**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỎ - ĐỊA CHẤT**

****

**BÁO CÁO SEMINAR**

**ĐỀ TÀI:**

**TÌM HIỂU KỸ NGHỆ ĐỘ TIN CẬY PHẦN MỀM**

Giảng viên: Ths Đặng Hữu Nghị

Sinh viên:

1. Nguyễn Tuấn Phúc MSSV:1821050134
2. Lý Trung Toàn MSSV: 2021050640

**Hà Nội, tháng 11/2023**

**Nội dung**

I. Giới thiệu đề tài

II. Nội dung

* Xác định tính tin cậy
* Sơ thảo hoạt động
* Dự đoán tính tin cậy
* Đảm bảo tính an toàn
* Những luận chứng về tính an toàn
* Đảm bảo quy trình
* Kiểm tra tính an toàn khi thực hiện
* Các trường hợp an toàn và tin cậy được

III. Tài liệu tham khảo

**I. Giới thiệu**

Rõ ràng việc thẩm định và xác nhận tích hợp của hệ thống quan trọng đã rất phổ biến trong việc xác nhận tính hợp lệ của bất kì hệ thống nào. Quá trình V & V mô tả các đặc tả hệ thống, dịch vụ hệ thống cách cư xử với các yêu cầu của người dùng. Tuy nhiên, với hệ thống quan trọng tính tin cậy được đòi hỏi cao, hơn nữa kiểm thử và phân tích được yêu cầu để đưa ra bằng chứng tỏ hệ thống đáng tin cậy. Có hai lý do tại sao bạn 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 định 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 tọa ra một trường hợp bình thường để cho các khách hàng thấy hệ thống có chứ các yêu cầu về tính tin cậy (tính sẵn sàng, tính tin cậy, tính an toàn và tính bảo mật). Để đánh giá đặc điểm tính tin cậy đòi hỏi sự hoạt động cụ thể V & V (được thảo luận ở phần sau trong chương này). Trong một vài trường hợp, người kiếm soát bên ngoài như người có thầm quyền trong ngành hàng không quốc gia có thể phải chứng nhận rằng hệ thống là an toàn trước khi nó có thể cất cánh. Đề đạt được chứng nhận này, bạn phải thiết kế và thực thi thủ tục đặc biệt V & V nhằm tập hợp chứng cứ vé tỉnh an toàn của hệ thống.

Tính toán tính tin cậy

Áp dụng các kiểm thử vào hệ thống

Chuẩn bị tập dữ liệu kiểm thử

Xác định sơ lược hoạt động

**Hình 6.1. Quá trình đo tính tin cậy**

Vì rất nhiều lý do, giá trị của V & V với hệ thống quan trọng luôn luôn cao hơn so với các loại hệ thống khác. Thông thường V & V chiếm hơn 50% tổng giá trị phát triển của hệ thống phần mềm quan trọng. Tất nhiên, đây là giá đã được điều chỉnh, khi thất bại của hệ thống đã được ngăn ngừa. Ví dụ, năm 1996, một hệ thống phần mềm quan trọng trên tên lửa Ariane 5 bị hỏng và một vài vệ tinh đã bị phá hủy, gây thiệt hại hàng trăm triệu đôla. Sau đó, những người có trách nhiệm đã khám phá ra rằng sự thiếu hụt trong hệ thống V & V có phần nào trách nhiệm trong việc này.

Mặc dù, quá trình xác nhận tính hợp lệ của hệ thống quan trọng phần lớn tập trung vào việc xác nhận tính hợp lệ của hệ thống, các hoạt động liên quan nên kiểm tra để xác nhận quá trình phát triển hệ thống đã được thực hiện. Tóm lại, quá trình tốt dẫn đến hệ thống tốt. Vì vậy, để cung cấp hệ thống tin cậy, bạn cần phải tin tưởng răng quá trình phát triển hợp lý đã được tiến hành.

**II. Xác nhận tính tin cậy**

Số lượng độ đo đã được phát triển để chỉ ra yêu cầu tin cậy của một hệ thống. Đề xác nhận hệ thống có các yêu cầu nào, chúng ta phải đo độ tin cậy của hệ thống như một người dùng hệ thống thông thường.

Quá trình đo độ tin cậy của hệ thống được minh họa trong hình 6.1. Quá trình này gồm 4 bước:

1. Đầu tiên, chúng ta 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, chúng ta thiết đặt tập các dữ liệu kiểm thử để phần ánh mô tả sơ lược hoạt động. Có nghĩa là chúng ta tạo ra dữ liệu kiểm thử với phân bố xác suất như nhau (như dữ liệu cho hệ thống mà bạn đã nghiên cứu). Thông thưởng, bạn sử dụng máy sinh dữ liệu kiểm thử để kiểm tra quá trình này.

3. Chúng ta kiểm thử hệ thống sử dụng các dữ liệu đã được sinh ở trên và đếm số lượng cùng các loại lỗi xảy ra. Số lần lỗi cũng được ghi nhận.

4. Cuối cùng, bạn tiến hành thống kê các lỗi quan trọng, bạn có thể tính toán độ tin cây của phần mềm và đưa ra giá trị độ do dộ 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.

Cách tiếp cận dựa trên độ đo tính tin cậy không đễ dàng áp dụng trong thực tế. Những khó khăn chủ yếu xuất hiện do:

4.1 Không chắc chẵn 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.

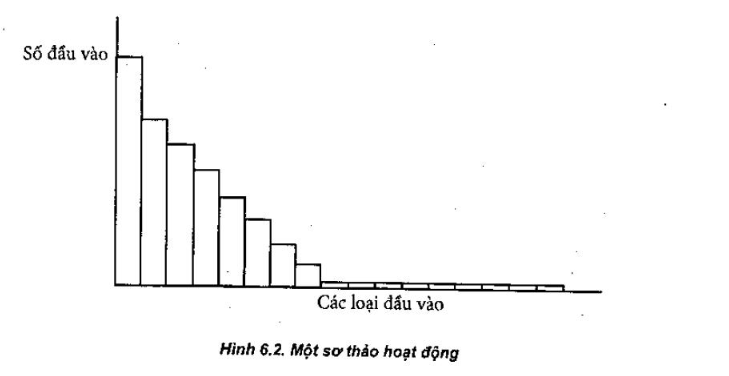
Giá trị cao của sự sinh ra dữ liệu kiểm tra: Có thể rất đắt để sinh một lượng lớn dữ liệu yêu cầu trong mô tả sơ lược hoạt động, trừ khi quá trình có thể hoàn toàn tự động.

4.3. Thống kê không chắc chẵn khi yêu cầu tính tin cậy cao được chi ra: Bạn 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 chân có thể với vài kiểu hệ thống, như hệ thống truyền thông có một mẫu tiêu chuẩn hóa được sử dụng. Tuy sái cức looi bộ 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 để chung trong việc đo độ tin cậy của hệ thống. Để tạo nên một dự đoán chính xác độ tin cậy, chúng ta cần phải làm nhiều hơn là chỉ phát hiện ra một lỗi hệ thống đơn lẻ. Chúng ta phải sinh ra một lượng lớn dữ liệu phù hợp, thống kê số các lỗi để chắc chắn rằng độ tin cậy của chúng ta là chính xác. Điều này tốt nhất khi bạn tìm ra rất ít lỗi trong hệ thống, khó khăn là nó trở thành thước đo sự hiệu quả của kỹ thuật it 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.



**III. Sơ thảo hoạt động**

**Sơ thảo hoạt động của phần mềm phản ánh cách phần mềm sẽ được sử dụng trong**

**thực tế. Sơ thảo 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 mềm 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. Ví dụ, một sơ thảo hoạt động có thể được xác định cho các hệ thống chuyển mạch viên thông bởi vì các công ty viễn thông biết tất cả các mẫu cuộc gọi mà các hệ thống đó phải điều khiển.**

**Musa (Musa, 1993; Musa, 1998) để xuất các nguyên tắc để phát triển các sơ thảo**

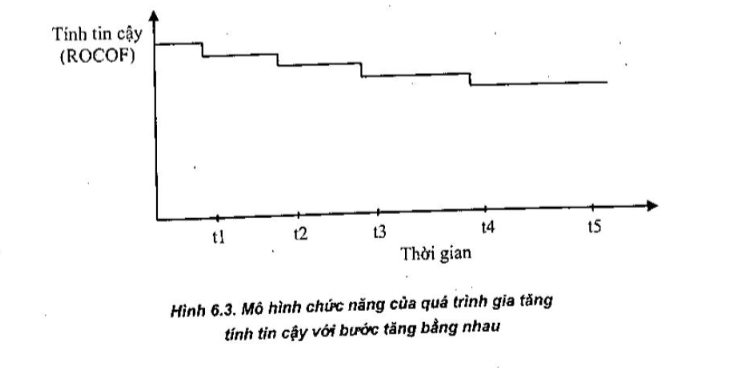
**hoạt động. Ông là một kỹ sư hệ thống viễn thông và ông đã có thời gian dài làm việc tập hợp dữ liệu người dùng trong lĩnh vực này. Do đó, ông đã rút ra kết luận: quá trình • phát triển sơ thảo hoạt động tương đối dễ làm. Với một hệ thống yêu cầu công sức phát triển của khoảng 15 người làm việc trong một năm, sơ thảo hoạt động đã được phát 108 triển trong khoảng 1 người/tháng. Trong các trường hợp khác, hệ sinh sơ thảo hoạt động cần nhiều thời gian hơn (2-3 người/năm), nhưng chi phí đã trải rộng ra hệ thống phát hành. Musa tính rằng công ty của ông (một công ty viễn thông) có ít nhất 10 nhóm để đầu tư vào việc phát triển sơ thảo hoạt động I1].**

**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 sơ thảo 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 để trở nên phức tạp hơn bởi vì các sơ thảo 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 sơ thảo hoạt động tin cậy. Nếu bạn không chắc chẵn rằng sơ thảo 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.**



**IV. Dự đoán tính tin cậy**

**Trong lúc thầm định phần mềm, người quản lý phải phân công quá trình kiểm thử**

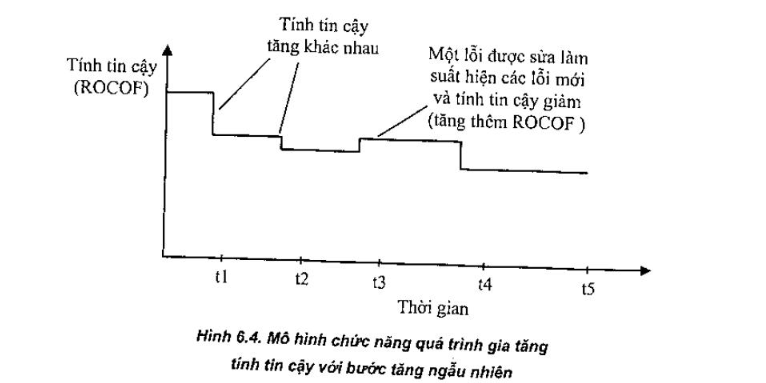
**hệ thống. Vì quá trình kiểm thử phần mềm rất tốn kém, nên nó sẽ được dừng ngay khi có thể và không "kiềm thử quá" hệ thống. Kiểm thử có thể dừng khi mức độ yêu cầu tính tin cậy của hệ thống đã được thực hiện. Tất nhiên, thỉnh thoảng, các dự đoán tính tin cậy có thể cho thấy mức độ yêu cầu tính tính 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 tà 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 diễn ra 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, vi vậy tính tin cậy của hệ thống có thể được cải thiện trong lúc kiểm thử và gỡ lỗi hệ thống. Để dự đoán tính tin cậy, mô hình quá trình gia tăng tính tin cậy nhận thức phải hiểu là mô hình toán học. Không đi vào các mức cụ thể của vấn để này, đơn giản chỉ thảo luận các nguyên tắc của quá trình gia tăng tính tin cậy.**

**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: hấu 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 (hình 6.5). Sau đó, sự tăng thêm nhỏ dần di và tiến tới trạng thái ổn định khi rất ít khiếm khuyết được phát hiện và loại bò trong lẫn kiểm thử cuối cùng.**



**Mô hình đơn giản nhất minh họa khải 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 lõi (hoặc một tập lỗi) được phát hiện và sửa chữa (hình 6.3) và một phiên bản mới của phần mềm dược tạo ra. Mô hình này giả sử rắng sự sửa chữa phần mềm luôn được thực hiện chính xác, vì vậy số lỗi và khiếm khuyết liên hợp của phần mềm giảm trong mỗi phiên bản mới của hệ thống. Khi sự sửa chữa được tạo ra, tỷ lệ xuất hiện lỗi của phần mếm (ROCOF) có thể giảm, như trên hình 6.3. 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 [2].**

**Tuy nhiên, trong thực tế, các lõi phần mềm không phải lúc nào cũng được sửa**

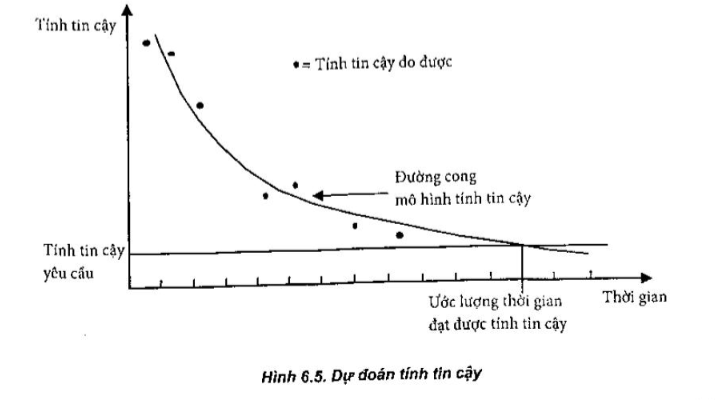
**trong lúc gỡ lỗi, khi bạn thay đổi một chương trình, thỉnh thoảng bạn đưa các lỗi mới vào chương trình đó. Khả năng xuất hiện các lỗi đó có thể cao hơn khả năng xuất hiện các lỗi đã được sửa chữa. Do đó, thỉnh thoảng tính tin dậy của hệ thống có thể trở nên tối hơn trong phiên bản mới.**

**Mô hình quá trình gia tăng tính tin cậy đơn giản bước bằng nhau cũng giả sử tât**

**cả các lỗi đóng góp như nhau vào tính tin cây 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 lồi 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 tim 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.**

**Cuối cùng, các mô hình như để xuất của Littlewood và Verrall dưa ra các vấn để**

**bằng cách đưa một thành phần ngẫu nhiên vào quá trình gia tăng tính tin cậy nhằm cải thiện tác động của một sửa chữa trong phần mềm. Do đó, mỗi sửa chữa không dẫn đến cùng một lượng tăng tính tin cậy bằng nhau trong phần mềm, các biến đổi phụ thuộc vào tính ngẫu nhiên (hình 6.4).**



**Mô hình của Littewood và Verrall cho phép phủ nhận quá trình gia tăng tính tin**

**cậy khi sự sửa chữa đưa vào nhiều lỗi hơn. Đó cũng là mô hình khi một lỗi được sửa chữa, tính tin cậy được cải thiện trung bình bởi mỗi sửa chữa giảm. Lí do là hầu hết các lôi có khả năng xảy ra có thể đã được phát hiện sớm trong quá trình kiếm thử. Sửa chữa các lỗi đó đóng góp phần lớn vào quá trình gia tăng tính tin cậy (2).**

**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 kiếm thử, nó nên có tỳ 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 nhận được từ các lĩnh vực ứng dụng khác nhau, đã được để xuất và so sánh.**

**Đơn giản, bạn có thể dự đoán tính tin cậy bằng cách kết hợp dữ liệu đo tính tin**

**cậy và mô hình nhận biết tính tin cậy. Sau đó, bạn ngoại suy mô hình đó với các mức yêu cầu tính tin cậy và quan sát khi một mức yêu cầu tính tin cậy sẽ đạt được (hình 6.5). Do đó, kiểm thử và gỡ lỗi phải được thực hiện liên tục cho đến khi thoa mãn các yêu cầu.**

**Dự đoán tính tin cây của hệ thống từ mô hinh 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, bạn có thể dự đoán khi nào quá trình kiểm thử được**

**hoàn thành. Nếu điều đó kết thức sau ngày dự kiến phát hành hệ thống, thì bạn phải triển khai bổ sung tài nguyên cho việc kiểm thử và gỡ lôi 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, bạn sẽ phải đàm phán lại với khách hàng về các yêu cầu về tính tin cậy của hệ thống.**

**V. Đảm bảo tính an toàn**

**Các quá trình đảm bảo tính an toàn và thầm định tính tin cậy có mục tiêu khác**

**nhau. Bạn 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, bạn 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, để 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 để với 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 đựa trên kinh nướng điếu khến mà iển tự đốg ản vắnh ht đ đt c nợp y đ chó ring hiếu hệp 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 V & V và các quá trình phát triển hệ thống đã được sử dụng. Với một số hệ thống, các chứng cứ rõ ràng được thu thập trong một hộp an toàn (xem phần 6.4), nó cho phép một người điếu chỉnh bên ngoài 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 V & V với các hệ thống quan trọng an toàn là phổ biến với các quá**

**trình có thế 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 lôi cực thấp được yêu cầu trong nhiều hệ thống quan trọng 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 của 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 cần thiết trong lúc quát trình phát triển hướng tính an**

**toàn để trưng bày phần mềm tới những người sẽ xem xét nó từ nhiều khung nhìn khác nhau. Có 5 loại xem xét lại nên được ủy thác với hệ thống quan trọng.**

**- Xem xét lại chính xác chức năng mong đợi.**

**- Xem xét lại cấu trúc có thể duy trì được và có thể hiểu được.**

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

**- Xem xét lại tính chắc chắn của thiết kể mã, thuật toán và cấu trúc dữ liệu.**

**- Xem xét lại sự đấy đủ của các trường hợp kiểm thử.**

**Một sự thửa 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 rùi 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 tiêm 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 tôi sẽ thảo luận trong phần tới.**

**VI. Những luận chứng về tính an toàn**

**Việc chứng minh một chương trình đúng đẫn, như thảo luận trong chương trước,**

**đã được để ra như một kỹ thuật thẩm định phần mềm khoảng hơn 30 năm trước. Việc chứng minh một chương trình bình thường chắc chẵn 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 là quá lớn và vài tổ chức xem xét việc chứng minh đúng đắn trở thành một chi phí. Tuy nhiên, với một số ứng dụng quan trọng, nó có thể rất kinh tế để đẩy mạnh việc chứng minh tính đúng dẫn 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 thê cô lập trong một hệ thống con mà có thể được xác định chính xác.**

**Mặc dù, nó có thể không mang lại lợi nhuận để phát triển việc chứng minh tính**

**đúng đẫn cho hầu hết các hệ thống, thỉnh thoảng nó vẫn có thể thực hiện được để phát triển những luận chứng đơn giản về tính an toàn nhằm chứng minh chương trình đáp ứng các yêu cầu về tính an toàn. Với một luận chứng về tính an toàn, nó có thế 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 trinh không thể dẫn tới một trạng thái không an toàn.**

**Hầu hết các kỹ thuật hiệu quả để chứng minh tính an toàn của hệ thống là chứng**

**minh bằng phản chứng. Bạn bắt đấu với giả thiết rằng một trạng thái không an toàn đã được xác định bằng phân tích rùi ro hệ thống, có thể được đi đến khi chương trình thực thì. Bạn viết một thuộc tính để xác định đó là trạng thái không an toàn. Sau đó, một cách có hệ thống, bạn 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 đấu của trạng thái không an toàn là không đúng. Nếu bạn lặp lại điều đó với tất cả định danh rủi ro, thì phần mềm là an toàn.**

**- Liếu lượng Insulin được phân phối là một hàm của mức độ đường trong máu, liêu lượng Insulin phân phối lần trước và thời gian phân phối liều thuốc trước.**

**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;**

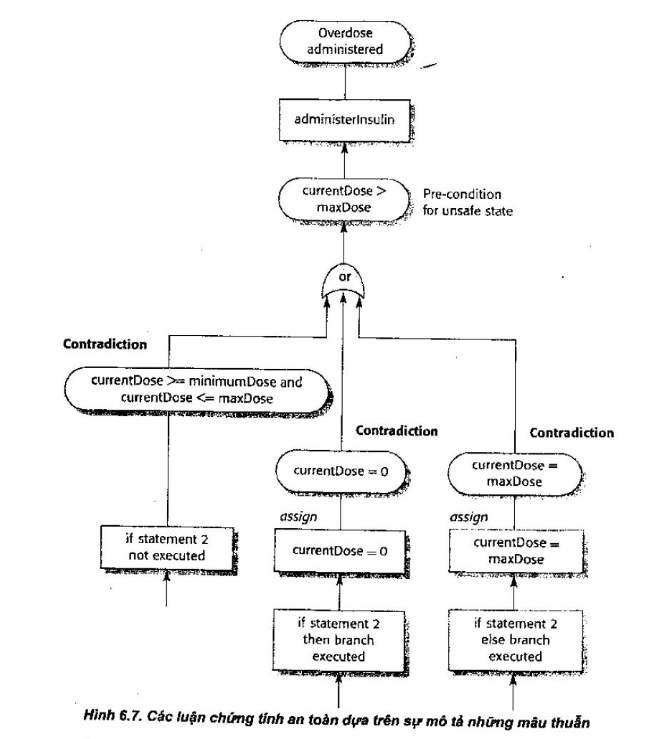
**administerInsulin(currentDose);**

**Hình 6.6. Mã phân phổi Insulin**

**Một ví dụ, xem xét mã trên hình 6.6, nó có thể là một phần thực thi của hệ thống**

**phân phối insulin. Phát triển một luận chứng cho mã này bao gồm việc chứng minh liêu 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 đế chứng minh rằng hệ thống phân phối đúng liếu lượng thuốc, mà chỉ đơn thuần là nó không bao giờ phân phối quá liều lượng cho bệnh nhân.**

**Để xây dựng những luận chứng về tính an toàn, bạn xác định tiến điều kiện cho trạng thái không an toàn, trong trường hợp đó như là currentDose > maxDose. Sau đó, bạn chứng mình rằng tất cả các đường dẫn 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. Bạn có thể lập cấu trúc và đưa ra những luận chứng về tính an toàn bằng đồ thị như trên hình 6.7.**



**Những luận chứng về tính an toàn, như được chỉ ra trên hình 6.7, là ngắn hơn**

**nhiều so với việc thẩm tra hệ thống theo đúng trình tự. Đấu tiên, bạn xác dịnh tất cả các đường dẫn có thể mà đưa tới trạng thái không an toàn tiêm năng. Bạn 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. Bạn có thể bỏ qua các tính toán trước đó (như câu lệnh if thứ nhất trong hình 6.7) trong các luận chứng tính an toàn. Trong ví dụ này, tất cả những gi bạn 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.**

**Trong các luận chứng về tính an toàn chỉ ra trên hình 6.7, có ba đường dẫn**

**chương trinh 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 minimum Dose và nhỏ hơn hoặc bằng maxDose.**

**2. Nhánh then của câu lệnh if thứ hai được thực thì. 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 hai được thực thì. 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 vẽ tính không an toàn là liếu lượng thuốc được phân phối nhiều hơn maxDose, vì vậy hệ thống là an toàn (2).**

**VII. Đảm bảo quy trình**

**Chúng ta đã thảo luận tẩm quan trọng của việc dảm bảo chất lượng của quá trình**

**phát triển hệ thống trong phần giới thiệu chương này. Đây là điều quan trọng với tất cả các hệ thống quan trọng nhưng nó đặc biệt quan trọng với các hệ thống tính an toàn quan trọng. Có hai lý do cho điều này:**

**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, dôi khi là những yêu cầu "sẽ không" (shall not)**

**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 yêu cấu đó đã được thực thi.**

**Mô hình vòng đời cho quá trình phát triển hệ thống tính an toàn quan trọng làm cho điều đó trở nên rõ ràng răng sự chú ý 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 định hệ thống.**

**2. Bổ nhiệm 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 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 từng bước với tài liệu kỹ thuật liên quan. Có một điểm nhỏ trong thủ tục thẩm dịnh nghiêm ngặt là nếu xuất hiện một lỗi trong cấu hình quản lý có nghĩa là một hệ thống không đúng được phân phối tới khách hàng.**

**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 Brownt Phiên bản Log: 1/3**

**Xác định rủi ro: Lượng Insulin được phân phối quá liêu lượng tới bệnh nhân**

**Xác định bởi: JaneWilliams**

**Mức quan trọng: 1**

**Xác định sự rúi 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 thứ mà sẽ kiểm tra hệ thống cảm biến, đồng hô 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 liêu 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.**

***Hình 6.8. Một trang đơn giản Hazard log [2]***

**Để minh họa việc đảm bảo tính an toàn, chúng ta đã sử dụng quá trình phân tích**

**rũi ro mà nó là một phần thiết yếu của quá trình phát triển các hệ thống tính tin cậy quan trọng. Phân tích rủi ro liên quan đến việc xác định các rùi 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. Điều này có thể được bổ sung vào các luận cứ về tính an toàn, như đã thảo luận trong phần 6.2.1. Khi sự xác nhận bên ngoài được yêu cầu trước khi hệ thống được sử dụng (ví dụ, một máy bay), nó thường là điều kiện xác nhận rằng các đấu vết này có thể được chứng minh.**

**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 tại mỗi giai đoạn trong quá trình phát triển phần mềm dể đánh giá rằng giai đoạn phát triển đó đã đưa các rúi ro đó vào bản kê khai. Một ví dụ đơn giản của một Hazard log đầu vào cho hệ thống phân phối insulin được chỉ ra trên hình 6.8. Mẫu tài liệu này chứng minh quá trình phân tích rủi ro và chỉ ra các yêu cầu thiết kế 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 đã chỉ ra trên hình 6.8, 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 bố nhiệm một kỹ sư về tính an toàn, người mà không liên quan trong việc phát triển hệ thống. Trách nhiệm của kỳ 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 thầm 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 để 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. Ở Anh, điều này có nghĩa là họ phải được thừa nhận như là một thành viên của một viện kỹ nghệ (về điện, cơ khí,...) và phải là các kỹ sư có đủ tư cách hành nghê. 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, trồng 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).**

**VIII. Kiểm tra tính an toàn khi thực hiện**

**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 vẽ tính an toàn nên luôn được giữ tại các diểm 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. Vi 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 (hình 6,9). Lượng tăng insulin cùng với liêu 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 liếu lượng insulin quá mức sẽ không được phân phối, khi phương thức kiểm tra dảm bảo răng bơm sẽ không phân phối nhiều hơn max Dose.**

**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);**

**eise**

**for (int i = 1; i <= increments; it+)**

{

**generateSignal():**

**if (i > maxincrements)**

**throw new SatefyException (Pump.incorrectincrements);**

**}// for loop**

**}// administerInsulin**

***Hình 6.9. Quản lý Insulin bằng cách kiểm tra lúc thực thi***

**Từ các xác nhận tính an toàn bao gồm các lời chú giải chương trình, viết mã để**

**kiểm tra các xác nhận đó có thể được sinh ra. Bạn có thể xem hình 6.9, 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.**

**IX. Các trường hợp an toàn và tin cậy được**

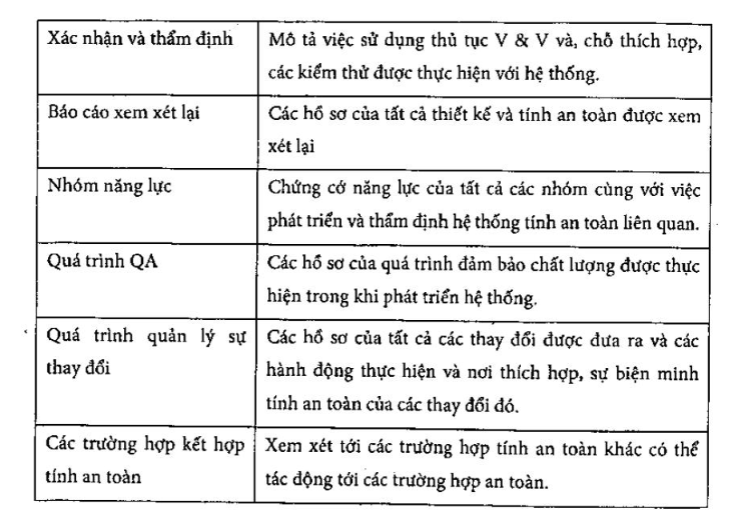
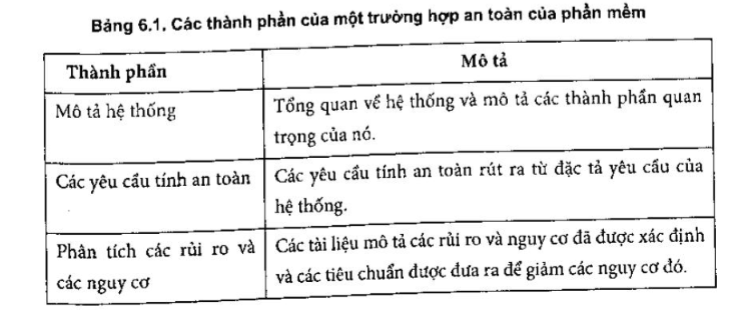
**Các trường hợp an toàn và, tổng quát hơn, các trường hợp tin cậy được cấu trúc**

**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.**

**Những người điều chỉnh được tạo ra bởi chính phù 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. Ví dụ, ngành hàng không được điều chỉnh bởi những người trong lĩnh vực hàng không quốc gia như FAA (tại Mỹ) và CAA (tại Anh). Những người điều chỉnh ngành đường sắt tốn tại để đảm bảo tính an toàn trong ngành đường sắt, 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. 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 tầng của các quốc gia, những người điểu chỉnh trở nên liên quan nhiều hơn tới các trường hợp an toàn và tin cậy được của các hệ thống phần mềm.**

***Bảng 6.1, Các thành phần của một trưởng 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 dến trong một trường hợp an toàn. Những người điều chinh và những người phát triển cùng nhau thầm tra các quá trình và các thủ tục để đảm bảo rằng nó đã được ban hành và chứng minh thỏa 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 tồn hại và 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 rằng hệ thống 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**

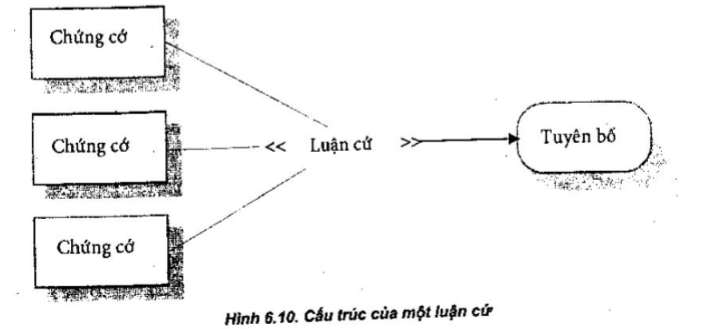
**rằng hệ thống thỏa mãn tính an toàn với ứng dụng đưa ra trong môi trường đứa ra.**

**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. Bảng 6.1 chỉ ra một tổ chức có thể xảy ra với một trường hợp an toàn của phần mềm.**

**Thành phần then chốt của một trường hợp an toàn là một tập các luận chứng hợp**

**logic về tính an toàn của hệ thống. Nó có thể là các luận chứng xác thực (sự kiện X sẽ xảy ra hoặc sẽ không xảy ra) hoặc các luận chứng khả năng (xác xuất của sự kiện X là O.Y); khi được kết hợp, nó có thể chứng minh được tính an toàn. Như trên hình 6.10, một luận chứng là một mối liên hệ giữa cái gì được nghĩ là một trường hợp (một tuyên bố) và khung chứng cớ đã được thu thập. Về cơ bản, luận chứng đó giải thích tại sao tuyên bố đó (nói chung điều gì đó là an toàn) có thể được suy ra từ các chứng cớ đó. Tất nhiên, đưa ra nhiều mức tự nhiên của các hệ thống, các tuyên bố được tổ chức trong một hệ thống phân cấp. Để chứng minh rằng một tuyên bố mức cao là hợp lệ, đầu tiên bạn phải thực hiện với các luận chứng mức thấp hơn. Hình 6.11 chỉ ra một phần của hệ thống phân cấp tuyên bố phân phối kim tiêm Insulin.**



**Như một thiết bị y tế, hệ thống bơm Insulin có thể phải được chứng nhận bên**

**ngoài. Ví dụ, ở Anh, MDD phải đưa ra một chứng nhận an toàn với bất kỳ thiết bị y tế**

**nào được bán tại Anh. Nhiều luận chứng khác nhau có thể phải đưa ra để chứng minh tính an toàn của hệ thống đó. Ví dụ, các luận chứng dưới đây có thể là một phần trường hợp an toàn của hệ thống bơm Insulin.**

**Tuyên bố: Một liếu thuốc lớn nhất được tính bởi hệ thống bơm Insulin sẽ không**

**vượt quá maxDose.**

**Chứng cớ: Luận chứng an toàn cho hệ thống Insulin (hình 6.7).**

**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 liếu lượng insulin lớn nhất có thể được tính bằng**

**maxDose.**

**Trong 400 thử nghiệm, giá trị của Dose được tính chính xác và không bao giờ**

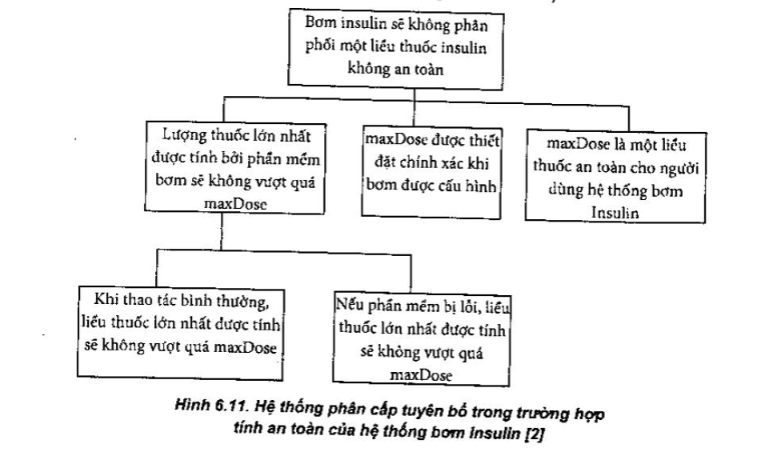
**vượt quá max Dose.**

**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.**

**Tất nhiên, đây là một luận chứng rất đơn giản và trong một trưởng hợp an toàn**

**thực tế cụ thể tham khảo đến các chứng cớ sẽ được đưa ra. Bởi vì, chi tiết tự nhiên của nó, do đó, các trường hợp an toàn là các tài liệu rất dài và phức tạp. Các công cụ phần mềm khác nhau có khả năng giúp xây dựng chúng và tôi đã bao gồm các liên kết tới các công cụ đó trong các trang web được đưa ra trong cuốn sách này.**



**Những vấn để trọng tâm**

**- 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 doán khi các yêu cầu tính tin cậy dạt 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 đẩy đủ đặc tả.**

**- Điều quan trọng là, để có một định nghĩa rõ ràng, phải 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 định tính bảo mật có thể được thực hiện bằng cách sử dụng phân tích dựa**

**trên kinh nghiệm, phân tích dựa trên công cụ hoặc "đội hồ" (tiger teams) 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.**

**"Best practices in code inspection for safety-critical software". Bài báo giới thiệu**

**một bản liệt kê các mục cần kiểm tra của các nguyên tắc để kiểm tra và xem xét phần mềm tính an toàn quan trọng. (J.R. de Almeida, IEEE Software, 20(3), May/June 2003).**

**"Statically scanning Java code: Finding security vulnerabilities". Đây là một bài**

**báo hay về vấn để ngăn ngừa việc tấn công tính bảo mật. Nó thảo luận cách các tấn công đó được thực hiện và cách phát hiện chúng bằng một phân tích tĩnh. J. Viega, JEEE Software, 17(5), September/Octorber 2000).**

**"Software Reliability Engineering: More Relable Software, Faster Development**

**and Testing". Đây chắc chấn là điều rõ ràng về việc sử dụng các sơ thảo thao tác và mô hình tính tin cậy để đánh giá tính tin cây. Nó bao gồm kinh nghiệm chi tiết về kiểm thử thống kê. (J.D. Musa, 1998, McGraw-Hill).**

**"Safety-critical Computer System". Đây là một cuốn sách giáo khoa rất hay bao**

**gồm một số chương hay về vị trí của những phương thức hình thức trong quá trình phát triển phần mềm tính tin cậy quan trọng. (N. Storcy, 1996, Addison-Wesley).**

**"Safeware: System Safety and Computers". Cuốn sách này bao gồm một số**

**chương về việc thầm định các hệ thống tính an toàn quan trọng với rất nhiều chi tiết hơn những vấn để tôi đã đưa ra ở đây về việc sử dụng những luận chứng vẽ tính an toàn dựa trên cây thiếu sót. (N. Leveson, 1995, Addison-Wesley)[2).**

**- THE END -**