lượng (Quality Activity)	**Hành động (Action)**
Theo dõi lỗi *(Defect tracking)*	Dùng một Một hệ thống kiểm soát lỗi (DCS) để ghi lại lỗi và theo dõi chúng cho đến khi đóng *(Use DCS for logging defects and tracking them to closure.)*
Xem xét lại (yêu cầu, thiết kế mức cao, thiết kế chi tiết) *(Reviews (requirements, high-level design, detailed design))*	Kiểm tra để đối chiếu với các mục tiêu của dự án trong kế hoạch chất lượng *(Check against project goals in quality plan)*
Xem xét lại mã *(Code review)*	Kiểm tra để đối chiếu với các giới hạn cho mỗi chương trình thông qua công cụ Kiểm soát quy trình dùng thống kê (SPC) *(Check against limits for each program through SPC tool)*
Kiểm thử đơn vị độc lập *(Independent unit testing)*	Kiểm tra để đối chiếu với các giới hạn cho mỗi chương trình thông qua công cụ Kiểm soát quy trình dùng thống kê (SPC)
Kiểm thử tích hợp/Kiểm thử hệ thống *(Integration testing/System testing)*	Kiểm tra để đối chiếu với các mục tiêu của dự án và kế hoạch chất lượng

3.6 Xem xét lại được làm bởi nhà quản lý cấp cao (Review by Senior Management (BM))

STT	Mục được xem xét lại (Item for Review)	Mức độ thường xuyên của việc xem xét lại (Frequency of Review)
1	Lịch *(Schedule)*	Mỗi khi thay đổi phiên bản *(Every version change)*
2	Kế hoạch dự án *(Project plan)*	Khi nào các thay đổi lớn được làm *(When significant changes are made)*
3	Báo cáo tại cột mốc *(Milestone report)*	Kết thúc mỗi cột mốc *(End of milestones)*

3.7 Báo cáo tình trạng (Status Reporting)

Báo cáo đến (Report To)	Mức độ thường xuyên (Frequency)
Người quản lý kinh doanh (Business manager)	Vào thứ Hai hàng tuần qua email (Weekly on Monday by e-mail)
Khách hàng (Customer)	Vào thứ Hai hàng tuần (Weekly on Monday)

3.8 Các giới hạn cho chênh lệch tại các cột mốc (Deviation Limits at Milestones)

Thực tế so sánh với ước lượng của (Actual vs Estimated of)	Cho 5 cột mốc đầu tiên (For the First Five Milestones)	Cho các cột mốc còn lại (For the Rest of the Milestones)
Nỗ lực (Effort)	10%	5%
Tiến độ (Schedule)	10%	5%
Lỗi (Defects)	20%	20%

3.9 Báo cáo cho khách hàng (Report to the Customer)

- Các báo cáo tại các cột mốc và các báo cáo tình trạng (trạng thái) hàng tuần
- Các vấn đề phát sinh cần phải làm rõ

- Leo thang, nếu có

3.10 Báo cáo cho người quản lý kinh doanh (Report to the BM)

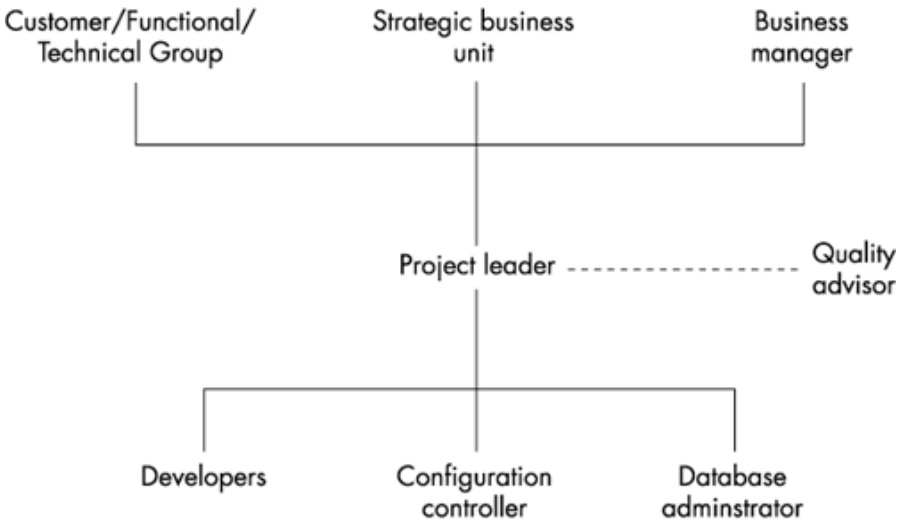
- Ý kiến của khách hàng
- Các báo cáo tại các cột mốc và các báo cáo tình trạng (trạng thái) hàng tuần
- Các vấn đề phát sinh cần phải làm rõ/chú ý
- Leo thang, nếu có
- Số lượng thay đổi yêu cầu và nỗ lực ước lượng (ước tính) cho chúng
- Những thay đổi lớn trong kế hoạch

3.11 Các thủ tục leo thang (Escalation Procedures)

Nơi leo thang (Escalate Where)	**Khoảng thời gian ngưỡng (Threshold Period)**	**Tên người (Name of the Person)**	**Chức vụ (Designation of the Person)**
Ở ACIC	3 ngày	Xxxx	Người quản lý dự án (Project manager)
Ở Infosys	3 ngày	Xxxx	Người quản lý quan hệ khách hàng (Account manager)
Ở Infosys	3 ngày	Xxxx	Người quản lý kinh doanh (Business manager)

4. NHÓM DỰ ÁN (PROJECT TEAM)

4.1 Tổ chức dự án (Project Organization)



4.2 Nhóm dự án (Project Team)

STT	Tên viết tắc (Initials)	Trách nhiệm (Responsibility)	Ngày bắt đầu (Start Date)	Ngày kết thúc dự kiến (Expected End Date)
1	BB	Người quản lý dự án (Project manager)	4 April 2000	3 November 2000
2	KP	Người điều phối tại chỗ (Onsite coordinator)	4 April 2000	3 November 2000
3	BJ	Người lãnh đạo mô-đun (Module leader), người lãnh đạo dự án dự phòng (backup project lead)	15 May 2000	3 November 2000
4	SP	Người kiểm soát cấu hình (Configuration controller)	22 May 2000	13 October 2000
5	DD	Nhà phát triển (Developer)	22 May 2000	29 September 2000
6	HP	Nhà phát triển (Developer), Người kiểm soát cấu hình dự phòng (backup configuration controller)	22 May 2000	29 September 2000

7	NA	Nhà phát triển (Developer)	17 July 2000	3 November 2000
8	SH	Nhà phát triển (Developer)	1 August 2000	15 September 2000
9	AL	Nhà phát triển (Developer)	14 August 2000	31 August 2000
10	JP	Nhà phát triển (Developer)	1 September 2000	22 September 2000
11	SDS	Người quản lý quan hệ khách hàng (Account manager)	4 April 2000	3 November 2000
12	SB	Cố vấn đảm bảo chất lượng (SQA)	15 May 2000	3 November 2000

4.3 Vai trò và trách nhiệm (Roles and Responsibilities)

Vai trò (Role)	**Trách nhiệm (Responsibilities)**
Người quản lý kinh doanh *(BM-Business manager)*	• Giải quyết các phát sinh leo thang *(Resolve escalated issues)* • Xem xét lại tình trạng (trạng thái) dự án *(Review project status)* • Tham gia vào trong các xem xét lại dùng kỹ thuật tới hạn *(Participate in critical technical reviews)*
Khách hàng *(Customer)*	• Xem xét lại thiết kế *(Review design)* • Giải quyết các phát sinh leo thang *(Resolve escalated issues)* • Lập kế hoạch kiểm thử chấp nhận và kiểm thử *(Acceptance test planning and testing)*
Người quản lý quan hệ với khách hàng *(AM-Account manager)*	• Thỏa mãn khách hàng *(Customer satisfaction)* • Phát triển kinh doanh *(Business growth)* • Kế hoạch về tài chính cho dự án *(Project financial plan)* • Giao diện với việc bán hàng và tiếp thị *(Interface with sales and marketing)* • Các phát sinh liên quan đến đào tạo *(Training-related issues)* • Các phát sinh liên quan đến nhân viên *(Employee-related issues)*
Người quản lý dự án *(PM- Project manager)*	• Lập kế hoạch cho dự án và thời gian biểu *(Project planning and scheduling)* • Thiết kế *(Design)*

	• Tương tác với khách hàng *(Customer interaction)* • Xem xét lại (Reviews) • Kiểm thử (Testing) • Báo cáo (Reporting) • Phân công và theo dõi công việc *(Task assignment and tracking)* • Tương tác với Cố vấn đảm bảo chất lượng từ nhóm quy trình công nghệ phần mềm *(Interact with software quality adviser from SEPG)* • Đảm bảo giao sản phẩm đúng như hợp đồng *(Ensure delivery as per contract)* • Tương tác với các phòng ban khác khi có nhu cầu *(Interface with other departments as per need)* • Đảm bảo các vấn đề phát sinh/than phiền của khách hàng được giải quyết thỏa đáng *(Ensure open issues/customer complaints are closed properly)* • Đảm bảo các thành viên của dự án được đào tạo đầy đủ *(Ensure project members are adequately trained)*
Người lãnh đạo mô-đun *(ML-Module leader)*	• Thiết kế *(Design)* • Phát triển *(Development)* • Kiểm thử *(Testing)* • Báo cáo *(Reporting)*
Nhóm ngăn chặn lỗi (DP) *(DP-Defect prevention team)*	• Nhận thức về sự lan truyền lỗi trong nhóm và ngăn chặn chúng *(Spread awareness in the team on defects and their prevention)* • Phân tích dữ liệu về lỗi *(Analyze defect data)* • Xác định các phương pháp để giảm tỷ lệ tiêm lỗi *(Identify methods to reduce defect injection)*
Nhà phát triển *(DV-Developer)*	• Thiết kế chi tiết các trường hợp sử dụng *(Detail design for use cases)* • Phát triển *(Development)* • Kiểm thử đơn vị và kiểm thử tích hợp *(Unit testing and integration testing)*
Người kiểm soát cấu hình	• Chuẩn bị kế hoạch quản lý cấu hình *(Prepare the CM plan)*

(CC-Configuration controller)	• Quản lý cấu hình là một phần của kế hoạch quản lý cấu hình *(Manage the configuration as per the CM plan)*
Cố vấn đảm bảo chất lượng (SQA) từ nhóm quy trình công nghệ phần mềm (SEPG)	• Tư vấn về quy trình *(Process consultancy)* • Đảm bảo chất lượng (kiểm toán) *(Quality assurance (audits))* • Cài đặt các công cụ đo lường và đào tạo cho nhân viên trong dự án *(Install measurement tools and train project personnel)* • Tham gia vào xem xét lại kế hoạch và các quy trình khi cần thiết *(Participate in reviews of project plan and processes as necessary)*
Người điều phối tại chỗ *(Onsite coordinator)*	• Giải quyết bất kỳ vấn đề phát sinh nào từ khách hàng/phát triển từ nước ngoài *(Resolve any issues from customer/offshore)* • Hỗ trợ xuyên suốt giai đoạn phát triển *(Support during development)*

5. CÁC THAM KHẢO

Bỏ qua.

6. CHỮ VIẾT TẮT ĐƯỢC DÙNG

Bỏ qua.

8.5 TÓM TẮT

Thật là quan trọng để những nhà quản lý dự án thực hiện lập kế hoạch cho nhóm và quản lý giao tiếp và chuẩn bị một tài liệu quản lý dự án toàn diện cuối cùng.

Sau đây là các bài học quan trọng từ chương này:

- Dự án phần mềm là một nỗ lực của nhóm. Trong việc quản lý nhóm, bạn phải xem xét các mục tiêu của dự án cũng như các mục tiêu của từng thành viên.

- Có giao tiếp thường xuyên trong phạm vi nhóm và với khách hàng. Lập kế hoạch rõ ràng cho các kênh leo thang ở cả hai phía để giải quyết các vấn đề phát sinh rất khó giải quyết.

- Lập tài liệu cho kết quả của các công việc khác nhau trong kế hoạch tổng thể. Tài liệu này phải được xem xét lại để xác định các lỗi tiềm tàng và để tận dụng kinh nghiệm của các nhà quản lý cấp cao (senior managers) để cải tiến nó.

- Kế hoạch được lập tài liệu (documented plan) phải chứa các thông tin về dự án, quy trình được sử dụng, phương pháp quản lý các thay đổi, nỗ lực được ước lượng, các cột mốc, các kế hoạch quản lý rủi ro, các kế hoạch chất lượng, theo dõi dự án và các kế hoạch báo cáo, các kênh leo thang để giải quyết các vấn đề phát sinh vượt ra ngoài dự án và nhóm.

Tương ứng với CMM, phải lập tài liệu cho kế hoạch quản lý dự án là một yêu cầu của KPA Lập Kế Hoạch Cho Dự Án (Project Planning) và KPA Theo Dõi và Giám Sát Dự Án (Project Tracking and Oversight) ở CMM mức 2. Các phương pháp được thảo luận trong chương này và các nội dung của kế hoạch quản lý dự án cũng thỏa mãn nhiều yêu cầu của KPA Quản Lý Dự Án Tích Hợp (Integrated Project Management) và KPA Phối Hợp Giữa Các Nhóm (Intergroup Coordination) ở CMM mức 3, hai KPA ở CMM mức 4, và một số khía cạnh của KPA Đào tạo (Training) ở CMM mức 3.

Chương 9. Quản lý cấu hình

Các thay đổi - do sự tiến triển (evolution) của các sản phẩm công việc (work products) và các thay đổi yêu cầu – diễn ra liên tục trong một dự án phần mềm. Tất cả những thay đổi này cuối cùng sẽ được thể hiện trong các tập tin có chứa mã nguồn, dữ liệu, hoặc các tài liệu. Khi nhiều người tạo ra và thay đổi một số lượng lớn các tập tin trong một dự án phần mềm, điều này có thể dẫn đến sự phức tạp trừ khi các thay đổi được kiểm soát đúng đắn. Hãy xem xét những tình huống này, được rút ra từ các dự án khác nhau.

- Khách hàng đã yêu cầu hai thay đổi khác nhau. Người quản lý dự án đã phân công Rao thực hiện một yêu cầu, và cái còn lại phân cho Meera. Cả hai đã phải sửa đổi mã cho mô-đun X. Khi Meera hoàn thành sửa đổi của mình, cô lưu trữ một tập tin cho X, vô tình ghi đè lên các thay đổi Rao đã thực hiện trước đó một ngày.

- Thứ Sáu đã là thời hạn (deadline) để giao một mô-đun mà nhóm gồm ba thành viên đang phát triển mã. Việc tích hợp mã (đã được kiểm thử đơn vị) đã được lên kế hoạch cho hai ngày cuối cùng. Vào đêm thứ Ba, Subbu - nhà phát triển của một số chức năng then chốt - rời thành phố để tham dự một công việc khẩn cấp. Ngày hôm sau, người lãnh đạo mô-đun và các thành viên của nhóm nghiên cứu đã dành nhiều giờ tìm kiếm các tập tin của Subbu. Chúng được quản lý để xác định một số tập tin đang chứa các chức năng mà Subbu đang phát triển, nhưng họ đã tìm thấy nhiều phiên bản của các tập tin này. Một trong những thành viên trong nhóm đã phải làm việc về vấn đề này suốt cuối tuần. Bắt đầu từ một số phiên bản của các chương trình của Subbu, anh ta đã phát triển và kiểm thử đơn vị, làm lại hầu hết các công việc mà Subbu gần như đã hoàn tất trước khi rời khỏi. Mô-đun cuối cùng đã bị giao trễ ba ngày.

- Nhóm của Srinath đã có tinh thần làm việc tốt. Họ đã hoàn thành việc phát triển đúng hạn, và kiểm thử cuối cùng đã cho thấy không có lỗi. Phần mềm này được giao cho khách hàng. Nhưng ngày hôm sau, Srinath nhận được e-mail tức giận từ những người sử dụng và khách hàng, báo về các vấn đề trong phần mềm. Sau nỗ lực điên cuồng của nhóm làm việc, nguyên nhân đã được tìm thấy: Phiên bản phát hành của phần mềm đã chứa một phiên bản cũ của một thành phần (component) then chốt.

Những câu chuyện như thế này có thể được tìm thấy trong hầu hết các công ty. Quản lý cấu hình (*CM - Configuration management*) - còn được gọi là quản lý cấu hình phần mềm (SCM - software configuration management) - là một khía cạnh của quản lý dự án để tập trung vào việc kiểm soát có hệ thống các thay đổi để mà những vấn đề như bên trên không xảy ra.

Chương này thảo luận về một số khái niệm liên quan đến quản lý cấu hình (CM) và sau đó mô tả quy trình quản lý cấu hình được sử dụng tại Infosys. Lập kế hoạch để quản lý cấu hình (CM plan) cũng được trình bày qua case study.

9.1 CÁC KHÁI NIỆM TRONG QUẢN LÝ CẤU HÌNH (CONCEPTS IN CONFIGURATION MANAGEMENT)

CM là việc làm cần thiết để đáp ứng một trong những mục tiêu cơ bản của dự án: giao sản phẩm phần mềm có chất lượng cao cho khách hàng. "Phần mềm" được giao phải như thế nào? Ít nhất, nó chứa các mã nguồn khác nhau hoặc các tập tin đối tượng để tạo ra mã nguồn (source) hoặc mã đối tượng (object code), các đoạn mã ngắn (scripts) để xây dựng nên hệ thống làm việc từ những tập tin này, và các tài liệu liên quan. Trong suốt dự án, các tập tin sẽ thay đổi, dẫn đến việc tạo ra các phiên bản khác nhau. Trong tình huống này, làm thế nào để một người quản lý chương trình đảm bảo rằng các phiên bản thích hợp của các tập tin mã nguồn (source code files) được kết hợp lại mà không làm bỏ sót bất kỳ mã nguồn nào, và rằng các phiên bản đúng của các tài liệu, nhất quán với mã nguồn cuối cùng, đã được gửi? Tất cả điều này sẽ được đảm bảo thông qua quản lý câu hình (CM) thích hợp.

Một mục tiêu chính của CM là quản lý cấu hình đang tiến triển (evolving) của phần mềm system [1][2]. Trong một dự án, một sự tiến triển của một chương trình trải qua nhiều trạng thái (states). Lúc bắt đầu, khi một lập trình viên phát triển nó, chương trình có trnagj thái là “đang được phát triển” (under development) hoặc "riêng" (private). Một khi lập trình viên hài lòng với chương trình, nó di chuyển sang trạng thái "sẵn sàng để thử thử đơn vị" (ready for unit testing). Chỉ khi chương trình đạt đến trạng thái này, nó mới được kiểm thử đơn vị. Sau khi nó vừa được kiểm thử đơn vị xong, lập trình viên phải sửa chữa bất kỳ lỗi đã được tìm thấy. Tuy nhiên, nếu kiểm thử đơn vị thành công, trạng thái của chương trình sẽ thay đổi sang “sẵn sàng để kiểm thử hệ thống" (ready for system testing). Chỉ khi nào tất cả các chương trình đạt được trạng thái này, việc kiểm thử hệ thống mới có thể bắt

đầu. Một lần nữa, nếu lỗi (defect) được tìm thấy trong quá trình kiểm thử hệ thống, trạng thái của chương trình sẽ trở lại là "riêng" (private); ngược lại, nó di chuyển đến trạng thái "sẵn sàng để kiểm thử chấp nhận" (ready for acceptance testing). Nếu kiểm thử chấp nhận thành công, trạng thái của tất cả các chương trình thay đổi thành "sẵn sàng để phát hành" (ready for release), ngụ ý rằng chúng có thể sẽ được phát hành cho "sử dụng." Khi một chương trình được phát hành và được sử dụng, tất cả các chương trình (và các tài liệu liên quan) chuyển sang trạng thái "baselined" để trình bày trạng thái của hệ thống sử dụng.

Ngoài những thay đổi diễn ra trong suốt quá trình phát triển bình thường, các yêu cầu thay đổi yêu cầu có thể được gửi, và việc thực hiện chúng có thể làm thay đổi các chương trình. Khi một dự án có một số lượng lớn các mục (item) có thể thay đổi, các nhà phát triển có thể được phân công để thực hiện nhiều hoạt động; những hoạt động này có thể được thực hiện nếu và chỉ nếu luôn luôn có sự hỗ trợ hợp lý từ quy trình CM.

Để hiểu rõ hơn về CM, chúng ta hãy xem xét các loại chức năng của CM mà các dự án yêu cầu. Mặc dù các yêu cầu này có thể phụ thuộc vào tính tự nhiên và hoàn cảnh cụ thể của dự án, nhưng một số chức năng chung có thể được xác định rõ. Sau đây là danh sách của các chức năng này cùng kèm theo các kịch bản cần đến chúng. Các chức năng này thì chi tiết hơn các chức năng CM được định nghĩa bởi Humphrey [3].

- Cho biết các trạng thái của các chương trình. Bạn cần thông tin này để quyết định khi nào bắt đầu kiểm thử hoặc khi nào giao (phát hành) phần mềm.

- Cho biết phiên bản mới nhất của một chương trình. Giả sử rằng một chương trình phải được sửa đổi. Rõ ràng rằng, việc sửa đổi phải được thực hiện trong phiên bản mới nhất; nếu không, các thay đổi trước đó có thể bị mất.

- Xử lý các yêu cầu cập nhật (update) đồng thời. Hai lập trình viên, tương ứng với hai yêu cầu thay đổi khác nhau, có thể thay đổi đồng thời cùng một chương trình. Một trong những thay đổi có thể có khả năng ghi đè lên các thay đổi khác. Để tránh tình trạng như vậy cần phải kiểm soát truy cập (access control) để chỉ có một người có thể làm thay đổi một chương trình tại một thời điểm. Nếu nhiều thay đổi song song được cho phép, các thủ tục hòa giải cần được thực hiện để đảm bảo rằng tất cả các thay đổi được phản ánh trong phiên bản cuối cùng (final version).

- Hủy bỏ thay đổi một chương trình (undo a program change). Một thay đổi được thực hiện lên một chương trình (để thực hiện một yêu cầu thay đổi), nhưng sau đó lại có nhu cầu phát sinh để đảo ngược (reverse) việc thay đổi yêu cầu này.

- Ngăn chặn các thay đổi hoặc xóa trái phép. Một lập trình viên có thể quyết định thay đổi một số chương trình, chỉ để khám phá ra rằng sự thay đổi đó có tác dụng phụ. Cần có cơ chế kiểm soát truy cập (access control mechanisms) để mà các thay đổi trái phép sẽ không được chấp thuận.

- Cung cấp khả năng theo dõi dấu vết nguồn gốc giữa các thay đổi yêu cầu (traceability between requirement change requests) và các thay đổi chương trình (program changes). Giả sử một yêu cầu thay đổi yêu cầu (requirement change request) ra lệnh rằng ba chương trình phải được sửa đổi, và những sửa đổi này đã được phân công cho ba thành viên trong nhóm. Làm thế nào để người quản lý dự án đảm bảo rằng yêu cầu thay đổi đã được thực hiện đúng cách – tức là, tất cả các chương trình đã được thay đổi và các chương trình vừa được thay đổi xong đã trải qua chu kỳ sống của chúng và đang ở trạng thái "sẵn sàng để phát hành"? Để trả lời câu hỏi này đòi hỏi phải có một cơ chế để theo dõi các yêu cầu thay đổi, cơ chế này có thể xác định tất cả các chương trình được thay đổi cũng như trạng thái của từng chương trình.

- Hủy bỏ thay đổi yêu cầu (undo a requirement change). Một thay đổi yêu cầu cần được thực hiện (bằng cách phải thay đổi nhiều chương trình), nhưng sau đó lại cần phải hủy bỏ thay đổi vừa được thực hiện (undo) (có lẽ do người sử dụng không thích các tính năng mới).

- Chỉ ra các thay đổi có liên quan. Giả sử một lỗi được tìm thấy trong một chương trình, và nó bị nghi ngờ rằng lỗi này đến từ việc thực hiện các yêu cầu thay đổi. Do đó, có mong muốn để xem xét lại tất cả các thay đổi đã được thực hiện cho yêu cầu thay đổi này.

- Thu thập tất cả các mã nguồn, tài liệu, và các thông tin khác cho hệ thống hiện tại. Chúng như một kết quả của việc sửa đổi làm sai lạc tập tin (file corruption) hay sự sụp đổ hệ thống (system crash), cần thiết để phục hồi lại tất cả các tập tin. Tương

tự như vậy, một thay đổi lên một hệ thống hiện có (một trong đó là quá trình hoạt động (operation)) có thể được cần, vì thế mà cần thiết để có được tất cả các tập tin nguồn và các tài liệu trình bày cho hệ thống hiện tại.

Đây là một vài trong số những kịch bản thường xuyên xảy ra hơn trong một dự án đang cần đến sự hỗ trợ từ quy trình CM. Hơn nữa, nếu nhiều phiên bản khác nhau của một sản phẩm phần mềm cùng tồn tại, mỗi phiên bản lại sử dụng một phiên bản khác nhau của các chương trình, các tình huống khác liên quan đến các thay đổi có thể dẫn đến đòi hỏi rằng CM phải có một chức năng bổ sung (ví dụ, xử lý sự thay đổi (handling variance) [4]). Tuy nhiên, mục đích chính của CM là để cung cấp các cơ chế xử lý để xử lý các kiểu kịch bản trong danh sách đã được trình bày bên trên. Những cơ chế này bao gồm:

- Quy ước để đặt tên và tổ chức các tập tin (conventions for naming and organization of files)
- Kiểm soát phiên bản (version control)
- Khả năng theo dõi dấu vết nguồn gốc yêu cầu thay đổi (change request traceability)
- Kiểm soát truy cập (access control)
- Các thủ tục hòa giải (reconciliation procedures)
- Các chương trình đăng nhập thay đổi (modification login programs)

Cách đặt tên (naming) các tập tin chương trình (program files) và các tập tin tài liệu (document files) theo một quy ước chuẩn và lưu giữ các tập tin vào trong các thư mục cụ thể để giúp cho việc tìm kiếm một tập tin mong muốn được nhanh chóng hơn. Đặt tên thích hợp (ví dụ, bằng cách sử dụng các phần mở rộng chuẩn) cũng giúp các nhà phát triển để dễ dàng hiểu được bản chất của nội dung tập tin mà không cần nhìn vào các tập tin. Ngoài ra, đặt cách ly các chương trình dựa theo trạng thái của chúng vào trong các thư mục riêng biệt giúp các nhà phát triển xác định được trạng thái của chương trình một cách dễ dàng.

Kiểm soát phiên bản (version control) là một vấn đề then chốt của CM [1][2][4] và hiện có nhiều công cụ có thể giúp quản lý nhiệm vụ này. Kiểm soát phiên bản giúp bảo tồn các

phiên bản cũ của các chương trình bất cứ khi nào chúng bị thay đổi. Nếu không có cơ chế như vậy, hệ thống không thể hỗ trợ được nhiều cho các chức năng mà CM yêu cầu.

Một cơ chế để theo dõi dấu vết nguồn gốc yêu cầu thay đổi (change request traceability mechanism) cung cấp một ánh xạ (mapping) từ một yêu cầu thay đổi yêu cầu (requirement change request) đến các thay đổi tiếp theo trong các chương trình, và giúp quản lý các thay đổi yêu cầu. Để lần ngược trở lại một thay đổi yêu cầu thay đổi (to trace a change back to the change request), việc ghi nhận lại các sửa đổi thì hữu ích.

Các cơ chế kiểm soát truy cập (access control mechanisms) đảm bảo rằng chỉ những người được phép (authorized people) mới có thể sửa đổi các tập tin nào đó và chỉ có một người có thể sửa đổi một tập tin tại một thời điểm nhất định. Các thủ tục hòa giải (reconciliation procedures) xác định rõ làm thế nào để hai thay đổi được thực hiện độc lập lên một chương trình có thể được trộn lại (merged) để tạo ra một phiên bản mới phản ánh cả hai.

Nếu các cơ chế này được cung cấp, các kịch bản được đưa ra trước đây có thể được xử lý một cách thỏa đáng. Một vài trong số các kịch bản này đòi hỏi phải sử dụng nhiều hơn một cơ chế. Ví dụ, việc hủy bỏ một thay đổi yêu cầu (undo a requirement change) sẽ có liên quan đến một cơ chế để chỉ ra (show) dấu vết nguồn gốc của một thay đổi yêu cầu cho các thay đổi tiếp theo trong các chương trình, và một cơ chế kiểm soát phiên bản để thực sự hủy bỏ các thay đổi (actually undo the changes).

Một số cơ chế CM có thể được hỗ trợ bởi công cụ (tool), nhưng ngược lại nhiều cơ chế khác có thể cần người sử dụng phải thực hiện chúng một cách rõ ràng. Ví dụ, kiểm soát phiên bản có thể được thực hiện bởi một công cụ, nhưng việc đoạt được (capture) trạng thái của một chương trình có thể cần lập trình viên lưu giữ (maintain) thông tin này một cách rõ ràng. Quy trình CM xác định tất cả các bước cần thiết để thực hiện các cơ chế như vậy và giải thích làm thế nào các cơ chế này được sử dụng trong một dự án.

Các cuộc thảo luận cho đến nay đã tập trung vào các chương trình. Các tài liệu được tạo ra trong một dự án (ví dụ như các tài liệu yêu cầu, các tài liệu thiết kế, và các kế hoạch) cũng cần quản lý cấu hình. Trong suốt quá trình phát triển bình thường của một dự án, một tài liệu trải qua ba trạng thái: "đang phát triển" (under development), "đang được xem xét lại" (under review), và "baselined". Việc chuyển trạng thái thì đơn giản như một đề xuất

được cho bởi Whitgift [4]. Quy trình CM (CM process) cũng phải cài đặt biểu đồ trạng thái (state diagram) cho các tài liệu.

9.2 QUY TRÌNH QUẢN LÝ CẤU HÌNH (CONFIGURATION MANAGEMENT PROCESS)

Quy trình CM xác định một trình tự các hoạt động phải được thực hiện dưới sự hỗ trợ của các cơ chế CM. Như với hầu hết các hoạt động trong quản lý dự án, giai đoạn đầu tiên trong quy trình CM tại Infosys là lập kế hoạch (planning) - xác định những mục (items) cần thiết cho CM (được gọi là các mục cấu hình - *configuration items*), các nơi (locations) để lưu trữ chúng, các thủ tục kiểm soát thay đổi, v.v. Người quản lý dự án hoặc người kiểm soát cấu hình (CC – configuration control) của dự án sẽ chuẩn bị kế hoạch này. Sau đó, quy trình này phải được thực thi, có lẽ bằng cách triển khai một công cụ (việc sử dụng nó cũng phải được lập kế hoạch). Cuối cùng, bởi vì bất kỳ kế hoạch CM nào cũng cần các quy tắc (discipline) cho các nhân viên của dự án – là những người làm các công việc liên quan tới bảo trì các phiên bản, lưu trữ các mục vào các nơi thích hợp, và thực hiện thay đổi một cách đúng đắn, theo dõi tình trạng (trạng thái) của các mục cấu hình và thực hiện kiểm tra CM (CM audits) là những hoạt động khác trong quy trình CM. Chương này thảo luận về ba hoạt động của quy trình CM này tại Infosys.

9.2.1 Lập kế hoạch và thiết lập quản lý cấu hình (planning and setting up configuration management)

Lập kế hoạch quản lý cấu hình liên quan đến việc xác định các mục cấu hình (configuration items) và xác định các thủ tục (procedures) được sử dụng để kiểm soát và thực hiện các thay đổi cho chúng. Xác định các mục cấu hình là một hoạt động cơ bản trong bất kỳ kiểu CM nào [2][3][4]. Các ví dụ điển hình về các mục cấu hình bao gồm các đặc tả yêu cầu (requirements specifications), các tài liệu thiết kế (design documents), mã nguồn (source code), các kế hoạch kiểm thử (test plans), các đoạn chương trình dùng để kiểm thử (test scripts), các thủ tục kiểm thử (test procedures), dữ liệu kiểm thử (test data), các tiêu chuẩn được sử dụng trong dự án (chẳng hạn như các tiêu chuẩn cài đặt mã (coding standards) và các tiêu chuẩn thiết kế (design standards)), kế hoạch chấp nhận (acceptance plan), các tài liệu như kế hoạch CM và kế hoạch dự án, tài liệu người dùng như hướng dẫn sử dụng (user manual), và các tài liệu như tài liệu đào tạo (training

material), tài liệu hợp đồng (contract documents) (bao gồm các công cụ hỗ trợ như trình biên dịch hoặc công cụ do công ty tự phát triển), các bản ghi chất lượng (quality records) (bản ghi về xem xét lại, bản ghi về kiểm thử), và các bản ghi quản lý cấu hình (CM records) (bản ghi về phát hành (release records), bản ghi theo dõi tình trạng (status tracking records)). Bất kỳ sản phẩm nào được cung cấp bởi khách hàng (customer-supplied products) hoặc các mục được mua (purchased items) từ bên ngoài mà chúng sẽ là một phần của sản phẩm được giao (được gọi là "sản phẩm phần mềm được bao gồm") cũng được xem là các mục cấu hình.

Trong suốt quá trình lập kế hoạch, các loại mục nằm dưới sự bảo vệ (aegis) của CM sẽ được xác định, nhưng một danh sách chi tiết các hạng mục sẽ không được chuẩn bị trước. Thiếu sót này phản ánh thực tế rằng một số mục có thể không được biết đến trong suốt quá trình lập kế hoạch CM (xảy ra vào lúc bắt đầu dự án). Để tạo dễ dàng cho việc đặt tên phù hợp cho các mục cấu hình, quy ước đặt tên cho các mục CM được thiết lập trong giai đoạn lập kế hoạch CM. Ngoài các tiêu chuẩn đặt tên (naming standards), những người quản lý dự án phải lập kế hoạch đánh số phiên bản (version numbering). Khi một mục cấu hình bị thay đổi, các mục cũ không được thay thế bằng bản sao mới; thay vào đó, bản sao cũ được duy trì (maintained) và một cái mới được tạo ra. Điều này tạo ra nhiều phiên bản của cùng một mục, do đó, các quy ước gán số phiên bản (version number assignment) được cần đến. Nếu một công cụ CM được sử dụng, đôi khi công cụ này quản lý việc đánh số phiên bản. Ngược lại, nó phải được quản lý rõ ràng trong dự án.

Trong suốt quá trình lập kế hoạch, người quản lý dự án phải quyết định làm thế nào để lưu giữ trạng thái của một chương trình. Một cách để tập trung các mục lại theo các trạng thái khác nhau là tạo ra các thư mục riêng biệt cho chúng. Tất cả các mục đang ở một trạng thái nhất định sẽ nằm trong cùng một thư mục cho trạng thái đó. Khi trạng thái của một chương trình thay đổi, chương trình được di chuyển từ thư mục cho các trạng thái cũ sang thư mục cho trạng thái mới. Đây là một tiếp cận tổng quát và không cần sử dụng bất kỳ công cụ nào để lưu giữ thông tin trạng thái (state information). Tuy nhiên, nếu sử dụng một công cụ CM sẵn có, cấu trúc thư mục (directory structure) - được cần đến để quản lý trạng thái của các chương trình – sẽ phụ thuộc vào công cụ này. Trong suốt giai đoạn lập kế hoạch, những người quản lý dự án phải thiết lập cấu trúc thư mục để dùng nó quản lý các trạng thái, và luôn ghi nhớ (trong đầu) các đòi hỏi của công cụ CM, nếu có thể.

Người kiểm soát cấu hình (configuration controller) hoặc người quản lý dự án sẽ thực hiện lập kế hoạch CM. Nó chỉ bắt đầu khi dự án đã được khởi xướng và môi trường hoạt động và các đặc tả yêu cầu được lập tài liệu rõ ràng. Các hoạt động trong giai đoạn này bao gồm:

- Xác định các mục cấu hình, bao gồm cả các mục do khách hàng cung cấp (customer-supplied) và các mục được mua (purchased items).
- Định nghĩa một cơ chế để đặt tên và đánh số (naming and numbering scheme) cho các mục cấu hình (configuration items).
- Định nghĩa cấu trúc thư mục (directory structure) cần thiết cho CM.
- Định nghĩa các hạn chế truy cập (access restrictions).
- Định nghĩa các thủ tục kiểm soát thay đổi (change control procedures).
- Xác định và định nghĩa trách nhiệm và thẩm quyền của người kiểm soát cấu hình (CC- Configuration Controler) hoặc Bảng Kiểm Soát Cấu Hình (CCB - Configuration Control Board).
- Xác định một phương pháp để theo dõi trạng thái (status) của các mục cấu hình.
- Định nghĩa một thủ tục để sao lưu phòng hờ (backup procedure).
- Định nghĩa một thủ tục hòa giải (reconciliation procedure), nếu cần thiết.
- Định nghĩa một thủ tục phát hành (giao) sản phẩm (release procedure).
- Định nghĩa một thủ tục lưu trữ (archival procedure).
- Xác định các điểm mà tại đó các mục cấu hình sẽ được chuyển đến baseline (đường cơ sở).

Đầu ra của giai đoạn này là kế hoạch CM. Người kiểm soát cấu hình (CC) là người chịu trách nhiệm cho việc thực hiện kế hoạch CM. Tùy thuộc vào kích thước của hệ thống được phát triển, vai trò của CC có thể là một công việc bán thời gian (part-time job) hoặc

toàn thời gian (full-time job). CC cũng có thể chịu trách nhiệm quản lý việc phát hành, lưu trữ phát hành, truy tìm và phát hành các phiên bản thích hợp khi cần thiết, và nhiều hơn nữa.

Trong một số trường hợp nhất định - khi có các nhóm lớn hoặc khi hai hoặc nhiều nhóm hơn cùng tham gia phát triển cùng một phần hoặc các phần khác nhau của phần mềm hoặc các hệ thống giao diện (interfacing systems) – thì cần thiết để có một Bảng Kiểm Soát Cấu Hình (CCB). Bảng này chứa các biểu diễn từ các nhóm. Một CCB (hoặc một CC) là cần thiết cho quản lý cấu hình (CM) [3], và kế hoạch CM phải định nghĩa rõ ràng các vai trò và các trách nhiệm của CC hoặc CCB. Những trách nhiệm này cũng dựa trên kiểu của hệ thống tập tin (file system) và các công cụ đang được sử dụng.

Như đã đề cập trước đây, CM yêu cầu truy cập vào một số mục ở một số trạng thái bị hạn chế. Ví dụ, việc truy cập và quyền của các lập trình viên để sửa đổi một chương trình trong baseline phải bị hạn chế. Do đó, các hoạt động lập kế hoạch phải xác định các quyền truy cập (access rights) của CC, người quản lý dự án, và các nhà phát triển.

Các chính sách và thủ tục để kiểm soát thay đổi (change control) cũng được thiết lập trong suốt quá trình lập kế hoạch. Điều này bao gồm việc kiểm soát các thay đổi bình thường diễn ra trong suốt vòng đời cũng như các thay đổi bị lèo lái bởi các thay đổi yêu cầu (requirement changes). Các thay đổi bình thường thường được quản lý thông qua một cơ chế thư viện (library mechanism) và cấu trúc thư mục (directory structure). Các thay đổi do yêu cầu thay đổi yêu cầu (requirement change requests), có thể làm nhiều chương trình bị thay đổi, thường xuyên được theo dõi thông qua một bảng (spreadsheet) liệt kê tất cả các mục phải được thay đổi cũng như các thư mục cho từng mục (do đó cho biết trạng thái của nó).

Nếu những người quản lý dự án cho phép các cập nhật đồng thời vào các chương trình, họ phải xác định các thủ tục hòa giải. Cập nhật đồng thời đôi khi là cần thiết. Ví dụ, giả sử một yêu cầu thay đổi có độ ưu tiên cao đến cùng lúc đang thực hiện một yêu cầu thay đổi khác. Rõ ràng, yêu cầu thay đổi mới không thể được đưa vào cho đến khi sự thực hiện thay đổi kia được hoàn thành. Tương tự như vậy, nếu có các vấn đề xảy ra trong hệ thống đang vận hành (working system) trong lúc một yêu cầu thay đổi đang được thực hiện, khi đó các thay đổi phải được thực hiện ngay lập tức để đảm bảo rằng hệ thống có thể tiếp tục hoạt động. Đối với những tình huống như vậy, các thủ tục hòa giải là cần thiết.

Một thủ tục có thể thực hiện được là: các điểm khác nhau giữa phiên bản gốc và các phiên bản mới sẽ được kiểm tra, và những thay đổi trong phiên bản đang có ít thay đổi hơn sẽ được trộn vào trong phiên bản kia. Nếu các thay đổi ảnh hưởng đến các bộ phận khác nhau của chương trình, việc trộn thì đơn giản, một số công cụ CM sẵn sàng hỗ trợ khả năng này. Ngược lại, lập trình viên phải xem xét lại sự chồng chéo, và sau đó điều chỉnh lại cho hợp lý cả hai thay đổi.

Tất cả các yếu tố của kế hoạch điều được lập tài liệu trong kế hoạch CM. Phần sau của chương này, chúng tôi trình bày về kế hoạch CM cho dự án ACIC.

9.2.2 Thực hiện kiểm soát cấu hình (Perform configuration control)

Mặc dù các hoạt động kiểm soát cấu hình được thực hiện trong suốt giai đoạn thực hiện dự án, nhưng chúng tôi thảo luận về chúng ở đây cùng với các khía cạnh khác của CM. Hai hoạt động kiểm soát cấu hình chính được thực hiện: 1) liên quan tới quản lý quá trình chuyển đổi trạng thái (state transition management) của các chương trình (và các tài liệu), và 2) liên quan tới quản lý các yêu cầu thay đổi (managing the change requests) phải được thực hiện.

Quản lý các quá trình chuyển đổi trạng thái liên quan đến việc di chuyển các mục từ một thư mục này sang một thư mục khác khi có các chuyển đổi trạng thái và sau đó tạo ra các phiên bản mới khi các thay đổi được thực hiện.

Thông thường, các công cụ này được sử dụng để quản lý các trạng thái và các phiên bản của các mục và sự truy cập vào chúng. Nhiều công cụ CM sử dụng các thủ tục đưa-vào/tách-ra (check-in/check-out) để kiểm soát truy cập (control access) và kiểm soát phiên bản (version control). Ý tưởng cơ bản đằng sau phương pháp này là như sau. Một chương trình được coi là đang ở trong một *môi trường có kiểm soát (controlled environment)* khi nó ở bất cứ trạng thái nào mà những người khác vẫn có thể sử dụng được nó. Một khi một chương trình đang ở trong một môi trường có kiểm soát, nó không thể bị sửa đổi, thậm chí bởi tác giả ban đầu, mà không có sự cho phép (*authorization*), bởi vì những người khác có thể đang sử dụng nó. Để làm một thay đổi đã được phê duyệt, nhà phát triển phải tách (check-out) chương trình ra khỏi mỗi môi trường có kiểm soát.

Nghĩa là cần tạo ra một bản sao (copy) của một mục (item) mà không làm phá hủy phiên bản trước đó, và làm một ghi nhận là chương trình vừa được tách ra.

Một mục bị sửa đổi sau khi nó đã được tách ra. Các thay đổi sau đó phải được phản ánh trong môi trường có kiểm soát để mà những người khác đạt được lợi ích của phiên bản mới và để mà yêu cầu thay đổi có thể được cài đặt (thực hiện) thực sự. Bởi vì các thành viên khác của nhóm có thể cũng đang sử dụng các mục, một số kiểm tra phải được thực hiện để đảm bảo rằng các mục bị thay đổi là hợp lý trước khi đưa nó trở lại vào môi trường (check-in). Khi một mục được kiểm tra trở lại vào môi trường có kiểm soát, bản sao cũ hơn không bị phá hủy; thay vào đó, một phiên bản mới sẽ được tạo ra. Thông thường, chỉ có người kiểm soát cấu hình (CC) hoặc người lãnh đạo dự án có thể đưa chương trình trở lại vào môi trường (check-in). Sự hạn chế này làm cho nó có thể quay trở lại trạng thái cũ (roll back) nếu có yêu cầu.

Để cung cấp thông tin về những thay đổi đã được thực hiện, những người quản lý dự án có thể lựa chọn để lưu giữ một ghi nhận về thay đổi (modification log) đã được làm lên chính chương trình nguồn. Ghi nhận này cần thiết xác định ngày bắt đầu và ngày kết thúc của một thay đổi và chứa một tham chiếu đến yêu cầu thay đổi đã gây ra nó.

Tất cả các công việc này – đưa-vào (check-in), tách-ra (check-out), bảo trì phiên bản (version maintenance), và tạo ra một ghi nhận về thay đổi (modification log) – có thể được xử lý thông qua việc sử dụng các công cụ CM thích hợp. Nhiều công cụ sẵn có khác nhau có thể thực hiện nhiều khía cạnh khác nhau của chức năng thư viện CM này (CM library function).

Để thực hiện (cài đặt) các thay đổi yêu cầu, tại đó lần lượt kích hoạt thay đổi đến các mục cấu hình, yêu cầu thay đổi trước tiên được phân tích bằng cách thực hiện một phân tích tác động (impact analysis) (được thảo luận trong Chương 3). Phân tích này xác định các chương trình và các tài liệu cần được thay đổi và ước lượng chi phí và thời gian để thực hiện thay đổi. Sau khi thay đổi được sự chấp thuận bởi người lãnh đạo dự án và CC, tất cả các chương trình và các tài liệu đã được xác định bởi hoạt động phân tích tác động phải được thay đổi một cách hợp lý. Các hoạt động sau đây là một phần của thực hiện (cài đặt) một yêu cầu thay đổi:

- Chấp nhận yêu cầu thay đổi (với tác động phân tích).

- Thiết lập một cơ chế theo dõi (tracking mechanism).
- Tách ra (check-out) các mục cấu hình cần thay đổi.
- Thực hiện các thay đổi.
- Đưa các mục cấu hình trở lại môi trường (check-in).
- Đưa mục qua chu kỳ sống (vòng đời) của nó.

Một cơ chế dựa vào bảng tính (spreadsheet-based mechanism) thường được sử dụng để theo dõi trạng thái của các yêu cầu thay đổi. Đối với mỗi yêu cầu thay đổi, một bảng tính được tạo ra để liệt kê danh sách tất cả các chương trình đang bị thay đổi và trạng thái của chúng. Để thực hiện một thay đổi, người kiểm soát cấu hình (CC) hoặc bảng kiểm soát cấu hình (CCB) phân công các thành viên của nhóm để thực hiện thay đổi (modify) các mục khác nhau. Các thành viên của nhóm sẽ tách các mục ra (check-out) ra khỏi môi trường của nó trước khi thực hiện các thay đổi này. Sau khi một thành viên trong nhóm làm một thay đổi, chương trình (hoặc tài liệu) bị thay đổi có thể được xem như là một chương trình mới – chương trình phải đi qua các trạng thái khác nhau (trong vòng đời của chương trình) trước khi nó có thể trở thành một phần của hệ thống hoạt động sau cùng. Một khi tất cả các chương trình và tài liệu liên quan đã được thay đổi xong, chúng sẽ đi đến trạng thái “baseline” (sau khi trải qua chu kỳ cuộc sống của chúng), và yêu cầu thay đổi được coi như là đã được thực hiện đầy đủ.

9.2.3 Giám sát tình trạng và kiểm tra (Status Monitoring and Audits)

Một mục cấu hình (configuration item) có thể có vài trạng thái. Tập hợp các trạng thái có thể thay đổi tùy thuộc vào mục đó là một chương trình hay một tài liệu và loại công cụ CM đang được sử dụng. Điều quan trọng là phải xác định chính xác trạng thái của mỗi mục bởi vì các lỗi liên quan đến trạng thái có thể dẫn đến nhiều vấn đề. Ví dụ, nếu một chương trình đã không được kiểm thử đơn vị nhưng được chuyển sang trạng thái "sẵn sàng cho phát hành" (ready for release), nó có thể gây ra vấn đề. Tương tự như vậy, nếu hệ thống bị hỏng sau khi một chương trình được tách ra (check-out) khỏi baseline để cài đặt một thay đổi, phần mềm này có thể được phân phối mà không có thay đổi. Vì vậy, khi một yêu

cầu thay đổi được thực hiện, cũng quan trọng để xác định các cơ chế được dùng để nắm bắt (capture) chính xác trạng thái của mục.

Nếu các dự án sử dụng một cơ chế dựa trên cấu trúc thư mục (directory structure) để trình bày trạng thái của một chương trình, nhiều lỗi có thể xảy ra. Cơ chế kiểu này đòi hỏi rằng các chương trình phải được di chuyển đúng từ một thư mục này sang một thư mục khác khi trạng thái của chúng thay đổi và sự thay đổi trạng thái này sẽ được trình bày trên một bảng tổng thể - lưu giữ trạng thái của nhiều mục khác nhau. Để giảm thiểu lỗi và sớm phát hiện được bất kỳ lỗi nào, các dự án phải thường xuyên thực hiện kiểm tra trạng thái của các mục cấu hình. Một báo cáo có thể được tạo ra để trình bày về sự không nhất quán, và tất cả các khác biệt này phải được giải quyết.

Ngoài việc kiểm tra trạng thái của các mục, các dự án phải kiểm tra trạng thái của các yêu cầu thay đổi. Để thực hiện mục tiêu này, các yêu cầu thay đổi vừa nhận được kể từ hoạt động giám sát trạng thái CM cuối cùng sẽ được kiểm tra. Đối với mỗi yêu cầu thay đổi, trạng thái của mục như đã được đề cập trong các bản ghi nhận yêu cầu thay đổi (change request records) được so sánh với trạng thái thực tế. Các kiểm tra cũng có thể được thực hiện để đảm bảo rằng tất cả các mục vừa được sửa đổi phải trải qua toàn bộ chu kỳ sống của chúng (tức là, biểu đồ trạng thái) trước khi chúng được kết hợp vào trong baseline.

Cuối cùng, các dự án có thể thực hiện một cuộc kiểm tra cấu hình (configuration audit). Giống như các kiểm tra khác, mối quan tâm chính ở đây là để đảm bảo rằng quy trình CM của dự án đang được thực hiện theo. Baseline cho hệ thống có thể cũng được kiểm tra để đảm bảo tính toàn vẹn (integrity) của nó không bị vi phạm và có những mục được chuyển đến và chuyển ra khỏi baseline theo một cách thức nhất quán (consistent) với các kế hoạch CM.

Người kiểm soát cấu hình (CC- configuration controller) của dự án thường tiến hành các cuộc kiểm tra CM (CM audits). Sau mỗi cuộc kiểm tra CM, một báo cáo thường được phát hành để liệt kê những gì cần phải được thực hiện để giữ cho các hoạt động CM nhất quán với kế hoạch CM. Bảng 9.1 trình bày một ví dụ về báo cáo kiểm tra CM cho dự án ACIC (kế hoạch CM của nó được đưa trình bày trong phần sau của chương này). Như bạn có thể thấy, ngoài việc xem xét quy trình, kiểm tra CM tập trung vào trạng thái và vị trí của các mục cấu hình.

Bảng 9.1. Báo cáo kiểm tra CM cho Dự án ACIC

STT	Theo dõi (Observation)	Người chịu trách nhiệm (Responsible)	Thời điểm sẽ hoàn thành (Complete By)
1	Các email được ghi vào trong thư mục Messages/User *(Mails to be logged in to the Messages/User folder)*	PL, DVs	06 Aug 2000
2	Biểu đồ tuần tự đã được xem xét lại sẽ được di chuyển vào VSS *(Reviewed sequence diagrams to be moved under VSS)*	PL	06 Aug 2000
3	Kế hoạch CM sẽ được cập nhật với những thay đổi ở cấu trúc thư mục *(CM plan to be updated with the changed directory structure)*	CC	05 Aug 2000
4	Các biểu đồ tuần tự đã hoàn thành được di chuyển từ khu vực của người dùng sang khu vực xem xét lại *(Completed sequence diagrams to be moved from users' area to the review area)*	PL	06 Aug 2000
5	Cập nhật lại VSS để phản ánh các thay đổi trong cấu trúc lưu trữ *(Update VSS to reflect the modifications in the storage structure)*	CC	06 Aug 2000
6	Thư mục con ProjectDocs trong thư mục Users phải được xóa sạch *(ProjectDocs subfolder in Users folder to be cleaned up)*	All	06 Aug 2000

9.3 KẾ HOẠCH QUẢN LÝ CẤU HÌNH Ở ACIC (THE ACIC CONFIGURATION MANAGEMENT PLAN)

Kế hoạch CM được trình bày ở đây xác định môi trường CM và cấu trúc thư mục được dùng trong dự án ACIC. Kế hoạch CM của ACIC xác định rõ rằng bên dưới khu vực dự án là một cấu trúc thư mục, được kiểm soát thông qua công cụ Visual Source Safe (VSS), nơi lưu trữ tất cả các tài liệu cần được kiểm soát. Các tập tin mã nguồn (source code files)

được kiểm soát thông qua Visual Age cho công cụ Java (VAJ). Trong phạm vi thư mục VSS, các thư mục khác nhau sẽ được xác định cụ thể; các tên chỉ rõ những gì diễn ra trong những thư mục này. Khu vực của người sử dụng, nơi không kiểm soát được, cũng được xác định; có một thư mục cho mỗi người dùng, và mỗi thành viên của nhóm có nghĩa vụ phải thực hiện theo các hướng dẫn thư mục trong khu vực của mình. Tương tự như vậy, khu vực để xem xét lại (reviews) cũng được xác định; đây là thư mục chứa các mục (items) sẽ được xem xét lại.

Sau đó, kế hoạch liệt kê các mục cấu hình (configuration items), tên của chúng, và các địa điểm lưu trữ của chúng (storage locations). Chỉ có các mục cấu hình được kiểm soát mấy được đề cập đến ở đây. Các mục không được kiểm soát - ví dụ, kết quả kiểm thử (testing results), kết quả xem xét lại (review results), các thông điệp (mesages), các mẫu (template), các chuẩn (standards), v.v. – sẽ được bỏ qua; chúng được lưu trữ trong các thư mục tương ứng của chúng trong khu vực không được kiểm soát. Đối với mỗi mục (item), khu vực lưu trữ nó khi nó đang được phát triển là khu vực làm việc (work area). Nếu mục (item) sẽ được xem xét lại, kế hoạch chỉ ra nơi mà nó sẽ được di chuyển đến để xem xét lại, cũng như khu vực baseline bên dưới VSS – nơi mà nó sẽ rời khỏi sau khi được phê duyệt. Nói cách khác, kế hoạch phát thảo ra các khu vực lưu trữ cho các mục này khi chúng tiến triển (evolve).

Tiếp theo, kế hoạch mô tả làm thế nào một tài liệu di chuyển qua những khu vực khác nhau – tức là, mô tả quy trình kiểm soát cấu hình cho các tài liệu. Nó thì khá đơn giản: người dùng làm việc trên tài liệu trong khu vực làm việc (work area). Khi tài liệu đã sẵn sàng để xem xét lại, nó được chuyển đến khu vực xem xét lại đã được xác định cho nó. Nếu việc xem xét lại không yêu cầu phái làm lại (rework) bất kỳ việc gì, tài liệu sẽ được baseline. Kế hoạch không xác định một quy trình tương tự cho mã (code) bởi vì các phương pháp của VAJ sẽ được sử dụng.

Sau đó các quyền truy cập cho các khu vực khác nhau và VSS được xác định. Bởi vì dự án này có một nhóm nhỏ (small team), theo sau là một cơ chế truy cập tương đối tự do, tất cả các thành viên trong nhóm điều có truy cập nhập-vào (check-in) và tách-ra (check-out) để vào khu vực có kiểm soát (controlled area).

Tiếp theo, quy trình kiểm soát thay đổi (change control process) sẽ được xác định. Đầu tiên, kế hoạch xác định người chịu trách nhiệm để đăng nhập vào các yêu cầu thay đổi

(tức là người thực hiện việc xem xét lại), v.v. Sau đó, nó cung cấp quy trình để thực hiện thay đổi (change implementation process).

Hòa giải (reconciliation) là một vấn đề lớn trong nhiều dự án. Như đã đề cập trước đây, cần thiết phải hòa giải vấn đề phát sinh khi có nhiều người đang đồng thời thay đổi cùng một cấu hình. Đối với các tài liệu, không có nhu cầu cho việc hòa giải chúng bởi vì không có dự kiến nào cho các thay đổi diễn ra song song lên chúng, và VSS có các thủ tục đưa-vào (check-in) và tách-ra (check-out) hình thức (formal) mà không cho phép các thay đổi diễn ra song song. Tuy nhiên, việc hòa giải cho mã nguồn (source code) có thể là cần thiết. Ở mức độ lớp (class level), trách nhiệm hòa giải thuộc về chủ sở hữu của lớp – tức là những người sử dụng tính năng VAJ cho nó. (Nếu nhiều người cùng thay đổi một lớp ở cùng một thời điểm, VAJ làm nổi bật sự khác biệt trong các lớp và các phương pháp trong các phiên bản khác nhau.) Tại mỗi một mốc quan trọng, hoặc bất cứ khi nào cần thiết, tất cả các hòa giải sẽ được thực hiện cho tất cả mã nguồn.

Hòa giải cũng được cần đến nếu như một số thay đổi đạng được thực hiện song song ở phía phát triển ở nước ngoài (offshore development) cũng như phía tại chỗ (onsite). Nếu điều này xảy ra, một lần nữa VAJ được sử dụng; các khác biệt được xác định, và sau đó các thay đổi sẽ được trộn lại.

Nếu nhiều dự án đang cùng sử dụng các mã nguồn chung, việc hòa giải giữs các dự án cũng được cần đến. Thông thường, tất cả các dự án gửi các tập tin VAJ của chúng đến một người điều phối ở trung tâm (central coordinator) – người sẽ hòa giải các tập tin và trả chúng về trở lại. Những tập tin đã được hòa giải này sau đó trở thành baseline cho mỗi dự án.

Tiếp theo, kế hoạch CM xác định các khu vực phát hành (release area) và các thủ tục sao lưu dự phòng (backup procedures). Khu vực phát hành cho mã nguồn là kho lưu trữ VAJ, và mã nguồn sẽ được phát hành tại cuối các cột mốc quan trọng. ("Release" ở đây có nghĩa là việc hòa giải đã hoàn thành và nghĩa là một baseline – cái cũng có thể được giao cho khách hàng). Thiết kế ở mức cao (HLD - high-level design) được phát hành từ khu vực VSS vào cuối của cột mốc có liên quan. Để thực hiện sao lưu dự phòng (backup), khu vực (area) và tần suất (frequency) được xác định. Sau đó, tính chất tự nhiên và tần suất của việc kiểm tra cấu hình được đề cập, cùng với các vai trò và trách nhiệm của người kiểm soát cấu hình (CC - configuration controller). Trong dự án này, CC có trách nhiệm bảo trì công cụ, thực hiện các sao lưu dự phòng, tiến hành xem xét lại, và giúp nhóm thực

hiện theo các thủ tục CM. (Kế hoạch CM hoàn chỉnh có một số yếu tố bổ sung được bỏ qua ở đây.)

Kế hoạch quản lý cấu hình của Dự án ACIC

1. GIỚI THIỆU

Bỏ qua

2. MÔI TRƯỜNG CM (CM ENVIRONMENT)

- **Hệ điều hành:** Windows NT trên các máy chủ (servers), Windows 98 trên máy tính cá nhân (PCs)
- **Các phần mềm/công cụ khác**: MS Project 4.0, Rational Rose, Requisite Pro
- **Các công cụ CM**: Visual Source Safe (VSS) cho các tài liệu (documents), và Visual Age cho Java (VAJ) cho các tập tin nguồn (source files)

3. CẤU TRÚC THƯ MỤC (DIRECTORY STRUCTURE)

Khu vực dự án (**Project area**): Itlkec02/ACIC. Tất cả các thư mục sẽ nằm bên trong (bên dưới) thư mục này.

Khu vực lưu trữ được kiểm soát cho các tài liệu (Controlled Storage Area for Documents): Itlkec02/ACIC/vss. Trong khu vực có kiểm soát này, có những thư mục như HLD, ProgSpecs, ProjectDocs, ProjectMgmt, Requirements, Scope, TestPlans, v.v.

Mã nguồn (Source Code): InfosysKEC/C://ivj.dat (tập tin được VAJ dùng để lưu giữ các tập tin nguồn).

Khu vực dự án không được kiểm soát (**Uncontrolled Project Area**): Ngoài thư mục có kiểm soát cho VSS, có các thư mục riêng biệt như ChangeRequest, ImpacAnalysis, Issues, MilestoneReports, ReviewReports, StandardsAndChecklists, StatusReports, Templates, v.v. Các thư mục này không có kiểm soát.

Khu vực người dùng (User Area): Các khu vực cho những người dùng khác nhau sẽ là ACIC /Users/<UserId>. Mỗi người dùng sẽ theo cấu trúc của khu vực có kiểm soát.

Khu vực xem xét lại (Review Area): ACIC / Review. Bên trong (bên dưới) thư mục này là các thư mục riêng biệt như ProjectDocs, ChangeRequest, TestPlans, v.v.

4. CÁC MỤC CẤU HÌNH, ĐẶT TÊN VÀ LƯU TRỮ

Chỉ có các mục trong khu vực có kiểm soát được đề cập ở đây. Quy ước đặt tên theo chuẩn Infosys sẽ được dùng để đặt tên cho tất cả các tài liệu và các tập tin nguồn.

Mục cấu hình (Configuration Item)	**Tên (Name)**	**Khu vực làm việc (Work Area)**	**Khu vực xem xét lại và kiểm thử (Review/Test Area)**	**Khu vực Baseline/ Phát hành (Baseline/Release Area)**
Project scope document	ScopeDocument.doc			/vss/Scope
Use case catalog	UseCaseCatalog			/vss/Requirements
Screens	Screens			/vss/Requirements
BAR document	BAR			/vss/Requirements
Project plan	ProjectPlan.doc	/Users/<UserName>/ProjectDocs	/Reviews/ProjectDocs	/vss/ProjectMgmt
CM plan	CM Plan.doc	/Users/<UserName>/ProjectDocs/	/Reviews/ProjectDocs	/vss/ProjectDocs
Project schedule	ProjectSchedule.mpp	/Users/<UserName>/ProjectDocs	/Reviews/ProjectDocs	/vss/ProjectMgmt
High-level design document	TAD3.0.doc	/Users/<UserName>/ProjectDocs	/Reviews/ProjectDocs	/vss/HLD
Program specifications	ProgSpec#<n>	/Users/<UserName>/ProgSpecs/	/Reviews/ProgSpecs	/vss/Prog Specs

Unit test plans	UnitTestPlan#<n>.doc	/Users/<UserName>/TestPlans/	/Reviews/TestPlans/	/vss/TestPlans
Sequence diagrams	<Description>SeqDiag	/Users/<UserName>/ProjectDocs/	/Reviews/ ProjectDocs	/VSS/RoseElements/ SequenceDiagrams
Class diagrams	<Description>ClassDiag	/Users/<UserName>/ProjectDocs/	/Reviews/ProjectDocs	S/RoseElements/Cla agrams
Activity diagrams	<Description>ActDiag	/users/<UserName>/ProjectDocs/	/Reviews/ProjectDocs	/VSS/RoseElements/ ActivityDiagrams
Source code	PackageName.Classname	VAJ Repository	VAJ Repository	VAJ Repository
Integration test plan	IntegrationTestPlan	/Users/ProjectDocs/TestPlans	/Reviews/TestPlans	/vss/TestPlans
Test plans	/UseCase#.tst	/Users/<UserName>/TestPlans/	/Reviews/TestPlans	/vss/TestPlans
Closure report	ClosureReport.doc	/Users/UserName/ProjectDocs/	/Reviews/ProjectDocs	/vss/ProjectDocs

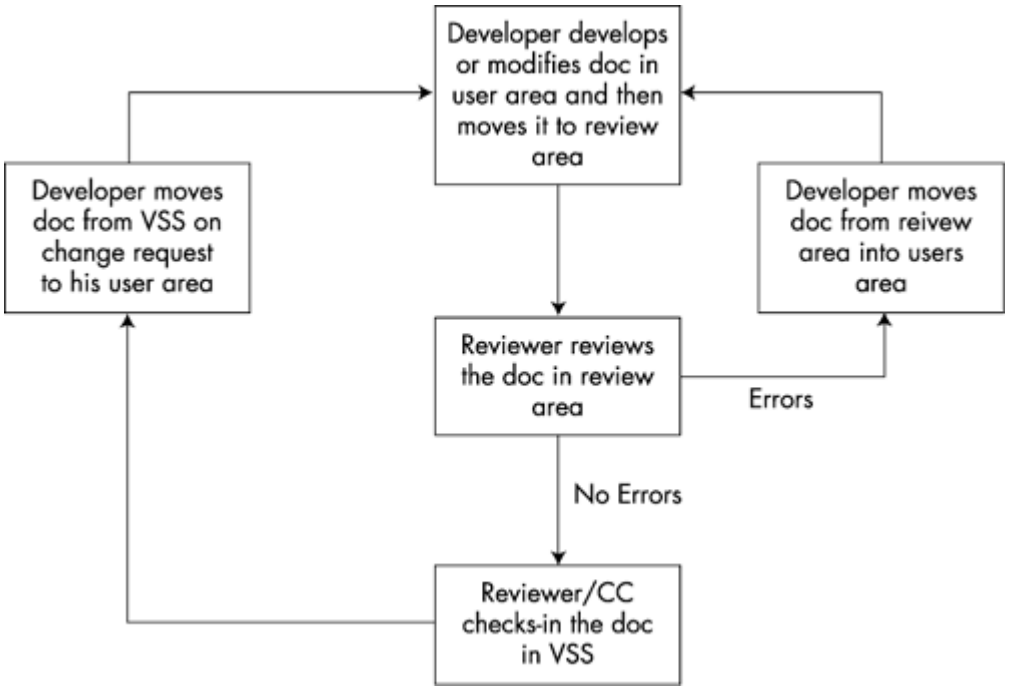
5. ĐÁNH SỐ PHIÊN BẢN/SỬA LẠI (VERSION / REVISION NUMBERING)

Đối với các tập tin nguồn (**Source Files**): Các tập tin chương trình nguồn có một cơ chế đánh số phiên bản tự động (automated versioning mechanism). Các phiên bản đầu sẽ là 1.0. Bất kỳ sự thay đổi lớn nào trong mã nguồn sẽ được gán số phiên bản là 1.1, 1.2, v.v., và những thay đổi nhỏ sẽ có số phiên bản là 1.1.1, 1.1.2, v.v.

Đối với các tài liệu (**Document**): Phiên bản gốc sẽ được đánh số là 0.0a. Các phiên bản sửa lại (revisions) tiếp theo sẽ được đánh số là 0.0b, 0.0c, v.v. Phiên bản baseline sẽ là 1.0. Các tài liệu có thể được thay đổi như là kết quả của việc thiết kế lại hoặc yêu cầu thay đổi của khách hàng. Các phiên bản mới được tạo ra được đánh số là 1.1, 1.2, v.v.

6. VIỆC DI CHUYỂN CỦA CÁC MỤC CẤU HÌNH ĐI QUA CÁC KHU VỰC LƯU TRỮ CỦA CHÚNG

Quy trình kiểm soát cấu hình cho các tài liệu như sau:



Quy trình kiểm soát cấu hình cho mã nguồn (**Configuration control process for source code**): Mã được xem xét trong VAJ. Nếu bất kỳ thay đổi nào được thực hiện, chúng được thực hiện trong phiên bản mở - nơi mà người dùng đang làm việc.

Các quyền truy cập (**Access Rights)**

Khu vực người dùng (User Area): Mỗi người dùng có truy cập R/W riêng biệt. Người lãnh đạo dự án có quyền truy cập R/W đến tất cả.

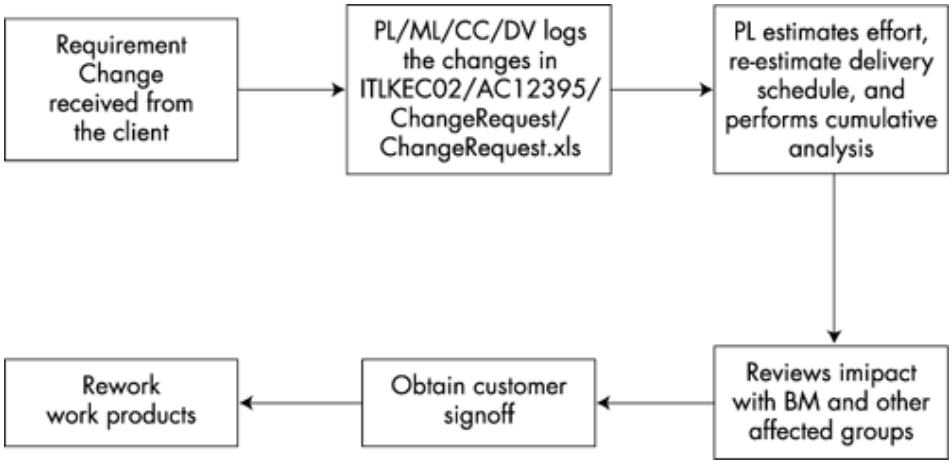
Khu vực xem xét lại (**Review Area**): Tất cả đều có truy cập R/W.

Khu vực VSS (VSS Area): Tất cả đều có quyền nhập-vào (check-in) và tách-ra (check-out).

Kiểm soát thay đổi (Change Control)

Các yêu cầu thay đổi sẽ được ghi lại vào đâu? *(Where is the change request logged?)*	ChangeRequest.xls
Ai ghi nhận các yêu cầu thay đổi? *(Who logs the change request?)*	Bất kỳ thành viên của nhóm (Any team members)
Ai xem xét lại các yêu cầu thay đổi? *(Who reviews the change request?)*	Người lãnh đạo dự án (Project leader)
Ai phê duyệt các yêu cầu thay đổi? *(Who approves the change request?)*	Người lãnh đạo hoặc người quản lý kinh doanh (Project leader or business manager)

Dòng công việc cho yêu cầu thay đổi (Workflow for Change Request)



Các tài liệu hòa giải (Reconciliation Documents)

Tất cả các tài liệu được lưu trong VSS. Chỉ có một tách-ra (check-out) duy nhất được phép tại một thời điểm nhất định.

Mã nguồn (Source Code)

Hòa giải cho tại chỗ/cho ở xa (Onsite/Offshore Reconciliation): Tham khảo đến tài liệu để biết phương pháp Cài đặt & Hòa giải VAJ (VAJ Setup & Reconciliation).

Hòa giải giữa các dự án (**Interproject Reconciliation**): hòa giải giữa các dự án sẽ được thực hiện ngay khi có một bản phát hành (release).

Hòa giải bên trong các dự án (**Intraproject Reconciliation**): Hòa giải mức độ lớp (class-level) sẽ là trách nhiệm của chủ sở hữu lớp. Các ngày hòa giải (reconciliation dates) sẽ được quyết định trong cuộc họp dự án hàng tuần. Các hướng dẫn VAJ cho hòa giải sẽ được sử dụng.

Phát hành (Release)

Mục cấu hình **(Configuration Item)**	**Khu vực phát hành** **(Release Area)**	**Trách nhiệm cho việc xây dựng phát hành và phát hành** **(Responsibility for Building the Release and Releasing)**	**Thời điểm phát hành** **(When Released)**
Source Code	VAJ Repository	Người chủ của không gian làm việc *(Workspace Owner)*	Ở cuối của phân tích cột mốc *(At the end of the milestone analysis)*
HLD	vss/HLD	Người lãnh đạo dự án (PL)	Ở cuối của giai đoạn thiết kế mức cao *(At the end of high-level design phase)*

Sao lưu (Backup)

Khu vực lưu trữ để được sao lưu **(Storage Area to Be Backed Up)**	**Phương tiện sao lưu** **(Backup Media)**	**Cơ chế đánh số** **(Numbering Scheme)**	**Sao lưu (Backup)**	
			Tấn suất **(Frequency)**	**Trách nhiệm)** **(Responsibility)**
Itlkec15 /	Băng (Tape)	As per Infosys stds.	Hàng tuần (Weekly)	Người quản trị hệ thống (system administrator)

VAJ repository (ivj.dat of server)	ITLKEC02\\\\ ProjectBackup	mmmddyyyy.dat	2 lần mỗi tuần (Twice a week)	CC
VAJ.icx and.ide files	ITLKEC02\\\\Users \\<Username> \\General	mmddyyyy.ide và mmmddyyyy.ic x	Hàng ngày (Daily)	Người sử dụng (Users)

Thủ tục lưu trữ (Archival Procedure)

Việc lưu trữ sẽ được thực hiện bởi người quản lý hệ thống ở Infosys bằng cách sử dụng các thủ tục chuẩn.

Kiểm tra cấu hình (Configuration Audit)

Kiểm tra (Type of Audit)	**Tần suất (Frequency)**
Kiểm tra CC (CC audit)	Mỗi 2 tuần (Every two weeks)
Kiểm tra baseline (Baseline audit)	Mỗi tháng 1 lần (Once a month)
Hoàn tất sản phẩm công việc (Work product completion)	Ở cuối mỗi cột mốc (At the end of a milestone)
Phát hành (Release)	Trước khi phát hành (Before release)
Kiểm tra bất ngờ (Surprise audit)	Bất kỳ lúc nào (Any time)

Trách nhiệm của người kiểm soát cấu hình

- Định hướng CM cho nhóm dự án (CM orientation for project team)
- Triển khai công cụ CM nếu có thể (CM tool deployment if applicable)
- Theo dõi trạng thái của mục cấu hình (Configuration item status tracking)·
- Đảm bảo sao lưu và lưu trữ dự phòng (Ensuring backups and archival)
- Tiến hành kiểm tra CM theo kế hoạch (Conducting CM audits as planned)

- Tạo các báo cáo kiểm tra CM (Generating CM audit reports)

- Theo dõi các khác biệt khi kiểm tra CM để kết thúc dự án (Tracking CM audit discrepancies to closure)

9.4 TÓM TẮT

Một sản phẩm phần mềm thông thường bao gồm nhiều chương trình và tài liệu, và các mục (items) này thay đổi và tiến triển trước khi chúng được hoàn thành cho hệ thống cuối cùng. Vì lý do này, quản lý cấu hình phần mềm là một vấn đề quan trọng.

Sau đây là một số bài học từ quy trình CM được sử dụng tại Infosys:

- Xác định quy trình CM để nó cho phép các dự án xử lý các cập nhật đồng thời, hủy bỏ các thay đổi (undo a change), đạt được phiên bản mới nhất của một chương trình, xác định trạng thái (tình trạng) của một chương trình, và ngăn chặn những thay đổi trái phép. Sử dụng việc kiểm soát phiên bản (version control), theo dõi yêu cầu thay đổi (change request tracking), và các cơ chế quản lý thư viện (library management mechanisms) để hỗ trợ những khả năng này.

- Phát triển một kế hoạch CM riêng biệt từ kế hoạch quản lý dự án. Kế hoạch CM phải xác định môi trường (environment), các mục cấu hình (configuration items), và quy ước đặt tên (naming convention) của chúng, các khu vực lưu trữ (storage areas) cho các mục ở các trạng thái khác nhau, và phương pháp quản lý các thay đổi đối với các mục, bao gồm việc đánh số phiên bản và hòa giải, kiểm soát truy cập, và phát hành và các chính sách sao lưu dự phòng.

- Thực hiện các cuộc kiểm tra CM (CM audits) và kiểm tra trạng thái (tình trạng) để đảm bảo rằng kế hoạch CM đang được thực hiện theo.

Tương ứng với CMM, các hoạt động được thảo luận trong chương này đáp ứng rất nhiều yêu cầu của KPA Quản Lý Cấu Hình Phần Mềm (Software Configuration Management) của CMM mức 2. Các phương pháp thực hiện các thay đổi yêu cầu đáp ứng một số khía cạnh của KPA Quản Lý Yêu Cầu (Requirements Management). Việc baseline và kiểm soát các sản phẩm công việc và mã cũng đáp ứng yêu cầu của các KPA như Quản Lý Yêu Cầu (Requirements Management) và Lập Kế Hoạch Cho Dự Án Phần Mềm (Software Project Planning).

9.5 CÁC THAM KHẢO

1. E.H. Bersoff, V.D. Henderson, and S.G. Siegel. Software configuration management: A tutorial. *IEEE Computer*, Jan. 1979.

2. E.H. Bersoff. Elements of software configuration management. *IEEE Transactions on Software Engineering*, Jan. 1984.

3. W. Humphrey. *Managing the Software Process*. Addison-Wesley, 1989.

4. D. Whitgift. *Methods and Tools for Software Configuration Management*. John Wiley and Sons, 1991.

5. P. Jalote. *CMM in Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley, 2000.

Phần II: Thực hiện và kết thúc dự án

Chương 10. Xem xét lại

Chương 11. Giám sát và kiểm soát dự án

Chương 12. Kết thúc dự án

Chương 10. Xem xét lại

Bạn có thể kiểm tra chất lượng một chương trình bằng cách kiểm thử. Nhưng làm thế nào để bạn kiểm tra được kế hoạch kiểm thử đang được sử dụng có chứa các ca kiểm thử (test case) đúng không? Và làm thế nào để bạn kiểm tra một thiết kế (design) để phát hiện các lỗi thiết kế (design errors) hoặc kiểm tra các đặc tả yêu cầu (requirement specification) để phát hiện lỗi (defects)? Xem xét lại (reviews) là phương pháp hiệu quả nhất và thường được sử dụng để xác định lỗi, không chỉ trong các tài liệu (documents) như kế hoạch kiểm thử (test plan) và tài liệu thiết kế (deisgn document) mà còn trong mã (code). Xem xét lại cung cấp cho các nhà quản lý thấy được (visibility) tiến bộ của dự án, một cái gì đó có thể giúp họ có những hành động chấn chỉnh kịp thời. Đối với một người quản lý dự án, xem xét lại cung cấp một số lợi thế khác sau đây:

- Thông qua xem xét lại, các tài năng giỏi nhất trong công ty có thể được sử dụng trong một dự án ngay cả khi họ không được phân công cho nó.
- Xem xét lại giúp duy trì động lực của nhóm (team motivation) bằng cách cho mọi người một cảm giác về thành tích, sự tham gia, và công nhận.
- Thông qua xem xét lại, các thành viên trong nhóm có thể phát triển các kỹ năng của họ và những người cấp cao có thể làm cố vấn cho những đồng nghiệp có ít kinh nghiệm.
- Xem xét lại giúp phòng ngừa lỗi bằng cách tạo ra nhận thức (awareness) nhiều hơn về chúng.

Chương này thảo luận về quy trình xem xét lại đã được sử dụng tại Infosys và giải thích làm thế nào dữ liệu được thu thập và được sử dụng để theo dõi và kiểm soát những xem xét lại này. Chúng tôi cũng xem xét ngắn gọn làm thế nào thử nghiệm được dùng để thuyết phục các nhà quản lý dự án về giá trị của hoạt động xem xét lại bởi nhóm (group reviews).

10.1 QUY TRÌNH XEM XÉT LẠI (THE REVIEW PROCESS)

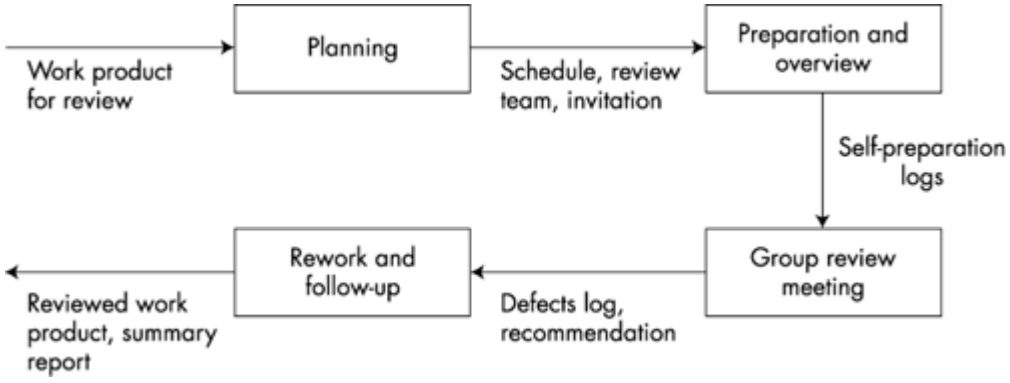
Xem xét lại (reviews) có thể được thực hiện bằng nhiều cách khác nhau. Một xem xét lại bởi nhóm theo đúng hình thức (formal group review), còn được gọi là *thanh tra*

(inspection), có lẽ là một lựa chọn tốt nhất để xác định lỗi. Thanh tra phần mềm (software inspection) lần đầu tiên được đề xuất bởi Fagan [1][2]. Các thanh tra trước đây chỉ tập trung vào mã, nhưng trong những năm qua kỹ thuật này đã lan rộng đến các sản phẩm công việc (work product) khác. Ngày nay, thanh tra phần mềm đã được công nhận là thực hành tốt nhất của công nghiệp, với nguồn dữ liệu lớn để chứng minh rằng chúng giúp nâng cao chất lượng và năng suất (ví dụ, xem các báo cáo trong Gilb và Graham [3] Grady và Slack [4], và Weller [5]). Một số cuốn sách về chủ đề này mô tả rất chi tiết về việc làm thế nào thực hiện thanh tra [3][6][7].

Quy trình xem xét lại cơ bản tại Infosys là việc xem xét lại bởi nhóm (group review), nó tương tự như thanh tra. Một xem xét lại bởi nhóm là một phân tích một sản phẩm công việc (work product) được làm bởi một nhóm các đồng nghiệp bằng cách theo một quy trình đã được định nghĩa (xác định) rõ ràng. Các mục tiêu của xem xét lại là để nâng cao chất lượng bằng cách tìm ra các lỗi (defects) và nâng cao năng suất bằng cách tìm các lỗi một cách hiệu quả về mặt chi phí. Ngoài ra, xem xét lại cung cấp các đầu vào (inputs) và khả năng nhìn thấy được tình trạng (visibility) để giúp ích cho quản lý dự án ở chất lượng của các sản phẩm công việc. Một xem xét lại bởi nhóm được xây dựng bởi những người kỹ thuật (technical people), và tập trung vào việc xác định các vấn đề (problem), nhưng không đi giải quyết chúng.

Quy trình xem xét lại bởi nhóm bao gồm nhiều giai đoạn: lập kế hoạch (planning), chuẩn bị và tổng quan (preparation and overview), họp xem xét lại bởi nhóm (group review meeting), và làm lại công việc và sự tiếp tục (rework and follow-up). Như được trình bày trong Hình 10.1, các giai đoạn này thường được thực hiện theo một thứ tự tuyến tính.

Hình 10.1. Quy trình xem xét lại bởi nhóm



Trong một số trường hợp, xem xét lại bởi nhóm có thể là quá mức cần thiết; cho nên một hình thức hạn chế hơn của xem xét lại bởi nhóm có thể mang lại hiệu quả hơn về chi phí. Ở Infosys, xem xét lại bởi một người (one-person review), được thảo luận trong phần sau, cũng được thực hiện.

10.1.1 Lập kế hoạch (Planning)

Mục tiêu của giai đoạn lập kế hoạch là để chuẩn bị cho việc xem xét lại bởi nhóm bằng cách chọn nhóm xem xét lại (group review team) và lập kế hoạch xem xét lại. Tác giả của sản phẩm công việc đảm bảo rằng các sản phẩm công việc đã sẵn sàng cho việc xem xét lại bởi nhóm và rằng tất cả các tiêu chuẩn thích hợp đã được đáp ứng. Người quản lý dự án, dưới sự đồng ý của tác giả, trước tiên sẽ chọn ra người điều phối (moderator); sau đó, dưới sự đồng ý của người điều phối, người quản lý dự án lựa chọn những người xem xét lại (reviewers) khác. Trách nhiệm tổng thể của người điều phối là đảm bảo rằng việc xem xét được thực hiện đúng và rằng tất cả các bước trong quy trình xem xét lại được thực hiện theo. Những người xem xét lại (reviewers) (còn được gọi là thanh tra viên/inspectors) có trách nhiệm xác định lỗi trong các sản phẩm công việc. Nói chung, tác giả cũng là một trong những người xem xét lại; người điều phối có thể cũng là người xem xét lại.

Cố gắng nếu có thể, không nên bổ sung thêm người cấp trên nào vào bởi vì sự hiện diện của họ có thể ngăn cản những người xem xét lại đưa ra các vấn đề hoặc lỗi. Tuy nhiên, nếu tác giả mong muốn, người lãnh đạo dự án cũng có thể tham gia.

Một khi nhóm xem xét lại (review team) đã được chọn ra, tác giả (author) chuẩn bị một gói (package) được phân phát để nhóm xem xét lại nó. Gói này bao gồm các sản phẩm công việc cần được xem xét lại, các đặc tả (specification) của nó, và các danh sách kiểm tra (checklists) và các tiêu chuẩn có liên quan. Thông thường, các đặc tả cho các sản phẩm công việc là đầu ra của giai đoạn trước đó và nó được cần để kiểm tra tính đúng đắn của sản phẩm công việc hiện tại.

10.1.2 Tổng quan và chuẩn bị (Overview and Preparation)

Mục đích của giai đoạn tổng quan và chuẩn bị là để phân phát gói (package) để được xem xét lại bởi những người xem xét lại và để giải thích về sản phẩm công việc, nếu cần thiết. Tài liệu (material) có thể được phân phối và giải thích trong một cuộc họp ban đầu. Người điều phối mở cuộc họp này với một trình bày ngắn gọn về các sản phẩm công việc, mục

tiêu xem xét lại bởi nhóm, và, nếu cần thiết, một tổng quan về quy trình xem xét lại bởi nhóm. Tác giả có thể cung cấp một hướng dẫn ngắn gọn về các sản phẩm công việc, bao gồm một bản tóm tắt về bất kỳ vấn đề nào cần được quan tâm đặc biệt hoặc các khu vực khó hiểu. Trong suôt giai đoạn tổng quan này, bất cứ điều gì đặc biệt về dự án hoặc các sản phẩm công việc sẽ được đánh dấu. Bước này là một tùy chọn và có thể được bỏ qua. Trong trường hợp đó, người điều phối chỉ cần cung cấp một bản sao của gói đến những người xem xét lại.

Để chuẩn bị cho cuộc họp xem xét lại bởi nhóm (group review meeting), những người xem xét lại *tự xem xét lại (self-review)* các sản phẩm công việc. Bất cứ khi nào sản phẩm công việc xuất hiện một lỗi, những người xem xét lại sẽ ghi chú các giải thích vào trong một *bản ghi tự chuẩn bị (self-preparation log)* (được mô tả chi tiết trong phần 10.2.1). Họ cũng ghi lại thời gian đã xài cho việc tự xem xét lại này. Trong suốt quá trình tự xem xét lại này, họ sử dụng bất kỳ danh sách kiểm tra (checklists), hướng dẫn (guidelines), và các tiêu chuẩn (standards) có liên quan.

Lý tưởng nhất, bước chuẩn bị này nên được thực hiện chỉ trong một lần ngồi làm (one sitting). Thời gian được đề nghị cho chuẩn bị này là một giờ cho mỗi lần có cuộc họp xem xét lại bởi nhóm.

10.1.3 Họp xem xét lại bởi nhóm (Group Review Meeting)

Mục đích cơ bản của cuộc họp xem xét lại bởi nhóm để đi đến thống nhất một danh sách lỗi cuối cùng; danh sách này dựa trên danh sách ban đầu chứa các lỗi và các vấn đề phát sinh được báo cáo bởi những người xem xét lại và những lỗi hoặc vấn đề mới được tìm thấy qua thảo luận trong suốt cuộc họp.

Trước khi cuộc họp được lên kế hoạch, người điều phối (moderator) trước tiên sẽ kiểm tra xem tất cả những người xem xét lại đã có chuẩn bị trước chưa. Xác minh này liên quan đến một cuộc kiểm tra ngắn về dữ liệu nỗ lực và dữ liệu về lỗi trong các bản ghi tự chuẩn bị (self-preparation logs) để xác nhận rằng đã có sự đầu tư đầy đủ về thời gian và sự quan tâm ở giai đoạn chuẩn bị. Nếu việc chuẩn bị chưa được làm đầy đủ, xem xét lại bởi nhóm sẽ bị hoãn lại cho đến khi tất cả những người tham gia đã chuẩn bị đầy đủ.

Khi mọi thứ đã sẵn sàng, cuộc họp xem xét lại bởi nhóm sẽ được tổ chức. Một người xem xét (reviewer) được chỉ định là người ghi chép và khác người đọc. Cuộc họp được tiến

hành như sau. Người đọc đi qua các sản phẩm công việc (work product) từng dòng từng dòng một (hoặc bất kỳ đơn vị nhỏ thuận tiện nào khác), diễn giải mỗi dòng cho nhóm. (Một số cuộc họp không bao gồm diễn giải, trong trường hợp này, vai trò của người đọc không được thể hiện.) Ở bất kỳ dòng nào, nếu bất kỳ một người xem xét lại nào trước đây đã xác định được các vấn đề phát sinh hoặc vừa tìm thấy những vấn đề mới trong cuộc họp này trong khi đang lắng nghe những người khác, anh ta sẽ giơ tay lên. Có thể có một cuộc thảo luận về vấn đề được đưa ra này, và những người xem xét lại khác có thể đồng ý hoặc không đồng ý với nó. Trong mọi trường hợp, tác giả xem xét lại vấn đề phát sinh đang được thảo luận và hoặc làm rõ lý do tại sao nó không phải là một vấn đề hoặc chấp nhận nó như là một lỗi (defect) hay là một vấn đề mở (issue). Người chịu trách nhiệm ghi chép sẽ ghi lại các vấn đề và các lỗi vừa được xác định.

Vào cuối cuộc họp, người ghi chép trình bày danh sách các vấn đề mở (open issues) và lỗi (defects) vừa được xác định trong cuộc họp, và phải trải qua một cuộc xem xét lại sau cùng bởi các thành viên trong nhóm. Lưu ý rằng quy trình xem xét lại chỉ đơn thuần đi xác định các lỗi và các vấn đề phát sinh. Nhóm không có mục đích đi xác định các giải pháp (solutions); hoạt động này sẽ được thực hiện sau này bởi tác giả (author).

Nếu có ít sửa đổi được yêu cầu, trạng thái của xem xét lại bởi nhóm là "chấp nhận" (accepted). Nếu nhiều sửa đổi được yêu cầu, một cuộc họp tiếp theo hoặc việc xem xét lại có thể là cần thiết để xác minh xem các thay đổi đã được kết hợp lại một cách đúng đắn chưa. Người điều phối gợi ý về những gì sẽ được thực hiện. Không giống như một lỗi (defect) – cái mà mà tác giả là người có trách nhiệm để sửa chữa - người điều phối có thể phân công các vấn đề (issues) cho những người khác nhau để giải quyết. Việc phân công được thực hiện trước khi cuộc họp kết thúc. Đôi khi, các gợi ý liên quan đến việc xem xét lại trong các giai đoạn tiếp theo cũng được làm. Ví dụ, để xem xét lại một thiết kế chi tiết (detailed design review), nhóm có thể đề nghị những mô-đun mã (code modules) nào nên được xem xét lại bởi nhóm trong giai đoạn xây dựng (build phase).

Người điều phối là người phụ trách cuộc họp và đảm bảo rằng cuộc họp luôn tập trung vào các mục đích cơ bản của nó - xác định lỗi – chứ không bị thoái hóa thành một cuộc họp đưa ra các ý kiến chung chung (general brainstorming session) hoặc là cuộc họp chỉ trích tác giả. Người điều phối phải được đào tạo chính quy về làm thế nào để chỉ đạo (hướng dẫn) xem xét lại hoặc phải có kinh nghiệm như là một người tham gia xem xét lại. Trong suốt cuộc họp, người điều phối phải đảm bảo rằng tất cả những người tham gia

đóng góp một cách có hiệu quả, tất cả mọi người được nghe nói, thỏa thuận để thống nhất về những phát hiện trong suốt quá trình xem xét lại, và mức độ quan tâm không bị giảm. Một trách nhiệm quan trọng là phải đảm bảo rằng trọng tâm vẫn luôn là đi xác định vấn đề chứ không để bị lôi cuốn vào việc đi giải quyết vấn đề. Nhìn chung, việc chỉ đạo (hướng dẫn) cuộc họp một cách có trình tự và thân tình là trách nhiệm của người điều phối. Sau cuộc họp, người điều phối phải đảm bảo rằng tất cả những người tham gia hài lòng, báo cáo xem xét lại thì đầy đủ và truyền tải vấn đề rõ ràng, và các hoạt động tiếp theo sẽ được thực hiện.

Trong một cuộc họp xem xét lại, một người có thể được phân công một số vai trò hợp lý, với các hạn chế là tác giả không được làm người điều phối hay người đọc (reader), và người điều phối không được làm người đọc. Sự hạn chế này có nghĩa là kích thước tối thiểu của nhóm xem xét lại là ba: tác giả, người điều phối, và người đọc. Cả ba người này cũng có thể là người xem xét lại, và một trong số họ có thể đóng vai trò làm người ghi chép.

10.1.4 Làm lại và các công việc tiếp theo (Rework and Follow-up)

Tác giả thực hiện làm lại (rework) để sửa chữa tất cả các lỗi đã được đưa ra trong cuộc họp xem xét lại. Tác giả cũng có thể phải làm lại sản phẩm công việc (work product) nếu người điều phối đề nghị. Ngoài ra, nếu những người xem xét lại đã được phân công các vấn đề mở (open issues), họ phải điều tra những vấn đề này và đưa ra kết quả điều tra cho tác giả và người điều phối.

Tác giả xem xét lại các điều chỉnh (corrections) cùng với người điều phối hoặc, nếu người điều phối vừa quyết định để xem xét lại một kết quả xem xém lại (re-review). Người ghi chép đảm bảo rằng biên bản báo cáo về cuộc họp xem xét lại được truyền đạt đến các thành viên của nhóm xem xét lại. Sau khi tất cả các vấn đề và lỗi được đóng lại, người điều phối đảm bảo rằng các kết quả của nhóm xem xét lại và dữ liệu đã được ghi lại và rằng biểu mẫu tóm tắc việc xem xét lại bởi nhóm (xem phần 10.2.3) phải được nộp cho nhóm quy trình công nghệ phần mềm (SEPG) và người lãnh đạo dự án.

10.1.5 Xem xét lại bởi một người (One-Person Review)

Xem xét lại bởi nhóm là cách rất hiệu quả để xác định lỗi. Tuy nhiên, chi phí của nó cũng cao: Nhiều người dành nhiều thời gian chuẩn bị cũng như trong cuộc họp xem xét lại.

Ngoài ra, việc tổ chức các cuộc họp xem xét lại cũng là một vấn đề phức tạp. Nếu sản phẩm công việc (work product) có nhiều lỗi hoặc quan trọng, chi phí này là hợp lý. Thật vậy, các xem xét lại bởi nhóm là một cách hiệu quả về mặt chi phí để phát hiện ra các lỗi. Nhưng sẽ như thế nào nếu các sản phẩm công việc là tương đối đơn giản, nó hầu như không có nhiều lỗi, và không phải là rất quan trọng? Trong trường hợp này, các nỗ lực (effort/chi phí) mà nhóm xem xét lại phải bỏ ra có thể không hợp lý. Tuy nhiên, thực hiện một số xem xét lại cho các sản phẩm công việc như vậy có thể là hữu ích - không chỉ để phát hiện lỗi mà còn để đạt được những lợi ích về tâm lý khi tác giả biết rằng sẽ có một ai đó khác xem xét lại sản phẩm của mình.

Đối với tình huống như vậy, xem xét lại bởi một người (one-person review) có lẽ là một giải pháp thích hợp hơn. Tức là, đối với các sản phẩm công việc có độ quan trọng trung bình hoặc phức tạp, xem xét lại bởi một người có thể thay thế cho xem xét lại bởi nhóm. *Các xem xét lại bởi một người (one-person reviews)* là các xem xét lại theo đúng hình thức (formal reviews) nhưng ít tốn kém hơn so với xem xét lại bởi nhóm bởi vì chúng không cần đến một nhóm người xem xét.

Quy trình cho các xem xét lại bởi một người là tương tự như quy trình xem xét lại bởi nhóm. Tác giả (author), tham khảo ý kiến của người lãnh đạo dự án, xác định người xem xét lại (reviewer). Xem xét lại này được xếp lịch (scheduled), và người xem xét lại nhận một gói (package) chi tiết. Người xem xét lại xem xét lại sản phẩm công việc một mình và chuẩn bị cho cuộc gặp gỡ với tác giả. Cuộc họp xem xét lại (review meeting) chỉ có hai người tham gia: tác giả và người xem xét lại. Trong suốt cuộc họp, một bản ghi chép lại các vấn đề phát sinh (issues) và lỗi (defects) được tạo ra. Người xem xét lại thông báo cho người lãnh đạo dự án (project leader) khi nào việc xem xét lại sẽ kết thúc. Người lãnh đạo dự án chịu trách nhiệm cho việc theo dõi các lỗi cho đến khi chúng đóng lại.

10.1.6 Các hướng dẫn cho việc xem xét lại trong các dự án (Guidelines for Reviews in Projects)

Không phải tất cả các sản phẩm công việc (work products) trong một dự án điều phải trải qua xem xét lại bởi nhóm bởi vì nó tốn kém và có thể không mang lại lợi ích tương xứng. Sau đây là một số hướng dẫn chung cho việc lựa chọn các sản phẩm công việc để xem xét lại.

Đối với một dự án cụ thể, tiêu chuẩn và quyết định thực tế liên quan đến các sản phẩm công việc để được xem xét lại được đưa ra bởi người quản lý dự án và nhóm đã xem xét lại các sản phẩm công việc của các giai đoạn trước. Xác định trước mục đích cho kết quả xem xét lại, nhóm có thể thể đưa ra quyết định tốt hơn về những gì cần được xem xét trong phần còn lại của dự án và phương pháp sử dụng. Bởi vì các sản phẩm công việc của phần đầu của chu kỳ sống là quan trọng và bởi vì lỗi trong chúng có nhiều ảnh hưởng tới các giai đoạn sau, các sản phẩm công việc sau đây được khuyến cáo để được xem xét lại bởi nhóm:

- Kế hoạch quản lý dự án (project management plan)
- Đặc tả yêu cầu (requirement specification)
- Kế hoạch kiểm thử hệ thống (system test plan)
- Thiết kế ở mức cao (high-level design)
- Kế hoạch kiểm thử tích hợp (integration test plan)

Vào cuối quy trình xem xét lại thiết kế ở mức cao, nhóm xem xét lại làm một đề nghị để xem xét lại các thiết kế chi tiết. Các đề nghị tương tự như vậy được thực hiện cho mã (code) vào cuối quy trình xem xét lại thiết kế chi tiết.

Mặc dù quy trình xem xét lại thì tương tự cho bất kỳ sản phẩm công việc nào, nhưng vẫn có một số khác biệt nằm ở mối quan tâm (focus) của việc xem xét lại, tiêu chuẩn đầu vào (entry criteria), và cấu trúc (makeup) của nhóm xem xét lại dựa theo tính chất (nature) của sản phẩm công việc. Các danh sách kiểm tra (checklists) được sử dụng cũng phụ thuộc vào tính chất của các sản phẩm công việc. Bảng 10.1 tóm tắt các hướng dẫn cho một số sản phẩm công việc. Các hướng dẫn cho các sản phẩm công việc khác thì tương tự.

Bảng 10.1. Các hướng dẫn xem xét lại cho một số sản phẩm công việc

Sản phẩm công việc (Work Product)	**Mối quan tâm (Focus)**	**Tiêu chuẩn đầu vào (Entry Criteria)**	**Những người tham gia (Participants)**
Đặc tả yêu cầu	• Các yêu cầu đáp ứng	• Tài liệu tuân theo các	• Khách hàng

(Requirement specification)	đúng yêu cầu khách hàng *(Requirements meet customer needs.)* • Các yêu cầu phải cài đặt được *(Requirements are implementable.)* • Bỏ sót, không nhất quán, và mơ hồ trong các yêu cầu *(Omissions, inconsistencies, and ambiguities in the requirements.)*	chuẩn *(The document conforms to the standards.)*	*(Customer)* • Những người thiết kế *(Designers)* • Người kiểm thử hệ thống *(System testing)* • Nhóm cài đặt *(Installation team member)* • Tác giả của tài lệu người dùng *(User documentation author)*
Thiết kế ở mức cao (High-level design)	• Thiết kế ở mức cao để cài đặt các yêu cầu *(High-level design implements the requirements.)* • Thiết kế phải cài đặt được *(The design is implementable.)* • Bỏ sót và các lỗi khác trong thiết kế *(Omissions and other defects in the design.)*	• Tài liệu tuân theo các chuẩn *(The document conforms to the standards.)* • Các yêu cầu vừa được xem xét lại và đã hoàn thành *(The requirements have been reviewed and finalized.)*	• Tác giả tài liệu yêu cầu *(Requirements author)* • Người thiết kế chi tiết *(Detailed designer)* • Nhà phát triển *(Developer)*
Mã *(Code)*	• Mã cài đặt cho thiết kế *(Code implements the design.)* • Mã phải đầy đủ và đúng đắn *(Code is complete and correct.)* • Các lỗi trong mã *(Defects in code.)*	• Mã biên dịch và đúng theo các kiểu và tiêu chuẩn khác *(The code compiles and passes style and other norms.)*	• Người thiết kế *(Designer)* • Người kiểm thử *(Tester)* • Nhà phát triển *(Developer)*

Các ca kiểm thử hệ thống *(System test cases)*	• Tập hợp các ca kiểm thử phải kiểm tra tất cả các điều kiện trong các yêu cầu *(The set of test cases checks all conditions in the requirements.)* • Các ca kiểm thử hệ thống là đúng đắn *(System test cases are correct.)* • Các ca kiểm thử phải thực hiện được *(Test cases are executable.)*	• Các yêu cầu vừa được baseline *(Requirements have been baselined.)* • Kế hoạch kiểm thử hệ thống thì nhất quán với các chuẩn *(System test plan is consistent with the standards.)*	• Tác giả tài liệu yêu cầu *(Requirements author)* • Người kiểm thử *(Tester)* • Người lãnh đạo dự án *(Project leader)*
Kế hoạch quản lý dự án *(Project management plan)*	• Kế hoạch quản lý dự án đáp ứng được yêu cầu quản lý và kiểm soát dự án *(Project management plan meets project management and control needs.)* • Sự đầy đủ *(Completeness.)* • Kế hoạch quản lý dự án phải cài đặt được *(Project management plan is implementable.)* • Bỏ sót và mơ hồ *(Omissions and ambiguities.)*	• Kế hoạch quản lý dự án sử dụng các bản mẫu chuẩn *(The project management plan follows the standard template.)*	• Người lãnh đạo dự án *(Project leader)* • Thành viên nhóm quy trình công nghệ phần mềm *(SEPG member)* • Một người lãnh đạo dự án khác *(Another project leader)*

10.2 THU THẬP DỮ LIỆU (DATA COLLECTION)

Thu thập dữ liệu trong suốt quy trình xem xét lại là rất quan trọng. Bạn đã nhìn thấy rằng ở các giai đoạn khác nhau trong quy trình này, dữ liệu được ghi nhận lại. Bởi vì xem xét lại là quy trình chủ yếu dựa vào con người (human processes), nếu dữ liệu không được ghi lại, thông tin có thể dễ dàng bị mất đi. Ngoài việc ghi nhận lại các lỗi (defects) đã được xác định bởi quy trình xem xét lại, dữ liệu về nỗ lực (chi phí/effort) cũng phải được ghi lại. Dữ liệu chi tiết về lỗi được cần để theo dõi lỗi trong dự án. Dữ liệu về lỗi tổng thể và nỗ lực (chi phí) được cần để phân tích hiệu quả của quy trình xem xét lại (effectiveness of the review) và để xây dựng baseline về khả năng xem xét lại (review capability baseline). Do đó, các dữ liệu tóm tắt (summary data) cho mỗi xem xét lại được lưu giữ trong một cơ sở dữ liệu về xem xét lại (review database). Phần này mô tả các biểu mẫu chủ chốt được sử dụng để thu thập dữ liệu trong suốt quy trình xem xét lại tại Infosys.

10.2.1 Bản ghi tự chuẩn bị (Self-Preparation Log)

Những người xem xét lại sử dụng một *bản ghi tự chuẩn bị* để ghi lại tất cả các lỗi (defects) hoặc các vấn đề (issues) được tìm thấy trong suốt quy trình xem xét lại một cách độc lập bởi mỗi cá nhân họ (individual review). Nỗ lực (chi phí) bỏ ra trong suốt quy trình xem xét lại cũng được ghi nhận lại. Mỗi người xem xét lại chuẩn bị bản ghi này. Ngoài việc xác định dự án (project code), sản phẩm công việc (work product), người xem xét lại (reviewer name), và các sự kiện khác, các mục trong bản ghi xác định vị trí (location) của vấn đề hoặc lỗi và đánh giá (assessment) của người xem xét lại về mức độ nghiêm trọng (seriousness) hoặc nguy kịch (criticality) của nó. Biểu mẫu được trình bày ở Hình 10.2 có thể được sử dụng như là hoặc được thực hiện thông qua một bảng tính (spreadsheet).

Hình 10.2. Bản ghi tự chuẩn bị

Project code:

Work product ID:

Reviewer name:

Effort spent for preparation (hours):

Issue list:

Sl #	Location	Description	Criticality/Seriousness

10.2.2 Bản ghi cuộc họp xem xét lại bởi nhóm (Group Review Meeting Log)

Người ghi chép chuẩn bị một bản ghi cho cuộc họp để liệt kê (list) các lỗi (defects) và các vấn đề (issues) vừa được xác định trong suốt cuộc họp. Do đó, nó bao gồm tất cả các lỗi đã được tìm thấy bởi từng người xem xét lại đơn lẻ khi họ tự xem xét lại và các lỗi này vừa được thảo luận trong suốt cuộc họp để xác nhận (validated) rằng chúng đúng là lỗi hoặc vấn đề, cũng như các lỗi bổ sung vừa được tìm thấy trong suốt cuộc họp. Không giống như bản ghi tự chuẩn bị, trong đó liệt kê các lỗi được phát hiện bởi một người xem xét lại, bản ghi cuộc họp xem xét lại bởi nhóm chỉ liệt kê các lỗi vừa được sự đồng ý của tác giả. Nói cách khác, nó chỉ liệt kê các lỗi thực sự đã được tìm thấy, và nó là danh sách lỗi chính thức (official defects list) của việc xem xét lại.

Ngoài ra vị trí (location) và mô tả (description) của mỗi lỗi, mức độ nghiêm trọng (severity) của nó cũng được ghi lại trong bản ghi này. Mức độ nghiêm trọng phản ánh sự đồng thuận của nhóm xem xét lại. Như đã thảo luận trong Chương 7, mức độ nghiêm trọng của một lỗi có thể là nghiêm trọng (critical), lớn (major), nhỏ (minor), hoặc nhẹ bên ngoài (cosmetic). Nếu có thể, hãy xác định giai đoạn (stage) mà lỗi đã được tiêm vào (injected) và loại lỗi (type).

Các vấn đề (issue) được nêu ra được liệt kê trong một bản ghi riêng biệt. Đối với từng vấn đề, một người sẽ được phân công (assigned to) để giải quyết nó. Bản ghi nhận này cũng chứa nỗ lực tổng cộng đã xài (tính theo người-giờ (person-hours)) trong cuộc họp.

Hình 10.3 cho thấy định dạng của bản ghi cuộc họp xem xét lại bởi nhóm. Nó có thể được thực hiện thông qua một bảng tính hoặc một số cơ chế khác.

Hình 10.3. Bản ghi cuộc họp xem xét lại bởi nhóm

Project code: Meeting type:
Moderator: Scribe:
Author: Reviewer(s):
SEPG member: Observer(s):
Date: Work product ID:
Effort spent on review meeting (person-hours):
Defects to be closed by (date):
Defect List:

Sl #	Defect Location	Type	Severity	Stage injected	Description

Open Issues Log:

Sl #	Issue Description	Assigned to	Targeted date	Closed date

10.2.3 Báo cáo tóm tắt xem xét lại bởi nhóm (Group Review Summary Report)

Bản ghi lỗi được sử dụng để theo dõi tất cả các lỗi cho đến khi đóng lỗi (closure). Tuy nhiên, để phân tích tính hiệu quả của một quy trình xem xét lại, chỉ các thông tin ở mức độ tóm tắt (summary-level information) được cần đến. Thông tin này cũng được cần đến để cập nhật lại baseline về xem xét lại (review baseline). Để cải tiến quy trình và hiểu về quy trình xem xét lại, báo cáo tóm tắt xem xét lại bởi nhóm là phần tử quan trọng nhất. Vì thế,

thông tin này được lưu giữ riêng biệt vào trong một cơ sở dữ liệu về xem xét lại (review database) để dùng cho việc phân tích.

Báo cáo tóm tắt (summary report) mô tả sản phẩm công việc (work product); nỗ lực tổng cộng (total effort) và số tiền (amounts) đã xài cho mỗi hoạt động (activity) trong quy trình xem xét lại, tổng số lỗi được tìm thấy theo mỗi loại lỗi (type); và kích thước (size) của sản phẩm công việc đang được xem xét lại. Nếu loại lỗi đã được ghi nhận, số lỗi theo mỗi loại có thể được ghi lại trong bản báo cáo tóm tắt. Ngoài các dữ liệu về nỗ lực và lỗi, bản báo cáo tóm tắt còn chứa các đề nghị cho giai đoạn tiếp theo. Cuối cùng, bản báo cáo tóm tắt cho biết có cần phải làm lại xem xét lại (re-review) hay không. Một bản báo cáo tóm tắt đầy đủ sẽ được trình bày ở phần sau trong chương này.

10.3 GIÁM SÁT VÀ KIỂM SOÁT (MONITORING AND CONTROL)

Hiệu quả của quy trình xem xét lại phụ thuộc vào việc quy trình này đã được triển khai tốt như thế nào. Ví dụ, nếu chỉ có hai lỗi được tìm thấy trong quá trình xem xét lại một chương trình 500 dòng mã hoặc một tài liệu thiết kế 20 trang, rõ ràng việc xem xét lại là không có hiệu quả. Lý do phổ biến nhất để đánh giá một xem xét lại là nghèo nàn là nó đã không được thực hiện với sự tập trung đầy đủ và nghiêm túc. Trừ khi việc xem xét lại được thực hiện nghiêm túc, chúng có thể sẽ là một sự lãng phí lớn về thời gian nếu không đem lại bất kỳ lợi ích nào, hoặc chúng có thể được xem như là một bước kiểm tra được thực hiện cầm chừng.

Làm thế nào để một người quản lý dự án hay người điều phối đánh giá xem một xem xét lại đã có hiệu quả không để mà cô ta có thể ra quyết định về các hoạt động tương lai? Một cách hiệu quả để theo dõi (monitoring) và kiểm soát (controlling) quy trình xem xét lại là sử dụng các nguyên lý của phương pháp Kiểm soát quy trình dùng thống kê (SPC - statistical process control) - được thực hiện thông qua các biểu đồ kiểm soát (control charts). Bởi vì số lượng điểm dữ liệu (data points) để xem xét lại - đặc biệt là để xem xét lại mã - có thể lớn, các kỹ thuật thống kê (statistical techniques) có thể được áp dụng với sự tự tin và chặt chẽ. Phần này thảo luận làm thế nào Infosys theo dõi, giám sát, và kiểm soát việc xem xét lại bằng cách sử dụng các kỹ thuật thống kê.

10.3.1 Baseline về khả năng xem xét lại (The Review Capability Baseline)

Làm thế nào SPC có thể được áp dụng để giám sát việc xem xét lại? Để áp dụng SPC, những người quản lý dự án phải xác định các tham số năng lực tới hạn (critical performance parameters), xác định các giới hạn kiểm soát (control limits) cho chúng, và sau đó theo dõi hiệu suất thực tế (actual performance). Chúng có thể xây dựng các biểu đồ kiểm soát (control charts) bằng cách vẽ (plotting) các tham số hiệu suất của các xem xét lại (reviewers) và sau đó sử dụng các biểu đồ để đánh giá hiệu quả của một xem xét lại. Một tiếp cận khác là đi thiết lập các giới hạn kiểm soát cho các tham số khác nhau và sau đó sử dụng phạm vi (range) để xác định hiệu quả. Mặc dù tiếp cận thứ hai này có những hạn chế bởi vì biểu đồ chạy (run chart) không có sẵn, nhưng nó rất dễ dàng để được áp dụng. Ở Infosys, tiếp cận thứ hai được sử dụng.

Ở Infosys, các giới hạn kiểm soát đã được xác định cho các tham số hiệu suất sau đây: tốc độ bao phủ trong suốt quá trình chuẩn bị (coverage rate during preparation), tốc độ bao phủ trong suốt cuộc họp xem xét lại bởi nhóm (coverage rate during the group review meeting), mật độ lỗi nhỏ hoặc nhẹ bên ngoài (defect density for minor or cosmetic defects), và mật độ lỗi nghiêm trọng hoặc lớn (defect density for serious or major defects) (mật độ lỗi tổng thể được tính bằng tổng của hai mật độ lỗi này). Các giới hạn này được xác định từ dữ liệu quá khứ (past data) và từ baseline về khả năng xem xét lại (review capability baseline). Tạo và bảo trì baseline này là những lý do quan trọng để thu thập dữ liệu tóm tắt của các xem xét lại. Bảng 10.2 cho thấy baseline về khả năng xem xét lại bởi nhóm.

Baseline về khả năng xem xét lại bởi nhóm trong Bảng 10.2 cung cấp, cho các loại sản phẩm công việc khác nhau, tốc độ bao phủ trong quá trình chuẩn bị, tốc độ bao phủ trong quá trình xem xét lại, và mật độ lỗi nhỏ và nghiêm trọng (mật độ lỗi tổng thể là tổng của hai mật độ này). Mật độ lỗi được chuẩn hóa (normalized) để tương ứng với kích thước, ở đây kích thước được đo bằng số lượng trang (pages) cho tất cả các sản phẩm công việc không phải mã (noncode) và bằng số dòng mã (LOC - lines of code) cho các sản phẩm công việc là mã (code). Đối với thiết kế, kích thước cũng có thể được đo bằng số lượng của các phát biểu đặc tả (specification statements). Tốc độ bao phủ được tính theo kích thước đã được thực hiện trên mỗi đơn vị nỗ lực (size per unit effort), ở đây nỗ lực được tính bằng người-giờ (person-hours). Như bạn có thể thấy, đối với các tài liệu, tốc độ bao

phủ và mật độ lỗi là khá giống nhau, nhưng chúng khác với mã (tại đây đơn vị của kích thước cũng khác).

Tốc độ (rates) xem xét lại bởi một người (one-person reviews) có thể sẽ khác đi. Các tài liệu thiết kế chi tiết, kế hoạch kiểm thử, và mã phải trải qua hình thức xem xét lại này một cách thường xuyên. Do đó, một baseline về xem xét lại bởi một người vừa được phát triển cho các sản phẩm công việc này. Trong baseline của việc xem xét lại các tài liệu (documents) bởi một người, tốc độ bao phủ mỗi giờ (LOC/hour hoặc pages/hour) thì lớn khoảng gấp đôi tốc độ bao phủ của việc xem xét lại được làm bởi nhóm; tốc độ phát hiện lỗi trên mỗi trang (defects/page) thì bằng khoảng một nửa tốc độ của xem xét lại bởi nhóm. Đối với mã, tốc độ bao phủ mỗi giờ thì giống nhau, nhưng tốc độ phát hiện lỗi trên mỗi LOC (defects/LOC) thì ít hơn khoảng 30%.

Baseline này là nền tảng để theo dõi quá trình xem xét lại trong dự án.

Bảng 10.2. Baseline về khả năng xem xét ở Infosys

Mục xem xét lại (Review Item)	**Tốc độ bao phủ ở giai đoạn chuẩn bị (Preparation Coverage Rate (If Different from Coverage Rate))**	**Tốc độ bao phủ của xem xét lại bởi nhóm (Group Review Coverage Rate)**	**Mật độ lỗi nhẹ bên ngoài/nhỏ (Defect Density Cosmetic/Minor)**	**Mật độ lỗi nghiêm trọng/lớn (Serious/Major Defect Density)**
Tài liệu yêu cầu *(Requirements)*	5–7 pages/hour	0.5–1.5 defects/page	0.1–0.3 defects/page	
Thiết kế mức cao *(High-level design)*	4–5 pages/hour (or 200–250 specification statements/hour)	0.5–1.5 defects/page	0.1–0.3 defects/page	
Thiết kế chi tiết *(Detailed design)*	3–4 pages/hour (or 70–100 specification statements/hour)	0.5–1.5 defects/page	0.2–0.6 defects/page	
Mã *(Code)*	160–200 LOC/hour	110–150 LOC/hour	0.01–0.06 defects/LOC	0.01–0.06 defects/LOC

Kiểm thử tích hợp *(Integration test plan)*	5–7 pages/hour	0.5–1.5 defects/page	0.1–0.3 defects/page	
Các ca kiểm thử tích hợp *(Integration test cases)*	3–4 pages/hour			
Kế hoạch kiểm thử hệ thống *(System test plan)*	5–7 pages/hour	0.5–1.5 defects/page	0.1–0.3 defects/page	
Các ca kiểm thử hệ thống *(System test cases)*	3–4 pages/hour			
Kế hoạch quản lý dự án và quản lý cấu hình *(Project management and configuration management plan)*	4–6 pages/hour	2–4 pages/hour	0.6–1.8 defects/page	0.1–0.3 defects/page

10.3.2 Hướng dẫn phân tích và kiểm soát (Analysis and Control Guidelines)

Các phạm vi được cung cấp bởi baseline được sử dụng để xác định xem hiệu suất của việc xem xét lại có nằm trong các giới hạn có thể chấp nhận được hay không. Kiểm tra này được xác định như một tiêu chuẩn xuất (exit criterion) cho quy trình xem xét lại (review process). Những người quản lý dự án có thể xác định tiêu chuẩn xuất bằng cách kiểm tra xem tất cả các tham số có nằm trong phạm vi (in-range) không, nhưng bởi vì phát hiện lỗi là mục đích trung tâm của việc xem xét lại, tiêu chuẩn xuất là mật độ lỗi tổng thể (overall defect density) nên nằm trong các giới hạn quy định. (Một lựa chọn khác là đi kiểm tra xem mật độ lỗi của hai loại lỗi có nằm trong các phạm vi hợp lý không.) Nếu số lượng lỗi được tìm thấy trong quá trình xem xét lại nằm trong phạm vi của baseline, việc

xem xét lại được coi là hiệu quả, các tiêu chuẩn xuất được thỏa, và không cần phải làm thêm bất cứ hoạt động nào cho việc xem xét lại này.

Thay vì sử dụng baseline về khả năng xem xét lại (review capability baseline), một người quản lý dự án có thể giám sát việc xem xét lại bằng cách sử dụng một công cụ Kiểm soát quy trình dùng thống kê (SPC tool) do công ty tự phát triển. Công cụ này về bản chất là một bảng tính (spreadsheet) có dữ liệu baseline về khả năng (capability baseline data) riêng cho nó. Nó cũng có dữ liệu về tỷ lệ tiêm lỗi (defect injection rate), hiệu quả loại bỏ lỗi (defect removal efficiencies), khả năng chất lượng toàn tổ chức (organization-wide quality capability), v.v. Từ những dữ liệu này, công cụ sẽ xác định các đặc tả hiệu suất (performance specifications) cho một quá trình xem xét lại – tức là, phạm vi mà kết quả của nó được dự kiến sẽ rơi vào nếu mục tiêu chất lượng được thỏa. Công cụ SPC cung cấp các cảnh báo (warnings) khi một phần dữ liệu của quá trình xem xét lại nằm bên ngoài các giới hạn kiểm soát hoặc bên ngoài các giới hạn dự kiến..

Nếu mật độ lỗi được tìm thấy của quá trình xem xét lại không nằm trong phạm vi được cung cấp trong baseline, không có nghĩa là quá trình xem xét lại đã thất bại. Người quản lý dự án hoặc người điều phối cần phải đánh giá tình hình và làm quyết định cho các bước tiếp theo. Tốc độ chuẩn bị (preparation rate) và tốc độ xem xét lại (review rate) trở nên rất hữu ích ở đây; nếu tốc độ xem xét lại là "quá nhanh" (too fast) so với baseline, lý do cho sự không hiệu quả của xem xét lại là tương đối rõ ràng. Mật độ các lỗi nhỏ (minor defects) và nghiêm trọng (critical defects) cũng có thể hữu ích cho phân tích này. Mặc dù người điều phối hoặc người quản lý dự án có thể sử dụng bất kỳ kỹ thuật nào để xác định nguyên nhân gây ra chênh lệch hiệu suất và thực hiện các hành động khắc phục và phòng ngừa cần thiết, các hướng dẫn được trình bày trong Bảng 10.3 sẽ giúp ích cho nỗ lực này.

Bảng 10.3 chứa hai nhóm hướng dẫn: 1) các hướng dẫn được áp dụng khi mật độ lỗi thấp hơn (nằm dưới) phạm vi, 2) các hướng dẫn được áp dụng khi mật độ lỗi cao hơn (nằm trên) phạm vi. Cả hai trường hợp đề nghị rằng có một điều gì đó bất thường có thể đã diễn ra, và tình hình này cần phải được kiểm tra một cách cẩn thận. Bảng 10.3 liệt kê một số nguyên nhân có thể; người lãnh đạo dự án hay người điều phối có thể sử dụng thông tin này để xác định nguyên nhân và sau đó quyết định các hoạt động cần thiết để khắc phục quy trình xem xét lại và cũng đề nghị các hoạt động phòng ngừa cho hoạt động xem xét lại trong tương lai.

Các giới hạn kiểm soát tĩnh (static control limits) làm việc tốt khi quy trình đang hoạt động trong một trạng thái ổn định. Tuy nhiên, nếu các thay đổi được thực hiện lên quy trình, các biểu đồ kiểm soát (control charts) phải được sử dụng cẩn thận bởi vì hiệu suất của quy trình (process performance) được dự kiến là sẽ thay đổi. Nếu những thay đổi trong quy trình xảy ra thường xuyên, tốt nhất là nên có các giới hạn kiểm soát động (dynamic control limits). Một cách để làm được điều này là thiết lập lại các giới hạn kiểm soát bằng các giá trị mới dựa trên n điểm dữ liệu hiệu suất trong quá khứ (past *n* performance data points) (n phải được lựa chọn; nó có thể là một con số lớn hơn khoảng 10 đến 15). Với phương pháp này, nếu quy trình này bị thay đổi, hiệu suất của nó sẽ thay đổi, và sau một vài điểm dữ liệu, các giới hạn kiểm soát sẽ phản ánh khả năng hiệu suất của quy trình vừa bị thay đổi.

Một phương pháp khác là điều chỉnh dữ liệu hiệu suất (performance data) hoặc các giới hạn kiểm soát (control limits) dựa vào tác động dự kiến của các thay đổi quy trình. Ví dụ, nếu quy trình xem xét lại bị thay đổi và nếu nó được dự kiến rằng quy trình xem xét lại sẽ sẽ phát hiện nhiều thêm 10% lỗi nữa, khi đó nếu một điểm nằm ngoài bên ngoài các giới hạn kiểm soát, sự thật này nên được chú ý để phân tích.

Bảng 10.3. Các hướng dẫn để phân tích cho xem xét lại

Lý do có thể (Possible Reason)	**Các hoạt động để xem xét (Actions to Consider)**
Nếu số lỗi được phát hiện ít hơn các tiêu chuẩn (If Defects Found Are Less Than Norms)	
Sản phẩm công việc đã rất đơn giản. *(Work product was very simple.)*	• Chuyển đổi việc xem xét lại bởi nhóm thành việc xem xét lại bởi một người cho cùng các sản phẩm công việc này. *(Convert group reviews of similar work products to one-person reviews.* • Kết hợp các xem xét lại. *(Combine reviews.)*
Quá trình xem xét lại có thể không kỹ lưỡng, triệt để. *(Reviews may not be thorough.)*	• Kiểm tra lại tốc độ bao phủ; nếu quá chậm, xếp lại lịch xem xét lại, có thể bổ sung thêm một nhóm khác. *(Check coverage rate; if too low, reschedule a review, perhaps with a different team.)*
Những người xem xét lại không được	• Xếp lịch hoặc hướng dẫn đào tạo về xem xét lại bởi

đào tạo đầy đủ về xem xét lại bởi nhóm hoặc thiếu kinh nghiệm về các tài liệu cần xem xét lại. *(Reviewers do not have sufficient training on group reviews or experience with the reviewed material.)*	nhóm. *(Schedule or conduct group review training.)* • Làm lại công việc xem xét lại bởi một nhóm khác. *(Re-review with a different team.)*
Sản phẩm công việc có chất lượng rất tốt. *(Work product is of very good quality.)*	• Xác nhận thực tế này bằng cách nhìn vào tốc độ bao phủ, kinh nghiệm của tác giả, những người xem xét lại, v.v.; xem chất lượng này có thể được lặp lại ở các phần khác của dự án hay không. *(Confirm this fact by coverage rate, experience of the author, reviewers, and so on; see if this quality can be duplicated in other parts of the project.)* • Sửa lại dự đoán lỗi trong dòng các hoạt động đang tiếp diễn, xem có thể rút ra những bài học gì để cải tiến quy trình chung không. *(Revise defect prediction in downstream activities; see if there are general process improvement lessons.)*
• **Nếu số lỗi được phát hiện nhiều hơn các tiêu chuẩn (If Defects Found Are More Than Norms)**	
Sản phẩm công việc có chất lượng thấp. *(Work product is of low quality.)*	• Kiểm tra nhu cầu đào tạo cho tác giả. *(Examine training needs for author.)* • Có các sản phẩm công việc làm lại. *(Have the work product redone.)* • Hãy phân công lại các công việc trong tương lai (ví dụ, chỉ phân công các công việc dễ hơn cho tác giả). *(Consider reassigning future tasks (e.g., assign easier tasks only to the author))*
Sản phẩm công việc rất phức tạp. *(Work product is very complex.)*	• Đảm bảo xem xét lại hoặc kiểm thử chính xác *(Ensure good review or testing downstream.)* • Tăng cường ước tính cho kiểm thử hệ thống *(Increase estimates for system testing.)* • Phân chia sản phẩm công việc ra thành những phần nhỏ hơn

	(Break the work product into smaller components.)
Có quá nhiều lỗi nhỏ (và quá ích lỗi lớn) *(There are too many minor defects (and too few major defects).)*	• Xác định nguyên nhân gây ra các lỗi nhỏ; sửa chữa lại trong tương lai bằng cách mở rộng danh sách kiểm tra hợp lý và giúp các tác giả nhận thức về các nguyên nhân phổ biến. *(Identify causes of minor defects; correct in the future by suitably enhancing checklists and making authors aware of the common causes.)* • Người xem xét lại có hiểu biết không đầy đủ về sản phẩm công việc. Nếu vậy, hãy tổ chức một cuộc họp tổng quan hoặc cần xem xét lại hoạt động xem xét lại này bởi những người khác. *(Reviewer may have insufficient understanding of the work product. If so, hold an overview meeting or have another review with different reviewers.)*
Tài liệu tham khảo được dùng cho hoạt động xem xét lại này thì không chính xác và rõ ràng. *(Reference document against which review was done is not precise and clear.)*	• Tài liệu tham khảo cần được xem xét lại và được phê duyệt. *(Get the reference document reviewed and approved.)*
Các mô-đun được xem xét lại là những mô-đun đầu tiên của dự án. *(Reviewed modules are the first ones in the project.)*	• Phân tích các lỗi, cập nhật lại danh sách kiểm tra cho xem xét lại, và thông báo cho các nhà phát triển biết. Xếp lịch trình đào tạo. *(Analyze the defects, update the review checklist, and inform developers. Schedule training.)*

Cách tiếp cận thứ hai này được sử dụng tại Infosys khi phòng ngừa lỗi (DP - defect prevention) được sử dụng trong suốt quá trình xem xét lại. Với DP, tỷ lệ tiêm lỗi (defect injection rate) dự kiến sẽ giảm. Do đó, mật độ lỗi được phát hiện trong phần xem xét lại có thể cũng sẽ giảm. Nếu một dự án đang sử dụng DP, khi đó trong suốt quá trình lập kế hoạch cho dự án, tác động dự kiến của DP cũng được ghi lại. Sử dụng tỷ lệ tiêm lỗi (defect injection rate), số lượng lỗi sẽ giảm theo dự kiến (expected reduction) từ việc sử dụng DP, và tốc độ phát hiện lỗi (defect detection rate), các *giới hạn được mong muốn* (*expected limits*) đối với hiệu suất (performance) được thiết lập. Nếu hiệu suất của hoạt

động xem xét lại nằm ngoài những giới hạn này, cần phải cẩn thận kiểm tra các nguyên nhân.

10.3.3 Ví dụ

Hãy xem xét báo cáo tóm tắt của hoạt động xem xét lại bởi nhóm cho một kế hoạch quản lý dự án được đưa ra ở bảng 10.4. Tóm tắt này chứa một xem xét lại bởi nhóm cho một một kế hoạch quản lý dự án 14-trang. Tổng số lỗi nhỏ và nhẹ bên ngoài (minor and cosmetic defects) được tìm thấy là 16, và tổng số lỗi lớn được tìm thấy là 3. Như vậy, mật độ lỗi được tìm thấy là 16/14 = 1.2 lỗi nhỏ trên mỗi trang, và 3/14 = 0.2 lỗi lớn trên mỗi trang. Cả hai mật độ này nằm trong phạm vi được cung cấp bởi baseline về khả năng (capability baseline), do đó, các tiêu chuẩn xuất được thỏa và có thể giả định rằng việc xem xét lại đã được thực hiện đúng cách.

Bảng 10.4. Báo cáo tóm tắt của hoạt động xem xét lại

Dự án (Project)	
Loại sản phẩm công việc (Work product type) Kích thước sản phẩm (Size of product) Người điều phối (Moderator) Người xem xét lại (Reviewers) Tác giả (Author)	Project plan, v. 1.0 14 trang (14 pages) Meera Biju, Meera JC
Nỗ lực (Người-giờ) (Effort (Person-Hours))	
a. Họp tổng quan (Overview meeting) b. Chuẩn bị (Preparation) c. Họp xem xét lại bởi nhóm (Group review meeting) Nỗ lực tổng cộng (Total Effort)	0 10 person-hours 10 person-hours 20 person-hours
Lỗi (Defects)	
Số lỗi nghiêm trọng (Number of critical defects) Số lỗi lớn (Number of major defects) Số lỗi nhỏ (Number of minor defects) Số lỗi nhẹ bên ngoài (Number of cosmetic defects) Số lỗi được phát hiện trong suốt giai đoạn chuẩn bị (Number of defects detected during preparation) Số lỗi được phát hiện trong suốt cuộc họp xem xét lại bởi nhóm	0 3 12 4 — —

(Number of defects detected during group review meeting) Số vấn đề mở (Number of open issues raised) Số lỗi tổng cộng (Total number of defects)	 1 19
Kết quả (Result)	Người điều phối kiểm tra lại (Moderator reexamination)
Các đề nghị cho giai đoạn kế tiếp (Recommendations for Next Phase)	
Các đơn vị (unit) cần được xem xét lại bởi nhóm *(Units to undergo group review)* Các đơn vị (unit) cần được xem xét lại bởi một người *(Units to undergo one-person review)*	N/A N/A
Chú thích (người điều phối) **(Comments (Moderator))**	Kế hoạch đã được lập tài liệu và được trình bày tốt *(The plan has been well documented and presented.)*
Được chuẩn bị bởi (Prepared by): Meera; **Ngày (Date):** xx-xx-xxxx	

Mặc dù không được cần cho hoạt động xem xét lại này bởi vì các tiêu chuẩn xuất đã được thỏa, nhưng các tỷ lệ khác có thể cũng cần được kiểm tra. Nhóm xem xét lại gồm có 4 thành viên, mỗi người trong số họ đã dành 2.5 giờ trong việc xem xét lại độc lập bởi mỗi cá nhân họ (individual review), cuộc họp xem xét lại bởi nhóm (review meeting) kéo dài 2.5 giờ. Như vậy, tốc độ bao phủ trong quá trình chuẩn bị và xem xét lại là 14/2.5 = 5.6 trang một giờ, giá trị này nằm trong phạm vi tốc độ của quá trình chuẩn bị (baseline là 4–6 pages/hour), nhưng cao hơn một chút so với tốc độ xem xét lại bởi nhóm (baseline là 2–4 pages/hour).

Khi dữ liệu tóm tắt được cung cấp cho công cụ Kiểm soát quy trình dùng thống kế (SPC tool), nó được hiển thị bằng đồ họa hiệu suất của xem xét lại này, cùng với các giới hạn kiểm soát và giới hạn được mong muốn.

10.4 GIỚI THIỆU VỀ XEM XÉT LẠI VÀ HỘI CHỨNG NAH (INTRODUCTION OF REVIEWS AND THE NAH SYNDROME)

Các xem xét lại, trong nhiều trường hợp, thì phản trực giác (khác thường). Một lập trình viên không thể hiểu nổi làm thế nào xem xét lại bởi một nhóm người có thể có hiệu quả nhiều hơn kiểm thử. Khi nỗ lực của con người (human effort) là nguồn tài nguyên quan trọng nhất trong một dự án, thì không phải là dễ dàng gì để chấp nhận một điều là quy trình xem xét lại tiêu tốn rất nhiều sức người lại có thể giúp quy trình tổng thể tăng cao năng xuất và chất lượng. Kết quả là, việc thuyết phục mọi người sử dụng xem xét lại là một trong những nhiệm vụ khó khăn nhất khi triển khai quy trình. Một báo cáo của Viện công nghệ phần mềm (SEI) chỉ ra rằng chỉ có 22% các công ty phần mềm sử dụng một số hình thức của thanh tra (inspections) [8].

Rõ ràng, dữ liệu cứng (hard data) là vô giá để chứng minh trường hợp này. Một số lượng khá lớn dữ liệu đã được công bố để chứng minh cho lời khẳng định rằng xem xét lại có thể mang lại hiệu quả về mặt chi phí (cost-effective) và có thể cải thiện đáng kể chất lượng. Tuy nhiên, dữ liệu như thế đã được công bố từ các công ty (và tổ chức) trên khắp thế giới thường thất bại trong việc thuyết phục các kỹ sư rằng việc xem xét lại như vậy có thể cũng tốt cho công ty (tổ chức) của họ. Một lý do cho sự hoài nghi này là hội chứng Không Áp Dụng Được Ở Đây (NAH - Not Applicable Here): Người ta tin rằng xem xét lại thì tốt cho các công ty (tổ chức) khác, nhưng hoàn cảnh (tình hình) ở công ty của họ thì có sự khác biệt và do đó xem xét lại không thể áp dụng được [9].

Nếu thanh tra (inspections) được triển khai trong một dự án, hội chứng NAH phải được vượt qua (overcome). Các nhà quản lý cũng như các nhà phát triển phải được chỉ cho thấy rằng thanh tra có thể có thể mang lại lợi ích thực sự. Theo định nghĩa, dữ liệu từ các công ty (tổ chức) khác không thể được sử dụng để vượt qua hội chứng NAH. Thay vào đó, dữ liệu từ bên trong của chính công ty (tổ chức) phải được sử dụng để xây dựng một trường hợp (case) cho thanh tra. Do đó, một thiết lập thử nghiệm được cần đến để nhanh chóng triển khai các kịch bản (tình huống) thực tế, từ đó đi đánh giá xem thanh tra có phù hợp với công ty (tổ chức) hay không.

Để vượt qua hội chứng NAH, Infosys đã sử dụng một thí nghiệm (thử nghiệm) đơn giản, được mô tả trong phần tiếp theo. Thí nghiệm này là nói chung và đơn giản và có thể được

thực hiện một cách dễ dàng bởi bất kỳ công ty (tổ chức) nào. Các thông tin chi tiết về thí nghiệm này đã được trình bày trong một một bài báo đồng tác giả [9].

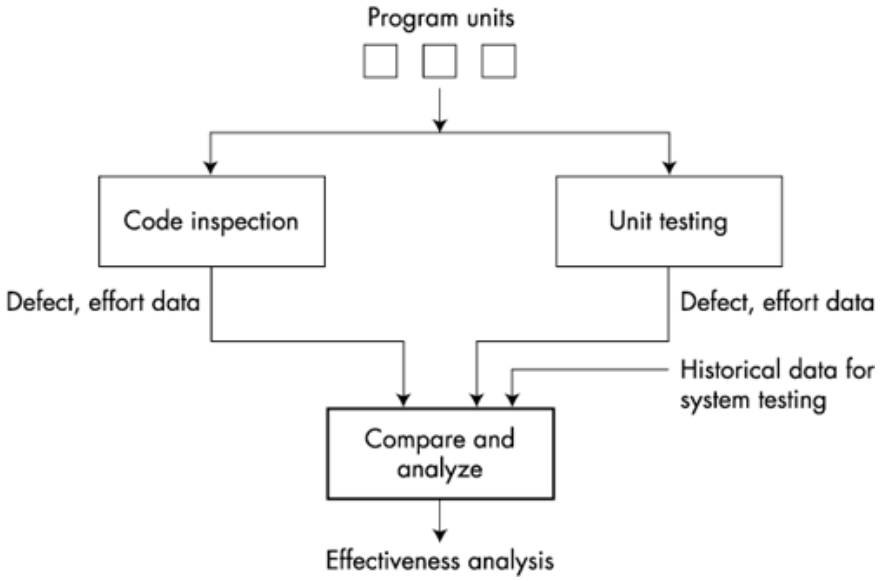
10.4.1 Thí nghiệm ở Infosys

Thí nghiệm của Infosys tập trung vào thanh tra mã (code inspection) bởi vì các nhà phát triển liên quan đến cài đặt mã và nó thường là nguồn chứa nhiều lỗi nhất. Trong lịch sử, thanh tra đã bắt đầu với mã và chỉ được mở rộng sau đó cho thiết kế (design), yêu cầu (requirements), kế hoạch kiểm thử (test plans), và các mục (items) khác. Một khi bạn xây dựng một trường hợp (case) mạnh mẽ để thanh tra mã và chúng được triển khai, các ưu điểm của thanh tra sẽ trở nên rõ ràng và từ đó sẽ xây dựng một trường hợp thanh tra cho các sản phẩm công việc khác.

Để dùng cho thí nghiệm của Infosys, sáu yêu cầu nâng cấp hệ thống (SERS - system enhancement requests) cho một sản phẩm ngân hàng (banking product) đã được lựa chọn. Sáu nhà phát triển đã được phân công: một người thực hiện một SER. Những nhà phát triển này trước tiên đã được đào tạo về quy trình thanh tra (inspection process); để thực hành, họ đã được yêu cầu thanh tra sự cài đặt (implementation) của một SER – mà trước đó nó đã được gieo trước các lỗi.

Các nhân viên được chia ra thành hai nhóm – mỗi nhóm gồm ba nhà phát triển. Mỗi nhóm là một nhóm thanh tra (inspection team). Trước khi nộp SER (submitting), mỗi nhà phát triển đã được yêu cầu để cài đặt SER, biên dịch mã của mình, và tự làm một số kiểm thử (self-testing). Một khi được nộp, SER đã đi qua hai con đường độc lập: thanh tra và kiểm thử đơn vị (unit testing). Trong suốt thí nghiệm, mỗi nhóm thanh tra đã thanh tra mã của ba SERS vừa được phát triển bởi các thành viên của nhóm. Trong mỗi thanh tra, tác giả, chứ không phải là người điều phối hoặc người đọc, đã làm công việc thanh tra (inspecter). Song song với hoạt động thanh tra, các SERS được kiểm thử đơn vị một cách độc lập bởi người lãnh đạo mô-đun (module leader) cho lĩnh vực ứng dụng (domain) mà SER thuộc về. Sơ đồ dòng công việc trong Hình 10.4 cho thấy các bước cơ bản của thí nghiệm.

Hình 10.4. Các bước trong thí nghiệm của Infosys



Đối với mỗi trong hai đường, nỗ lực (công sức) bỏ ra và lỗi được tìm thấy đã được ghi nhận lại. Những dữ liệu này sau đó được sử dụng để đánh giá hiệu quả về mặt chi phí (cost-effectiveness) và chất lượng (quality) của các xem xét lại. Nếu tập hợp các lỗi được tìm thấy bởi hai phương pháp này là không giống nhau và nếu một tập hợp lỗi không là một tập hợp con của tập kia, khi đó sẽ có một khẳng định rằng các thanh tra đã tìm ra một tập lỗi khác với tập lỗi do kiểm thử đơn vị tìm thấy và rằng sử dụng thêm thanh tra sẽ mang lại lợi ích. Việc hiểu rõ tác động đến chi phí thì khó khăn hơn (và là đây là chỗ mà hầu hết các nghi ngờ phát sinh). Chi phí này đã được ước lượng (ước tính) dựa vào dữ liệu quá khứ cho thử nghiệm hệ thống (system testing).

10.4.2 Dữ liệu từ thí nghiệm (Data from the Experiment)

Bảng 10.5 cung cấp kích thước của các SERs (size SER), nỗ lực tổng cộng (total effort), và số lượng các lỗi (number of defects) đã được tìm thấy theo hai đường. Rõ ràng, đường thanh tra (inspections) đã xác định được nhiều lỗi hơn so với đường kiểm thử đơn vị (unit testing). Kết quả này đã được quan sát một cách nhất quán ở tất cả các SERS. Nhìn chung, thanh tra bắt được khoảng 2.5 lần nhiều lỗi hơn kiểm thử đơn vị, mặc dù thanh tra dùng nhiều nỗ lực hơn so với kiểm thử đơn vị. Tuy nhiên, số lỗi được phát hiện trên mỗi người-giờ (person-hour) thì giống nhau ở thanh tra và kiểm thử đơn vị, cả hai đều phát hiện khoảng 1.9 lỗi trên mỗi người-giờ.

Bây giờ chúng ta hãy nhìn vào bản chất của các lỗi đã được tìm thấy bằng hai phương pháp, được trình bày trong Bảng 10.6. Hầu như trong tất cả các loại lỗi, thanh tra bắt được nhiều lỗi hơn so với kiểm thử đơn vị, đặc biệt là đối với các loại lỗi liên quan đến các thuộc tính chất lượng (quality attributes) như khả năng di chuyển, khả năng bảo trì, v.v. (như dự kiến thì kiểm thử thường tập trung vào các lỗi chức năng). Dữ liệu cũng cho thấy rằng, ngay cả trong các lỗi lôgic và lỗi giao diện (là mối quan tâm của kiểm thử), thanh tra cũng làm tốt hơn so với kiểm thử đơn vị. Từ những dữ liệu này, việc bổ sung thanh tra để cải thiện khả năng phát hiện lỗi thì rõ ràng và thuyết phục.

Bảng 10.5. Dữ liệu về nỗ lực và lỗi

	Thanh tra (Inspections)			**Kiểm thử đơn vị (Unit Testing)**	
Kích thước của SER (Size SER)	**Nỗ lực tổng cộng (Total Effort)**		**Tổng số lỗi được phát hiện (Total Number of Defects)**	**Nỗ lực tổng cộng (Total Effort)**	**Tổng số lỗi được phát hiện (Total Number of Defects)**
	(LOC)	**(Hours)**		**(Hours)**	
1	968	8.0	8	2.0	4
2	432	5.0	8	1.5	3
3	85	4.0	4	1.5	1
4	667	6.5	26	1.5	7
5	50	12.5	3	1.5	0
6	408	2.5	5	2.5	5
Tổng (Total)	**2,610**	**27.5**	**54**	**10.5**	**20**

Bảng 10.6. Phân phối lỗi

Loại lỗi (Defect Type)	**Thanh tra (Inspections)**	**Kiểm thử đơn vị (Unit Testing)**	**Lỗi chung (Common Defects)**
Dữ liệu (Data)	3	1	0
Chức năng (Function)	4	2	0
Giao diện (Interface)	14	11	7

Lôgic (Logic)	12	5	4
Khả năng bảo trì (Maintainability)	11	0	0
Khả năng di chuyển (Portability)	5	0	0
Khác (Other)	5	1	1
Tổng (Total)	**54**	**20**	**12**

Theo dữ liệu này, chi phí cho mỗi lỗi thì gần giống nhau. Tuy nhiên, xem xét lại (reviews) mang đến các lợi ích về chi phí. Kinh nghiệm của quá khứ và dữ liệu đã chỉ ra rằng mất khoảng 4 người-giờ để xác định và loại bỏ một lỗi trong quá trình kiểm thử hệ thống. Nếu một lỗi bị bỏ sót trong quá trình kiểm thử hệ thống, phải mất khoảng 2.5 người-ngày (17 người-giờ) để xác định và loại bỏ nó.

Kiểm thử hiếm khi bắt được các loại lỗi về khả năng bảo trì và khả năng di chuyển. Chúng tôi giả định rằng tất cả các lỗi dữ liệu, chức năng, giao diện, và lôgic nếu không được bắt được bằng kiểm thử đơn vị thì sẽ được tìm thấy sau này. Số lượng các lỗi như thế trong Bảng 10.6 tương ứng là 3, 4, 14, và 12. Sau khi loại bỏ các lỗi thông thường – những lỗi này cũng được bắt bằng kiểm thử đơn vị - số lượng là 3, 4, 7, và 8. Nếu tất cả các lỗi này được bắt trong kiểm thử hệ thống, khi đó chi phí của kiểm thử hệ thống sẽ tăng lên 22 * 4 = 88 giờ, hoặc khoảng 11 người-ngày. Nếu 75% của các lỗi này được bắt trong kiểm thử hệ thống và 25% bị bắt sau đó, chi phí cho kiểm thử hệ thống là 0.75 * 22 * 4 = 66 giờ (9.5 người-ngày); chi phí bổ sung để sửa lỗi được tìm thấy sau đó là 0.25 * 22 * 2.5 = 11 người-ngày. Tức là, nếu không có thanh tra được thực hiện, sẽ cần thêm 20.5 người-ngày để sửa chữa các lỗi còn sót lại.

Vì vậy, chi phí tiết kiệm nhờ thanh tra là 11 người-ngày nếu tất cả các lỗi được bắt trong kiểm thử hệ thống và 20.5 người-ngày nếu 25% lỗi không được bắt trong kiểm thử hệ thống. Chi phí của thanh tra, đã mang lại các khoản tiết kiệm này, là khoảng 3.5 người-ngày. Trường hợp này là rõ ràng: Nếu chúng ta xài thêm một ngày để thanh tra mã, cho sản phẩm này, chúng ta có thể mong đợi sẽ tiết kiệm được 3 đến 6 ngày để sửa chữa lỗi sau này trong chu kỳ phát triển!

Thí nghiệm đã chỉ kéo dài khoảng hai tuần, nhưng nó đã có một tác động đáng kể. Các kết quả đã thuyết phục các nhà phát triển và người quản lý. Các dữ liệu từ thí nghiệm cũng chỉ ra rằng việc thanh tra sẽ mang lại ít lợi ích hơn nếu mã thì đơn giản hoặc nhỏ

(trong các SERS nhỏ hơn, lợi ích đem lại thì không mấy ấn tượng). Do đó, nhóm phát triển phần mềm cho ngân hàng (banking product team) đã làm các quyết định mang tính chính sách là đi phân loại các SERS ra làm ba loại (đơn giản, trung bình, và phức tạp) và thực hiện thanh tra theo đúng cách thức (formal inspections) cho tất cả các mô-đun phức tạp.

10.5 TÓM TẮT

Mục đích của xem xét lại (review) là để xác định lỗi và các vấn đề trong một sản phẩm công việc (work product) thông qua một quy trình xem xét lại hình thức (formal review) và có cấu trúc (structured review) được thực hiện bởi một nhóm người. Xem xét lại mang lại hiệu quả về mặt chi phí và thậm chí có thể áp dụng được cho các sản phẩm công việc không thể thực thi (chạy) được. Xem xét lại là một kỹ thuật quan trọng để cải thiện cả về chất lượng và năng suất cũng như cung cấp khả năng nhìn thấy được tình trạng (trạng thái) của dự án.

Sau đây là một số bài học kinh nghiệm từ quy trình xem xét lại tại Infosys:

- Cần có các chuyên gia bên ngoài tham gia vào nhóm xem xét lại để làm tăng thêm năng lực của nhóm.

- Sử dụng một quy trình xem xét lại được xác định rõ ràng (well-defined review process) và có cấu trúc (structured review process) cùng với các hướng dẫn rõ ràng (clear guidelines) và thu thập dữ liệu hình thức (formal data collection). Quy trình nên chứa các kế hoạch, tự xem xét lại (self-review), và một cuộc họp nhóm.

- Trong suốt quy trình xem xét lại, tập trung chính vào việc tìm kiếm các lỗi và các vấn đề. Lỗi và các vấn đề sẽ được giải quyết sau đó.

- Tùy vào thực tế, sử dụng xem xét lại bởi một người cho các sản phẩm công việc. Để thực hiện xem xét lại bởi một người, hãy theo (follow) cùng một quy trình và các hướng dẫn thu thập dữ liệu như của xem xét lại bởi nhóm.

- Giám sát hiệu quả của mỗi lần xem xét lại. Tạo hiệu suất mong đợi (performance expectations) từ các dữ liệu quá khứ, và sử dụng chúng để đánh giá hiệu quả của quy trình xem xét lại hiện tại.

- Nếu hiệu suất của một xem xét lại không như mong đợi, phân tích nguyên nhân và có những hành động khắc phục và phòng ngừa.

- Để hiểu được tác động của việc xem xét lại, tiến hành các thí nghiệm đơn giản trong phạm vi dự án. Dữ liệu được thu thập từ chính công ty nên được dùng để thuyết phục những người trong công ty về lợi ích của việc xem xét lại.

Tương ứng với CMM, các thực hành xem xét lại được mô tả ở đây đáp ứng các KPA Xem Xét Lại (Peer Review KPA) của CMM mức 3. Phương pháp theo dõi và kiểm soát (monitoring and control method) đáp ứng một số yêu cầu của các KPA Quản Lý Dự Án Định Lượng (Quantitative Project Management) và Quản Lý Chất Lượng Phần Mềm (Software Quality Management) của CMM mức 4. Việc xem xét lại các sản phẩm công việc khác nhau đáp ứng các yêu cầu xem xét lại của nhiều KPAs.

10.6 CÁC THAM KHẢO

1. M.E. Fagan. Design and code inspections to reduce errors in program development. *IBM System Journal*, (3), 1976.
2. M.E. Fagan. Advances in software inspections. *IEEE Transactions on Software Engineering*, SE-12(7), 1986.
3. T. Gilb and D. Graham. *Software Inspection*. Addison-Wesley, 1993.
4. 4. R.B. Grady and T.V. Slack. Key lessons learned in achieving widespread inspection use. *IEEE Software*, July 1994.
5. E.F. Weller. Lessons learned from three years of inspection data. *IEEE Software*, Sept. 1993.
6. R.G. Ebenau and S.H. Strauss. *Software Inspection Process*. McGraw Hill, 1993.
7. D.P. Freedman and G.M. Weinberg. *Handbook of Walkthroughs, Inspections, and Technical Reviews: Evaluating Programs, Projects, and Products*. Dorset House, 1990.
8. D.H. Kitson and S.M. Masters. An analysis of SEI software process assessment results: 1987–1991. *Proceedings of the 15th International Conference on Software Engineering*, 1993.
9. P. Jalote and M. Haragopal. Overcoming the NAH syndrome for inspection deployment. *Proceedings of the 20th International Conference on Software Engineering*, 1998

Chương 11. Giám sát và kiểm soát dự án

Một kế hoạch dự án, dù cho được chuẩn bị kỹ lưỡng tới đâu, vẫn chỉ là một mảnh giấy. Trong suốt quá trình thực hiện dự án, bác sĩ quản lý dự án phải cẩn thận theo dõi sức khỏe của dự án - bệnh nhân của mình - và cung cấp cho đúng liều thuốc là các hành động khắc phục khi cần thiết. Nếu đúng liều được đưa ra vào đúng thời điểm, bệnh nhân sẽ sống sót. Tuy nhiên, nếu thất bại trong việc xác định đúng triệu chứng và không để cung cấp đúng những viên thuốc đắng kịp thời thì có thể dẫn đến biến chứng nhiều hơn nữa và bệnh nhân có thể tử vong. Bên dưới là hai trường hợp nhỏ minh họa cho điểm này.

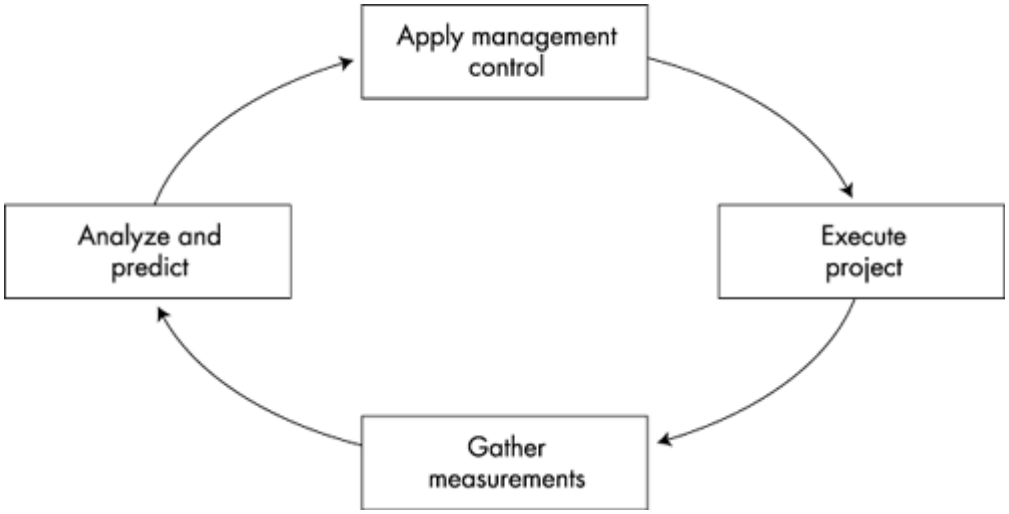
Trường hợp A: Shiva là người quản lý dự án chịu trách nhiệm phát triển một hệ thống giao dịch an toàn (secure transaction system) cho một công ty quản lý quỹ lớn. Nhóm của anh ta, bao gồm nhiều người tận tụy nhưng còn trẻ tuổi và có ít kinh nghiệm về bảo mật máy tính (computer security). Kiểm thử thử đơn vị (unit testing) cho vài mô-đun đầu tiên đã tìm thấy một số lượng lớn lỗi – rất nhiều hơn so với dự kiến. Dựa trên kết quả phân tích, Shiva kết luận rằng bởi vì các chương trình thì khó và sự thiếu kinh nghiệm của các lập trình viên, mã được tạo ra có nhiều lỗi. Do đó, ông đã thấy, nhiều lỗi hơn đã xuất hiện ở giai đoạn tích hợp và kiểm thử hệ thống (integration and system testing), và nếu không có những hoạt động nào đó được thực hiện, nhiều lỗi sẽ được giao cho khách hàng. Các hoạt động khắc phục (sửa chữa) và phòng ngừa lỗi mà Shiva đã thực hiện là đã thành lập một nhóm kiểm thử riêng biệt, đứng đầu nhóm là một người thông thạo về các vấn đề trong kiểm thử. Thông qua thảo luận với khách hàng, ông đã kéo dài giai đoạn kiểm thử hệ thống (system testing) lên 15 ngày. Cuối cùng, mặc dù đã bị trễ hạn 15 ngày, nhưng hệ thống đã được giao đáp ứng đúng mục tiêu chất lượng – điều này đã được thể hiện ở chỗ nó có ít lỗi hơn so với dự kiến ở giai đoạn kiểm thử chấp nhận.

Trường hợp B: Sau giai đoạn thiết kế chi tiết, phân tích tại một cột mốc (milestone) trong dự án của Bala cho thấy rằng mặc dù không có sự trượt lịch, nhưng nỗ lực (chi phí) đã vượt 40%. Bởi vì lịch đã không bị trượt, không có hành động nào được thực hiện. Tuy nhiện, tại cột mốc ở một tháng sau đó, dự án đã cho thấy có sự trễ hạn một tuần - mà Bala đã giải thích nguyên nhân là do có những tình huống (trường hợp) đặc biệt. Cuối cùng, việc phát triển cũng được hoàn tất nhưng bị trễ hạn một tháng. Để hoàn thành nó, số lượng lỗi được tìm thấy trong kiểm thử chấp nhận (acceptance testing) đã nhiều lần lớn hơn mục tiêu chất lượng ban đầu mà Bala đã đưa ra. Cuối cùng, dự án của ông đã không thành công trên cả ba phương diện: nỗ lực (chi phí), tiến độ và chất lượng. Phân tích sau

đó đã cho thấy rằng ông đã hiểu lầm phạm vi (scope) của hệ thống và vì thế đã hiển nhiên ước lượng thấp (underestimate) nỗ lực. Nếu phạm vi hoặc thời gian biểu đã được thương lượng lại khi vấn đề đầu tiên xuất hiện sau cột mốc thiết kế (design milestone), kết quả cuối cùng có lẽ đã hoàn toàn khác.

Những trường hợp nhỏ này đưa ra hai khía cạnh then chốt của việc giám sát dự án. Thứ nhất, những người quản lý dự án phải có khả năng nhìn thấy được tình trạng thực sự của dự án, mà cách tốt nhất là phải đo lường định lượng các tham số then chốt (key parameters) [1][2]. Đối với những người quản lý dự án, việc sử dụng các số đo phần mềm sẽ cung cấp khả năng nhìn thấy được (visibility) tình trạng của dự án [3]. Thứ hai, khả năng nhìn thấy được tự nó không giải quyết bất kỳ vấn đề nào. Những người quản lý dự án đúng phải hiểu được dữ liệu, và nếu họ nhận thấy rằng dự án không đang di chuyển theo con đường như kế hoạch, họ phải áp dụng các hành động khắc phục thích hợp để đưa nó trở lại đường ray (track). Việc thu thập dữ liệu để cung cấp thông tin phản hồi (feedback) về tình trạng hiện tại và có bất kỳ hành động khắc phục cần thiết đã hình thành nên một mô hình cơ bản của quản lý dự án. Hình 11.1 minh họa chu trình kiểm soát này [4].

Hình 11.1. Chu trình giám sát và kiểm soát dự án



Chương này mô tả làm thế nào chu trình giám sát và kiểm soát dự án được áp dụng tại Infosys. Đây là chương dài nhất trong cuốn sách và chứa một loạt các hoạt động giám sát (monitoring), bao gồm cả báo cáo tình trạng (status reporting), phân tích tại cột mốc (milestone analysis), kiểm soát sự kiện (event-level control) bằng cách dùng phương pháp

Kiểm soát quy trình dùng thống kê (SPC), kiểm tra quy trình (process audits), và phân tích để phòng ngừa lỗi (defect prevention).

11.1 THEO DÕI DỰ ÁN (PROJECT TRACKING)

Trước tiên, hãy kiểm tra lại quá trình theo dõi dự án tại Infosys. Một khi kế hoạch chi tiết của dự án được làm xong - trong đó tất cả các công việc đã được liệt kê cùng với ngày bắt đầu và kết thúc của chúng và mọi người đã được phân công để thực hiện chúng - người quản lý dự án phải theo dõi việc thực hiện các công việc này. Cố gắng xây dựng các hoạt động theo dõi. Ngoài ra, một người quản lý dự án phải theo dõi những vấn đề chưa được giải quyết và các lỗi vừa được phát hiện. Kết quả của việc theo dõi này được ghi lại trong bản báo cáo tình trạng (trạng thái) của dự án. Bản báo cáo tình trạng này cũng được phát cho người quản lý cao cấp và khách hàng, vì thế đảm bảo rằng họ có khả năng nhìn thấy được tình trạng (trạng thái) của dự án.

11.1.1 Theo dõi các hoạt động (Activities Tracking)

Một trong những nhiệm vụ đầu tiên trong việc theo dõi dự án là đảm bảo rằng các hoạt động đã được hoạch định sẽ được thực hiện đúng thời gian. Như đã đề cập trước đây, các hoạt động ở Infosys thường được xếp lịch bằng cách sử dụng Microsoft Project (MSP). Do đó, MSP cũng được sử dụng cho các hoạt động theo dõi.

Mỗi ngày (hoặc lâu hoặc ít thường xuyên hơn), người quản lý dự án kiểm tra tình trạng của các nhiệm vụ đã được xếp lịch và cập nhật lại tình trạng vào trong MSP. Mặc dù MSP cho phép người sử dụng có thể xác định một hoạt động vừa được hoàn thành một phần, nhưng khi dùng cho mục đích theo dõi, một hoạt động thường được xác định là đã được thực hiện 0% (0% done) cho đến khi nó hoàn thành (completed). Khi một hoạt động kết thúc, nó được đánh dấu là đã được thực hiện 100% (100% done). Mức độ theo dõi này là đủ bởi vì các nhiệm vụ đã được phân chia để mà nó đã ở mức thấp nhất, các nhiệm vụ cần ít hơn một hoặc hai ngày để hoàn thành. Sử dụng các tính năng của MSP, một người quản lý dự án khi đó có thể xác định được những nhiệm vụ mức cao hơn đang bị trễ lại phía sau, bao nhiêu phần trăm của một nhiệm vụ đã được thực hiện, ảnh hưởng của trượt lịch lên dự án tổng thể, v.v. Đối với giai đoạn hiện hành (ở cột mốc quan tiếp theo), dựa trên các hoạt động đã hoàn thành và thời gian thực tế mà chúng đã xài, người quản lý dự án xếp lại lịch cho các nhiệm vụ còn lại để đảm bảo cột mốc được hoàn thành đúng hạn.

11.1.2 Theo dõi lỗi (Defect Tracking)

Theo dõi lỗi là một điểm then chốt khác trong việc theo dõi. Như đã đề cập trong Chương 7, Infosys sử dụng một hệ thống kiểm soát lỗi (defect control system) để theo dõi lỗi. Một khi thông tin về một lỗi được nhập vào hệ thống này, nó vẫn mở (open) cho đến khi lỗi được sửa chữa (fixed) xong. Lỗi được đánh dấu là "đóng" (closed) khi việc gỡ bỏ nó đã được xác nhận. Bằng cách này, từng lỗi được ghi nhận và theo dõi cho đến khi được "đóng" lại. Đôi khi, các nhà quản lý dự án theo dõi tình trạng (trạng thái) của một lỗi bằng cách ghi nhận lại ngày nộp, ngày đóng lỗi. Vào cuối dự án, tốt nhất là không còn lại lỗi mở (open defects) (hoặc làm một quyết định thận trọng là bỏ lại vài lỗi chưa được sửa chữa). Tại bất kỳ thời điểm nào, người quản lý dự án có thể kiểm tra tỷ lệ tiêm lỗi tổng thể (overall rate of defect injection) và tỷ lệ đóng lỗi (overall rate of defect closure), thông tin này sẽ hữu ích cho việc nhìn thấy được khoảng cách giữa số lượng lỗi đã được nộp và số lượng lỗi đã được đóng lại có nằm trong tầm kiểm soát hay không.

Bảng 11.1 cho thấy các phần của dữ liệu về lỗi của case study - dự án ACIC. Ngoài ra, còn có các cột khác (không được hiển thị ở đây) lưu lại ID của người nộp (submitter ID), ngày nộp (submit date), ID của chủ sở hữu (owner ID) (tức là, ID của người được phân công để sửa chữa lỗi đó), và ngày đóng (closing date). Như bạn có thể thấy, thông tin chuẩn, chẳng hạn như giai đoạn tiêm (stage injected), giai đoạn phát hiện (stage detected), mức độ nghiêm trọng (severity level), v.v. được ghi lại cho mỗi lỗi.

Bảng 11.1. Dữ liệu về lỗi của dự án ACIC

ID	Tên (Title)	Mô tả (Description)	Giai đoạn tiêm vào (Stage Injected)	Giai đoạn phát hiện (Stage Detected)	Loại lỗi (Defect Type)	Mức độ nghiêm trọng (Severity)
7	Câu lệnh import dư thừa *(Redundant import statement.)*	Trong ActivitiesMaintenanceJPanel, lớp AccountSearchResult được import hai lần.	Cài đặt mã *(Coding)*	Xem xét lại mã *(Code review)*	Mã dư thừa *(Redundant code)*	Nhỏ *(Minor)*
10	Quy ước đặt tên	Panel vừa được đặt tên là	Cài đặt	Xem xét lại	Chuẩn	Nhẹ bên

	không được áp dụng theo *(Naming convention not followed.)*	ActivityMaintJPanel, và nó nên là ActivitiesMaintenanceJPanel *(The panel has been named ActivityMaintJPanel, and it should be ActivitiesMaintenanceJPanel.)*	mã *(Coding)*	mã *(Code review)*	*(Standards)*	ngoài *(Cosmetic)*
15	Quản lý giao dịch *(Transaction handling.)*	Giao dịch đã bắt đầu đọc mà không cần được cam kết như các phương pháp khác có thể làm. Nó được thực hiện trong addTask và updateTask. *(Transaction started for reading need not be committed as other methods can make use of it. It is done in addTask and updateTask.)*	Cài đặt mã *(Coding)*	Xem xét lại mã *(Code review)*	Logic	Nhỏ *(Minor)*
16	Cấu trúc Panel không phù hợp *(Panel structure not proper.)*	Trong panel Options/Services của View History, Regy cản trở panel và panel Options/Services panel được đặt trên một panel cha. Tuy nhiên, phương pháp đúng đắn là phải đặt nó lên panel Option Services. *(In Options/Services panel of View History, Regy block panel and Options/Services panel were put on a parent panel. However, the correct method would have been to place it on the Option Services Panel).*	Thiết kế chi tiết *(Detailed design)*	Xem xét lại thiết kế chi tiết *(Detailed design review)*	UI	Nhỏ *(Minor)*

21	Việc phân lớp không được thực hiện theo *(Layering not followed.)*	Mô hình nghiệp vụ Party được import trong ViewPartyHistoryPanel *(Business model Party is imported in ViewPartyHistoryPanel.)*	Cài đặt mã *(Coding)*	Xem xét lại mã *(Code review)*	Kiến trúc *(Architecture)*	Lớn *(Major)*
22	Chuẩn UI không được theo *(UI standards not followed.)*	Các nhãn trong màn hình không có màu đen *(The labels in the screens are not in black.)*	Cài đặt mã *(Coding)*	Xem xét lại mã *(Code review)*	Chuẩn *(Standards)*	Nhẹ bên ngoài *(Cosmetic)*
26	Ép kiểu không đúng *(TypeCasting is incorrect.)*	Ép kiểu thành IIndividualName và IOrganizationName trong các phương thức getIndividualNames và getOrganizationNames nên được đổi thành IName. *(TypeCasting to IIndividualName and IOrganizationName in methods getIndividualNames and getOrganizationNames should be changed to IName.)*	Cài đặt mã *(Coding)*	Xem xét lại mã *(Code review)*	Logic	Lớn *(Major)*
27	Gán không đúng *(Incorrect assigning)*	Tác giả muốn gán giá trị của vector đang tồn tại cho một vector khác. Việc gán không được làm đúng đắn. Nó tạo ra một vector mới có kích thước như vector đang tồn tại *(Author wanted to assign the value of existing vector to*	Cài đặt mã *(Coding)*	Xem xét lại mã *(Code review)*	Logic	Nghiêm trọng *(Critical)*

		another vector. The assigning is not done correctly. It creates a new vector of the existing vector size.)				
30	Chấp nhận giá trị Null *(Null Value acceptance)*	Cần kiểm tra lại xem các giá trị null có được phép không *(Need to check whether null values are allowed.)*	Cài đặt mã *(Coding)*	Kiểm thử đơn vị *(Unit testing)*	Logic	Lớn *(Major)*
35	Thông tin tài chính *(Financial info)*	Việc click vào các panel khác không nên xóa bỏ các trường trên màn hình đang chứa thông tin tài chính *(Clicking on other panels should not clear the fields on the screen for financial info.)*	Cài đặt mã *(Coding)*	Kiểm thử đơn vị *(Unit testing)*	UI	Lớn *(Major)*
39	Thông tin tài chính *(Financial info)*	Việc quản lý biệt lệ trong panel làm tươi từ ApplicationContext chưa được làm *(Exception handling in refresh panel from ApplicationContext not done.)*	Cài đặt mã *(Coding)*	Kiểm thử đơn vị *(Unit testing)*	UI	Lớn *(Major)*
152	Thiết đặt Tab *(Tab setting)*	Việc di chuyển Tab không được thiết đặt trong panel *(Tab movement is not set in the panel.)*	Cài đặt mã *(Coding)*	Kiểm thử tích hợp *(Integration testing)*	Hiệu suất *(Performance)*	Nhẹ bên ngoài *(Cosmetic)*
153	Nút Add bị làm mất khả năng dùng *(Disabled Add*	Nút Add bị làm mất khả năng dùng trong Address sau khi một vai trò được chọn trong bảng Account *(Add button is disabled in*	Cài đặt mã *(Coding)*	Kiểm thử tích hợp *(Integration*	Hiệu suất *(Performance)*	Lớn *(Major)*

	button)	*Address after a role is selected in the Account Roles Table.)*		*testing)*		
154	Nút Clear không xóa bảng AssociatedAccounts *(Clear button not clearing AssociatedAccounts table)*	Nút Clear trong panel bảo trì địa chỉ không xóa được bảng Associated Accounts *(Clear button in address maintenance panel is not clearing the Associated Accounts table.)*	Cài đặt mã *(Coding)*	Kiểm thử tích hợp *(Integration testing)*	Hiệu suất *(Performance)*	Nhỏ *(Minor)*
155	Tách điện thoại ra không làm việc *(Phone dissociation not working)*	Trong panel phoneMaintenance, việc tách điện thoại ra không làm việc được *(In phoneMaintenance panel, dissociating the phone is not working.)*	Cài đặt mã *(Coding)*	Kiểm thử tích hợp *(Integration testing)*	Hiệu suất *(Performance)*	Lớn *(Major)*
156	Cập nhật điện thoại không làm việc được *(Updating phone not working)*	Khi bạn cập nhật một số điện thoại, các thay đổi không được lưu lại *(When you update a phone number, the changes are not saved.)*	Cài đặt mã *(Coding)*	Kiểm thử tích hợp *(Integration testing)*	Hiệu suất *(Performance)*	Lớn *(Major)*
157	Cập nhật E-mail không làm việc được	Theo một trình tự nhất định các hành động – cập nhật một e-mail không có click vào nút Update, chọn một vai trò khác, click vào Yes trên hộp thoại xác nhận – và sau đó chọn vai trò trước đây, chỉ cập nhật e-mail được trình bày; bỏ sót các e-mail khác	Cài đặt mã	Kiểm thử tích hợp	Hiệu suất	Lớn

	(E-mail updating not working)	*(Following a certain sequence of actions—updating an e-mail without clicking the button Update, selecting another role, clicking Yes on the confirmation dialog—and then selecting the previous role, only the updated e-mail is shown; other e-mails are missing.)*	*(Coding)*	*(Integration testing)*	*(Performance)*	*(Major)*
158	Loại tài khoản không được cập nhật *(Account type not updated)*	Khi bạn thay đổi loại tài khoản và click Update, loại tài khoản không được lưu lại *(When you change the account type and click Update, the account type is not saved.)*	Cài đặt mã *(Coding)*	Kiểm thử tích hợp *(Integration testing)*	Hiệu suất *(Performance)*	Lớn *(Major)*

11.1.3 Theo dõi các vấn đề (Issues Tracking)

Chắc chắn, nhiều công việc nhỏ (small jobs) hoặc giải thích chi tiết (clarifications) xảy ra trong suốt quá trình thực hiện dự án. Những việc này được gọi là *các vấn đề (issues)*. Quản lý các vấn đề này là một nhiệm vụ quan trọng đối với bất kỳ nhà quản lý dự án nào bởi vì chúng có thể rất nhiều và có khả năng làm trì hoãn một dự án. Ví dụ, việc giải thích chi tiết để làm rõ một yêu cầu của khách hàng (một vấn đề) có thể làm trì hoãn nhiều hoạt động khác, trừ khi nó đã được giải quyết. Nhiều vấn đề như vậy có khả năng làm dừng một số hoạt động. Do đó, sẽ rất quan trọng cho một người quản lý dự án để theo dõi và quản lý các vấn đề một cách đúng đắn.

Một phương pháp đơn giản là hãy viết lại những vấn đề và kiểm tra chúng trong suốt quá trình phát triển. Nhưng bởi vì bất kỳ thành viên nhóm nào cũng có thể đưa ra một vấn đề, danh sách các vấn đề có thể trở nên lớn hơn. Do đó, các phương pháp hình thức (formal methods) sẽ hữu ích cho việc theo dõi chúng. Để đạt được mục đích này, các dự án thường mở *một bản ghi các vấn đề (issues log).*

Trong bản ghi các vấn đề, các vấn đề được ghi nhận lại khi chúng phát sinh, cùng với thông tin liên quan. Khi các vấn đề được đóng lại, chúng được đánh dấu là đã được đóng (closed). Để tạo bản ghi vấn đề, những người quản lý dự án có thể sử dụng một bảng tính (spreadsheet), lưu giữ tài liệu (document), dùng một tiện ích theo dõi vấn đề (issue tracker utility), hoặc một hệ thống theo dõi lỗi (defect tracking system). Nếu một công cụ tự động được sử dụng, các truy vấn đơn giản là khả thi. Bảng 11.2 cung cấp một phần của bản ghi vấn đề của dự án ACIC sẽ cung cấp cho bạn ý tưởng về các loại mục (type of items) được ghi nhận vào trong nó (đây là bản ghi thực tế chứa ngày nộp và ngày đóng và các thông tin khác để được theo dõi).

Những người quản lý dự án phải thường xuyên theo dõi tình trạng (trạng thái) của vấn đề, đặc biệt là khi tất cả các thành viên trong nhóm đều có thể nhập chúng vào. Như đã đề cập trong Chương 8, các vấn đề chưa được giải quyết có thể gây ra rủi ro cho dự án. Nếu vấn đề vẫn còn mở quá lâu, các kênh leo thang (escalation channels) cần được sử dụng để giải quyết chúng, như đã được xác định trước trong kế hoạch quản lý dự án.

Bảng 11.2. Mẫu về một bản ghi vấn đề cho dự án ACIC

Mô tả vấn đề (Issue Description)	**Chú thích (Comments/Closure Comments)**	**Tình trạng (Status)**
Những trường nào của mẫu tin sẽ được gửi cho đối tác kinh doanh của ACIC? *(For what fields will the record be sent to the business partner of ACIC?)*	Trường hợp sử dụng đã được cập nhật. *(Use case updated.)*	Đóng *(Closed)*
Kịch bản thay thế số 1 trong Trường hợp sử dụng số 3 là không rõ ràng. Tại sao 9 có nghĩa là cập nhật thủ công? *(Alternate scenario 1 in UC 3 is not clear. Why does 9 mean manual update?)*	Kịch bản thay thế được thêm vào trong trường hợp sử dụng 4. *(Added alternate scenario in use case 4.)*	Đóng *(Closed)*
Wizard cho phép chỉ một giá trị cho cả Stmts và Confirms. Tại sao nó phải là khác với các màn hình *Maintenance*?	Giải thích được chứa trong Trường hợp sử dụng.	Đóng

(Wizard allows only one value for both Stmts and Confirms. Why should it be different for the Maintenance screens?)	*(Included the explanation in the use case.)*	*(Closed)*
Hãy giải thích rõ ràng tất cả các điều kiện khi Address Change được gửi cho đối tác kinh doanh của ACIC. *(Clarify all the conditions when the Address Change should be sent to the business partner of ACIC.)*		Đóng *(Closed)*
Chỉ số quan trọng đã được gỡ bỏ trong Synergy. Nếu nó được đưa trở lại, nó phải được đề cập như là một yêu cầu trong trường hợp sử dụng. *(Primary indicator was removed in Synergy. If it must be put back, it must be mentioned as a requirement in the use case.)*	Tham khảo đến PPT đã được gửi ngày hôm qua. Nó có các trường trên màn hình. *(Refer to the PPT sent yesterday. It has the field on the screen.)*	Đóng *(Closed)*
Trong Update, các kịch bản thay thế không được thảo luận. Điều gì xảy ra nếu e-mail được cập nhật có liên kết với các tài khoản khác? *(In Update, the alternate scenarios are not discussed. What if the updated e-mail is associated with other accounts?)*	Việc làm cho rõ ràng dễ hiểu được thêm vào cho các Trường hợp sử dụng. *(Clarification added to the use cases.)*	Đóng *(Closed)*
Trong bảng chứa các dịch vụ đã được lựa chọn, có một dấu ghi thời gian cho cả ba loại dịch vụ tùy chọn. Điều này sẽ không gây ra vấn đề gì khi thực hiện các tùy chọn riêng biệt? *(In the table in which history of service options is displayed, there is one timestamp for all three service options types. Will this not cause problems in dealing with the options separately?)*		Đóng *(Closed)*
Đối với các bảng, chúng nên chứa dữ liệu có ý nghĩa		Đóng

(Regarding populating the tables, they should be populated with some meaningful data.)		*(Closed)*
Nếu bất kỳ một thay đổi nào được thực hiện lên vai trò chịu trách nhiệm thuế hoặc bên quan tâm, nó sẽ ảnh hưởng đến các tài khoản khác? *(If any change is made on tax liable role or interested party, should it affect other accounts?)*	Điều này được đề cập trong trường hợp sử dụng. Vui lòng nhìn ở "Flow of Events" trong kịch bản chính. *(This is mentioned in the use case. Please look at "Flow of Events" in the main scenario.)*	Đóng *(Closed)*
Combo Box Country có cần được điền vào sẵn các giá trị mã quốc gia, hoặc nó phải được thay đổi sang một trường văn bản? *(Does Country Combo Box need to be prefilled with country code values, or does it have to be changed to a text field?)*		Mở *(Open)*
Trường văn bản ForeignAddressLine4 có thể bị xóa không? *(Can the text field ForeignAddressLine4 be deleted?)*		Mở *(Open)*
Foreign Address và Foreign Phone từ nhóm Bảo trì có thể chưa có để tích hợp vào. (Foreign Address and Foreign Phone number from the Maintenance team may not be ready for integration.)		Mở *(Open)*
Chuyển đổi sang VAJ 3.0 sẽ tác động đến dự án này. *(VAJ 3.0 conversion is going to impact this project.)*	Chuyển đổi này sẽ được thực hiện chỉ sau khi dự án này đã hoàn thành. *(This conversion will be done only after this project is completed.)*	Đóng *(Closed)*

11.1.4 Các báo cáo tình trạng (Status Reports)

Các báo cáo tình trạng là cơ chế chính để thường xuyên chung cấp thông tin về tình trạng (tình trạng) của dự án đến người quản lý cấp cao và khách hàng. Các bên sẽ nhận được các báo cáo tình trạng như đã được quy định cụ thể trong kế hoạch quản lý dự án. Thông thường, báo cáo tình trạng được tạo ra hàng tuần và chứa các mục sau đây:

- Các khiếu nại của khách hàng
- Các cột mốc đã đạt được trong tuần này
- Các cột mốc đã bị trượt (trễ) trong tuần này và các lý do của chúng
- Các cột mốc đã được lập kế hoạch cho tuần sau
- Các vấn đề cần phải làm được làm rõ hoặc được chú ý
- Leo thang, nếu cần thiết
- Thời gian dự kiến để thực hiện công việc so sánh với thời gian còn lại từ đây cho đến cột mốc kế tiếp

Rõ ràng, mối quan tâm nằm ở việc đảm bảo là dự án sẽ tiếp tục tiến triển theo kế hoạch và ở việc giải quyết các vấn đề chưa được giải quyết xong. Những người quản lý dự án cũng có thể kiểm tra "mức độ thoải mái” (comfort level) của dự án bằng cách xem thời gian còn lại có đủ để thực hiện dự án hay không. Người quản lý dự án có thể thêm vào các mục khác, chẳng hạn như số lượng thay đổi yêu cầu, và cũng có thể xóa bỏ bớt một trong những mục này.

Hình 11.2 cung cấp một mẫu bản báo cáo tình trạng của dự án ACIC. Báo cáo tình trạng hàng tuần này tập trung vào những gì đã được thực hiện ở tuần trước, những gì cần phải làm thêm, và còn lại những vấn đề mở nào, v.v.

Hình 11.2. Báo cáo tình trạng của dự án ACIC cho tuần kết thúc vào ngày 3 tháng 7

STATUS SUMMARY

Project	Life-Cycle Stage	Next Milestone Date	% Complete	Number of Resources	Remarks
Release 2.0	Build	7 July 2000	90%	4	
Release 3.0	Elaboration 1, Elaboration 2	28 June 2000	Elaboration 1: 95% Elaboration 2: 20%	5.5	There are two elaboration phases and three construction phases.

TASKS COMPLETED AND MISSED

	Tasks Completed	Tasks Missed
Release 2.0	• Construction of iteration 2 complete. • Code review done. • Test plans for iteration 2 and 3 done. • Independent test environment set up.	None
Release 3.0	• As part of elaboration 1, 10 use cases were taken up for design and 4 critical use cases for implementation. This is 95% complete. • Elaboration 2 started.	Review of the requirements will be taken up next week.

TASKS PLANNED FOR THE WEEK 4 JULY 2000 TO 9 JULY 2000

Release 2.0	• Construction for iteration 3 to be complete. • Independent testing of iteration 1 to continue. • Code review of iterations 2 and 3 to be performed. • Test plans for iterations 2 and 3 to be done.
Release 3.0	• Elaboration 2 • Complete architecture document. • Review BARs. • Schedule PMR review.

ISSUES/MISCELLANEOUS ITEMS

Open Issues	None
Misc. Items	None

11.2 PHÂN TÍCH TẠI CỘT MỐC (MILESTONE ANALYSIS)

Một bản báo cáo tình trạng cung cấp các cơ chế giám sát dự án thường xuyên. Nó tập trung chủ yếu vào việc xem dự án có kịp tiên độ không. Ưu điểm chính của các báo cáo tình trạng là chúng không cần nhiều dữ liệu đo (metrics) hoặc phân tích (analysis). Phần này thảo luận làm thế nào các số đo được sử dụng tại Infosys để đánh giá tình trạng (trạng thái) của dự án tại các cột mốc.

Nếu kế hoạch dự án đã được xây dựng và được đánh giá cẩn thận, dự án sẽ thành công nếu kế hoạch này được theo. Vì vậy, phân tích tại cột mốc nên sử dụng kế hoạch và lịch của dự án làm baseline và so sánh chúng với tiến độ thực tế. Chiến lược này được sử dụng trong hai phương pháp phổ biến để theo dõi dựa trên số đo (metrics-based tracking):

biểu đồ giá-lịch-cột mốc (cost-schedule-milestone chart) và phương pháp giá trị đạt được (EVM - Earned Value method) [5]. Phân tích tiến độ đã được hoạch định so với tiến bộ thực tế (planned versus actual progress) được coi là thực hành tốt nhất cho quản lý dự án [3]. Tại Infosys, tiến độ trong kế hoạch so với thực tế cũng được theo dõi tại các cột mốc.

Bởi vì mục tiêu chính của việc phân tích số đo (metrics analysis) đối với theo dõi (monitoring) và kiểm soát (control) là để có những hành động khắc phục và phòng ngừa kịp thời, phân tích như vậy cần được thực hiện đều đặn. Vì lý do này, một người quản lý dự án có thể xác định các cột mốc trong dự án theo hướng khách hàng (customer-driven milestones) để mà các cột mốc liên tiếp chỉ cách nhau một vài tuần.

11.2.1 Phân tích nỗ lực và lịch thực tế so sánh với dự kiến (Actual Versus Estimated Analysis of Effort and Schedule)

Đối với lịch và nỗ lực, hầu hết dự án điều có sự chênh lệch giữa tiến độ được hoạch định và tiến độ thực tế của nó. Tuy nhiên, các chênh lệch nhỏ ở nỗ lực có thể được coi là “bình thường" và không đáng được chú ý đặc biệt. Ngược lại, các chênh lệch "đáng kể" có nghĩa là dự án có thể đang hướng tới sự thất bại và do đó cần phải có thêm các hoạt động để phân tích và kiểm soát nó.

Để phân biệt sự khác nhau giữ “bình thường” và “đáng kể”, các giới hạn chênh lệch có thể chấp nhận được phải được thiết lập trong kế hoạch dự án, như được thảo luận trong Chương 7. Nhiều dự án tại Infosys có giới hạn chênh lệch nỗ lực (effort deviation) vào khoảng 20% và giới hạn chêch lệch lịch (schedule deviation) là 10%. Các giới hạn thực tế cho một dự án được quy định cụ thể trong kế hoạch quản lý của nó.

Nếu chênh lệch tại một mốc vượt quá các giới hạn này, nó hàm ý rằng dự án có thể sẽ đi vào rắc rối và có thể không đáp ứng được các mục tiêu của nó; do đó dưới áp lực về thời gian, các thành viên của dự án có thể bắt đầu dùng các con đường tắt (biện pháp nhanh chóng trực tiếp hơn) không mong muốn. Tình trạng này đòi hỏi người quản lý dự án phải hiểu được lý do cho sự thay đổi và áp dụng các hành động khắc phục và phòng ngừa nếu cần thiết. Bảng 11.3 liệt kê các hướng dẫn cho các hoạt động phân tích và kiểm soát mà một người quản lý dự án có thể xem xét.

Bảng 11.3 chứa các đề xuất cho cả hai loại thay đổi: nỗ lực ước lượng quá thấp hoặc nỗ lực ước lượng quá cao. Một số lý do được đưa ra, cùng với những hành động kiểm soát

có thể. Ví dụ, nếu ước lượng là quá thấp, các lý do có thể là quá lạc quan, sự tận dụng nhân sự thấp (có nghĩa là, có lãng phí trong dự án), một thành viên quan trọng vắng mặt, hoặc các thành viên trong nhóm có mức chuyên môn thấp. Đối với mỗi lý do, một số hoạt động kiểm soát đã được đề xuất. Ví dụ, nếu ước lượng đã là quá lạc quan, người quản lý dự án có thể xem xét lại dự toán (ước lượng), yêu cầu về nhân sự, hoặc cố gắng để thu hẹp quy mô dự án lại. Hầu hết các mục trong Bảng 11.3 thì có thể hiểu được (không cần giải thích) (self-explanatory).

Bảng 11.3. Các hướng dẫn cho hiệu suất nỗ lực/lịch

Lý do có thể (Possible Reason)	**Các hoạt động xem xét (Actions to Consider)**
Nếu thực tế nhỏ hơn ước lượng một lượng lớn hơn giới hạn cho phép ***(If Actual Is Less Than Estimate by More Than the Allowable Limit)***	
Các ước lượng cho các chương trình thì quá cao hoặc nhóm dự án có nhiều kiến thức và kinh nghiệm chuyên môn hơn dự kiến. *(Estimates for programs were too high or project team has more domain knowledge or experience than expected.)*	• Ước lượng lại các mô-đun tương lai. *(Reestimate for future modules.)* • Ước lượng lại nhân công *(Release resources.)*
Các nhiệm vụ đến thời điểm hiện tại vẫn chưa được thực hiện triệt để kỹ lưỡng. *(The tasks so far have not been thoroughly)* performed.	• Xem xét lại các nhiệm vụ đã được làm đến thời điểm hiện tại và xem xét lại lịch của các sản phẩm công việc vẫn chưa được xem xét lại. *(Review tasks done so far and schedule reviews for work products not reviewed.)* • Kiểm tra lại bản ghi vấn đề. *(Examine issues log.)*
• **Nếu thực tế lớn hơn ước lượng một lượng lớn hơn giới hạn cho phép** ***(If Actual Is More Than Estimate by More Than the Allowable Limit)***	
Kiến thức chuyên môn thấp. *(Low domain knowledge.)*	• Xếp lịch để đào tạo. *(Schedule training.)*
Kinh nghiệm phần mềm/viết mã của tác giả thấp. *(Low software/coding experience of*	• Phân công lại để tận dụng kinh nghiệm sẵn có. *(Reassign to leverage existing experience.)*

author.)	
Lĩnh vực công nghệ mới. *(New technology area.)*	• Ước lượng lại hoặc yêu cầu về nhân sự. *(Reestimate or request resources.)* • Đàm phán ngày giao sản phẩm. *(Negotiate delivery dates.)*
Các ước lượng đã quá lạc quan. *(Estimates were too aggressive.)*	• Xác định các thành phần cần bổ sung thêm nỗ lực, và xem lại ước lượng cho các hoạt động tương lai. *(Identify main components for extra effort, and revise estimate for future activities.)* • Yêu cầu về nhân sự. *(Request resources.)* • Đàm phán để thu hẹp quy mô dự án lại. (Negotiate to scale down project objectives.)
Tối ưu hóa nguồn nhân sự thấp. *(Resource optimization is low.)*	• Xếp lại lịch và xếp lại độ ưu tiên cho các nhiệm vụ, và xác định và loại bỏ "thời gian đang chờ đợi". *(Reschedule and reprioritize tasks, and identify and eliminate "waiting times.")*
Một thành viên quan trọng vắng mặt. *(Nonavailability of a critical resource.)*	• Leo thang vấn đề *(Escalate the issue.)* • Lấy nhân sự dự phòng *(Get a backup resource.)* • Xếp lại lịch, luôn nghĩ tới (các) nhân sự quan trọng trong đầu. *(Reschedule, keeping critical resource(s) in mind.)*
Quá nhiều công việc bị làm lại do chất lượng kém của các giai đoạn trước. *(Too much rework due to poor-quality of output of earlier phases.)*	• Xác định nguồn gốc của vấn đề và sửa lại chúng. *(Identify sources of problems and rectify them.)* • Thay đổi lịch của dự án. *(Change project schedule.)*

Các mẫu của sự chênh lệch nỗ lực (pattern of effort deviation) cũng có thể hữu ích cho việc phân tích. Nếu sự chênh nỗ lực luôn tiếp tục tăng với một tốc độ không thay đổi (consistently increasing) từ cột mốc này tới mốc khác, thậm chí nếu nó ở dưới ngưỡng, một số hành động có thể được thực hiện. Tương tự như vậy, nếu hành động kiểm soát được thực hiện và phần trăm chênh lệch đã được giảm ở cột mốc kế tiếp, nó cho thấy rằng các hành động đã mang lại hiệu quả như mong muốn. Ngược lại, nếu chênh lệch nỗ lực thậm chí còn tăng cao hơn nữa sau khi các hành động đã được thực hiện, nó ngụ ý

rằng các hành động trước đó đã không làm việc tốt và các hành động quyết liệt hơn có thể là cần thiết.

Ngoài hiệu suất quá khứ, tại mỗi cột mốc, người quản lý dự án có thể kiểm tra lại dự đoán cho phần còn lại của dự án. Đối với nỗ lực, trong phân tích tại cột mốc, số lượng nhân sự sẵn có và số lượng cần thiết cũng được báo cáo. Nếu số lượng nỗ lực có sẵn (dựa trên số nhân sự) là thấp hơn đáng kể hơn so với lượng cần thiết, rõ ràng là cần phải có các hành động nào đó. Việc đạt được khả năng nhìn thấy được trình trạng này (visibility) để mà có hành động kịp thời chính là mục đích của phân tích này.

11.2.2 Giám sát chất lượng (Monitoring Quality)

Để giám sát phương diện thứ ba - chất lượng - số đo chính là số lượng lỗi được tìm thấy. Ngoài ra, số lượng và tình trạng của các xem xét lại (reviews) đã được thực hiện và tình trạng và hiệu quả của các hoạt động phòng ngừa lỗi cũng được theo dõi. Bởi vì số lượng lỗi cũng được ước lượng trong suốt quá trình lập kế hoạch, lỗi được đánh giá theo cùng một cách thức như lịch và nỗ lực. Tức là, bằng cách sử dụng các mức độ lỗi (defect levels) được dự đoán trong kế hoạch dự án cho các giai đoạn khác nhau và số lượng lỗi thực tế được tìm thấy, người quản lý dự án chuẩn bị một phân tích so sánh giá trị thực tế với giá trị ước lượng. Nếu chênh lệch này vượt quá ngưỡng của dự án, phải phân tích nguyên nhân của sự chênh lệch này dựa trên quyết định mà cô ấy đã làm lên các hành động để mang dự án trở lại tầm kiểm soát. Bảng 11.4 đưa ra các hướng dẫn liên quan đến các lý do và các hành động kiểm soát có thể. (Chương 10 mô tả các hướng dẫn để đánh giá quá trình xem xét lại).

Đối với các nhiệm vụ kiểm soát chất lượng của kiểm thử và xem xét lại, bên cạnh số lượng lỗi được tìm thấy, người quản lý dự án cũng phân tích so sánh giá trị thực tế với giá trị ước lượng cho nỗ lực (effort) đã xài trong các nhiệm vụ này. Những dữ liệu này rất hữu ích cho việc phân tích tình hình khi mà số lượng lỗi thực tế đã được tìm thấy có khác biệt đáng kể so với số lượng lỗi ước lượng. Ví dụ, nếu có quá ít lỗi được tìm thấy sau khi kiểm thử hệ thống (system testing) và nếu lượng thời gian đã xài trong kiểm thử hệ thống cũng là quá ít, lý do để thực hiện kiểm tra suất trở nên rõ ràng hơn.

Bảng 11.4. Các hướng dẫn khi số lỗi được tìm thấy khác với ước lượng

Lý do có thể (Possible Reason)	Các hoạt động xem xét (Actions to Consider)
Nếu thấp hơn ước lượng (If Fewer than Estimate)	
Sản phẩm công việc có chất lượng cao. *(Work product is of high quality.)*	• Xác định lý do và xem có bài học nào được rút ra cho dự án hoặc cho quy trình hay không. *(Identify reason and see whether there are possible lessons for the project or for the process.)*
Kiểm thử chưa đủ. *(Inadequate testing.)*	• Kiểm tra lại nỗ lực đã xài cho kiểm thử; xem xét lại kế hoạch kiểm thử và mở rộng nó. *(Check the effort spent on testing; review the test plan and enhance it.)* • Xếp lịch cho kiểm thử thêm *(Schedule further testing*.)
Các hoạt động kiểm soát chất lượng trước đây đã được thực hiện rất triệt để, kỹ lưỡng. *(Very thorough execution of earlier quality control activities.)*	• Kiểm tra lại tất cả các bản ghi về xem xét lại và về kiểm thử. *(Examine all review and testing records for the project.)* • Kiểm tra xem có bài học nào được rút ra cho dự án hoặc cho quy trình hay không. *(Check whether there are possible lessons for the project or the process.)*
Các ước lượng về số lỗi thì quá cao. *(Defect estimates are too high.)*	• Xác định nguyên nhân và sửa chữa lại các ước lượng. *(Identify cause and correct estimates.)*
• **Nếu nhiều hơn ước lượng (If More than Estimate)**	
Cho đến thời điểm hiện tại, các hoạt động kiểm soát chất lượng chưa được thực hiện đầy đủ. *(Inadequate execution of quality control activities so far.)*	• Kiểm tra lại tất cả các bản ghi về kiểm thử và về xem xét lại. *(Examine all testing and review records.)* • Xếp lịch để xem xét lại các mô-đun quan trọng trước khi tiếp tục kiểm thử. *(Schedule reviews of critical modules before continuing with testing.)*

Việc xem xét lại và kiểm thử đơn vi đã được lập kế hoạch chưa đủ. *(Insufficient reviews and unit tests planned.)*	• Mở rộng kế hoạch kiểm thử và xếp lịch để kiểm thử thêm. *(Enhance test plan and schedule further testing.)* • Xem xét lại các ước lượng và các kế hoạch cho chấp nhận. *(Review estimates and plans for acceptance.)*
Các ước lượng về số lỗi thì quá thấp. *(Defect estimates are too low.)*	• Xác định nguyên nhân và sửa chữa lại các ước lượng. *(Identify cause and correct estimates.)*

Số đo thứ hai được giám sát là số lượng các xem xét lại đã được thực hiện. Khi có áp lực về thời gian, có xu hướng bỏ qua các xem xét lại đã được lên hoạch. Bởi vì những xem xét lại này là một phần quan trọng của kế hoạch chất lượng đã được phát triển để đạt được mục tiêu chất lượng của dự án, việc thực hiện để thỏa chúng là cần thiết. Do đó, báo cáo tại cột mốc chỉ ra những xem xét lại nào đã được lập kế hoạch và những xem xét lại nào đã thực sự được thực hiện. Hiệu suất thực tế của mỗi xem xét lại sẽ được theo dõi tại thời điểm hoàn thành xem xét lại đó, như được thảo luận trong Chương 10.

11.2.3 Giám sát rủi ro liên quan (Risk-Related Monitoring)

Rủi ro và các hoạt động liên quan cũng được theo dõi tại các cột mốc. Như đã thảo luận trong Chương 7, rủi ro của một dự án không phải là tĩnh; nhận thức về rủi ro thay đổi theo thời gian và khi các bước để giảm thiểu rủi ro (risk mitigation steps) được thực hiện. Vì vậy, rất quan trọng để đánh giá rủi ro và lưu ý về tác động của các bước để giảm thiểu rủi ro cho đến thời điểm hiện tại. Vì lý do này, nhận thức về rủi ro ở hiện tại, cùng với tình trạng của các bước để giảm thiểu rủi ro ở hiện tại, được báo cáo lại trong bản báo cáo phân tích tại cột mốc. Rõ ràng, nếu nguy cơ rủi ro (risk exposure) do nhiều rủi ro chưa được giảm, nó ngụ ý rằng các bước để giảm thiểu rủi ro đã không mang lại hiệu quả như mong muốn. Do đó, rủi ro và các bước để giảm thiểu tác động của nó phải được đánh giá lại.

Một bước để giảm thiểu rủi ro quan trọng là đào tạo (training), được đề nghị để đối phó với nhiều loại rủi ro. Đào tạo cũng là một hoạt động quan trọng trong bất kỳ dự án nào liên quan đến công nghệ mới hoặc những người mới và nếu không được thực hiện đúng cách

có thể sẽ tạo ra những rủi ro mới. Để tránh giám sát việc đào tạo khỏi bị trượt khỏi kế hoạch, báo cáo tại cột mốc sẽ mô tả các công việc đào tạo liên quan đến dự án đã được lên kế hoạch và đã thực sự được thực hiện.

Các thay đổi yêu cầu (requirement changes) cũng gây ra rủi ro cho dự án bởi vì chúng có tác động tiêu cực đến chi phí, tiến độ và chất lượng. Chương 3 giải thích quy trình quản lý thay đổi yêu cầu. Trong phân tích tại cột mốc, một bản báo cáo tóm tắt những thay đổi vừa được yêu cầu và tác động của chúng lên nỗ lực và tiến độ.

Các vấn đề (issue) đã phát sinh từ lâu mà vẫn chưa được giải quyết, cũng có thể trở thành rủi ro. Do đó, tình trạng của vấn đề cũng được báo cáo trong phân tích tại cột mốc. Ngoài ra, khiếu nại của khách hàng, trong đó nêu rõ các rủi ro nghiêm trọng của dự án, cũng được báo cáo. Đặc biệt, số lượng khiếu nại của khách hàng mới nhận được, số lượng khiếu nại của khách hàng đã được giải quyết, và số lượng khiếu nại của khách hàng đang chờ được giải quyết đều phải được báo cáo.

11.2.4 Phân tích tại cột mốc cho dự án ACIC

Chúng ta hãy nhìn vào một bản báo cáo phân tích tại một cột mốc của dự án ACIC. Phân tích này, được trình bày trong Hình 11.3, được thực hiện khi vòng lặp Xây dựng đầu tiên (first construction iteration) của dự án ACIC đã được hoàn thành. Nó cho thấy hầu như không có sự trượt ở tiến độ và nỗ lực, và lượng nỗ lực có sẵn gần bằng với lượng nỗ lực cần thiết để hoàn thành dự án. Do đó, dự án đang nằm trong tầm kiểm soát ở những phương diện này, và không có hành động nào cần được thực hiện thêm. Tuy nhiên, đối với phương diện chất lượng, báo cáo đã chỉ ra rằng mức độ lỗi (defects) đã cao hơn so với dự kiến bởi vì có nhiều lỗi hơn đã tiêm vào. Do đó, một hoạt động phòng ngừa lỗi (defect prevention activity) đã được thực hiện. Ngoài ra, tỷ lệ tiêm lỗi cho phần còn lại của dự án cũng đã được xem lại.

Hình 11.3. Báo cáo phân tích tại cột mốc của dự án ACIC

MILESTONE NAME: Construction: Iteration-1 **DATE:** Aug 18, 2000

Schedule (for the milestone)

Planned date	Actual date	Slippage
July 21, 2000	July 21, 2000	0

Reasons for deviation: N/A
Actions taken to bring schedule back in control: Nil
Overall impact on project: Nil

Effort (person-hours)

Estimated Effort Before This Milestone	Actual Effort Before This Milestone	Deviation	Estimated Effort from Milestone to Project End	Effort Available Until Project End
2019	2191	9	2231	2059

Reasons for Deviation: N/A
Actions Taken: Nil
Overall impact on project: Nil

Defects

No.	QC Activity	Size of Work Product	Estimated Effort from MSP (Person-Hours)	Actual Effort (Person-Hours)	Estimated Defects	Actual Defects	Defect Deviation
1	Code review and unit testing	Appx. 4000	31	25.5	25	57	125%

Reasons for Deviation: One of the use cases had too many defects.

Action Taken: A number of steps have been taken to prevent these defects from occurring. More details are given in the causal analysis report.

Impact on Quality Goals: The defect injection rate in the project may be higher than projected. It has been revised by 20%, and the quality plan has been suitably modified. Project goals remain unchanged, however. To minimize the future impact, DP activities will be done more vigorously.

Defect prevention activities

One causal analysis was done followed by a root cause analysis, based on which some preventive actions were recommended. These are being implemented. The potential impact of these, and the analysis, is given in the causal analysis report.

Requirements Change Tracking

Total number of major requirement changes to date	0
Total number of minor requirements changes to date	0

Risks

	Previous State					Current Status		
Sl. No.	Proba-bility	Im-pact	Expo-sure	Risk Item	Status of Mitigation Action	Proba-bility	Im-pact	Expo-sure
1	0.9	9	8.1	VAJ 3.0 conversion time line not known and could impact this project if planned prior to Release 3.0 delivery.	Escalated to customer and they are aware.	0.9	9	8.1
2	0.9	9	8.1	Time line for screen resizing not known.	Effort sized and given to the customer. No response yet.	0.9	9	8.1
3	0.8	8	6.4	Dependency on certain components by the onsite maintenance team for Rel 3.0.	Issue raised with the maintenance team. They may not have the bandwidth to work on this.	0.9	8	7.2
4	0.3	7	2.1	Dependency on Rel 2.0 for certain components used by Rel 3.0.	2.0 looks to be on time.	0.2	7	1.4
5	0.9	1	0.9	There is some mainframe work that must be taken up, and this has not been planned.	Customer has approved a mainframe resource. Not a risk anymore.	0	0	0

Customer Complaints

Number of customer complaints raised	0
Number of customer complaints closed	0
Number of customer complaints open	0

Training

Planned	Architecting Distributed Applications in J2EE for Bhaskar and Balajee
Actual	The same was completed

Group Reviews

Planned	None
Actual held	None

Issues

Risks 1, 2, and 3 are still issues. Follow up with the stakeholders for a resolution to these.

Người quản lý dự án đánh giá lại những rủi ro và phát hiện ra rằng, mặc dù các rủi ro mà dự án đang phải đối mặt thì khác với các rủi ro đã được xác định vào lúc bắt đầu dự án, nhưng không có thay đổi nào xảy ra ở các rủi ro và độ ưu tiên kể từ cột mốc trước đó. Báo cáo phân tích cũng chỉ ra rằng các hoạt động đào tạo đã được thực hiện như kế hoạch.

11.3 PHÂN TÍCH Ở MỨC HOẠT ĐỘNG BẰNG CÁCH SỬ DỤNG SPC

Phân tích tại cột mốc dùng các số đo (metrics) để theo dõi trạng thái của dự án tại các các cột mốc đã được xác định và để đề xuất các hành động khắc phục nếu cần thiết. Tuy nhiên, việc phân tích tại các cột mốc thì khó xác định được hoạt động nào trong nhiều hoạt động đã được thực hiện kể từ cột mốc trước đó có thể là nguyên nhân đã làm giảm hiệu suất. Tình trạng này cũng tương tự như thực hiện kiểm thử hệ thống; gỡ lỗi (debugging) thì khó khăn hơn nhiều khi hệ thống đang chứa nhiều đơn vị (units). Gỡ lỗi trong suốt quá trình kiểm thử đơn vị (unit testing) sẽ dễ dàng hơn nhiều. Khó khăn trong việc xác nguyên nhân gây ra giảm hiệu suất làm giới hạn các hành động khắc phục có thể được thực hiện.

Để cung cấp kiểm soát tốt hơn, phân tích ở mức hoạt động (activity-level analysis) cũng được thực hiện tại Infosys. Trong phân tích ở mức hoạt động, vài số đo được phân tích ngay lập tức sau khi một công việc vừa được thực hiện xong. Phân tích này được sử dụng để đánh giá hiệu quả của hiệu suất công việc (task performance). Nếu công việc đã không được thực hiện thỏa mãn, ngay lập tức sẽ có hành động được thực hiện để sửa nó và để ngăn chặn nó lặp lại.

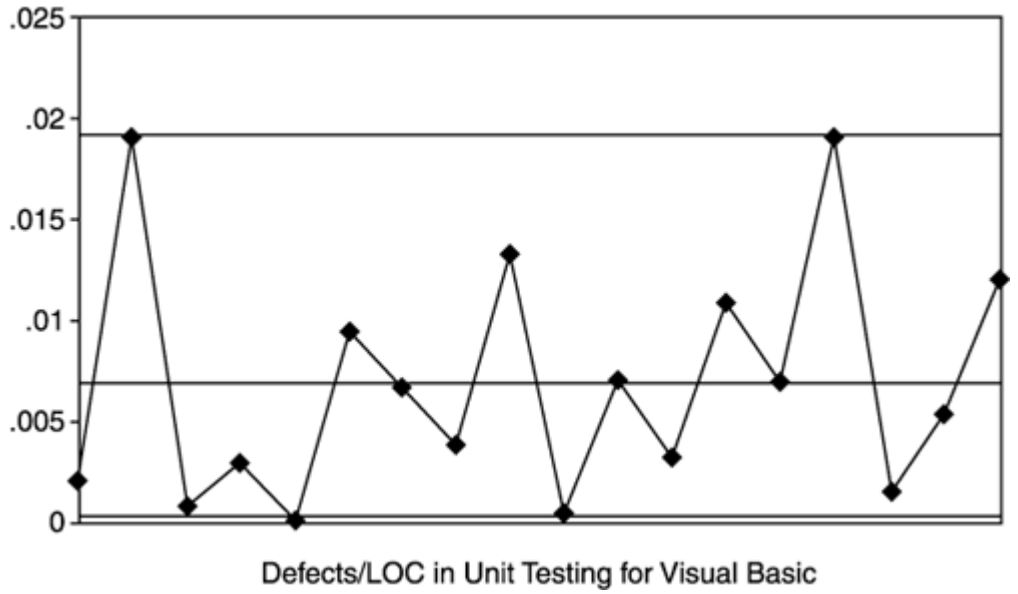
Phân tích ở mức hoạt động thường được thực hiện bằng cách sử dụng phương pháp Kiểm soát quy trình dùng thống kê (SPC - statistical process control). Ở Infosys, xem xét lại (reviews) và kiểm thử đơn vị (unit testing) được xác định là hai nhiệm vụ đã áp dụng SPC. Sử dụng biểu đồ kiểm soát (control charts), người quản lý dự án có thể đánh giá ngay lập tức tính hiệu quả của một xem xét lại hoặc một kiểm thử đơn vị. Nếu hiệu suất không được thỏa mãn, sẽ có hành động cần thiết được thực hiện.

Lưu ý rằng các hoạt động thiết kế, thiết kế chi tiết, kiểm thử hệ thống, v.v. đã được đánh giá riêng thông qua phân tích tại cột mốc bởi vì chúng thường bắt đầu và kết thúc tại các cột mốc. Do đó, trọng tâm của giám sát ở mức hoạt động (activity-level monitoring) là ở các hoạt động có liên quan đến mã (coding-related activities) của xem xét lại và kiểm thử đơn vị.

Chương 7 thảo luận về những vấn đề cơ bản của phương pháp Kiểm soát quy trình dùng thống kê (SPC), và Chương 10 giải thích quá trình theo dõi và kiểm soát các xem xét lại bằng cách sử dụng SPC. Ở đây chúng tôi thảo luận quá trình giám sát và kiểm soát pử mức hoạt động của kiểm thử đơn vị - một hoạt động có áp dụng SPC. Tiếp cận cho kiểm thử đơn vị thì tương tự như tiếp cận được sử dụng cho giám sát xem xét lại. Đặc trưng hiệu suất (performance characteristic) chính được giám sát là mật độ lỗi được phát hiện (density of defects detected) trong kiểm thử đơn vị. Từ các dữ liệu quá khứ về kiểm thử đơn vị, các giới hạn kiểm soát (control limits) (tức là, phạm vi của mật độ lỗi chấp nhận được) được đạt được. Vào cuối của mỗi kiểm thử đơn vị, mật độ lỗi được so sánh với phạm vi giới hạn. Nếu nó rơi bên ngoài các giới hạn, các hành động khắc phục hoặc phòng ngừa cần được thực hiện. Các hướng dẫn được đưa ra trước đây cho kiểm thử có thể được sử dụng để quyết định hành động nào là cần thiết (để phân tích, các nhà quản lý dự án có thể sử dụng các thông tin hỗ trợ như là nỗ lực được xài cho mỗi công việc, nỗ lực đã tiêu tốn cho các công việc trước đây, và lỗi được tìm thấy). Hình 11.4 cho thấy biểu đồ kiểm soát mật độ lỗi cho kiểm thử đơn vị của một ngôn ngữ lập trình. Mật độ lỗi có thể

được vẽ dưới dạng số lượng lỗi trên mỗi LOC (number of defects per LOC) hoặc số lượng lỗi trên mỗi người-giờ (defects per person-hour). Bảng 11.5 cho thấy baseline về khả năng của kiểm thử đơn vị, cung cấp tỷ lệ lỗi của vài ngôn ngữ.

Hình 11.4. Biểu đồ kiểm soát của kiểm thử đơn vị của các chương trình Visual Basic



Bảng 11.5. Tỷ lệ lỗi trong kiểm thử đơn vị của vài ngôn ngữ

Ngôn ngữ (Language)	Phạm vi tỷ lệ lỗi (trung bình) (Defect Rate Range (Average))
PB	0.0003–0.0266 defects/LOC (Avg: 0.008)
C	0.0004–0.0206 defects/LOC (Avg: 0.0052)
C++	0.00009–0.0067 defects/LOC (Avg: 0.0017)
RPG	0.0006–0.0075 defects/LOC (Avg: 0.0025)

Giống như với xem xét lại, những người quản lý dự án có thể sử dụng công cụ SPC để cài đặt kiểm soát ở mức hoạt động (activity-level control) của kiểm thử đơn vị. Công cụ này có chứa dữ liệu hiệu suất trong quá khứ chẳng hạn như tỷ lệ tiêm lỗi, tỷ lệ loại bỏ lỗi, v.v. Sử dụng dữ liệu này, nó xác định được các giới hạn kiểm soát (control limits) cũng như các giới hạn được mong đợi (expected limits). Khi dữ liệu hiệu suất của một kiểm thử đơn vị được nhập vào, công cụ này ngay lập tức nói cho biết là hiệu suất (performance) có nằm bên ngoài các giới hạn hay không. Nếu nó nằm ngoài giới hạn, phân tích sâu hơn phải được thực hiện để xác định những hành động cần được thực hiện, nếu có.

11.4 PHÂN TÍCH VÀ PHÒNG NGỪA LỖI (DEFECT ANALYSIS AND PREVENTION)

Phòng ngừa lỗi nhằm mục đích để học hỏi từ các lỗi được tìm thấy trong dự án cho đến thời điểm hiện tại và để phòng ngừa lỗi trong phần còn lại của dự án. Như đã thảo luận trong Chương 5, các hoạt động phòng ngừa lỗi thường được thực hiện hai lần trong một dự án: một lần khi khoảng 20% các mô-đun đã được cài đặt mã và kiểm thử đơn vị, và một lần nữa khi 50% các mô-đun đã được cài đặt mã và kiểm thử đơn vị. Các công việc chính của công tác phòng ngừa lỗi là thực hiện phân tích Pareto để xác định các loại lỗi chính, thực hiện phân tích nguyên nhân (causal analysis) để xác định nguyên nhân của lỗi, và xác định các giải pháp để loại bỏ các nguyên nhân đó. Ở đây chúng tôi thảo luận làm thế nào những công việc này được thực hiện trong một dự án.

11.4.1 Thực hiện phân tích Pareto

Kỹ thuật thống kê (statistical technique) thường được sử dụng để phân tích nguyên nhân, *phân tích Pareto* (*Pareto analysis*) là một trong những công cụ chính để quản lý chất lượng [7][8]. Đôi khi nó cũng được gọi là quy tắc 80-20: 80% các vấn đề đến từ 20% trong những nguồn có thể. Trong phần mềm, nó có thể có nghĩa là 80% lỗi xuất phát từ 20% các nguyên nhân gốc hoặc 80% lỗi được tìm thấy trong 20% của mã.

Bước đầu tiên trong công tác phòng ngừa lỗi là vẽ một biểu đồ Pareto từ dữ liệu về lỗi. Số lượng các lỗi được tìm thấy theo các loại khác nhau được tính toán từ các dữ liệu về lỗi và được vẽ thành một biểu đồ thanh (bar chart) theo thứ tự giảm dần. Cùng với biểu đồ thanh, một biểu đồ khác được vẽ trên cùng một đồ thị để cho thấy số lượng lỗi tích lũy khi chúng ta di chuyển từ các loại lỗi nằm bên trái của trục x sang bên phải của trục x. Biểu đồ Pareto trình bày rõ ràng qua hình ảnh cũng như qua định lượng các loại lỗi chính, và cho biết những loại lỗi nào đã chiếm 80% -85% tổng số lỗi. Thay vì vẽ biểu đồ số lượng lỗi, bạn có thể vẽ một tổng trọng lượng lỗi (weighted sum) bằng cách gán các trọng lượng (weights) khác nhau cho các loại lỗi khác nhau.

Toàn bộ thủ tục chung để thực hiện phân tích Pareto như sau:

1. Liệt kê tất cả các lỗi đã được xác định cho đến nay.

2. Tính tổng số các lỗi theo mỗi loại.

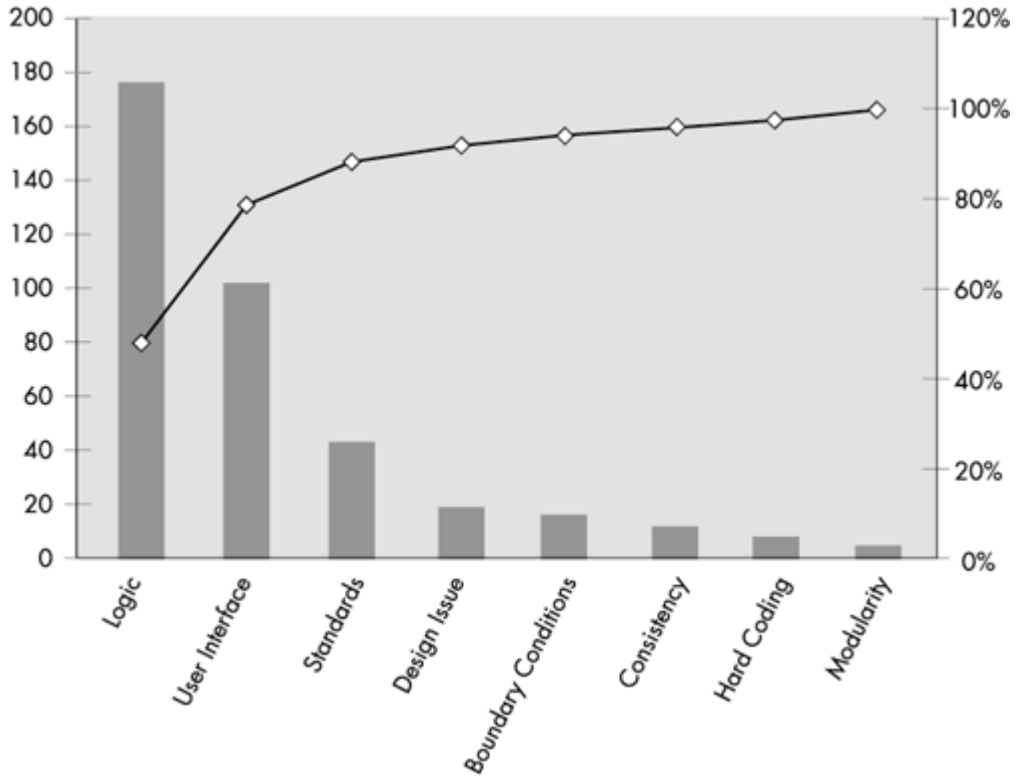
3. Sắp xếp lỗi loại lỗi thứ tự giảm dần của số lỗi.

4. Tính toán tỷ lệ phần trăm của từng loại lỗi tương ứng với tổng số các lỗi phát hiện.

5. Xác định loại lỗi đó là nguyên nhân gây ra khoảng 80% tổng số các lỗi.

Ví dụ, hãy xem xét biểu đồ Pareto cho dữ liệu về lỗi của dự án ACE được hiển thị trong hình 11.5. Trong dự án này, các tính năng mới được thêm vào hệ thống hiện có. Các dữ liệu về lỗi từ tất cả các công việc nâng cấp trước đó (all previous enhancements) đã được sử dụng cho phân tích này. Như bạn có thể thấy, số lượng lỗi cao nhất là các lỗi lôgic (logic defects), tiếp theo là các lỗi giao diện người dùng (user interface defects) và các lỗi chuẩn (standards defects). Tổng ba loại lỗi này chiếm hơn 88% tổng số các lỗi, trong đó hai loại đầu chiếm hơn 75%. Rõ ràng, mục tiêu phòng ngừa lỗi nên có hai hoặc ba loại đầu này.

Hình 11.5. Biểu đồ Pareto cho các lỗi được tìm thấy trong dự án ACE



11.4.2 Thực hiện phân tích nguyên nhân

Biểu đồ Pareto giúp nhận diện các loại lỗi chính đã được tìm thấy trong dự án cho đến thời điểm hiện tại và có thể được tìm thấy trong phần còn lại của dự án, trừ khi có hành động nào đó được thực. Những lỗi này có thể được coi như là "hậu quả" (effects) mà bạn muốn giảm thiểu trong tương lai. Để giảm lỗi, bạn phải tìm ra nguyên nhân chính của chúng và sau đó cố gắng loại bỏ chúng. Một sơ đồ nhân quả (CE - cause effect diagram) có thể được sử dụng để xác định nguyên nhân của hậu quả đã được quan sát thấy (observed effects) [7][8]. Ví dụ, sơ đồ nhân quả có thể được sử dụng để xác định nguyên nhân chính làm cho số lỗi giao diện GUI cao (hoặc lỗi lôgic) trong dự án ACE. Mục đích chính của sơ đồ CE là trình bày bằng đồ họa mối quan hệ giữa một hậu quả (effect) và các nguyên nhân (causes) khác nhau có thể có của nó. Hiểu được nguyên nhân sẽ giúp xác định các giải pháp để loại bỏ chúng.

Bước đầu tiên trong việc xây dựng một sơ đồ nhân quả (cause-effect diagram) là đi xác định hậu quả (effect) để được phân tích. Trong ví dụ ACE, hậu quả có thể là "quá nhiều lỗi giao diện GUI". Để xác định nguyên nhân, trước tiên bạn thiết lập một số loại nguyên nhân chính. Đối với quá trình sản xuất (manufacturing), các nguyên nhân chính thường là nhân lực (manpower), máy móc, phương pháp, vật liệu (materials), đo lường (measurement), và môi trường (environment). Để phân tích nguyên nhân tại Infosys, một tập hợp chuẩn các nguyên nhân chính gây ra các lỗi này là quy trình (process), con người (people), công nghệ (technology), và đào tạo (training) (đào tạo được tách ra khỏi con người bởi vì nó xuất hiện rất thường xuyên). Cấu trúc chính của sơ đồ cho thấy hậu quả là một hộp (box) nằm ở bên phải; một đường ngang thẳng (đường chính) kéo dài từ hộp này, và một đường cho mỗi nguyên nhân chính gây ra kết nối vào đường chính.

Để phân tích các nguyên nhân, câu hỏi chính là, "Tại sao nguyên nhân này gây ra hậu quả này?" cho mỗi nguyên nhân chính. Những câu trả lời cho những câu hỏi này trở thành các nguyên nhân con (subcauses) và được biểu diễn như là các đường ngang ngắn ngang nối vào đường đang trình bày nguyên nhân chính. Sau đó, cùng một câu hỏi sẽ được hỏi cho những nguyên nhân đã được xác định. Quá trình "Tại sao-Tại sao-Tại sao" này được lặp đi lặp lại cho đến khi tất cả các *nguyên nhân gốc rễ* (*root causes*) đã được xác định – tức là, các nguyên nhân mà câu hỏi "Tại sao" cho nó thì không còn có câu trả

lời nào khác nữa. Khi tất cả các nguyên nhân được đánh dấu trong biểu đồ, hình ảnh cuối cùng trông giống như một cấu trúc xương cá (fish-bone structure), và vì thế sơ đồ nhân quả còn được gọi là một sơ đồ xương cá (*fish-bone* diagram), hoặc sơ đồ Ishikawa (tên của người đã phát minh ra nó).

Các bước chính để vẽ một sơ đồ nhân quả như sau [8]:

1. Xác định rõ các vấn đề (hậu quả) để được nghiên cứu. Để phòng ngừa lỗi, nó thường là "có quá nhiều lỗi loại X."

2. Vẽ một mũi tên từ trái sang phải với một hộp (box) chứa hậu quả ở đầu bên phải. Đây là xương sống (backbone) của sơ đồ.

3. Xác định các loại nguyên nhân chính. Đây có thể là các loại nguyên nhân chuẩn hoặc một số thay đổi (biến thể) để phù hợp với vấn đề.

4. Viết những nguyên nhân chính này vào trong các hộp (box) và kết nối chúng bằng các mũi tên chéo đến xương sống. Những nguyên nhân này hình thành các xương của biểu đồ.

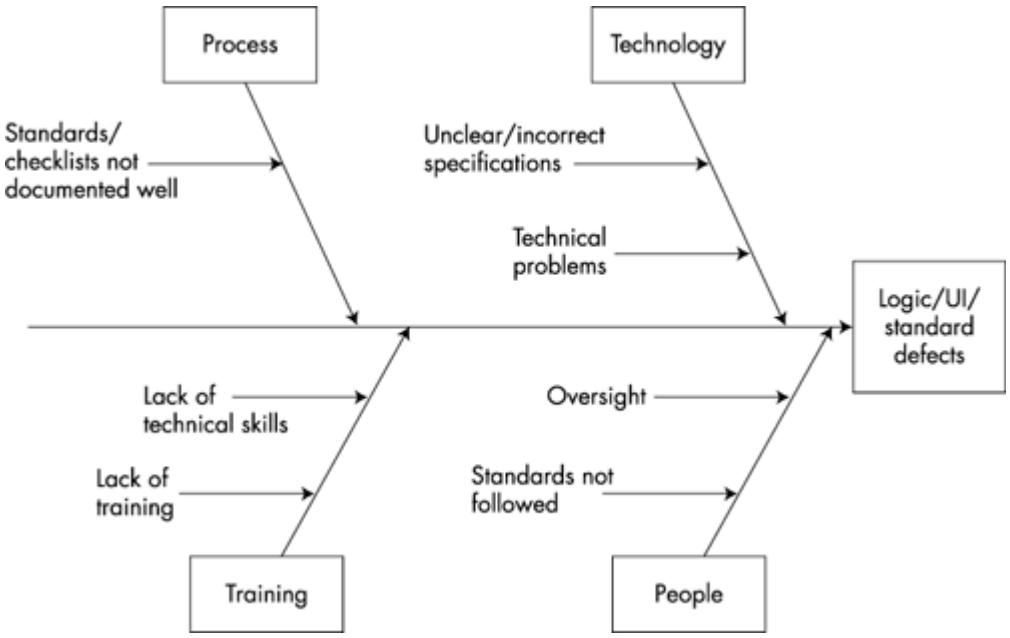
5. Thu thập ý kiến của nhóm (team) về các nguyên nhân con (subcauses) của các nguyên nhân chính bằng cách hỏi nhiều lần, cho mỗi nguyên nhân chính, "Tại sao nguyên nhân chính này tạo ra hậu quả?"

6. Thêm các nguyên nhân con vào sơ đồ nhóm xung quanh xương của nguyên nhân chính. Nếu cần thiết, chia nhỏ hơn nữa những nguyên nhân này. Dừng lại khi không còn tìm thấy câu trả lời nào đáng giá cho câu hỏi.

Khi sơ đồ xương cá được vẽ xong, bạn đã xác định được tất cả các nguyên nhân của hậu quả đang được nghiên cứu. Tuy nhiên, nhiều khả năng sơ đồ xương cá ban đầu sẽ có quá nhiều nguyên nhân. Rõ ràng, một số nguyên nhân có tác động lớn hơn so với những nguyên nhân khác. Do đó, trước khi hoàn tất việc phân tích nguyên nhân gốc rễ, bạn hãy xác định những nguyên nhân hàng đầu, chủ yếu là thông qua thảo luận. Để phòng ngừa lỗi, bạn có thể thực hiện toàn bộ thực hành này cho một hoặc hai loại lỗi đầu đã được tìm thấy trong phân tích Pareto.

Hình 11.6 cho thấy sơ đồ xương cá của dự án ACE. Trong phân tích này, các nguyên nhân của ba loại lỗi chính đã được thảo luận trong một cuộc họp tập thể. Do đó, hậu quả của chúng tôi là "có quá nhiều lỗi logic/GUI/lỗi chuẩn" (too many logic/GUI/standards defects). Khi chúng tôi hỏi câu hỏi, "Tại sao con người gây ra quá nhiều lỗi lôgic hoặc GUI hoặc lỗi chuẩn?", chúng tôi đã xác định được một số lý do (gần như rõ ràng): thiếu đào tạo (lack of training), sơ suất (oversight) (có nghĩa là, thiếu tập trung), thiếu các kỹ năng về công nghệ (lack of technical skills). Tương tự như vậy, khi chúng tôi hỏi, "Tại sao các quy trình gây ra quá nhiều lỗi lôgic/GUI/lỗi chuẩn?", các câu trả lời là "tiêu chuẩn không được lập tài liệu đày đủ" ("standards not comprehensively documented") và "mọi người không nhận thức được các tiêu chuẩn" (people not aware of standards). Đối với công nghệ (technology), các nguyên nhân là "các đặc tả không rõ ràng" (unclear specifications) và "các vấn đề kỹ thuật của các công cụ" (technical problems of tools). Các cuộc họp thảo luận tập thể để phân tích nguyên nhân đã tạo ra nhiều nguyên nhân hơn. Sau khi liệt kê tất cả những lời đề xuất được đưa ra trong suốt cuộc họp, nhóm phòng ngừa lỗi đã xếp hạng ưu tiên chúng bằng cách xem xét từng lỗi một và xác định các nguyên nhân của nó. Các nguyên nhân xuất hiện thường xuyên nhất sẽ có độ ưu tiên cao. Chúng được trình bày trong Hình 11.6.

Hình 11.6. Sơ đồ nhân-quả cho dự án ACE



11.4.3 Phát triển và thực hiện các giải pháp

Cho đến thời điểm này, chúng ta đã thảo luận làm thế nào để xác định các loại lỗi thường xuyên xảy ra và nguyên nhân gốc rễ của chúng. Giai đoạn tiếp theo là phải hành động để làm giảm sự xuất hiện lỗi.

Mô hình cơ bản là câu ngạn ngữ "Một Ao-xơ (ounce) phòng ngừa có trị giá bằng một cân (pound) chữa bệnh." Với phòng ngừa lỗi, bạn không cố gắng để "chữa trị" các lỗi phần mềm, thay vào đó, bạn đang dùng các hoạt động phòng ngừa để mà phần mềm không "bị ốm" từ lỗi. Các hoạt động phòng ngừa phổ biến là tạo ra hoặc cải tiến danh sách kiểm tra (checklists), tổ chức các chương trình đào tạo (training) và xem xét lại (reviews), và bằng cách sử dụng một công cụ cụ thể. Tất nhiên, đôi khi bạn phải có những hành động quyết liệt chẳng hạn như thay đổi quy trình (process) hoặc công nghệ (technology).

Các giải pháp, như phân tích nhân-quả (cause-effect analysis), được phát triển thông qua một cuộc họp thảo luận tập thể. Do đó, hai bước này thường được thực hiện trong cùng một cuộc họp. Đây là cách được thực hiện ở Infosys.

Các giải pháp phòng ngừa được xác định rõ dưới dạng các hoạt động (actions) mà một người nào đó phải thực hiện. Do đó, việc thực hiện (implementation) các giải pháp là yêu cầu then chốt. Nếu các giải pháp không được thực hiện, chúng sẽ không được sử dụng. Ở Infosys, bên cạnh chỉ ra giải pháp, người chịu trách nhiệm để thực hiện nó cũng được chỉ định. Những hoạt động này sau đó được bổ sung vào lịch chi tiết của cho dự án, và quá trình thực hiện nó được theo dõi như các công việc khác. Bảng 11.6 cho thấy nguyên nhân gốc rễ và các hoạt động phòng ngừa đã được phát triển cho dự án ACE. Các hoạt động phòng ngừa được đề xuất thì có thể hiểu được (không cần giải thích) (self-explanatory). Chúng đã được lên kế hoạch trong lịch Microsoft Project (MSP) của dự án.

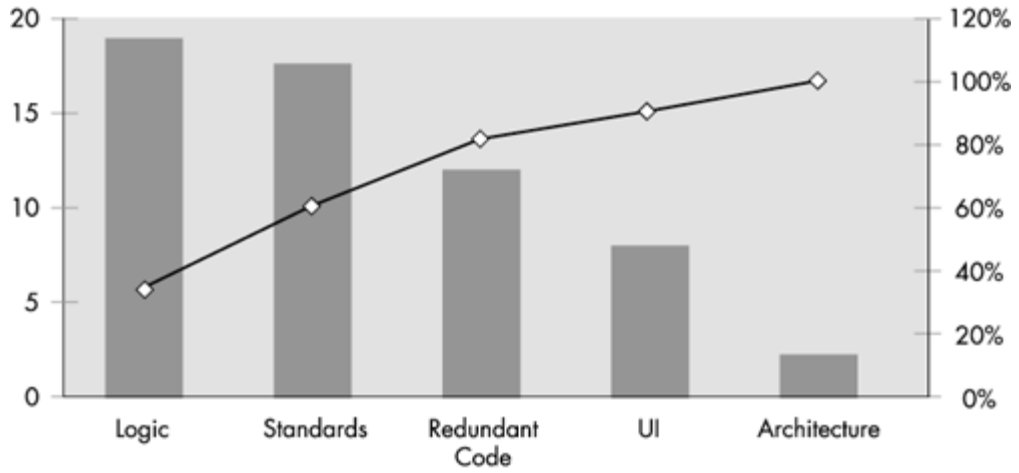
Một phần quan trọng trong việc thực hiện các giải pháp này là phải xem chúng có mang lại hiệu quả như mong muốn trong việc giúp làm giảm tỷ lệ tiêm lỗi (injection of defects) và do đó giúp làm giảm nỗ lực làm lại công việc (rework) hay không. Tiếp tục phân tích các lỗi được tìm thấy sau khi các giải pháp đã được thực hiện có thể cung cấp cái nhìn sâu sắc

cho câu hỏi này. Nói chung, các phân tích tiếp theo để phòng ngừa lỗi có thể được sử dụng cho mục đích này. Ngoài việc theo dõi tác động, tiếp tục phân tích như vậy có hiệu quả rất lớn. Việc nhìn thấy được các lợi ích sẽ thuyết phục mọi người. Do đó, ngoài việc thực hiện, tác động (impact) của việc thực hiện cũng cần được phân tích.

11.4.4 Phòng ngừa lỗi trong dự án ACIC

Bây giờ chúng ta hãy nhìn vào quy trình phòng ngừa lỗi (DP - defect prevention process) cho case study ACIC. Dữ liệu về lỗi sau khi vòng lặp Xây dựng đầu tiên (first construction iteration) đã được phân tích, và tần số của các loại lỗi khác nhau được trình bày trong Bảng 11.7. Hình 11.7 cho thấy biểu đồ Pareto của dữ liệu về lỗi.

Hình 11.7. Biểu đồ Pareto của dữ liệu về lỗi



Mục đích chính của các hoạt động phòng ngừa lỗi là để làm giảm tỷ lệ tiêm lỗi. Trong vòng lặp đầu tiên (first iteration), người quản lý dự án ACIC đã biết rằng có ít nhất 57 lỗi đã được tiêm vào. Từ dữ liệu nỗ lực, ông đã tính toán tỷ lệ tiêm lỗi cho giai đoạn xây dựng là 0.33 lỗi trên mỗi giờ. Theo kế hoạch, nó đã được dự kiến khoảng 70% các lỗi sẽ được tiêm vào trong hoạt động xây dựng (build activity), nỗ lực ước lượng (dự toán) là khoảng 110 ngày (không bao gồm ước lượng để làm lại công việc). Tức là, theo kế hoạch chất lượng và nỗ lực, tỷ lệ tiêm lỗi trong suốt quá trình cài đặt mã (coding) dự kiến là khoảng 0.1 lỗi trên mỗi người-giờ. Tuy nhiên, sau vòng lặp đầu tiên, tỷ lệ tiêm lỗi đã ba lần nhiều hơn con số này! Rõ ràng, các hoạt động phòng ngừa lỗi cần được thực hiện để đạt được các mục tiêu.

Bảng 11.6. Các nguyên nhân gốc và giải pháp đề xuất cho dự án ACE

Nguyên nhân gốc (Root Cause)	**Các hoạt động phòng ngừa (Preventive Actions)**	**Phân công cho (Assigned to)**	**Ngày thực hiện (Implementation Date)**
Các chuẩn không được theo. *(Standards not followed)*	• Thực hiện đọc lại các chuẩn bởi nhóm (sau khi chúng đã được cập nhật). *(Do a group reading of the standards (after they have been updated).)* • Đảm bảo rằng các chuẩn phải được theo. *(Ensure that standards are followed in the mock projects done.)*	Tất cả mọi người *(All)*	15/12/00
Chuẩn/danh sách kiểm tra được lập tài liệu không tốt. (Standards/checklists not documented well)	• Thực hiện xem xét lại các chuẩn bởi nhóm và có chuyên gia bên ngoài tham gia và sau đó cập nhật lại các chuẩn. *(Do a group review of the standards with expert from outside and then update the standards.)*	Xxxx	Tuần tới *(Next week)*
Sơ suất (thiếu tập trung) *(Oversight (incomplete attention))*	• Tự xem xét lại *(Effective self-review)* • Xem xét lại mã một cách nghiêm ngặt *(Rigorous code reviews)*	All	Ảnh hưởng ngay *(Immediate effect)* Ảnh hưởng ngay *(Immediate effect)*
Đặc tả không rõ ràng/không đúng.	• Xem xét lại các đặc tả. *(Specification reviews)*	All	Ảnh hưởng ngay *(Immediate effect)*

(Unclear/incorrect specifications)			
Thiếu đào tạo. *(Lack of training)*	• Mọi người tham gia sẽ làm một dự án giả, mã của nó được xem xét lại và được kiểm thử kỹ lưỡng. *(Every new entrant will do a mock project, whose code will be reviewed and tested thoroughly).* • Đặc tả chi tiết và kế hoạch kiểm thử sẽ được làm tương tự. *(A detailed specification and test plan will be made for the same.)*	xxxx	29/12/00
Các vấn đề về kỹ thuật. *(Technical problems)*	• Cung cấp kiến thức cho mọi người về các vấn đề và làm thế nào để tránh chúng. *(Create awareness in people about the problems with the tools and how to avoid them.)* • Viết một BOK (cơ sở tri thức) về vấn đề và để mọi người có thể dùng được nó. *(Write a BOK on this and make it available.)*		
Thiếu kỹ năng về công nghệ. *(Lack of technical skills)*	• Lập tài liệu một BOK về các chủ đề như Sheridan grids, recordsets, Active Reports. *(Document a BOK on topics like Sheridan grids, recordsets, Active Reports.)*	xxxx	31/01/01

Để làm giảm đáng kể tỷ lệ tiêm lỗi, người quản lý dự án đã quyết định để giải quyết ba loại lỗi: lôgic, chuẩn, và mã dư thừa. Một cuộc họp thảo luận tập thể đã được tổ chức để xác

định các nguyên nhân gốc rễ và các hoạt động phòng ngừa có thể. Thủ tục thường xuyên thu thập ý kiến (brainstorming) được sử dụng. Đầu tiên, tất cả các nguyên nhân có thể mà mọi người đã đề nghị đã được liệt kê, và sau đó là những nguyên nhân được xác định là thủ phạm chính được tách ra. Đối với các nguyên nhân này, các hoạt động phòng ngừa được thảo luận và cuối cùng được nhất trí. Bảng 11.8 cho thấy kết quả cuối cùng của cuộc họp phân tích nguyên nhân – nó gồm: các nguyên nhân gốc rễ chính và các hoạt động phòng ngừa sẽ được thực hiện. Nhiều trong số các hoạt động phòng ngừa này đã trở thành các hoạt động xếp lịch được (schedulable activities) và đã được thêm vào lịch của dự án và sau đó được thực hiện.

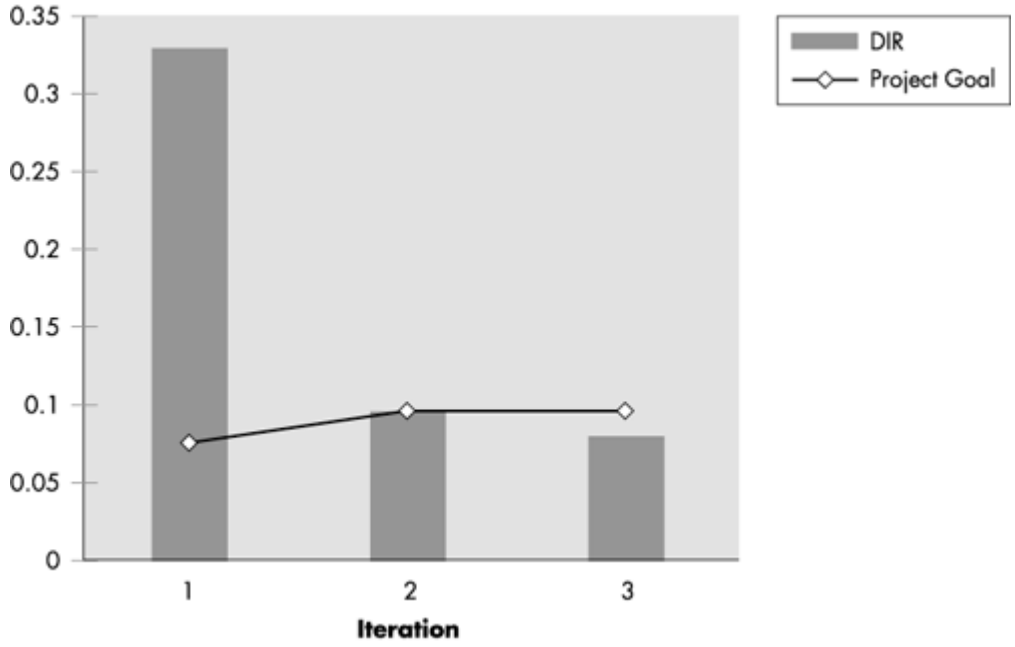
Bảng 11.7. Tóm tắt về dữ liệu về lỗi sau vòng lặp đầu tiên, dự án ACIC

Loại lỗi (Defect Type)	**Số lượng lỗi (Number of Defects)**
Logic	19
Chuẩn (Standards)	17
Mã dư thừa (Redundant code)	11
UI	8
Kiến trúc (Architecture)	2
Tổng (Total)	**57**

Các hoạt động phòng ngừa được đưa ra trong bảng là đề nghị của các thành viên trong nhóm; người quản lý dự án phải xác định chắc chắn rằng chúng đã cho kết quả như mong muốn. Để biết xem những số đo này đã thành công hay không trong việc làm tỷ lệ tiêm lỗi, cách duy nhất để kiểm tra điều này là dùng dữ liệu về lỗi.

Các hoạt động phòng ngừa lỗi được thực hiện sau vòng lặp Xây dựng đầu tiên (first construction iteration) đã được thực hiện xong. Vì dự án ACIC có ba vòng lặp như vậy, tỷ lệ tiêm lỗi sau hai vòng lặp tiếp theo đã được tính toán. Hình 11.8 cho thấy kết quả phân tích được thực hiện sau hai vòng lặp còn lại. Biểu đồ này cho thấy rõ ràng tác động của việc thực hiện các hoạt động phòng ngừa lên tỷ lệ tiêm lỗi: Nó đã giảm từ hơn 0.33 xuống dưới 0.1!

Hình 11.8. Tỷ lệ tiêm lỗi trong các vòng lặp khác nhau, dự án ACIC



Bảng 11.8. Các nguyên nhân gốc rễ và các hoạt động phòng ngừa cho dự án ACIC

Loại lỗi (số lượng) (Defect Type (Number of Defects))	**Nguyên nhân gốc rễ (Root Cause)**	**Hoạt động ngăn chặn (Preventive Action)**	**Phân công cho (Assigned To)**
Chuẩn (Standards) (17)	Thiếu kinh nghiệm lập trình. *(Lack of programming experience)*	Đào tạo *(Training.)*	Tự (Self)
	Sơ suất *(Oversight)*	Các nhà phát triển nên đọc các chuẩn cài đặt mã cẩn thận và thực hiện theo chúng. *(Developers should read the coding standards carefully and adhere to them strictly.)*	Tự (Self)
	Thiếu hiểu rõ về các	(i) Xây dựng một phương pháp để sinh ra	xxxx

	đặc tả chương trình. *(Lack of understanding of program specs use and need)*	đặc tả chương trình từ Rational Rose. (ii) Chuẩn bị một danh sách kiểm tra để xem xét lại đặc tả chương trình. (iii) Chuẩn bị các hướng dẫn để viết đặc tả chương trình. *((i) Come up with a method to generate program specs from Rational Rose. (ii) Prepare a checklist for reviewing program specs. (iii) Prepare guidelines for writing program specs.)*	
	Các chuẩn cài đặt mã chưa được cập nhật *(Coding standards not updated)*	Cập nhật lại các chuẩn cài đặt mã và chuẩn bị một tài liệu để liệt kê các chuẩn UI cho dự án. *(Update coding standards and prepare a document listing the applicable project-specific UI standards.)*	xxxx
Mã dư thừa (Redundant Code) (11)	Thiếu sự hiểu rõ về ngôn ngữ. *(Lack of understanding of language)*	Đào tạo *(Training.)*	xxxx
	Thiếu sự hiểu rõ về mô hình đối tượng và cơ sở dữ liệu. *(Lack of understanding of object model and database)*	(i) Đào tạo về cấu trúc cơ sở dữ liệu. (ii) Nhà phát triển nên xem kỹ lưỡng mô hình đối tượng. *((i) Training on database structure. (ii) Developer should go through the object model thoroughly.)*	Cuộc họp về DB được thực hiện bởi xxxxx *(Session on DB to be taken by xxxxx)*
	Thiếu sự hiểu rõ về mã hiện có. *(Lack of*	Thảo luận nhóm và hoàn thành tập hợp các lời gọi phương thức chung và xác định chúng được gọi từ đâu. *(Group to discuss in a meeting and finalize*	Nhóm (Team)

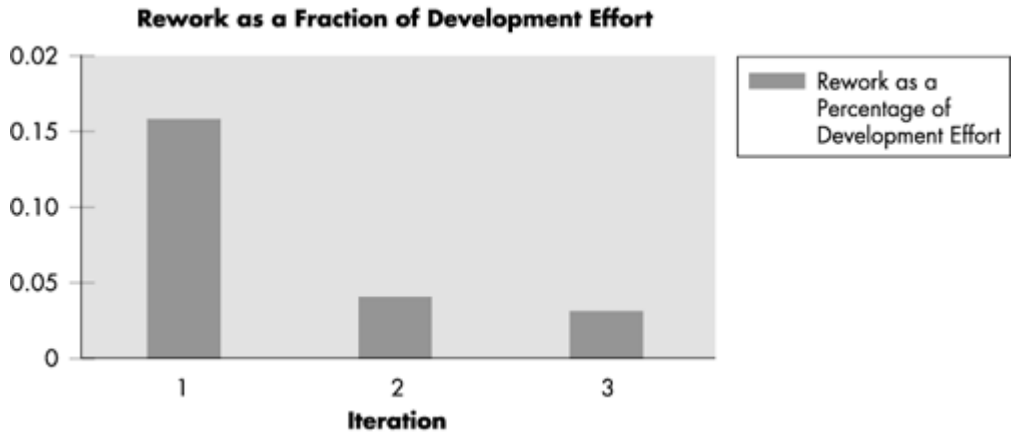
	understanding of existing code)	*the set of general method calls and identify where they should be called from.)*	
	Thiếu sự hiểu rõ về mô hình bảng. *(Lack of understanding of table model)*	Hiểu chức năng của mô hình bảng và sự phụ thuộc lên Table Selection và nói cho nhóm biết về nó. *(Understand the functionality of table model and dependency on Table Selection and inform the team about it.)*	xxxx
Logic (19)	Thiếu sự hiểu rõ về mã hiện có. *(Lack of understanding of existing code)*	Tổ chức các cuộc họp đọc mã. *(Arrange code reading sessions.)*	Tự *(Self)*
	Thiếu kinh nghiệm lập trình. *(Lack of programming experience)*	Đào tạo *(Training.)*	xxxx
	Thiếu sự hiểu biết về biểu đồ tuần tự. *(Lack of understanding of sequence diagrams representations)*	Đào tạo về Rational Rose *(Give training in Rational Rose.)*	xxxx
	Thiếu sự hiểu rõ về cơ sở dữ liệu và các xử lý liên kết. *(Lack of understanding of database and associated processes.)*	Giống như trước đây *(Same as earlier.)*	xxxx

	Thiếu sự hiểu rõ về mô hình đối tượng. *(Lack of understanding of object model)*	Giống như trước đây *(Same as earlier.)*	Tự *(Self)*
	Sơ suất *(Oversight)*	Tự kiểm thử bởi lập trình viên nên được làm kỹ lưỡng. Tổ chức một cuộc họp về làm thế nào để kiểm thử một phần nhỏ mã. *(Self-testing by programmer should be made more thorough. A session on how to test a small part of code to be taken.)*	Đào tạo bởi xxxx *(Training by xxxx)*
	Thiếu hiểu biết về các trường hợp sử dụng. *(Lack of understanding of use cases)*	Người phát triển sẽ xem qua tài liệu yêu cầu. *(Developers will do a requirement walkthrough.)*	Tự *(Self)*
	Thiếu hiểu rõ về các quy tắc nghiệp vụ. *(Lack of understanding of business rules)*	Nhà phát triển tham khảo đến các ma trận có sẵn mà chúng đang dùng các quy tắc khác nhau. (ii) Nhà phát triển xem xét lại các trường hợp sử dụng của các ứng dụng trước đây để hiểu rõ về các quy tắc nghiệp vụ. *((i) Developer to refer to the matrix available that deals with various rules. (ii) Developer to review use cases of earlier application for better understanding of business rules.)*	Tự *(Self)*
	Thiếu sự hiểu rõ về lỗi. *(Lack of understanding of*	Việc tiếp tục phân tích với người xem xét lại nên được thực hiện bởi người chủ lỗi. Một cố gắng nên được làm để làm giảm bất kỳ lỗ hổng giao tiếp nào bằng cách thường xuyên tiếp tục phân tích các vấn đề	Nhóm (Team)

	defect)	phát sinh của nhóm/thành viên liên quan. *(Follow-up with the reviewer should be taken by the owner of the defect. An attempt shall be made to reduce any existing communication gaps by more frequent follow-ups of the issues with the team/member concerned.)*	

Giảm tỷ lệ tiêm lỗi ngụ ý rằng sẽ có ít lỗi hơn được phát hiện và sửa chữa. Do đó, hoạt động phòng chống lỗi thành công sẽ dẫn đến giảm nỗ lực làm lại các công việc (rework effort) sau giai đoạn kiểm thử. Hình 11.9 cho thấy nỗ lực làm lại công việc trong ba vòng. (Nỗ lực làm lại này được thu được từ WAR bởi vì có một mã khác nhau cho một công việc làm lại, và chương trình và mô-đun cũng được xác định). Nỗ lực (chi phí) làm lại sau vòng lặp Xây dựng đầu tiên là khoảng 16% tổng nỗ lực của vòng lặp đó. Nỗ lực này đã giảm xuống còn khoảng 5% và 3% trong hai vòng lặp tiếp theo. Nỗ lực đã bỏ ra trong việc phân tích nguyên nhân là một vài giờ để phân tích dữ liệu, cùng với một cuộc họp thảo luận nhóm về phòng ngừa lỗi.

Hình 11.9. Nỗ lực làm lại trong dự án ACIC giảm do phòng ngừa lỗi



11.5 QUY TRÌNH GIÁM SOÁT VÀ KIỂM TRA (PROCESS MONITORING AND AUDIT)

Để đạt được những lợi ích của các kế hoạch và các quy trình, các dự án được thực hiện theo chúng một cách đúng đắn. Mọi người có thể làm cho những sai lầm, và dưới áp lực của thời hạn hoàn thành (deadline), họ có xu hướng dùng các con đường tắt (biện pháp nhanh chóng trực tiếp hơn) hoặc các phép đo có lợi, thường không tuân thủ (compliance) theo các quy trình. Để đảm bảo việc tuân thủ theo các quy trình đã được xác định, một nỗ lực tích cực là cần thiết. Kiểm tra (audit) nhằm mục đích đáp ứng yêu cầu này.

Mục tiêu cơ bản của kiểm tra (audits) là để đảm bảo tuân thủ (compliance) theo các quy trình đã được xác định và để cung cấp cho người quản lý cao cấp với khả năng nhìn thấy được (visibility) trong quá trình sử dụng của các quy trình. Để đảm bảo một mức độ hợp lý của việc tuân thủ, kiểm tra phải được thực hiện thường xuyên. Chúng cũng phải có hình thức (formal), cùng với một thông báo có hình thức (formal notice) của việc không tuân thủ (noncompliance) sẽ được phát hành và sau đó theo dõi cho đến khi kết thúc thỏa đáng . Hình thức đảm bảo rằng "bình đẳng cá nhân" không đóng vai trò chính và người quản lý cấp cao đạt được khả năng nhìn thấy được sự tuân thủ quy trình thông qua các bản báo cáo kiểm tra tóm tắt.

Ai nên tiến hành cuộc kiểm tra? Lý tưởng nhất, các kiểm tra viên phải hiểu các quy trình của công ty và tầm quan trọng của chúng, và cần phải có sự trưởng thành và tầm vóc cần thiết để có thể đánh giá việc thực hiện dự án một cách khách quan. Kiểm toán viên cũng cần được đào tạo về quy trình kiểm tra.

Ở Infosys, thành viên đến từ các dự án khác sẽ thực hiện kiểm tra một dự án, và SEPG (nhóm quy trình công nghệ phần mềm) sẽ chịu trách nhiệm cung cấp khóa đào tạo. Hàng tháng, điều phối viên về kiểm tra sẽ công bố một lịch trình kiểm tra – lịch này xác định dự án nào sẽ được kiểm tra bởi ai và khu vực nào kiểm tra sẽ tập trung vào (việc kiểm tra nên tập trung vào khía cạnh nào của quy trình). Bình thường, một nhóm gồm hai người tiến hành kiểm tra.

11.5.1 Thực hiện kiểm tra (Conducting the Audit)

Trong kiểm tra, kiểm tra viên tập trung vào việc xem dự án có thực hiện theo đúng quy trình đã được định nghĩa (xác định) - quan tâm nhiều nhất đến các khu vực trong quy trình mà việc kiểm tra cần quan tâm. Họ đặt câu hỏi để biết một hoạt động được thực hiện như thế nào, và họ nhìn vào các bằng chứng hoặc kết quả đầu ra của các hoạt động này. Họ có thể sử dụng các danh sách kiểm tra (*audit checklists*) để xác định các câu hỏi. Các danh sách kiểm tra này được xây dựng nên từ các quy trình đã được phê duyệt và kinh nghiệm trong quá khứ, và họ cố gắng để tối đa hóa lợi ích từ một cuộc kiểm tra bằng cách tập trung vào các khía cạnh chủ chốt nhiều hơn vào các khía cạnh ít quan trọng. Dưới đây là các phần của một danh sách kiểm tra để lập kế hoạch cho dự án:

- Kế hoạch dự án (project plan) có được lập tài liệu bằng cách dùng mẫu kế hoạch dự án chuẩn (standard project plan template) không?
- Kế hoạch dự án có được nhóm xem xét lại không?
- Kế hoạch dự án đã được phê duyệt và được baseline chưa? Và có quản lý cấu hình không?
- Có một hợp đồng được ký kết không?
- Các cam kết với khách hàng hoặc các nhóm khác đã được xem xét lại?
- Nỗ lực dùng cho dự án có được ước lượng dựa trên dữ liệu lịch sử không?
- Ước lượng (dự toán) nỗ lực và lịch đã được xem xét lại?
- Kế hoạch chất lượng đã đầy đủ, và nó đã được xem xét lại?
- Vòng đời được sử dụng trong dự án đã được xác định và được lập tài liệu?
- Các thành viên trong dự án đã được và trách nhiệm cho từng công việc đã được xác định và được theo dõi?
- Các báo hiệu (triggers) để ước lượng lại (reestimation), chẳng hạn như khi có thay đổi phạm vi (scope changes) và các hoạt động khắc phục cần thiết (required corrective actions), đã được xác định chưa?

- Các phần sản phẩm để giao cho khách hàng, bao gồm cả tài liệu hướng dẫn người sử dụng, đã được xác định rõ ràng chưa?
- Rủi ro và kế hoạch giảm thiểu rủi ro đã được xác định và được lập tài liệu một cách đúng đắn chưa?
- Xem xét lại, báo cáo tiến độ, theo dõi, và các cơ chế chấp thuận (approval mechanisms) đã được xác định chưa?

Việc kiểm tra được coi là hoàn thành khi nhóm kiểm tra (audit team) đã hỏi tất cả các câu hỏi và đã nhìn vào tất cả các đối tượng (artifacts) mong muốn. Một báo cáo không tuân thủ (NCR - noncompliance report) được phát hành như là bằng chứng cho thấy rằng các quy trình đã được phê duyệt không được thực hiện theo, hoặc rằng một số khuyết điểm (weakness) tồn tại trong quy trình có thể dẫn đến mất kiểm soát hay chưa tối ưu.

Một khía cạnh then chốt của kiểm tra (và một trong đó là nhấn mạnh trong đào tạo) là thủ tục tìm kiếm để kiểm tra việc tuân thủ quy trình và không kiểm tra con người. Ý tưởng này là nền tảng cho tiếp cận theo quy trình (entire process-oriented approach); sự tập trung nên luôn luôn nằm ở quy trình và cải tiến quy trình, và các vấn đề được tìm thấy trong một dự án nên được đặc trưng bởi các yếu tố quy trình (process factors) chứ không bởi yếu tố con người. Rõ ràng, báo cáo không tuân thủ (NCR) chỉ ra các loại chênh lệch đã được tìm thấy.

Việc xác định sự không tuân thủ (noncompliance) là mục tiêu của kiểm tra (và phân tích kết quả kiểm tra được sử dụng để đánh giá hiệu quả của các quy trình), nhưng khi 2 chuyên gia phần mềm đánh giá một dự án, họ thường phát triển một số ý tưởng liên quan đến các khía cạnh kỹ thuật của dự án hoặc các khía cạnh quản lý – mà chúng có thể hữu ích trong việc cải thiện dự án. Những vấn đề này thường không tạo thành sự không tuân thủ, nhưng người ta không muốn bỏ qua nó. Vì vậy, những điều được quan sát thấy như vậy được ghi lại trên một biểu mẫu riêng để được dùng như là các đề nghị của kiểm tra viên. Các báo cáo kiểm tra, bao gồm cả NCRs và những đề nghị, được gửi cho người điều phối (coordinator) của các cuộc kiểm tra.

11.5.2 Các hoạt động tiếp tục theo dõi (Follow-up Actions)

Việc nộp báo cáo không tuân thủ (NCR) không có nghĩa là đã kết thúc hoạt động kiểm tra. Bởi vì mục đích cơ bản của kiểm tra là để đảm bảo rằng các dự án triển khai quy trình đã được phê duyệt của công ty, các báo cáo này nên được sử dụng để thực hiện những thay đổi cần thiết trong dự án để mà bất kỳ vấn đề nào được đưa ra trong NCR cũng được giải quyết thoả đáng. Bước này được gọi là một hành động khắc phục (*corrective action*). Đối với mỗi NCR, một số hành động khắc phục phải được thực hiện. Một khi đã được thực hiện xong, nó sẽ được ghi lại vào trong biểu mẫu của chính NCR.

Để đảm bảo rằng các vấn đề trong NCR được giải quyết thỏa đáng, người điều phối việc kiểm tra đảm bảo rằng hành động phải được chấp thuận bởi kiểm tra viên. Nếu các nhân viên này không rảnh, cố vấn chất lượng (quality adviser) của dự án hoặc người điều phối việc kiểm tra có thể chấp thuận hành động. Sau khi hành động được phê duyệt, NCR được coi là kết thúc (closed).

Hình 11.10. Một báo cáo về sự không tuân thủ cùng với hành động khắc phục

INFOSYS	**NON-CONFORMITY REPORT**
Project/ Dept.: Projyyy	**Date:** 21 Oct 97
QSD Ref.: Req. chg. Process	**Severity:** **Serious/**Minor
Non-Conformity	
Requirement changes in the development project are not being tracked/recorded. E.g., 5 programs sent for modification. The mails pertaining to those changes were not logged.	
Corrective action:	**Action by:** PL **Action date:** 10 Dec 97
A spreadsheet will be created in which all changes, along with their impact analysis, will be recorded.	
Preventive action:	**Action by:** **Action date:**
Auditor **Auditee** **(Signature)** **(Signature)**	
Follow-up action	
Done.	
Closed by (Signature)	**Recommendation:** Closed

Hình 11.10 cho một ví dụ về một NCR và hành động khắc phục của nó. NCR này xác định dự án, ngày, và mức độ nghiêm trọng của sự không tuân thủ. Mức độ nghiêm trọng cho thấy mức độ nghiêm trọng của vấn đề và hậu quả của nó (lớn hoặc nhỏ). Thành viên của dự án thường thực hiện các hành động khắc phục. Trong trường hợp này, vấn đề phát sinh (issue) có liên quan đến các thay đổi yêu cầu (requirements changes) – cái này thường được quản lý bởi người quản lý dự án - do đó người quản lý dự án sẽ thiết lập các hành động khắc phục. Ngày của các hành động khắc phục cũng được đề cập.

Đôi khi, một vấn đề có khả năng tái diễn, hoặc trong cùng một dự án hoặc ở các dự án khác. Trong trường hợp đó, ngoài việc thực hiện một hành động khắc phục, dự án có thể cần có một hoạt động phòng ngừa để đảm bảo rằng một vấn đề tương tự sẽ không xảy ra thêm một lần nào nữa. Do đó, các hoạt động phòng ngừa có thể cần được thực hiện trong một số trường hợp. NCR chứa một phần mà trong đó ghi lại các hoạt động phòng ngừa được thực hiện và người đã thực hiện chúng. Hình 11.11 cho thấy một ví dụ về một NCR cần cả hai hoạt động khắc phục và phòng ngừa. Hoạt động phòng ngừa cho NCR này là đi nâng cấp một danh sách kiểm tra để đảm bảo rằng một vấn đề tương tự sẽ không xảy ra trong tương lai.

Hình 11.11. Một báo cáo về sự không tuân thủ cùng với hoạt động phòng ngừa

INFOSYS — **NON-CONFORMITY REPORT**

Project/ Dept.: Projxxx — **Date:** 15 Dec 97
ISO 9001 clause : — **Severity:** **Serious**/Minor
QSD Ref.:

Non-Conformity
When a review finds no defects, no evidence exists to show that the review was done.

Corrective action: — **Action by:** PL
Action date: 28 Dec 97
Reviewed documents will be marked as **'reviewed'** with reviewer's signature.

Preventive action: — **Action by:** Manager – SEPG
Action date: 31 Jan 98
The review checklist will be enhanced to ensure that some review record is created even if no defects are found during the review.

Auditor
Auditee
(Signature)
(Signature)

Follow-up action
Done.

Closed by
(Signature) — **Recommendation:** Closed

Ở Infosys, một NCR phải được đóng lại trong vòng 60 ngày của cuộc kiểm tra. Thông thường, người điều phối việc kiểm tra sẽ gửi một lời nhắc nhở vào cuối tháng và một nhắc nhở khác vào lúc một tuần trước khi hết thời hạn. NCRs cũ hơn 60 ngày sẽ được báo cáo cho người quản lý cao cấp của dự án. Kiểu tiếp tục theo dõi này đảm bảo rằng các báo cáo kiểm tra đã được thực hiện nghiêm túc và các vấn đề phát sinh đã được giải quyết đúng đắn.

11.6 TÓM TẮT

Khi một kế hoạch được thực hiện, bất kể kế hoạch đã được thực hiện cẩn thận đến mức nào, nhiều thứ thường không diễn ra giống như kế hoạch. Với việc giám sát đúng cách, một người quản lý dự án có thể kiểm tra xem dự án có đang tiến triển theo kế hoạch hay không. Nếu nó không tiến triển theo con đường mong muốn, kiểm soát phải được áp dụng để đảm bảo rằng dự án vẫn đáp ứng được các mục tiêu của nó.

Tại Infosys, giám sát dự án xảy ra ở nhiều cấp độ khác nhau. Sau đây là những bài học từ cách tiếp cận này:

- Theo dõi sự hoàn thành của các hoạt động đã được xếp lịch, các lỗi được tìm thấy, và các vấn đề phát sinh. Sử dụng một bản báo cáo trạng thái hàng tuần để theo dõi và báo cáo thường xuyên.

- Tại các cột mốc của dự án, so sánh các giá trị thực tế của nỗ lực, lịch, và lỗi với các giá trị ước lượng. Nếu chênh lệch này vượt quá ngưỡng định trước, hãy thực hiện những hành động khắc phục và phòng ngừa nếu tình hình đã được xác nhận. Ngoài ra, hãy xem lại những rủi ro và các tình huống có ảnh hưởng đến rủi ro.

- Đánh giá lại ngay lập tức một số nhiệm vụ sau khi chúng vừa được thực hiện xong và thực hiện các hành động sửa chữa nếu hiệu suất không nằm trong phạm vi mong đợi - như đã được xác định từ các dữ liệu quá khứ. Xem xét lại và kiểm thử đơn vị là thích hợp nhất cho mức độ theo dõi.

- Phân tích dữ liệu về lỗi từ các mô-đun đầu tiên trong dự án để hiểu rõ nguyên nhân gốc rễ của các lỗi. Sau đó, có những hành động để loại trừ các nguyên nhân gốc rễ. Sau đó, lặp lại phân tích này để hiểu được tác động của công tác phòng ngừa lỗi.

- Kiểm tra dự án một cách hình thức xem nó có tuân thủ theo các quy trình đã được xác định. Dựa trên các báo cáo không tuân thủ, thực hiện các hành động khắc phục và phòng ngừa.

Từ quan điểm CMM, các kỹ thuật được thảo luận trong chương này đáp ứng một số yêu cầu của KPA Theo Dõi và Giám Sát Dự Án(Project Tracking and Oversight KPA) ở CMM mức 2, KPA Quản Lý Dự Án Tích Hợp (Integrated Project Management KPA) ở CMM mức 3, và KPA Quản Lý Quy Trình Định Lượng (Quantitative Process Management KPA) ở CMM mức 4. Việc giám sát quy trình và các hoạt động kiểm tra đáp ứng một số yêu cầu của KPA Đảm Bảo Chất Lượng Phần Mềm (Software Quality Assurance KPA) ở CMM mức 2 và các yêu cầu kiểm tra của một số KPAs khác. Phân tích lỗi và các hoạt động phòng ngừa đáp ứng một số yêu cầu của KPA Phòng Ngừa Lỗi (Defect Prevention KPA) ở CMM mức 5.

11.7 CÁC THAM KHẢO

1. P. Hsia. Making software development visible. *IEEE Software*, May 1996.

2. D.P. Youll. *Making Software Development Visible: Effective Project Control*. John Wiley and Sons, 1990.

3. N. Brown. Industrial-strength management strategies. *IEEE Software*, July 1996.

4. D.B. Simmons, N.C. Ellis, H. Fujihara, and W. Kuo. *Software Measurement: A Visualization Toolkit*. Prentice Hall PTR, 1998.

5. B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

6. P. Jalote. *CMM in Practice: Processes for Executing Software Projects at Infosys*. Addison-Wesley, 2000.

7. D.C. Montgomery. *Introduction to Statistical Quality Control, third edition*. John Wiley and Sons, 1996.

8. J.A. Swift. *Introduction to Modern Statistical Quality Control and Management*. St. Lucie Press, Delray Beach, Florida, 1995.

Chương 12. Kết thúc dự án

Phần mềm đã được giao và đã được cài đặt thành công. Sau khi làm việc nhiều giờ vào các ngày cuối tuần trong nhiều tháng cho dự án này, người quản lý dự án và nhóm của ông thốt ra một tiếng thở dài rằng dự án này không thành công lắm. Nhưng người quản lý dự án đã học được những bài học nào? Ông và các thành viên trong nhóm có thể tránh lặp lại những vấn đề mà họ đã gặp trong dự án này hay không? Nếu dự án kết thúc bây giờ, rất có thể câu chuyện sẽ được lặp lại trong một dự án khác - có lẽ chỉ có những cải tiến rất nhỏ.

Đối với người quản lý dự án, nhóm phát triển, và công ty, thì dự án chưa thể kết thúc cho đến khi việc mổ sẻ phân tích (sự rút kinh nghiệm) (postmortem) đã được thực hiện xong để phát hiện ra những gì đã sai và nguyên nhân của chúng, và những gì đúng đắn và tại sao đúng. Phân tích này cho phép người quản lý dự án và các thành viên trong nhóm rút ra những bài học quan trọng về quá trình thực hiện dự án. Ngoài việc giúp ích cho các thành viên của nhóm trong các dự án tương lai của họ, những bài học này cũng sẽ giúp các dự án khác cải thiện quá trình thực hiện chúng.

Một phân tích kết thúc dự án (project closure analysis), hoặc phân tích rút kinh nghiệm (postmortem), là một cơ hội vàng để cải tiến quy trình vì vậy không nên bị bỏ qua [1][2][3][4]. Thật vậy, thực hành này được coi là một thực hành tốt nhất của công nghệ phần mềm [5]. Một bước trong mô hình cải thiện chất lượng dựa trên kinh nghiệm (quality improvement paradigm of the experience factory) [6] là đi phân tích các dữ liệu sau khi kết thúc dự án để đánh giá lại các thực hành hiện tại, xác định vấn đề, v.v. Tuy nhiên, bất chấp những lợi ích này, một phân tích rút kinh nghiệm như vậy vẫn chưa phải là một hoạt động "chuẩn" [7].

Chương này mô tả nội dung của bản báo cáo phân tích kết thúc dự án (project closure analysis report) tại Infosys và cung cấp các báo kết thúc của case study ACIC.

12.1 PHÂN TÍCH KẾT THÚC DỰ ÁN (PROJECT CLOSURE ANALYSIS)

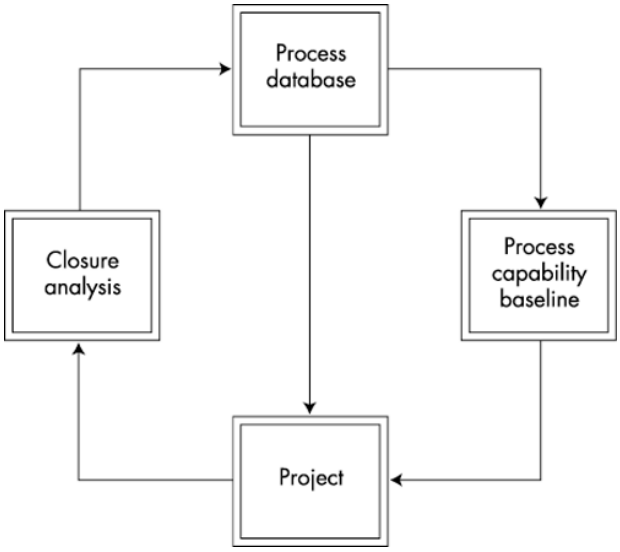
Phân tích kết thúc dự án là chìa khóa để học hỏi từ quá khứ và từ đó cung cấp các cải tiến trong tương lai. Để đạt được mục tiêu này, nó phải được thực hiện một cách cẩn thận

trong một bầu không khí an toàn để mà những bài học có thể được ghi nhận lại và được sử dụng để cải thiện quy trình và các dự án trong tương lai. Trước khi chúng tôi mô tả chi tiết về báo cáo phân tích kết thúc, chúng tôi thảo luận ngắn gọn về vai trò của phân tích kết thúc và việc thực hiện nó.

12.1.1 Vai trò của phân tích kết thúc (Role of Closure Analysis)

Mục tiêu của một phân tích rút ra bài học kinh nghiệm hoặc phân tích kết thúc là "để xác định những gì đã đi đúng, những gì đã đi sai, những gì đã hoạt động tốt, những gì không, và làm thế nào nó có thể được thực hiện tốt hơn trong thời gian tới" [2]. Thông tin liên quan phải được thu thập từ dự án, chủ yếu là để sử dụng cho các dự án trong tương lai. Tức là, mục đích của hoạt động phân tích, nhiều hơn là chỉ đơn giản nói “Dự án đã được làm xong”, không phải để giúp cho dự án này mà là để cải thiện công ty bằng cách tận dụng các bài học kinh nghiệm. Học kiểu này có thể được hỗ trợ một cách hiệu quả bằng việc phân tích dữ liệu từ các dự án đã hoàn thành. Phân tích này cũng cần hiểu rõ hiệu suất của quy trình đã được dùng trong dự án, từ đó đi xác định khả năng của quy trình.

Hình 12.1. Vai trò của phân tích kết thúc



Như đã đề cập trước đây, các dữ liệu thu được trong suốt quá trình phân tích kết thúc được sử dụng để lưu giữ vào cơ sở dữ liệu quy trình (PDB - process database). Các dữ liệu từ PDB có thể được sử dụng trực tiếp bởi các dự án tiếp theo cho mục đích lập kế hoạch. Thông tin này cũng được sử dụng trong việc tính toán khả năng quy trình (process

capability) - được sử dụng để lập kế hoạch (planning) và phân tích khuynh hướng (trends) cho các dự án. Hình 12.1 minh họa vai trò của phân tích kết thúc.

Các chương trước đã thảo luận về các loại dữ liệu thường được thu thập trong một dự án và mô tả các phương pháp thu thập. Số lượng dữ liệu thô được thu thập trong một dự án có thể khá lớn. Ví dụ, một dự án có năm người và kéo dài trong 25 tuần sẽ có 125 mục (entries) được thực hiện hàng tuần (weekly effort), dữ liệu có khoảng 250 lỗi (giả sử khoảng 0.05 lỗi được tiêm vào mỗi người-giờ), dữ liệu về các yêu cầu thay đổi, các kết quả đầu ra khác nhau, v.v. Rõ ràng, những dữ liệu này sẽ được sử dụng một cách hạn chế, trừ khi chúng được phân tích và được trình bày theo một khung làm việc (framework) và ở một mức trừu tượng phù hợp. Phân tích kết thúc nhằm mục đích để thực hiện mục tiêu này.

Sau khi phân tích dữ liệu và rút ra tất cả các bài học kinh nghiệm từ những phân tích, kết quả phải được đóng gói để mà chúng có thể được sử dụng bởi những người khác (đóng gói (packaging) là bước cuối cùng trong mô hình cải thiện chất lượng (quality improvement paradigm) [6]). Hơn nữa, để tận dụng thông tin này, các quy trình của dự án (project processes) phải được xây dựng để mà quá trình thực hiện chúng sử dụng hiệu quả dữ liệu. Tuy nhiên, có thể lập luận rằng ngay cả khi những người khác không học từ các thông tin đã được đóng gói (packaged information) này, các thành viên trong dự án sẽ củng cố kinh nghiệm của mình và sẽ thực hiện các bài học thu được từ việc phân tích vào các dự án tương lai [2]. Nói cách khác, việc phân tích kết thúc là hữu ích ngay cả khi những người khác không trực tiếp thu được từ nó.

12.1.2 Thực hiện phân tích kết thúc (Performing Closure Analysis)

Ở Infosys, người quản lý dự án thực hiện phân tích kết thúc với sự giúp đỡ từ các cố vấn chất lượng (SEPG - nhóm quy trình công nghệ phần mềm) có liên hệ đến dự án. Một mẫu (template) cho báo cáo phân tích đã được xác định. Người thực hiện phân tích kết thúc phải điền vào mẫu này một cách đúng đắn, chủ yếu bằng cách sử dụng là các dữ liệu số đo (metrics data), do đó chỉ tập trung vào các thông tin mục tiêu.

Như đã thảo luận trước đây, dữ liệu về nỗ lực (effort data) đã có sẵn từ cơ sở dữ liệu báo cáo hoạt động hàng tuần (weekly activity report database). Các dữ liệu về lỗi (defect data) có thể được thu thập từ hệ thống kiểm soát lỗi (defect control system). Dữ liệu về kích thước (size data) thu được từ dự án. Dữ liệu về hoạch định (planning data) xuất hiện

trong kế hoạch quản lý dự án (project management plan). Những dữ liệu này cấu thành những thông tin chính cần thiết cho việc phân tích các số đo (metrics analysis).

Trước tiên, các dữ liệu được phân tích bởi cố vấn chất lượng (quality adviser) – người trình bày một diễn giải ban đầu về các kết quả. Một cuộc họp sau đó được tổ chức cho cố vấn chất lượng, người lãnh đạo dự án, và các thành viên khác của dự án. Báo cáo khởi tạo (initial report) được dùng làm cơ sở cho cuộc thảo luận, và các ý kiến cá nhân và lời nhận xét từ cuộc họp cũng được lưu ý. Cuộc họp này tạo ra những phần quan trọng nhất của bản báo cáo phân tích kết thúc.

Bản báo cáo sau cùng (final report) được gửi cho SEPG và người quản lý kinh doanh của dự án và nó được gửi cho các thành viên trong nhóm dự án. Báo cáo này cũng được nhập vào trong PDB (cơ sở dữ liệu quy trình), để nó có thể được dùng lại được cho các dự án và phân tích khác trong tương lai.

12.1.3 Báo cáo phân tích kết thúc (Closure Analysis Report)

Phần này thảo luận ngắn gọn về các yếu tố chính trong một báo cáo phân tích kết thúc dự án tại Infosys; sau đó, chúng tôi trình bày báo cáo kết thúc cho dự án ACIC. Các nội dung của báo cáo phân tích này tạo thành một tập dữ liệu để đặt vào trong PDB. PDB chỉ chứa những dữ liệu số đo (metrics data) này – thường được cần đến bởi các dự án khác và cũng được dung bến bởi các quy trình hiện tại. Tuy nhiên, báo cáo phân tích có thể thu thập các dữ liệu khác giúp làm sáng tỏ hiệu suất của quy trình hiện tại hoặc giúp giải thích tốt hơn về quy trình.

Thông tin chung và thông tin liên quan đến quy trình (General and Process-Related Information)

Báo cáo kết thúc trước tiên cung cấp các thông tin chung về dự án, năng suất tổng thể đạt được và chất lượng được giao, quy trình được sử dụng và chênh lệch quy trình, ngày bắt đầu và kết thúc theo ước lượng và thực tế, các công cụ được sử dụng, v.v. Phần này cũng chứa một mô tả ngắn gọn về kinh nghiệm sử dụng các công cụ ("báo cáo kinh nghiệm" chi tiết được đưa vào hệ thống BOK). Các thông tin về các công cụ có thể được sử dụng bởi các dự án khác để quyết định xem việc sử dụng công cụ này có đảm bảo không. Ưu điểm của các công cụ cũng được kiểm tra và từ đó tuyên truyền để dùng chúng cho cả công ty.

Quản lý rủi ro (risk management)

Phần quản lý rủi ro tạo ra các rủi ro khởi tạo lúc ban đầu (dự kiến) cho dự án cùng với các bước giảm thiểu rủi ro được lên kế hoạch. Ngoài ra, phần này liệt kê những rủi ro hàng đầu (top risks) như được thấy trong phân tích hậu dự án (post-project analysis) (chúng là những rủi ro thực của dự án vừa hoàn thành). Những thông tin này có thể được sử dụng bởi các dự án sau này và có thể được sử dụng để cập nhật lại các hướng dẫn quản lý rủi ro (risk management guidelines). Hiệu quả của các bước giảm thiểu rủi ro đã được triển khai cũng được ghi nhận lại.

Kích thước (size)

Như đã thảo luận trong Chương 6, nhiều dự án sử dụng tiếp cận từ dưới lên (bottom-up) để ước lượng. Trong tiếp cận này, kích thước của phần mềm được ước lượng dưới dạng số lượng các mô-đun đơn giản, trung bình, hoặc phức tạp. Do đó, kích thước này được lưu trữ lại cùng với các tiêu chí (criteria) được sử dụng để phân loại (các dự án khác nhau có thể sử dụng các tiêu chí khác nhau). Dữ liệu của cả hai loại kích thước: kích thước ước lượng (estimated size) và kích thước thực tế (actual size) sẽ được lưu lại.

Cho các mục đích bình thường, năng suất của một dự án được đo bằng số Điểm chức năng (FP - function points) mỗi người-tháng. Mặc dù FP có thể được tính bằng cách phân tích các chức năng của hệ thống, nhưng tại thời điểm kết thúc dự án nó có thể được suy ra từ số dòng mã (LOC - lines of code) được đo được. Nếu nhiều ngôn ngữ được sử dụng, chúng tôi đơn giản thêm các kích thước (dùng FP) của các mô-đun trong các ngôn ngữ khác nhau (FP). Ở đây, Điểm chức năng (không giống như dòng mã) không phải là một số đo phụ trợ. Bởi vì chúng ta đang đo kích thước của hệ thống đã hoàn thành bằng FP, tuy nhiên, phương pháp này thì tương đương với việc chuyển đổi tất cả số lượng LOC sang số lượng LOC của một số ngôn ngữ "chung" (universal) và từ đó chuyển đổi kích thước này sang FP. Hơn nữa, do các hạn chế cố hữu của các số đo phần mềm (software metrics) và sự sử dụng của chúng, một ít không chính xác thì chấp nhận được, với điều kiện là các phương pháp được sử dụng phải nhất quán. Kích thước tính bằng FP cũng được ghi lại vào báo cáo phân tích kết thúc.

Nỗ lực (effort)

Báo cáo phân tích kết thúc cũng chứa nỗ lực được ước lượng (estimated effort) tổng cộng và nỗ lực thực tế (actual effort) tính theo người-giờ. Nỗ lực ước lượng tổng cộng có được từ kế hoạch quản lý dự án. Nỗ lực thực tế tổng cộng là của tổng các nỗ lực được báo cáo trong tất cả các WARs đã được gửi bởi các thành viên của dự án, bao gồm cả người lãnh đạo dự án. Nếu chênh lệch giữa giá trị thực tế và ước lượng là lớn, lý do cho sự khác biệt này sẽ được ghi lại.

Cho từng bước chính của quy trình, nỗ lực thực tế tổng cộng (total actual effort) và nỗ lực ước lượng tổng cộng (total estimated effort) cho giai đoạn cũng được ghi lại. Thông tin này có thể hữu ích trong việc lập kế hoạch, và nó là một đầu vào quan trọng để hình thành cơ sở dữ liệu về khả năng của quy trình (PCB). Cho từng giai đoạn, nếu có thể, nỗ lực này được tách ra thành nỗ lực cho công việc (task), cho việc xem xét lại (review), và cho làm lại (rework). Các mã WAR (đã được mô tả trước đây trong cuốn sách này) cho phép tách ra như thế này. Sau đó, sự phân phối nỗ lực cho các giai đoạn khác nhau có thể được tính và ghi lại. Việc tách nỗ lực đã được xài riêng cho công việc (task), xem xét lại (review), và làm lại (rework) nhằm hỗ trợ cho việc xác định phạm vi cải thiện năng suất (scope of productivity improvement).

Chi phí của chất lượng của dự án cũng được tính toán. Nó đo lường chi phí của tất cả các hoạt động trực tiếp tạo ra chất lượng. Chi phí của chất lượng có thể được định nghĩa theo nhiều cách; ở đây nó được định nghĩa như là tỷ lệ phần trăm của tổng nỗ lực đã chi tiêu trong việc xem xét lại, kiểm thử, làm lại công việc để để loại bỏ lỗi, và đào tạo (trainning) cho dự án.

Lỗi (defects)

Phần về lỗi của báo cáo phân tích kết thúc có chứa một bản tóm tắt các lỗi đã được tìm thấy trong suốt quá trình thực hiện dự án. Các lỗi có thể được phân tích tương ứng với mức độ nghiêm trọng (severity) (tỷ lệ phần trăm của các lỗi lớn, nhỏ, hoặc nhẹ bên ngoài), giai đoạn phát hiện (stage detected) (tỷ lệ phần trăm của tổng số lỗi đã được phát hiện bởi từng hoạt động (by activity) đảm bảo chất lượng), giai đoạn tiêm (stage injected) (hoạt động đã gây ra bao nhiêu phần trăm của tổng số lỗi), v.v. Tỷ lệ tiêm và phân phối lỗi cũng được xác định.

Hiệu quả loại bỏ khiếm khuyết (DRE - **Defect Removal Efficiency**) là một thước đo về hiệu quả hoạt động SQA của bạn. Ví dụ, nếu DRE thấp trong quá trình phân tích và thiết kế, nó có nghĩa là bạn nên dành thời gian cải thiện cách bạn tiến hành xem xét lại một cách cách hình thức (formal reviews).

DRE = E / (E + D)

Trong đó, E = Số Lỗi được tìm thấy và xóa bỏ trước khi giao sản phẩm phần mềm và D = Số Lỗi được tìm thấy và xóa bỏ sau khi giao sản phẩm phần mềm.

Giá trị lý tưởng của DRE nên là 1, có nghĩa là không có lỗi được tìm thấy. Nếu bạn có một DRE giá trị thấp, nó có nghĩa là bạn cần phải xem lại quy trình hiện tại của bạn. Về bản chất DRE là một chỉ số cho biết khả năng của hoạt động kiểm soát chất lượng và đảm bảo chất lượng (DRE is a indicator of the filtering ability of quality control and quality assurance activity). Nó khuyến khích các nhóm cố gắng tìm ra nhiều lỗi nhất nếu có thể trước khi chúng bị chuyển qua giai đoạn hoạt động tiếp theo (passed to the next activity stage).

Do đó, số đo này rất hữu ích để xác định các hoạt động chất lượng cần được cải thiện thêm. Báo cáo kết thúc cho biết hiệu quả loại bỏ lỗi của những hoạt động kiểm soát chất lượng chủ yếu, cũng như hiệu quả loại bỏ lỗi tổng thể của quy trình. Các phân tích khác được làm lên dữ liệu về lỗi cũng có thể được đưa vào bản báo cáo. Đôi khi, một phân tích riêng biệt lên dữ liệu về xem xét lại (review data) có thể được thực hiện. Các mức độ lỗi ước lượng so với mức độ lỗi thực tế cũng được phân tích.

Phân tích nhân quả (Causal Analysis)

Khi dự án hoàn thành, hiệu suất của quy trình tổng thể của dự án này sẽ được biết. Nếu hiệu suất nằm bên ngoài phạm vi được cung cấp bởi baseline về khả năng (capability baseline), đây là một cơ hội tốt vì ta có thể tìm ra được nguyên nhân cho biến đổi (variability) này. Phân tích nhân quả liên quan đến các biến đổi lớn và sau đó xác định các nguyên nhân của chúng, thường được thực hiện thông qua thảo luận và ý kiến cá nhân trong cuộc họp.

Các thành phần của quy trình (Process Assets)

Ngoài các dữ liệu số đo (metrics data), các đối tượng khác của dự án (project artifact) cũng có tiềm năng hữu ích cho các dự án trong tương lai. Chương 2 thảo luận về việc sử

dụng các thành phần của quy trình (process assets). Những thành phần của quy trình này được thu thập sau khi dự án kết thúc. Các mục (entries) tiềm năng cho BOK cũng được xác định trong suốt quá trình kết thúc (closure) dự án, mặc dù chúng sẽ được gửi sau đó.

12.2 BÁO CÁO PHÂN TÍCH KẾT THÚC DỰ ÁN ACIC

Phần này trình bày báo cáo phân tích kết thúc dự án ACIC. Đầu tiên, báo cáo cung cấp một số thông tin chung về dự án. Tóm tắt hiệu suất cho thấy dự án đã xài một lượng nỗ lực vượt 19% - bị gây ra bởi hai yêu cầu thay đổi lớn. Nó cũng cung cấp các so sánh giữa dữ liệu của kế hoạch (planned data) với dữ liệu thực tế (actual data) cho kích thước nhóm, bắt đầu và ngày kết thúc, chất lượng, năng suất, chi phí của chất lượng, tỷ lệ tiêm lỗi, và hiệu quả loại bỏ lỗi. Hầu như trong tất cả các tham số này, hiệu suất thực tế đã rất gần với hiệu suất ước lượng. Tỷ lệ tiêm lỗi thực tế là khoảng 26% thấp hơn ước lượng, chủ yếu là nhờ vào các hoạt động phòng ngừa lỗi.

Báo cáo cung cấp một cái nhìn tổng quan về việc điều chỉnh quy trình đã được thực hiện trong dự án và xác định cụ thể các công cụ nội bộ và bên ngoài đã được sử dụng. Để quản lý rủi ro, bản báo cáo thảo luận về những rủi ro đã được xác định lúc ban đầu cũng như những rủi ro thực tế mà người lãnh đạo dự án và SEPG cảm thấy đã tồn tại trong dự án. Như bạn có thể thấy, có sự khác nhau; một rủi ro mới - chuyển đổi sang VAJ 3.0 - phát sinh trong quá trình thực hiện dự án. Các ghi chú về trạng thái giảm thiểu rủi ro, rủi ro này đã được quản lý một cách hiệu quả bằng cách chỉ ra tác động của các thay đổi đến khách hàng và sau đó là sự đồng ý để trì hoãn việc chuyển đổi sang VAJ 3.0 này để thực hiện nó ở một phiên bản tương lai. Đối với các rủi ro khác, các ghi chú sẽ đánh giá hiệu quả của các chiến lược giảm thiểu rủi ro.

Bản báo cáo ghi lại kích thước ước lượng và kích thước thực tế dưới dạng số lượng các chương trình theo độ phức tạp khác nhau. Kích thước của hệ thống đầu ra cuối cùng cũng được cho dưới dạng LOC, cùng với ngôn ngữ. Đối với dự án này, kích thước khoảng 33 KLOC trong mã Java - được chuyển đổi tương ứng thành khoảng 1612 FP, và khoảng 1K mã COBOL – được chuyển đổi tương ứng thành khoảng 12FP. Kích thước này được sử dụng để tính toán năng suất và chất lượng của dự án.

Kế tiếp, thời gian hoàn thành theo ước lượng (estimated schedules) và thực tế (actual schedules) của các giai đoạn khác nhau được cung cấp. Ở những nơi có chênh lệch lớn (ví dụ, trong kiểm thử chấp nhận), báo cáo sẽ cung cấp lý do cho sự trượt.

Tiếp theo được trình bày là dữ liệu về nỗ lực. Đầu tiên, báo cáo cung cấp sự phân phối nỗ lực thực tế qua các giai đoạn khác nhau của dự án, cùng với lượng nỗ lực đã được dùng cho công việc (task), xem xét lại (review), và làm lại (rework) ở từng giai đoạn. Dùng sự tách ra này, chi phí cho chất lượng được tính toán là đã chiếm 31.4% của dự án này. Sau đó, nỗ lực ước lượng và thực tế của các giai đoạn khác nhau của dự án được cung cấp, ở cùng với lý do của sự chênh lệch. Như bạn có thể thấy, trong dự án này chênh lệch tổng thể không phải là quá lớn, mặc dù trong một số giai đoạn, chênh lệch thì đáng kể và đôi khi tiêu cực.

Tiếp theo, báo cáo chứa một phân tích về các lỗi. Sự phân bố lỗi (defects distribution) trong các giai đoạn phát hiện lỗi khác nhau được cung cấp, cùng với ước lượng cho những giai đoạn này. Như đối với các tham số khác, phần trăm chênh lệch (deviation) cũng được chỉ ra. Ở đây, chênh lệch tổng thể là -20%, mặc dù có chênh lệch đáng kể ở một số giai đoạn. Báo cáo nêu rõ lý do của sự giảm lỗi tổng thể và của sự chênh lệch đáng kể trong phân phối lỗi thực tế. Sau đó, dữ liệu về lỗi được cung cấp theo giai đoạn phát hiện (stage detected) và giai đoạn tiêm (stage injected). Sử dụng những dữ liệu này, hiệu quả loại bỏ lỗi cho từng giai đoạn loại bỏ lỗi sẽ được tính toán. Trong dự án này, hiệu quả loại bỏ là 100% cho các yêu cầu (requirements) và xem xét lại thiết kế (design review), chỉ có 55% cho xem xét lại mã (code review), chỉ có 32% cho kiểm thử đơn vị, 91% cho kiểm thử hệ thống, và 100% cho kiểm thử chấp nhận (bởi vì chỉ có các lỗi được loại bỏ trước khi kết thúc kiểm thử chấp nhận là được biết đến). Hiệu quả loại bỏ các lỗi tổng thể là 97.5%, được thỏa; mục tiêu là 97%. Sự phân bố lỗi tương ứng với mức độ nghiêm trọng và loại lỗi cũng đã được tính toán.

Cuối cùng, một phân tích nhân quả (causal analysis) cung cấp các lý do có thể có của quy trình này cho các tình huống mà tại đó các mục tiêu đã lên kế hoạch nhưng không đạt được. Trong dự án này, lý do cho sự chênh lệch hiệu suất được thảo luận, cùng với các dữ liệu về hiệu suất. Các bài học kinh nghiệm từ dự án này được tóm tắt ở đây. Các thành phần của quy trình (process assets) vừa được nộp (submitted) cũng được ghi lại.

Báo cáo kết thúc của dự án ACIC (Closure Report of the ACIC Project)

1. THÔNG TIN CHUNG (GENERAL INFORMATION)

Mã dự án *(Project Code)*	Xxxxx
Chu kỳ sống *(Life Cycle)*	Phát triển, toàn bộ chu kỳ sống *(Development, Full life cycle)*
Lĩnh vực nghiệp vụ *(Business Domain)*	Tài chính. Ứng dụng Web để quản lý tài khoản. *(Finance. Web-based application for managing accounts.)*
Người lãnh đạo dự án/lãnh đạo mô-đun *(Project leader/Module Leader)*	Xxxxxx
Người quản lý nghiệp vụ (Business Manager)	
Cố vấn chất lượng *(Software Quality Adviser)*	Xxxxx

2. TÓM TẮT HIỆU SUẤT (PERFORMANCE SUMMARY)

Tham số hiệu suất *(Performance Parameter)*	**Thực tế *(Actual)***	**Ước lượng *(Estimated)***	**Chênh lệch *(Deviation)***	**Lý do chênh lệch (nếu lớn) *(Reasons for Deviation (If Large))***
Nỗ lực tổng cộng *(Total Effort (person-days))*	597	501	19%	Hai yêu cầu thay đổi lớn đến. *(Two major change requests that came.)*
Kích thước nhóm lúc cao điểm *(Peak Team Size)*	9	9	0	N/A
Ngày bắt đầu *(Start Date)*	03 Apr 2000	03 Apr 2000	0	N/A
Ngày kết thúc *(End Date)*	03 Nov 2000	30 Nov 2000	27 ngày (Days)	Hai yêu cầu thay đổi lớn tiêu tốn hơn 5% nỗ lực. *(Two major change requests consumed more than 5% of the effort.*)

Chất lượng (số lỗi được giao/FP) *(Quality (number of defects delivered per FP))*	0.002	0.0125		Chất lượng được cải tiến bởi vì phòng ngừa lỗi và sử dụng quy trình tăng dần. *(Quality improved because of defect prevention and use of incremental process.)*
Năng suất *(Productivity)*	58	57	2%	N/A
Chi phí của chất lượng *(Cost of quality)*	31.4%	33%	5%	N/A
Tỷ lệ tiêm lỗi *(Defect injection rate)*	0.022	0.03	–26%	Đã được cải tiến bởi vì phòng ngừa lỗi. *(Improved because of defect prevention.)*
Hiệu quả loại bỏ lỗi *(Defect removal efficiency)*	97.4	97	ít (Small)	N/A

3. CHI TIẾT QUY TRÌNH (PROCESS DETAILS)

Điều chỉnh quy trình *(Process Tailoring)*	• Quy trình RUP được sử dụng (Rational Unified Process was employed.) • Phát triển và phân tích được làm một cách lặp lại – 3 vòng lặp phát triển và 2 vòng lặp thiết kế và phân tích. *(Development and analysis were done iteratively—3 iterations for development and 2 for design and analysis.)* • Việc theo dõi dấu vết nguồn gốc yêu cầu được làm bằng cách dùng công cụ Requisite Pro. *(Requirement traceability was done through Requisite Pro tool.)*

4. CÔNG CỤ ĐƯỢC SỬ DỤNG (TOOLS USED)

Ghi chú về các công cụ được sử dụng *(Notes on Tools Used)*	• Công cụ bên ngoài: VSS, VJA, Requisite Pro, MSP *(External Tools: VSS, VJA, Requisite Pro, MSP)* • Công cụ bên trong: BugsBunny, WAR *(Internal Tools: BugsBunny, WAR)*

5. QUẢN LÝ RỦI RO (RISK MANAGEMENT)

Các rủi ro đã được xác định lúc bắt đầu dự án (Risks identified at the start of the project)

Risk 1	Thiếu sự hỗ trợ từ kiến trúc sư cơ sở dữ liệu và người quản trị cơ sở dữ liệu của khách hàng. *(Lack of support from database architect and database administrator of the customer)*
Risk 2	Dùng sai RUP bởi vì nó được sử dụng lần đầu tiên. *(Improper use of RUP, as it is being used for the first time)*
Risk 3	Mất nhân sự *(Personnel attrition)*
Risk 4	Các vấn đề về việc sẽ làm việc với cơ sở dữ liệu của khách hàng thông qua liên kết. *(Problems with working on customer's database over the link)*

Các rủi ro gặp phải trong suốt quá trình thực hiện dự án (Risks encountered during the project)

Risk 1	Tác động của việc chuyển đổi sang VAJ 3.0 *(Impact of conversion to VAJ 3.0)*
Risk 2	Thiếu sự hỗ trợ từ kiến trúc sư cơ sở dữ liệu và người quản trị cơ sở dữ liệu của khách hàng. *(Lack of support from database architect and database administrator of the customer)*
Risk 3	Dùng sai RUP bởi vì nó được sử dụng lần đầu tiên. *(Improper use of RUP, as it is being used for the first time)*
Risk 4	Mất nhân sự *(Personnel attrition)*

Các ghi chú về giảm thiểu rủi ro (**Notes on Risk Mitigation**)

Risk1: Nói rõ ràng về rủi ro đã giúp khách hàng đồng ý trì hoãn việc chuyển đổi với chi phí phù hợp cho tác động của nó.

Risk2: Các chiến lược giảm thiểu đã được lập kế hoạch chi tiết và cẩn thận và sử dụng người điều phối tại chỗ (on-site coordinator) một cách có hiệu quả.

Risk3: Đào tạo đội ngũ về RUP có hiệu quả. Vì vậy, đã luôn được giữ liên lạc với khách hàng.

Risk 4: Vẫn là một rủi ro, mặc dù nó đã không xảy ra. Tác động có thể đã được giảm thiểu bởi vì nhiều người đã luôn được thông báo cho biết (informed) về từng hoạt động quan trọng (critical activity).

6. KÍCH THƯỚC (SIZE)

	Ước lượng (Estimated)	**Thực tế (Actual)**
Số trường hợp sử dụng đơn giản *(Number of simple use cases)*	5	5
Số trường hợp sử dụng trung bình *(Number of medium use cases)*	9	9
Số trường hợp sử dụng phức tạp *(Number of complex use cases)*	12	12

Ghi chú về ước lượng (Notes on Estimation)

Tiêu chí phân loại (Classification Criteria): Định nghĩa tiêu chuẩn về đơn giản, trung bình, và phức tạp đã được sử dụng để phân loại các trường hợp sử dụng. Điều này đã được thực hiện tốt.

Kích thước hệ thống tính theo FP

Kích thước của mã nguồn sau cùng được đo theo LOC. Nó được chuẩn hóa sang FP bằng cách sử dụng các bảng chuyển đổi đã được công bố. Đối với Java, các bảng đã được công bố cho thấy rằng 21 LOC bằng 1 FP và đối với COBOL, 107 LOC bằng 1 FP.

Ngôn ngữ kết quả ***(Output Language)***	**Kích thước tính theo LOC** ***(Size in LOC)***	**Kích thước tính theo FP** ***(Size in FP)***
Java	33,865	1612
COBOL	1241	12

7. THỜI GIAN BIỂU (SCHEDULE)

Giai đoạn (Phase)	Thời gian thực tế đã dùng (ngày) (Actual Elapsed Time)	Thời gian ước lượng (ngày) (Estimated Time)	% trượt (Slippage)	Lý Do trượt (Reasons for Slippage)
Yêu cầu *(Requirements)*	28.67	31	–6.5	
Thiết kế mức cao *(High-level design)*	0	0	0.0	
Thiết kế chi tiết *(Detailed design)*	38.8	42	–6.7	
Cài đặt mã *(Coding)*	132	135	–1.6	
Kiểm thử đơn vị *(Unit testing)*	9	10	–9.3	
Toàn bộ - Xây dựng *(Total – Build)*	141	144	–2.1	
Kiểm thử tích hợp *(Integration test)*	40	40	0	
Kiểm thử hệ thống (System testing)	15	0	0.0	
Kiểm thử chấp nhận *(AT - Acceptance testing)*	30	10	200.0	AT đầy đủ được mở rộng trên yêu cầu khách hàng *(AT completion was extended on customer's request.)*

8. NỖ LỰC (EFFORT)

Phân phối qua các giai đoạn của chu kỳ sống (Distribution over Life-Cycle Stages)

Giai đoạn (Stage)	**Công việc (Task)**	**Xem xét lại (Review)**	**Làm lại (Rework)**	**Tổng (Total)**
Yêu cầu *(Requirements)*	210.0	10.0	60.0	280.0
Thiết kế mức cao *(High-level design)*	0.0	0.0	0.0	0.0
Thiết kế chi tiết *(Detailed design)*	652.0	14.0	29.5	695.5
Cài đặt mã *(Coding)*	1188.0	39.5	76.5	1304.0
Kiểm thử đơn vị *(Unit testing)*	129.5	0.0	17.0	146.5
Kiểm thử tích hợp *(Integration testing)*	567.5	6.0	160.5	734.0
Kiểm thử hệ thống *(System testing)*	90.0	0.0	0.0	90.0
Kiểm thử chấp nhận *(Acceptance testing)*	336.5	0.0	0.0	336.5
Tổng số - Các giai đoạn của chu kỳ sống *(Total - LC stages)*	3173.5	69.5	343.5	3586.5
Quản lý dự án *(Project management)*	733.1	0.0	0.0	733.1
Đào tạo (Training)	104.5	0.0	0.0	104.5
Quản lý cấu hình (CM)	317.0	0.0	0.0	317.0
Misc.	488.5	0.0	0.0	488.5
Total – mgmt, training, and misc.	1643.0	0.0	0.0	1643.0
Tổng nỗ lực (người-giờ) *(Total Effort (Person-hours)))*	**4816.50**	**69.50**	**343.50**	**5229.50**
Tổng nỗ lực (người-tháng) *(Total Effort (Person-months)))*	**25.76**	**0.37**	**1.84**	**27.97**

Chi phí của chất lượng (Cost of Quality)

$$COQ = \frac{\text{Review effort} + \text{rework effort} + \text{test effort} + \text{training effort}}{\text{total effort}} = 100$$

$$= (69.5 + 343.5 + 129.5 + 567.5 + 90 + 336.5 + 104.5)/5229.5 \times 100$$

$$= 31.4\%$$

Phân phối nỗ lực thực tế so với ước lượng (Effort Distribution and Actual Versus Estimated)

	Thực tế (Actual)		**Ước lượng (Estimated)**			
Giai đoạn (Stage)	**Nỗ lực (người-giờ) (person-hours)**	**%**	**Nỗ lực (người-giờ) (person-hours)**	**%**	**% Chênh lệch (Deviation)**	**Lý do chênh lệch (Reasons for Deviation)**
Yêu cầu *(Requirements)*	280	5.35	475.0	10	–30	Ước tính quá cao nỗ lực (dữ liệu từ các dự án trước đây đã không giúp ích gì cho ước tính này bởi vì nó không có giai đoạn này.) *(Overestimated this effort (data from earlier project did not help because it did not have this phase.))*
Thiết kế (mức cao và chi tiết) *(Design (HLD and detailed))*	695.5	13.30	569.0	12	22	Thiết kế cần nhiều thời gian hơn bởi vì nhóm không có kinh nghiệm về Rational Rose và OOAD. *(Design took more time because team was inexperienced with Rational Rose and OOAD.)*

Cài đặt mã *(Coding)*	1304.0	24.94	1235.3	26	6	
Kiểm thử đơn vị *(Unit testing)*	146.5	2.80	142.5	3	3	
Kiểm thử tích hợp *(Integration testing)*	734.0	14.04	331.0	7	120	Nhiều nỗ lực được xài để sửa chữa lỗi xuất hiện trong suốt quá trình hòa giải với Synergy và mã Window Resized. *(Much effort spent on fixing bugs introduced during reconciliation with Synergy and Window Resized code.)*
Kiểm thử hệ thống *(System testing)*	90.0	1.72	95.0	2	–5	
Kiểm thử chấp nhận *(Acceptance testing)*	336.5	6.43	285.0	6	18	Các kiểm thử chấp nhận đã không hoàn thành vào này 3 tháng 11 và đã được gia hạn đến 23 tháng 11 do các trì hoãn từ khách hàng *(Acceptance testing was not completed on Nov 3 and was extended until Nov 23 due to delays from the customer.)*
Tổng – các giai đoạn của chu kỳ sống *(Total—LC stages)*	3586.5	68.58	3132.8	66	14.5	
Quản lý dự án	733.1	14.02	713.0	15	3	

(Project management)						
Đào tạo *(Training)*	104.5	2.00	455.0	10	–77	
Quản lý cấu hình *(CM)*	317.0	6.06	142.0	3	123	Chênh lệch do các vấn đề khi hòa giải *(Deviation due to reconciliation issues.)*
Misc.	488.5	9.34	285.0	6	71	Nhiều hơn bởi vì đào tạo *(More because of training.)*
Tổng cộng – quản lý, đào tạo, và misc.) *(Total—mgmt, training, and misc.)*	1643.0	31.42	1595.0	34	3.01	
Tổng *(Total)*	**5229.5**	**100**	**4727.8**	**100**	**10.6**	

9. LỖI (DEFECTS)

Phân phối lỗi (Defect Distribution)

Giai đoạn phát hiện (Stage Detected)	**Số lỗi thực tế (Actual Number of Defects)**	**% của tổng số lỗi được phát hiện (% of Total Defects Found)**	**Số lỗi được ước lượng (Estimated Number of Defects)**	**% của tổng số lỗi được ước lượng (% of Total Estimated Defects)**	**% chênh lệch (Deviation)**
Xem xét lại yêu cầu và thiết kế *(Req. and design review)*	11	10	29	20	–62
Xem xét lại mã *(Code review)*	58	50	29	20	100

Kiểm thử đơn vị *(Unit testing)*	15	13	57	40	–73
Tích hợp và kiểm thử *(Integration and system testing)*	29	25	25	17	16
Kiểm thử chấp nhận *(Acceptance testing)*	3	2	5	3	–40
Tổng *(Total)*	**116**	**100**	**145**	**100**	**–20**

Lý do cho chênh lệch (Reasons for Deviation)

1. Phòng ngừa lỗi làm giảm tỷ lệ tiêm lỗi trong các giai đoạn sau đó, kết quả là làm giảm tỷ lệ tiêm lỗi tổng thể.

2. Trong dự án trước đó – dự án mà dữ liệu của nó được dùng để làm các ước lượng (dự toán) cho dự án này – việc xem xét lại mã (code reviews) đã được thực hiện ít hơn và đã có một sự phụ thuộc lớn vào kiểm thử đơn vị (UT- unit testing). Ngược lại, trong dự án này, bởi vì việc xem xét lại mã đã được thực hiện một cách nghiêm ngặt và rộng hơn, nhiều lỗi hơn đã được tìm thấy trong quá trình xem xét lại, dẫn đến sự sụt giảm đáng kể số lỗi được tìm thấy trong kiểm thử đơn vị.

Hiệu quả loại bỏ lỗi (Defect Removal Efficiencies)

Giai đoạn phát hiện lỗi *(Defects Detection Stage)*	**Giai đoạn tiêm lỗi *(Defects Injection Stage)***			**Hiệu quả loại bỏ lỗi *(Defect Removal Efficiency)***
	Yêu cầu (Req.)	**Xây dựng (Build)**	**Thiết kế (Design)**	
Xem xét lại yêu cầu *(Req. review)*	5			100%
Xem xét lại thiết kế *(Design review)*	0	6		100%

Xem xét lại mã *(Code review)*	0	0	58	55% (58 / 58 + 15 + 29 + 3)
Kiểm thử đơn vị *(Unit testing)*	0	0	15	32% (15 / 15 + 29 + 3)
Kiểm thử và tích hợp hệ thống *(Integration/system testing)*	0	0	29	91% (29 / 29 + 3)
Kiểm thử chấp nhận *(Acceptance testing)*	0	0	3	100%

Hiệu quả loại bỏ lỗi tổng thể (Overall Defect Removal Efficiency) = 113 / 116 = 97.4 %

Trong đó hiệu quả loại bỏ lỗi của một giai đoạn/hoạt động SQA được tính theo công thức sau:

$$\textbf{DRE \%} = \frac{\text{Defects Removed during a Phase}}{\text{Defects Removed till date}} \times 100$$

Và hiệu quả loại bỏ lỗi của cả dự án là:

$$\textbf{DRE \%} = \frac{\text{Total No. Of Defects before Release/Delivery}}{\text{Total No. Of Defects for the Project}} \times 100$$

Sự phân phối theo mức độ nghiêm trọng (Distribution by Severity)

STT	**Mức độ nghiêm trọng (Severity)**	**Số lỗi (Number of Defects)**	**% tổng số lỗi (% of Total Defects)**
1	Nhẹ bên ngoài *(Cosmetic)*	26	22.4
2	Nhỏ *(Minor)*	51	44
3	Lớn *(Major)*	36	31
4	Nghiêm trọng *(Critical)*	3	2.6
5	Khác *(Others)*	—	—
	Tổng (Total)	**116**	

Sự phân phối theo loại lỗi (Distribution by Defect Type)

STT	Loại lỗi (Defect Type)	Số lượng lỗi (Number of Defects)	% của tổng lỗi (% of Total Defects)
1.	Lôgic *(Logic)*	33	28.4
2.	Chuẩn *(Standards)*	29	25
3.	Hiệu suất *(Performance)*	24	20.7
4.	Mã dư thừa *(Redundant code)*	14	12
5.	Giao diện người dùng *(User interface)*	9	7.7
6.	Kiến trúc *(Architecture)*	4	3.5
7.	Sự nhất quán *(Consistency)*	2	1.7
8.	Khả năng tái sử dụng *(Reusability)*	1	0.9
	Tổng (Total)	**365**	

10. PHÂN TÍCH NHÂN QUẢ VÀ BÀI HỌC KINH NGHIỆM (CAUSAL ANALYSIS AND LESSONS LEARNED)

Có rất ít các chênh lệch lớn ở hiệu suất quy trình; hiệu suất thực tế đã gần bằng với những gì đã được dự kiến. Những lý do chênh lệch, đối với những chênh lệch lớn, được đưa ra kèm theo. Một số bài học quan trọng đã học được là:

1. Phát triển tăng dần (incremental development) hoặc phát triển theo từng giai đoạn (phased development) là vô cùng hữu ích để đạt được chất lượng và năng suất cao hơn bởi vì dữ liệu từ giai đoạn đầu tiên có thể được sử dụng để cải thiện các giai đoạn còn lại thông qua việc phòng ngừa lỗi (defect prevention).

2. Phòng ngừa lỗi có thể làm giảm đáng kể tỷ lệ tiêm lỗi. Về mặt nỗ lực (effort), phòng ngừa lỗi giúp tiết kiệm nhiều nỗ lực; bằng cách xài ít nỗ lực (giờ), có thể đạt đến từ 5 đến 10 lần tiết kiệm nỗ lực thông qua hình thức giảm nỗ lực làm lại (rework effort).

3. Nếu yêu cầu thay đổi có ảnh hưởng lớn, việc thảo luận với khách hàng bằng cách sử dụng một phân tích chi tiết các tác động có thể rất hữu ích trong việc thiết lập những kỳ vọng và làm một phân tích lợi ích về mặt chi phí (cost-benefit) một cách đúng cách thức (có thể dẫn đến việc phải hoãn trì hoãn các thay đổi, như đã xảy ra trong dự án này).

4. Lỗi hiệu quả loại bỏ (defect removal efficiencies) của việc xem xét lại mã (code reviews) và kiểm thử đơn vị (unit testing) thì rất thấp. Quy trình cho cả hai, và việc cài đặt (thực hiện) các quy trình này, cần phải được xem xét lại để cải thiện những con số này. Trong dự án này, kiểm thử và tích hợp hệ thống đã bù đắp cho sự kém hiệu quả của các hoạt động xem xét lại và kiểm thử đơn vị. Tuy nhiên, đối với các dự án lớn, điều này thì không thể và sự kém hiệu quả trong các hoạt động xem xét lại và kiểm thử đơn vị có thể có ảnh hưởng xấu đến chất lượng.

11. CÁC THÀNH PHẦN CỦA QUY TRÌNH TRÌNH ĐƯỢC NỘP (PROCESS ASSETS SUBMITTED)

Kế hoạch quản lý dự án (project management plan), tiến độ (biểu thời gian) của dự án (project schedule), kế hoạch quản lý cấu hình (configuration management plan), các tiêu chuẩn cài đặt mã Java (Java coding standards), danh sách kiểm tra để xem xét lại mã (code review checklist), danh sách kiểm tra để xem xét lại kế hoạch tích hợp (integration plan review checklist), danh sách kiểm tra để phân tích tác động (impact analysis checklist), các báo cáo phân tích nhân quả để phòng ngừa lỗi (causal analysis reports for defect prevention).

12. TÀI LIỆU THAM KHẢO

Bỏ qua.

12.3 TÓM TẮT

Một dự án không kết thúc sau khi giao và cài đặt phần mềm; trước khi nó được đóng lại, nó phải được sử dụng cho việc học tập. Phân tích kết thúc dự án là một trong những phương pháp để đạt được mục tiêu này.

Sau đây là một số bài học quan trọng được rút ra từ cách tiếp cận của Infosys để kết thúc dự án:

- Phân tích kết thúc dự án dựa vào các số đo (metrics). Phân tích dữ liệu để hiểu về hiệu suất của dự án và những nguyên nhân đã gây ra bất kỳ chênh lệch lớn nào. Những nguyên nhân này có thể được dùng như là một nguồn để tạo ra các sáng kiến cải tiến.

- Việc phân tích các số đo nên báo cáo về chất lượng cuối cùng được giao cho khách hàng (productivity achieved), sự phân phối nỗ lực (distribution of effort) và phân phối lỗi (distribution of defects), hiệu quả của việc việc loại bỏ lỗi (defect removal efficiency) của các hoạt động chất lượng khác nhau, và chi phí của chất lượng (cost of quality).

- Thu thập các thành phần của quy trình như là kế hoạch (plans), danh sách kiểm tra (checklists), các tiêu chuẩn (standards), và các hướng dẫn (guidelines), và làm cho chúng có thể được dùng lại được (available) bởi những người khác.

Đối với CMM, kết thúc dự án (project closure) không phải là một yêu cầu trực tiếp của các KPAs của quản lý dự án. Tuy nhiên, báo cáo kết thúc (closure report) được dung để cung cấp dữ liệu cho cơ sở dữ liệu quy trình (process database) và để tạo baseline về khả năng của quy trình (process capability baseline) – được cần đến để để đáp ứng nhiều yêu cầu của KPA Lập Kế Hoạch Cho Dự Án (Project Planning KPA) và KPA Quản Lý Phần Mềm Tổng Hợp (Integrated Software Management KPA). Chúng cũng hỗ trợ cho việc học tập (learning) và lưu trữ hồ sơ (record keeping), được yêu cầu ở CMM mức 3.

12.4 CÁC THAM KHẢO

1. S. Brady and T. DeMarco. Management-aided software engineering. *IEEE Software*, Nov. 1994.

2. E.J. Chikofsky. Changing your endgame strategy. *IEEE Software*, Nov. 1990.

3. B. Collier, T. DeMarco, and P. Fearey. A defined process for project postmortem review. *IEEE Software*, July 1996.

4. R. Grady. *Successful Software Process Improvement*. Prentice Hall PTR, 1997.

5. K. Caputo. *CMM Implementation Guide: Choreographing Software Process Improvement*. Addison-Wesley, 1998.

6. V.R. Basili and H.D. Rombach. The experience factory. In *The Encyclopedia of Software Engineering, John J. Marciniak, editor*. John Wiley and Sons, 1994.

7. K. Kumar. Post implementation evaluation of computer-based information systems: Current practices. *Communications of the ACM*, Feb. 1990.

HẾT