Chương 2

ĐỘ TIN CẬY PHẦN MỀM

2.1. Giới thiệu độ tin cậy phần mềm

Kiểm tra chất lượng phần mềm là một hoạt động khó khăn để chấp nhận về mặt ý thức vì chúng ta đang cân nhắc công việc của chúng ta hoặc của đồng nghiệp để tìm lỗi. Sau quá trình làm việc trong nhóm và trở thành thành viên, chúng ta ngại tìm ra lỗi và không phát hiện được ra chúng thông qua kiểm tra. Khi một người nào đó tiến hành kiểm tra lại không phải là thành viên của dự án, ví dụ một chuyên gia kiểm tra, họ được nhìn nhận như là một kẻ thù.

Thêm vào đó, kiểm tra chất lượng phần mềm lại là một hoạt động khó được chấp nhận đối với việc quản lý vì nó tốn kém, mất thời gian và hiếm khi phát hiện được lỗi. Kết quả là phần lớn các ứng dụng không được kiểm tra đầy đủ và được phát hành với lỗi tiềm ẩn.

Tuy vậy, chất lượng phần mềm cao là một mục tiêu quan trọng của nhóm phát triển phần mềm. Do vậy, cần và phải đảm bảo các tiêu chuẩn của phần mềm như đã đề cập ở các giáo trình nhập môn kỷ nghệ phần mềm. Đảm bảo chất lượng phần mềm là một hoạt động có hệ thống và kế hoạch. Nó bao gồm nhiều nhiệm vụ liên kết với các hoạt động chính sau, nhằm xây dựng sản phẩm phần mềm có độ tin cậy cao. Đó là:

+ Áp dụng các phương pháp kỹ thuật,

+ Tiến hành các cuộc xét duyệt kỹ thuật chính thức,

+ Kiểm thử phần mềm,

+ Buộc tôn trọng các chuẩn,

+ Kiểm soat thay đổi,

+ Đo chất lượng,

+ Báo cáo, lưu giữ kết quả.

Như vậy, độ tin cậy của một hệ phần mềm là độ đo về mức độ tốt của các dịch vụ mà hệ cung cấp cho máy tính. Cần chú ý là người dùng không xét rằng các dịch vụ là quạn trọng như nhau: chẳng hạn một hệ điều khiển máy bay có thể rất, rất hiếm khi thất bại, nhưng nếu chúng có thất bại gây ra tai nạn máy bay thì các người bị nạn và thân nhân người bị nạn không thể xem hệ đó là đáng tin.

Độ tin cậy là một đặc trưng động của hệ thống, nó là một hàm của số các thất bại phần mềm. Một thất bại phần mềm là một sự kiện thi hành mà khi đó phần mềm hành xử không như người ta mong đợi. Chú ý rằng một thất bại phần mềm khác nột hư hỏng phần mềm. Hư hỏng phần mềm là một đặc trưng tĩnh, và nó sẽ gây ra thất bại phần mềm khi mà mã lỗi được thi hành với một tập hợp đặc biệt các thông tin vào. Các hư hỏng không phải luôn luôn xuất đầu lộ diện, vì vậy đọ tin cậy phụ thuộc vào việc sử dụng hệ thống như thế nào. Không thể đưa ra một phát biểu đơn giản và khái quát về độ tin cậy phần mềm.Các hư hỏng phần mềm không phải là các khuyết tật của chương trình. Một hành xử bất ngờ có thể xảy ra khi mà phần mềm phù hợp với các yêu cầu của nó, nhưng mà chính các yếu tố đó lại không đầy đủ. Các sai sót trong các tư liệu phần mềm cũng có thể dẫn đến các hành vi bất ngờ mặc dầu rằng phần mềm không có khiếm khuyết. Có công trình nghiên cứu đã chỉ ra rằng có thể rút bỏ 60% các khiếm khuyết mà chỉ có thể cải tạo được 3% độ tin cậy. Cũng có người đã chú ý rằng nhiều khiếm khuyết trong sản phẩm chỉ là kết quả của hàng trăm hoặc hàng nghìn tháng sử dụng.

Với hệ thống quan trọng, tính tin cậy được đòi hỏi cao, việc thẩm tra và xác nhận tính hợp lệ của hệ thống quan trọng đã rất phổ biến trong việc xác nhận tính hợp lệ của bất kỳ hệ thống nào. Có hai lý do tại sao ta nên làm việc đó:

1. Chi phí thất bại: Chi phí và hậu quả của việc thất bại trong hệ thống quan trọng có khả năng cao hơn trong hệ thống không quan trọng. Bạn giảm nguy cơ thất bại của hệ thống bằng cách sử dụng việc thẩm tra và xác nhận tính hợp lệ của hệ thống. Việc tìm và loại bỏ lỗi trước khi hệ thống được phân phối luôn luôn rẻ hơn chi phí do các sự cố của hệ thống.

2. Xác nhận tính hợp lệ của các thuộc tính tin cậy: Bạn có thể phải tạo ra một trường hợp bình thường để cho các khách hàng thấy hệ thống có chứa các yêu cầu về tính tin cậy (tính sẵn

sàng, tính tin cậy, tính an toàn và tính bảo mật). Để đánh giá đặc tính tin cậy đòi hỏi sự hoạt động cụ thể của thẩm tra và xác nhận.

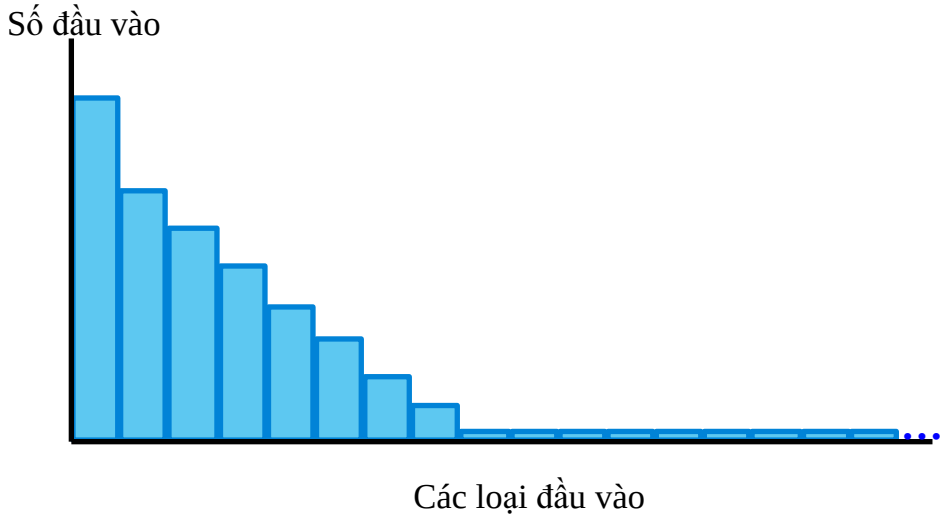
Ví dụ: Người có thẩm quyền trong ngành hàng không phải chứng nhận rằng hệ thống là an toàn trước khi nó có thể cất cánh. Để đạt được chứng nhận này, bạn phải thiết kế và thực thi thủ tục đặc biệt về thẩm tra và xác nhận nhằm tập hợp chứng cớ về tính an toàn của hệ thống.

2.2. Xác định tính tin cậy

2.2.1. Dự đoán hoạt động phần mềm

Dự đoán hoạt động của phần mềm phản ánh cách phần mềm sẽ được sử dụng trong thực tế. Dự đoán hoạt động gồm đặc tả các loại đầu vào và khả năng xuất hiện của chúng. Khi một hệ thống phần mềm mới thay thế một hệ thống bằng tay hoặc một hệ thống tự động hóa, điều đó là dễ thuyết phục để đánh giá các mẫu cách dùng có thể có của phần mêm mới. Nó nên phù hợp với cách sử dụng hiện có, với một vài sự thừa nhận được tạo bởi chức năng mới có thể có trong phần mềm mới.

Thông thường, dự đoán hoạt động như là các đầu vào có khả năng cao nhất được sinh ra và được phân vào một lượng nhỏ các lớp, như chỉ ra ở bên trái hình sau. Có một lượng lớn các lớp có các đầu vào có khả năng không xảy ra cao nhưng không phải là không thể xảy ra. Nó được chỉ ra ở bên phải hình vẽ. Dấu (…) có nghĩa là còn có rất nhiều đầu vào khác.



Musa (Musa 1993; Musa 1998) - một kỹ sư hệ thống viễn thông, và ông đã có thời gian dài làm việc tập hợp dữ liệu người dùng trong lĩnh vực này - đề xuất các nguyên tắc để phát triển các dự đoán hoạt động. Đó là: Với một hệ thống yêu cầu công sức phát triển của khoảng 15 người làm việc trong một năm, dự đoán hoạt động được phát triển trong khoảng 1 người/tháng cho đến 2,3 người/năm, nhưng chi phí đã trải rộng ra hệ thống phát hành. Musa tính rằng công ty của ông cần ít nhất 10 nhóm để đầu tư vào việc phát triển dự đoán hoạt động.

Tuy nhiên, khi một hệ thống phần mềm mới và tiên tiến được phát hành, ta rất khó đoán trước được nó sẽ được sử dụng như thế nào để đưa ra dự đoán hoạt động chính xác. Rất nhiều người dùng với trình độ, kinh nghiệm và sự mong muốn khác nhau có thể sử dụng hệ thống mới. Nó không có cở sở dữ liệu lịch sử cách dùng. Người dùng có thể sử dụng hệ thống theo nhiều cách mà người phát triển hệ thống đã không dự đoán trước.

Vấn đề trở nên phức tạp hơn bởi vì các dự đoán hoạt động có thể thay đổi lúc hệ thống đã được sử dụng. Khi người dùng nghiên cứu một hệ thống mới và trở nên tin tưởng hơn về nó, họ thường sử dụng nó theo những cách phức tạp. Do những khó khăn đó, Hamlet (Hamlet, 1992) cho rằng nó không có khả năng để phát triển một dự đoán hoạt động tin cậy. Nếu bạn không chắc chắn rằng dự đoán hoạt động của bạn là chính xác, thì bạn có thể không tin tưởng về sự chính xác của độ đo tính tin cậy của bạn.

2.2.2. Dự đoán tính tin cậy

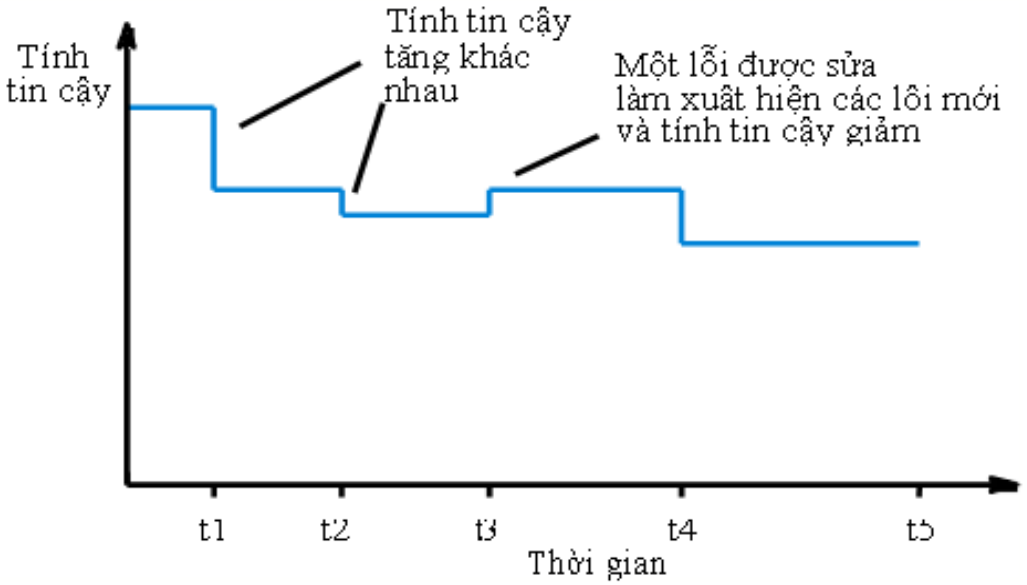
Trong lúc thẩm tra phần mềm, người quản lý phải phân công quá trình kiểm thử hệ thống. Vì quá trình kiểm thử phần mềm rất tốn kém, nên nó sẽ được dừng ngay khi có thể và không “kiểm thử quá” hệ thống. Kiểm thử có thể dừng khi mức độ yêu cầu tính tin cậy của hệ thống đã được thực hiện. Tất nhiên, thỉnh thoảng, các dự đoán tính tin cậy có thể cho thấy mức độ yêu cầu tính tính tin cậy của hệ thống sẽ không bao giờ được thực hiện. Trong trường hợp đó, người quản lý phải đưa ra quyết định khó khăn: viết lại các phần của phần mềm hoặc sửa lại mô tả hệ thống.

Mô hình quá trình gia tăng tính tin cậy là một mô hình mà tính tin cậy của hệ thống thay đổi quá giờ trong thời gian quá trình kiểm thử. Khi các lỗi hệ thống được phát hiện, các khiếm

khuyết cơ sở dẫn đến các lỗi đó đã được sửa chữa, vì vậy tính tin cậy của hệ thống có thể được cải thiện trong lúc kiểm thử và gỡ lỗi hệ thống.

Mô hình đơn giản nhất minh họa khái niệm gia tăng tính tin cậy là mô hình bước chức năng (Jelinski và Moranda, 1972). Tính tin cậy tăng liên tiếp mỗi khi một lỗi (hoặc một tập lỗi) được phát hiện và sửa chữa và một phiên bản mới của phần mềm được tạo ra. Khi sự sửa chữa được tạo ra, tỷ lệ xuất hiện lỗi của phần mềm có thể giảm. Chú ý các chu kỳ thời gian trên trục hoành phản ánh thời gian giữa các lần phát hành hệ thống để kiểm thử, vì vậy nó thường có chiều dài không bằng nhau.

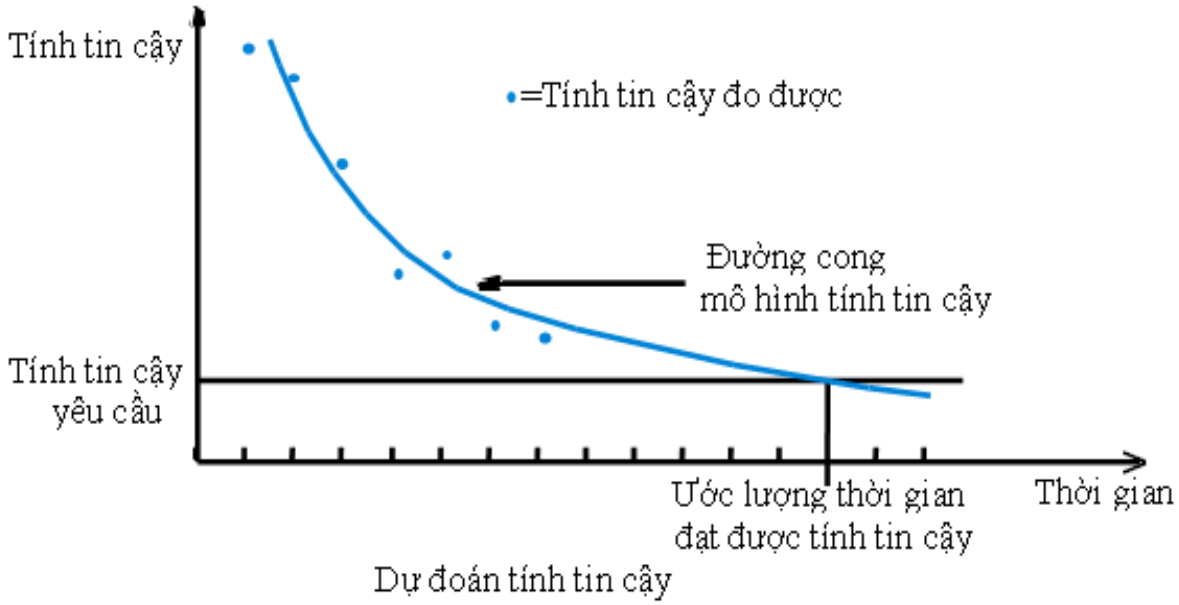
Để dự đoán tính tin cậy, mô hình quá trình gia tăng tính tin cậy nhận thức phải hiểu là mô hình toán học. Có nhiều mô hình quá trình gia tăng tính tin cậy đã được bắt nguồn từ kinh nghiệm trong các lĩnh vực ứng dụng khác nhau. Như Kan (Kan, 2003) đã phát biểu rằng: hầu hết các mô hình đó theo luật số mũ, với tính tin cậy tăng nhanh khi các khiếm khuyết được phát hiện và loại bỏ, thể hiện ở hình sau. Sau đó, sự tăng thêm nhỏ dần đi và tiến tới trạng thái ổn định khi rất ít khiếm khuyết được phát hiện và loại bỏ trong lần kiểm thử cuối cùng.



Mô hình chức năng quá trình gia tăng tính tin cậy với bước tăng ngẫu nhiên

Tuy nhiên, trong thực tế, các lỗi phần mềm không lúc nào cũng được sửa trong lúc gỡ lỗi, và khi bạn thay đổi một chương trình, thỉnh thoảng bạn đưa các lỗi mới vào chương trình đó. Khả năng xuất hiện các lỗi đó có thể cao hơn khả năng xuất hiện các lỗi đã được sửa chữa. Do đó, thỉnh thoảng tính tin cậy của hệ thống có thể trở nên tồi hơn trong phiên bản mới.

Mô hình quá trình gia tăng tính tin cậy đơn giản bước bằng nhau cũng giả sử tất cả các lỗi đóng góp như nhau vào tính tin cậy của hệ thống, và mỗi lỗi được sửa chữa đóng góp một lượng như nhau vào việc gia tăng tính tin cậy. Tuy nhiên, không phải tất cả các lỗi có khả năng xảy ra như nhau. Sửa chữa các lỗi phổ biến đóng góp vào việc gia tăng tính tin cậy nhiều hơn là sửa chữa các lỗi chỉ thỉnh thoảng xảy ra. Có lẽ bạn cũng cho rằng dễ dàng tìm kiếm các lỗi có khả năng xảy ra trong quá trình kiểm thử, vì thế tính tin cậy có thể tăng nhiều hơn khi các lỗi ít có khả năng xảy ra được phát hiện.



Các mô hình ở trên là các mô hình rời rạc phản ánh quá trình gia tăng tính tin cậy. Khi một phiên bản mới của phần mềm đã được sửa lỗi được đưa đi kiểm thử, nó nên có tỷ lệ lỗi xuất hiện thấp hơn phiên bản trước. Tuy nhiên, để dự đoán tính tin cậy sẽ đạt được sau khi thực hiện kiểm thử, chúng ta cần mô hình toán học liên tục. Nhiều mô hình này nhận được từ các lĩnh vực ứng dụng khác nhau, đã được đề xuất và so sánh. Do đó, kiểm thử và gỡ lỗi phải được thực hiện liên tục cho đến khi thỏa mãn các yêu cầu.

Dự đoán tính tin cậy của hệ thống từ mô hình quá trình gia tăng tính tin cậy có hai lợi ích chính:

1. *Lập kế hoạch kiểm thử*: đưa ra lịch kiểm thử hiện tại, ta có thể dự đoán khi nào quá trình kiểm thử được hoàn thành. Nếu điều đó kết thúc sau ngày dự kiến phát hành hệ thống, thì bạn phải triển khai bổ sung tài nguyên cho việc kiểm thử và gỡ lỗi để tăng nhanh tỷ lệ phát triển tính tin cậy.

2. *Sự đàm phán khách hàng*: Đôi khi mô hình tính tin cậy cho thấy sự tăng lên của tính tin cậy rất chậm và sự thiếu cân xứng của các cố gắng kiểm thử được yêu cầu với lợi ích đạt được tương đối ít. Nó có thể đáng giá để đàm phán lại các yêu cầu về tính tin cậy với khách hàng. Một sự lựa chọn khác, mô hình đó dự đoán tính các yêu cầu về tính tin cậy có thể sẽ không bao giờ đạt được. Trong trường hợp đó, ta sẽ phải đàm phán lại với khách hàng về các yêu cầu về tính tin cậy của hệ thống.

2.2.3. Xác định tính tin cậy

Khái niệm độ đo đã được phát triển để chỉ ra yêu cầu tin cậy của một hệ thống. Để xác nhận hệ thống có các yêu cầu nào, bạn phải đo độ tin cậy của hệ thống như bởi một người dùng hệ thống thông thường.

Quá trình đo độ tin cậy của hệ thống gồm 4 bước:

1. Đầu tiên, nghiên cứu các hệ thống tồn tại của các kiểu như nhau để đưa ra mô tả sơ lược hoạt động. Mô tả sơ lược hoạt động nhận biết loại của đầu vào hệ thống và xác suất xuất hiện các đầu vào này trong trường hợp bình thường.

2. Sau đó, thiết đặt tập các dữ liệu kiểm thử để phản ánh mô tả sơ lược hoạt động.Có nghĩa là ta tạo ra dữ liệu kiểm thử với phân bố xác suất như nhau. Thông thường, ta sử dụng máy sinh dữ liệu kiểm thử để kiểm tra quá trình này.

3. Kiểm thử hệ thống sử dụng các dữ liệu đã được sinh ở trên và đếm số lượng và các loại lỗi xảy ra. Số lần lỗi cũng được ghi nhận. Các đơn vị thời gian ta nên chọn phù hợp với độ đo tính tin cậy đã dùng.

4. Cuối cùng, ta tiến hành thống kê các lỗi quan trọng, có thể tính toán độ tin cậy của phần mềm và đưa ra giá trị độ đo độ tin cậy.

Cách tiếp cận này được gọi là kiểm thử thống kê. Mục đích của kiểm thử thống kê là đánh giá độ tin cậy của hệ thống. Việc đánh giá cùng với kiểm thử sai sót, có cùng mục đích là tìm ra lỗi của hệ thống. Prowell et Al.(1999) đưa ra một mô tả tốt của kiểm thử thống kê trong sách của ông - Kỹ nghệ phần mềm phòng sạch.

Tuy vậy, cách tiếp cận dựa trên độ đo tính tin cậy không dễ dàng áp dụng trong thực tế. Những khó khăn chủ yếu xuất hiện do:

1. Không chắc chắn mô tả sơ lược hoạt động: Mô tả sơ lược hoạt động dựa trên kinh nghiệm, với các hệ thống khác có thể không phản ánh chính xác thực tế sử dụng của hệ thống.

2. Giá trị cao của sự sinh ra dữ liệu kiểm tra: có thể rất đắt để sinh một lượng lớn dữ liệu yêu cầu trong mô tả sơ lược hoạt động trừ khi quá trình có thể hoàn toàn tự động.

3. Thống kê không chắc chắn khi yêu cầu tính tin cậy cao được chỉ ra: ta phải sinh một số lượng thống kê quan trọng các sai sót để cho phép đo độ tin cậy chính xác. Khi phần mềm đã được xác thực tính tin cậy, một vài sai sót liên quan xuất hiện và nó khó khăn để sinh sai sót mới.

Phát triển mô tả sơ lược “thao tác chính xác chắc chắn” có thể tốt với vài các hệ thống có “mẫu tiêu chuẩn hóa” được sử dụng. Tuy nhiên, với các loại hệ thống khác có rất nhiều người sử dụng khác nhau, mỗi người có một cách riêng khi sử dụng hệ thống. Từ đó, cách tốt nhất để sinh lượng lớn dữ liệu để đáp ứng yêu cầu đo độ tin cậy là sử dụng một hệ sinh dữ liệu kiểm thử mà có thể thiết đặt tự động sinh đầu vào phù hợp với mô tả sơ lược hoạt động. Tuy nhiên, nó thường không thể tự động sinh ra tất cả dữ liệu thử nghiệm cho các hệ thống tương tác bởi vì các đầu vào thường là câu trả lời tới đầu ra hệ thống. Tập dữ liệu cho các hệ thống đó phải được sinh ra bằng tay, do đó chi phi cao hơn. Ngay cả khi điều đó có thể hoàn toàn tự động, viết lệnh cho hệ sinh dữ liệu thử nghiệm có thể tiết kiệm nhiều thời gian.

Thống kê không chắc chắn là một vấn đề chung trong việc đo độ tin cậy của hệ thống. Để tạo nên một dự đoán chính xác độ tin cậy, ta cần phải làm nhiều hơn là chỉ phát hiện ra một lỗi hệ thống đơn lẻ. Phải sinh ra một lượng lớn dữ liệu phù hợp, thống kê số các lỗi để chắc chắn rằng độ tin cậy của bạn là chính xác. Điều này tốt nhất khi ta tìm ra rất ít lỗi trong hệ thống, khó khăn là nó trở thành đo sự hiệu quả của kỹ thuật ít lỗi. Nếu tính tin cậy được xác định ở mức rất cao, nó thường không thực tế để sinh đủ lỗi hệ thống để kiểm tra các đặc tả đó.

2.3. Tính an toàn phần mềm

Các quá trình đảm bảo tính an toàn và thẩm tra tính tin cậy có các mục tiêu khác nhau. Ta có thể xác định số lượng tính tin cậy bằng cách sử dụng một vài độ đo, do đó đo được tính tin cậy của hệ thống. Với các giới hạn của quá trình đo, ta biết các mức yêu cầu của tính tin cậy có thể đạt được hay không. Tuy nhiên, tính an toàn không thể xác định đầy đủ ý nghĩa theo số lượng, và do đó không thể được đo khi kiểm thử hệ thống.

Vì vậy, đảm bảo tính an toàn liên quan tới việc chứng minh mức độ tin cậy của hệ thống, nó có thể thay đổi từ rất thấp đến rất cao. Đây là một chủ đề với tri thức từ các chuyên gia phán đoán dựa trên các dấu hiệu của hệ thống, môi trường và các quá trình phát triển hệ thống. Trong nhiều trường hợp, sự tin cậy này phần nào dựa trên kinh nghiệm tổ chức phát triển hệ thống. Nếu trước đó một công ty đã phát triển nhiều hệ thống điều khiển mà đã hoạt động an toàn, thì đó là điều hợp lý để cho rằng họ sẽ tiếp tục phát triển các hệ thống an toàn. Tuy nhiên, sự đánh giá phải đảm bảo bởi những chứng cớ rõ ràng từ thiết kế hệ thống, các kết quả của hệ thống được kiểm tra và xác nhận và các quá trình phát triển hệ thống đã được sử dụng. Các vấn đề này đi đến kết luận sự tin tưởng của người phát triển về tính an toàn của hệ thống được chứng minh là đúng.

Các quá trình thẩm tra và xác nhận với các hệ thống quan trọng an toàn là phổ biến với các quá trình có thể so sánh được của bất kỳ hệ thống nào với các yêu cầu tính tin cậy cao. Đó phải là quá trình kiểm thử bao quát để phát hiện các khiếm khuyết có thể xảy ra, và tại những

chỗ thích hợp, kiểm thử thống kê có thể được sử dụng để đánh giá tính tin cậy của hệ thống. Tuy nhiên, bởi vì tỷ lệ lỗi cực thấp được yêu cầu trong nhiều hệ thống quan trong an toàn, kiểm thử thống kê không thể luôn cung cấp sự đánh giá về số lượng tính an toàn của hệ thống. Các thử nghiệm cung cấp một vài bằng chứng, mà đã được sử dụng cùng với các bằng chứng khác như các kết quả cua sự xem xét lại và kiểm tra tĩnh, để đưa ra kết luận về tính an toàn của hệ thống.

Sự xem xét lại bao quát là thật sự cần thiết trong lúc bao quát trình phát triển hướng tính an toàn để đưa phần mềm tới được những người mà sẽ xem xét nó từ nhiều khung nhìn khác nhau. Pernas đề xuất 5 loại xem xét lại mà nên được ủy thác với hệ thống quan trọng an toàn.

1. Xem xét lại chính xác chức năng mong đợi.

2. Xem xét lại cấu trúc có thể duy trì được, và có thể hiểu được.

3. Xem xét lại để kiểm tra lại thiết kế thuật toán và cấu trúc dữ liệu là thích hợp với hành vi xác định.

4. Xem xét lại tính chắc chắn của thiết kế mã, thuật toán và cấu trúc dữ liệu.

5. Xem xét lại sự đầy đủ của các trường hợp kiểm thử.

Một sự thừa nhận làm cơ sở của hoạt động về tính an toàn hệ thống là nhiều thiếu sót của hệ thống có thể dẫn tới những rủi ro về tính an toàn quan trọng là ít hơn đáng kể so với tổng số thiếu sót có thể tồn tại trong hệ thống đó. Đảm bảo tính an toàn có thể tập trung vào những lỗi có tiềm năng gây rủi ro. Nếu nó có thể được chứng minh rằng những lỗi đó không thể xuất hiện, hoặc nếu những lỗi đó xuất hiện, sự rủi ro kết hợp sẽ không đưa đến một tai nạn, thì hệ thống là an toàn. Đây là những luận chứng cơ bản về tính an toàn mà ta cần xem xét.

2.3.1. Luận chứng về tính an toàn phần mềm

1. Việc chứng minh một chương đúng đắn đã được đề ra như một kỹ thuật thẩm tra phần mềm khoảng hơn 30 năm trước. Việc chứng minh một chương trình bình thường chắc chắn có thể được xây dựng cho các hệ thống nhỏ. Tuy nhiên, những khó khăn thực tế của việc chứng minh rằng một hệ thống đáp ứng đầy đủ các đặc tả nó là quá lớn, vài tổ chức xem xét việc chứng

minh đúng đắn trở thành một gánh nặng về chi phí. Tuy nhiên, với một số ứng dụng quan trọng, cần phát triển việc chứng minh tính đúng đắn nhằm tăng sự tin tưởng rằng hệ thống đáp ứng các yêu cầu về tính an toàn và tính bảo mật. Đây là trường hợp đặc biệt khi chức năng tính an toàn quan trọng có thể cô lập trong một hệ thống con nhỏ mà có thể được xác định chính xác.

2. Mặc dù, có thể không mang lại lợi nhuận để phát triển việc chứng minh tính đúng đắn cho hầu hết các hệ thống, thỉnh thoảng nó có thể thực hiện được để phát triển những luận chứng đơn giản về tính an toàn để chứng minh chương trình đáp ứng các yêu cầu về tính an toàn. Với một luận chứng về tính an toàn, nó có thể không cần thiết để chứng minh các chức năng của chương trình được xác định. Nó chỉ cần thiết để chứng minh rằng sự thực thi của chương trình không thể dẫn tới một trạng thái không an toàn.

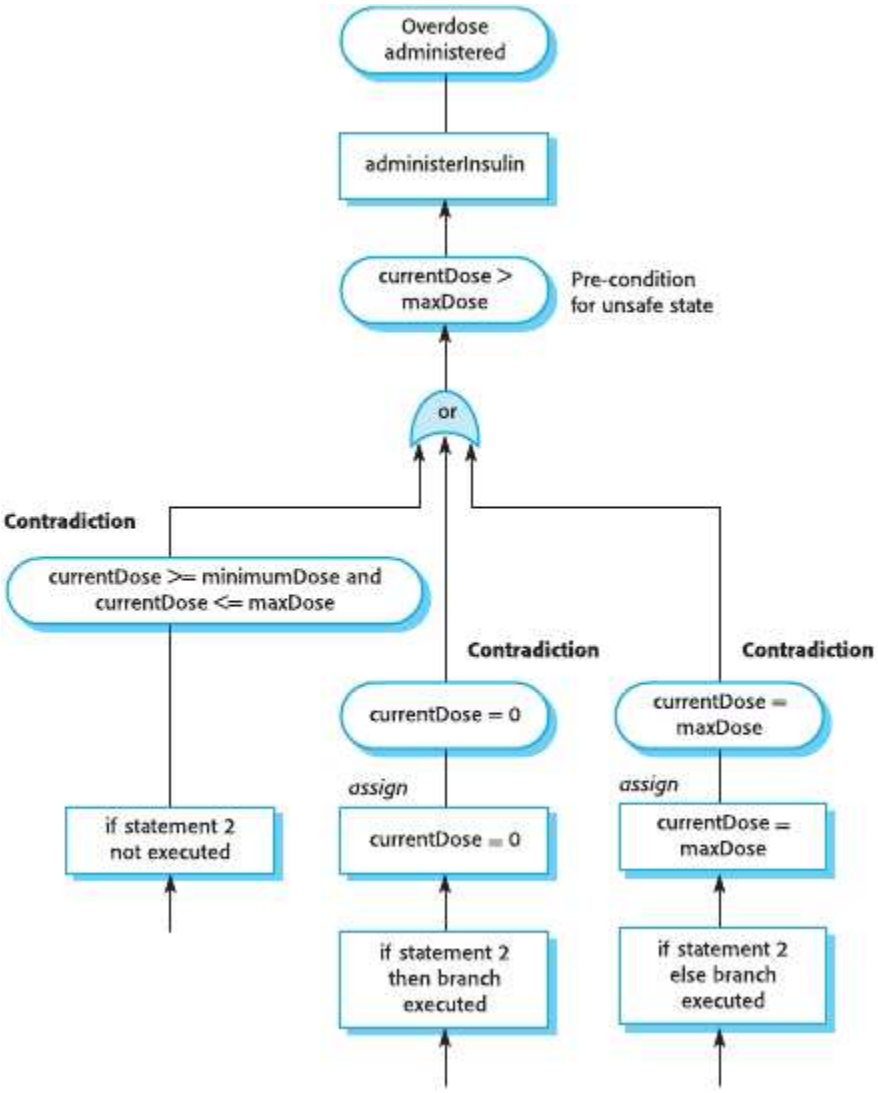
3. Hầu hết các kỹ thuật hiệu quả để chứng minh tính an toàn của hệ thống là chứng minh bằng phản chứng. Ta bắt đầu với giả thiết rằng một trạng thái không an toàn đã được xác định bằng phân tích rủi ro hệ thống, có thể được đi đến khi chương trình thực thi. Ta viết một thuộc tính để xác định đó là trạng thái không an toàn. Sau đó, một cách có hệ thống, ta phân tích mã chương trình và chỉ ra, với tất cả các đường dẫn chương trình dẫn tới trạng thái đó, điều kiện kết thúc của các đường dẫn đó mâu thuẫn với thuộc tính trạng thái không an toàn. Nếu có trường hợp đó, giả thiết ban đầu của trạng thái không an toàn là không đúng. Nếu ta lặp lại điều đó với tất cả định danh rủi ro, thì phần mềm là an toàn.

Ví dụ: Bài toán phân phối thuốc Insulin cho bệnh nhân. Liều lượng Insulin được phân phối là một hàm của mức độ đường trong máu, liều lượng Insulin phân phối lần trước và thời gian phân phối liều thuốc trước. Đoạn mã được cho nư sau, Hãy phát triển một luận chứng cho mã này bao gồm việc chứng minh liều lượng thuốc được quản lý không bao giờ nhiều hơn mức lớn nhất đã được lập cho mỗi bệnh nhân. Do đó, không cần thiết để chứng minh rằng hệ thống phân phối đúng liều lượng thuốc, mà chỉ đơn thuần là nó không bao giò phân phối quá liều lượng cho bệnh nhân.

```
currentDose = computeInsulin();
// Tính an toàn kiểm tra và điều chỉnh currentDose nếu cần thiết
// Câu lệnh if-1
if (previousDose == 0)
        {
        if (currentDose > 16)
            currentDose = 16;
        }
else
        if (currentDose > (previousDose * 2))
            currentDose = previousDose * 2;
// Câu lệnh if-2
if (currentDose < minimumDose)
        currentDose = 2;
else if (currentDose > maxDose)
        currentDose = maxDose;
administerInsulin(currentDose);
```

Để xây dựng những luận chứng về tính an toàn, ta xác định tiền điều kiện cho trạng thái không an toàn, trong trường hợp mày là currentDose > maxDose. Sau đó, chứng minh rằng tất cả các đường dẫn của chương trình đưa đến sự mâu thuẫn của điều khẳng định tính không an toàn đó. Nếu đó là một trường hợp, điều kiện không an toàn không thể là đúng. Do đó, hệ thống là an toàn, đồ thị miêu tả như trên hình dưới đây.

Những luận chứng về tính an toàn, như được chỉ ra ở hình trên. Đầu tiên, ta xác định tất cả các đường dẫn có thể mà đưa tới trạng thái không an toàn tiềm năng. Chúng ta làm việc về phía sau từ trạng thái không an toàn và xem xét kết quả cuối cùng tới tất cả các biến trạng thái trên mỗi đường dẫn dẫn tới nó. Trong ví dụ này, tất cả những gì cần liên quan tới là một tập các giá trị có thể xảy ra của currentDose ngay lập tức trước khi phương thức administerInsulin được thực thi.



Các luận chứng tính an toàn dựa trên sự mô tả những mâu thuẫn.

Trong các luận chứng về tính an toàn chỉ ra ở hình trên, có ba đường dẫn chương trình có thể dẫn tới việc gọi tới phương thức administerInsulin. Chúng tôi muốn chứng minh rằng lượng insulin phân phối không bao giờ vượt quá maxDose. Tất cả các đường dẫn chương trình có thể dẫn tới phương thức administerInsulin đã được xem xét:

1. Không có nhánh nào của câu lệnh if thứ hai được thực thi. Nó chỉ có thể xảy ra nếu một trong hai điều sau xảy ra: currentDose là lớn hơn hoặc bằng minimumDose và nhỏ hơn hoặc bằng maxDose.

2. Nhánh then của câu lệnh if thứ 2 được thực thi. Trong trường hợp đó, việc gán

currentDose bằng 0 được thực thi. Do đó, hậu điều kiện đó là currentDose = 0.

3. Nhánh else-if của câu lệnh if thứ 2 được thực thi. Trong trường hợp đó, việc gán currentDose bằng maxDose được thực thi. Do đó, hậu điều kiện đó là currentDose=maxDose.

=> Trong cả ba trường hợp trên, các hậu điều kiện mâu thuẫn với các tiền điều kiện về tính không an toàn là liều lượng thuốc được phân phối là nhiều hơn maxDose, vì vậy hệ thống là an toàn.

2.3.2. Đảm bảo quy trình

Chúng ta đã thảo luận tầm quan trọng của việc đảm bảo chất lượng của quá trình phát triển hệ thống. Đây là điều quan trọng với tất cả các hệ thống nhưng nó đặc biệt quan trọng với các hệ thống tính an toàn quan trọng. Có 2 lý do:

1. Các rủi ro ít xảy ra trong các hệ thống quan trọng và nó có thể là không thể xảy ra được trong thực tế để mô phỏng chúng trong lúc kiểm thử hệ thống. Bạn không thể dựa trên kiểm thử bao quát để tạo ra các điều kiện có thể dẫn tới một tai nạn.

2. Các yêu cầu về tính an toàn, đôi khi là những yêu cầu "sẽ không" loại trừ hành vi không an toàn của hệ thống. Nó không thể được chứng minh thuyết phục thông qua kiểm thử và các hoạt động thẩm định khác rằng các yều cầu đó đã được thực thi.

Mô hình vòng đời cho quá trình phát triển hệ thống nói lên rằng: nên dành cho tính an toàn trong tất cả các bước của quy trình phần mềm. Điều đó có nghĩa là các hoạt động đảm bảo tính an toàn phải được bao gồm trong quy trình. Nó bao gồm:

1. Việc tạo thành một đoạn rủi ro và giám sát hệ thống lần theo những rủi ro từ phân tích tính rủi ro ban đầu thông qua kiểm thử và thẩm tra hệ thống.

2. Thêm vào dự án các kỹ sư về tính an toàn, những người có trách nhiệm rõ ràng về tính an toàn của hệ thống.

3. Sử dụng tính an toàn bao quát xem xét lại trong toàn bộ quy trình phát triển.

4. Tạo thành sự chứng nhận tính an toàn hệ thống bởi tính an toàn của các thành phần quan trọng về tính an toàn đã chính thức được chứng nhận.

5. Sử dụng hệ thống quản lý cấu hình rất chi tiết, mà đã được sử dụng để theo dõi tất cả các tài liệu về tính an toàn liên quan và giữ nó trong tât cả các bước với tài liệu kỹ thuật liên quan. Chú ý rằng nếu có một lỗi trong cấu hình quản lý có nghĩa là một hệ thống không đúng được phân phối tới khác hàng.

Để minh họa việc đảm bảo tính an toàn, chúng ta sử dụng quá trình phân tích rủi ro mà nó là một phần thiết yếu của quá trình phát triển các hệ thống. Phân tích rủi ro liên quan đến việc xác định các rủi ro, khả năng có thể xảy ra của chúng và khả năng mà các rủi ro đó sẽ dẫn đến tai nạn. Nếu quá trình phát triển bao gồm các dấu hiệu rõ ràng từ nhận dạng rủi ro trong chính hệ thống đó, thì một luận cứ có thể chứng minh được tại sao các rủi ro đó không dẫn đến các tai nạn. Khi sự xác nhận bên ngoài được yêu cầu trước khi hệ thống được sử dụng, nó thường là điều kiện xác nhận rằng các dấu vết này có thể được chứng minh.

Ví dụ: Xác định rủi ro cho hệ thống tiêm quá liều Insulin của hệ thống ở ví dụ trên.

----------Hazard Log------Trang 4: được in ngày 20.02.2003----------
Hệ thống: Hệ thống bơm Insulin File: Insulin/Safety/HazardLog
Kỹ sư đảm bảo: James Brown
Phiên bản Log: 1/3
Xác định rủi ro: Lượng Insulin được phân phối quá liều lượng tới bệnh nhân
Xác định bởi: JaneWilliams
Mức quan trọng: 1
Xác định sự rủi ro: Cao
Xác định cây khiếm khuyết: Có ngày 24.01.99
Vị trí: Hazard Log, trang 5
Người tạo cây khiếm khuyết: Jane Williams và Bill Smith
Kiểm tra cây khiếm khuyết: Ngày 28.01.99
Người kiểm tra: James Brown.

---------Các yêu cầu thiết kế tính an toàn của hệ thống--------

1. Hệ thống sẽ bao gồm phần mềm tự kiểm thử mà sẽ kiểm tra hệ thống cảm biến, đồng hồ và hệ thống phân phối Insulin.

2. Phần mềm tự kiểm tra sẽ được thực thi ít nhất một lần mỗi phút.

3. Khi phần mềm tự kiểm tra phát hiện một sai sót trong bất kỳ một thành phần nào, một cảnh báo sẽ được phát ra và bơm hiển thị sẽ cho biết tên của thành phần mà sai sót đã được phát hiện. Việc phân phối Insulin sẽ bị trì hoãn.

4. Hệ thống sẽ kết hợp với hệ thống ghi đè để cho phép người sử dụng hệ thống sửa đổi liều lượng Insulin đã tính để phân phối bởi hệ thống.

5. Lượng ghi đè nên được giới hạn không lớn hơn một giá trị định trước là một tập mà hệ thống đã được cấu hình bởi các chuyên gia y tế.

Các tài liệu tính an toàn trung tâm là Hazard Log, nơi mà các rủi ro được xác định trong quá trình đặc tả được chứng minh và chỉ ra. Sau đó, Hazard Log được sử dụng tai mỗi giai đoạn trong quá trình phát triển phần mềm để đánh giá rằng giai đoạn phát triển đó đã đưa các rủi ro đó vào bản kê khai. Mẫu tài liệu này chứng minh quá trình phân tích rủi ro và chỉ ra các yêu cầu thiết kế mà đã được sinh ra trong quá trình này. Các yêu cầu thiết kế đó được dự định để đảm bảo rằng hệ thống điều khiển có thể không bao giờ phân phối quá liều lượng insulin tới người dùng. Như chỉ ra ở trên, các cá nhân chịu trách nhiệm về tính an toàn nên được xác định rõ ràng. Các dự án phát triển hệ thống tính an toàn quan trọng nên bổ nhiệm một kỹ sư về tính an toàn, người mà không liên quan trong việc phát triển hệ thống. Trách nhiệm của kỹ sư này là đảm bảo việc kiểm tra tính an toàn thích hợp đã được tạo thực hiện và chứng minh. Hệ thống đó cũng có thể yêu cầu một người thẩm định tính an toàn độc lập được bổ nhiệm từ một tổ chức bên ngoài, người này sẽ báo cáo trực tiếp tới khách hàng các vấn đề về tính an toàn.

Trong một số lĩnh vực, các kỹ sư hệ thống có trách nhiệm về tính an toàn phải được cấp giấy chứng nhận. Các kỹ sư thiếu kinh nghiệm, chất lượng kém có thể không đảm bảo trách

nhiệm về tính an toàn. Hiện nay, điều này không được áp dụng với các kỹ sư phần mềm, mặc dù nó đã được thảo luận rộng rãi về giấy phép của các kỹ sư ở một số bang của nước Mỹ (Knight và Leveson, 2002; Begert, 2002). Tuy nhiên, trong tương lai, các tiêu chuẩn của quá trình phát triển phần mềm tính an toàn quan trọng có thể yêu cầu các kỹ sư về tính an toàn của dự án nên là các kỹ sư đã được cấp giấy chứng nhận chính thức với một cấp độ thấp nhất của quá trình đào tạo.

2.3.3. Kiểm tra tính an toàn khi thực hiện phần mềm

Kỹ thuật tương tự có thể được sử dụng để giám sát động các hệ thống tính an toàn quan trọng. Các mã kiểm tra có thể được thêm vào hệ thống để kiểm tra một ràng buộc về tính an toàn. Nó đưa một ngoại lệ nếu ràng buộc đó bị vi phạm. Các ràng buộc về tính an toàn nên luôn được giữ tại các điểm cụ thể trong một chương trình có thể được biểu thị như các xác nhận. Các xác nhận đó mô tả các điều kiện phải được đảm bảo trước khi các câu lệnh tiếp theo có thể được thực hiện. Trong các hệ thống tính an toàn quan trọng, các xác nhận nên được sinh ra từ các đặc tả tính an toàn. Nó được dự định để đảm bảo hành vi an toàn hơn hành vi theo các đặc tả. Các xác nhận có thể có giá trị đặc biệt trong việc đảm bảo tính an toàn trong giao tiếp giữa các thành phần của hệ thống.

Ví dụ, trong hệ thống phân phối Insulin, liều thuốc của người quản lý Insulin cùng với các tín hiệu được sinh ra tới bơm Insulin để phân phối lượng tăng xác định Insulin. Lượng tăng insulin cùng với liều lượng insulin lớn nhất cho phép có thể được tính toán trước và được tính đến như một xác nhận trong hệ thống. Nếu có một lỗi trong việc tính toán *currentDose*, *currentDose* là một biến trạng thái giữ lượng insulin được phân phối, hoặc nếu giá trị này đã bị sửa đổi theo một cách nào đó, thì nó sẽ bị chặn lại tại bước này. Một liều lượng insulin quá mức sẽ không được phân phối, khi phương thức kiểm tra đảm bảo rằng bơm sẽ không phân phối nhiều hơn *maxDose*.

```
static void administerInsulin() throw SafetyException
{
        int maxIncrements = InsulinPump.maxDose / 8;
        int increments = InsulinPump.currentDose / 8;
```

```
        // xác nhận currentDose <= InsulinPump.maxDose;

        if (InsulinPump.currentDose > InsulinPump.maxDose)
                throw new SatefyException (Pump.doseHigh);
        else
                for (int i = 1; i <= increments; i++)
                {
                        generateSignal();
                        if (i > maxIncrements)
                                throw new SatefyException (Pump.incorrectIncrements);
                } // for loop
} // administerInsulin
```

Từ các xác nhận tính an toàn được bao gồm như các lời chú giải chương trình, viết mã để kiểm tra các xác nhận đó có thể được sinh ra. Câu lệnh if sau các chú giải xác nhận kiểm tra xác nhận đó. Về nguyên tắc, việc sinh các mã này có thể được sinh ra một cách tự động bằng cách sử dụng bộ tiền xử lý xác nhận. Tuy nhiên, các công cụ thường phải được viết riêng biệt và thông thường mã xác nhận được sinh bằng tay.

2.3.4. Một số trường hợp an toàn và tin cậy được của phần mềm

Các trường hợp an toàn và, tổng quát hơn, các trường hợp tin cậy được cấu trúc thành tài liệu, đưa ra các luận chứng và chứng cớ chi tiết để chứng minh hệ thống là an toàn hoặc mức yêu cầu của tính tin cậy được đã đạt được. Với nhiều loại hệ thống quan trọng, đưa ra một trường hợp an toàn là một yêu cầu theo luật định, và trường hợp đó phải thỏa mãn một số chứng nhận chính trước khi hệ thống có thể được triển khai.

Ví dụ: Những người điều chỉnh được tạo ra bởi chính phủ để đảm bảo rằng kỹ nghệ mật không được lợi dụng sự thiếu các tiêu chuẩn quốc gia về tính an toàn, tính bảo mật,… Có nhiều người điều chỉnh trong nhiều lĩnh vực kỹ nghệ khác nhau.

1. Ngành hàng không được điều chỉnh bởi những người trong lĩnh vực hàng không quốc gia.

2. Những người điều chỉnh ngành đường sắt tồn tại để đảm bảo tính an toàn trong ngành đường sắt.

3. Những người điều chỉnh hạt nhân phải chứng nhận tính an toàn của khu vực xử lý hạt nhân trước khi nó có thể hoạt động.

4. Trong lĩnh vực ngân hàng, các ngân hàng nhà nước phụ vụ với vai trò như những người điều chỉnh, thiết lập các thủ tục và các bài tập để giảm khả năng gian lận và bảo vệ khách hàng trước những rủi ro.

Khi các hệ thống phần mềm ngày càng tăng tầm quan trọng trong cơ sở hạ tần của các quốc gia, những người điều chỉnh trở nên liên quan nhiều hơn tới các trường hợp an toàn và tin cậy được của các hệ thống phần mềm.

Thành phần	Mô tả
Mô tả hệ thống	Tổng quan về hệ thống và mô tả các thành phần quan trọng của nó.
Các yêu cầu tính an toàn	Các yêu cầu tính an toàn rút ra từ đặc tả yêu cầu của hệ thống.
Phân tích các rủi ro và các nguy cơ	Các tài liệu mô tả các rủi ro và nguy cơ đã được xác định và các tiêu chuẩn được đưa ra để giảm các
Xác nhận và thẩm định	Mô tả việc sử dụng thủ tục V & V và, chỗ thích hợp, các kiểm thử được thực hiện với hệ thống.
Báo cáo xem xét lại	Các hồ sơ của tất cả thiết kế và tính an toàn được xem xét lại
Nhóm năng lực	Chứng cớ năng lực của tất cả các nhóm cùng với việc phát triển và thẩm tra hệ thống tính an toàn

Quá trình QA	Các hồ sơ của quá trình đảm bảo chất lượng được thực hiện trong khi phát triển hệ thống.
Quá trình quản lý sự thay đổi	Các hồ sơ của tất cả các thay đổi được đưa ra và các hành động thực hiện và nơi thích hợp, sự biện minh Tính an toàn của các thay đổi đó.
Các trường hợp kết hợp tính an toàn	Xem xét tới các trường hợp tính an toàn khác có thể tác động tới các trường hợp an toàn.

Các thành phần của một trường hợp an toàn của phần mềm

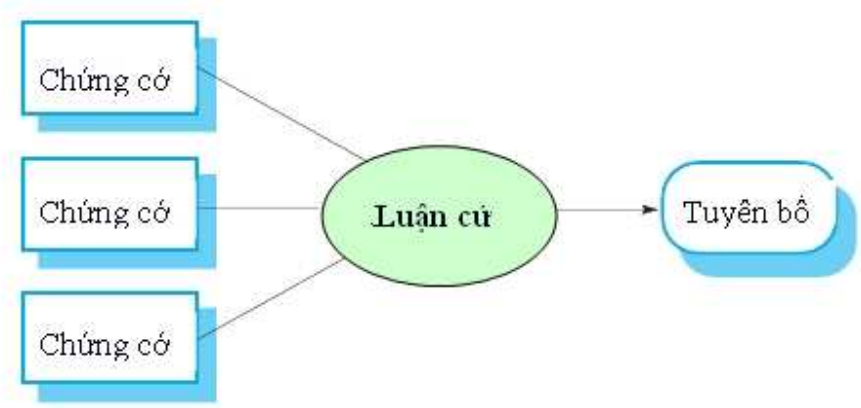
Vai trò của những người điều chỉnh là kiểm tra xem hệ thống đã hoàn thành là an toàn và có thể thực hiện được, vì vậy họ chủ yếu được tập hợp khi một dự án phát triển được hoàn thành. Tuy nhiên, những người điều chỉnh và những người phát triển hiếm khi làm việc độc lập, họ liên lạc với đội phát triển để xác minh những gì phải tính đến trong một trường hợp an toàn. Những người điều chỉnh và những người phát triển cùng nhau thẩm tra các quá trình và các thủ tục để đảm bảo rằng nó đã được ban hành và chứng minh thảo mãn người điều khiển.

Tất nhiên, bản thân phần mềm không nguy hiểm. Nó chỉ nguy hiểm khi nó được nhúng vào trong một hệ thống lớn, hệ thống dựa trên máy tính hoặc hệ thống chuyên môn xã hội mà những sai sót của phần mềm có thể dẫn đến sai sót của các thiết bị khác hoặc của các quá trình mà có thể gây ra tổn hại và dẫn đến cái chết. Vì vậy, một trường hợp an toàn của phần mềm luôn luôn là một phần của trường hợp an toàn hệ thống rộng hơn để chứng minh tính an toàn của toàn bộ hệ thống. Khi xây dựng một trường hợp an toàn của phần mềm, bạn phải liên hệ những sai sót của phần mềm với những sai sót của hệ thống lớn hơn và chứng minh rằng hoặc những sai sót của phần mềm sẽ không xảy ra hoặc nó sẽ không làm lan rộng ra theo cách làm cho các sai sót nguy hiểm của hệ thống có thể xảy ra.

Một trường hợp an toàn là một tập các tài liệu bao gồm mô tả hệ thống đã được chứng nhận, thông tin về các quá trình sử dụng để phát triển hệ thống và các luận chứng hợp logic để chứng minh răng hệ thống là có khả năng an toàn. Bishop và Bloomfield đã đưa ra định nghĩa

ngắn gọn về một trường hợp an toàn như sau: Một tài liệu nhiều bằng chứng cung cấp một luận chứng thuyết phục và hợp lệ rằng hệ thống thỏa mãn tính an toàn với ứng dụng đưa ra trong môi trường đưa ra.

Sự tổ chức và nội dung của một trường hợp an toàn phụ thuộc vào kiểu của hệ thống đã được chứng nhận và ngữ cảnh hoạt động của nó. Thành phần then chốt của một trường hợp an toàn là một tập các luận chứng hợp logíc về tính an toàn của hệ thống. Nó có thể là các luận chứng xác thực (sự kiện X sẽ hoặc sẽ không xảy ra) hoặc các luận chứng khả năng (xác xuất của sự kiện X là 0.Y); khi được kết hợp, nó có thể chứng minh được tính an toàn. Về cơ bản, luận chứng đó giải thích tại sao tuyên bố an toàn có thể được suy ra từ các chứng cớ đã có.



Ví dụ: Các luận chứng dưới đây có thể là một phần rường hợp an toàn của hệ thống bơm Insulin.

Chứng cớ: Luận chứng an toàn cho hệ thống Insulin.
Chứng cớ: Các tập dữ liệu thử nghiệm cho hệ thống Insulin.
Chứng cớ: Báo cáo phân tích tĩnh cho phần mềm bơm Insulin.
Luận cứ: - Luận cứ an toàn chỉ ra liều lượng insulin lớn nhất có thể được tính bằng với axDose.
- Trong 400 thử nghiệm, giá trị của Dose được tính chính xác và không bao giờ vượt quá maxDose.
- Phân tích tĩnh phần mềm điều khiển không xuất hiện dị thường.
- Tất cả đều hợp lý để thừa nhận rằng tuyên bố đã được khẳng định.

=> **Tuyên bố**: Một liều thuốc lớn nhất được tính bởi hệ thống bơm Insulin sẽ không vượt quá maxDose.

Như vậy:

- Kiểm thử thống kê được sử dụng đánh giá tính tin cậy của phần mềm. Nó dựa vào kiểm thử hệ thống với một tập dữ liệu thử nghiệm phản ánh sơ thảo hoạt động của phần mềm. Dữ liệu thử nghiệm có thể được sinh ra tự động.
- Các mô hình phát triển tính tin cậy biểu thị sự thay đổi tính tin cậy khi các thiếu sót được tháo gỡ từ phần mềm trong quá trình kiểm thử. Các mô hình tính tin cậy có thể được sử dụng để dự đoán khi các yêu cầu tính tin cậy sẽ đạt được.
- Chứng minh tính tin cậy là một kỹ thuật hiệu quả đảm bảo tính tin cậy của sản phẩm. Nó chỉ ra các điều kiện có tính rủi ro xác định có thể không bao giờ xuất hiện. Nó thường đơn giản hơn việc chứng minh rằng chương trình đáp ứng đầy đủ đặc tả.
- Điều quan trọng để có một định nghĩa rõ ràng, chứng nhận quá trình phát triển các hệ thống tính tin cậy quan trọng. Quá trình đó phải bao gồm sự xác nhận và giám sát các rủi ro tiềm năng.
- Thẩm tra tính bảo mật có thể được thực hiện bằng cách sử dụng phân tích dựa trên kinh nghiệm, phân tích dựa trên công cụ mà mô phỏng việc tấn công vào hệ thống.
- Các trường hợp tính an toàn tập hợp tất cả các chứng cớ để chứng minh một hệ thống là an toàn. Các trường hợp an toàn đó được yêu cầu khi một bộ điều khiển bên ngoài phải xác nhận hệ thống trước khi nó được sử dụng.