CHƯƠNG 3: YÊU CẦU NÂNG CAO

Những phần sẽ nói trong chương này

* Actor hệ thống
* Use-case hệ thống
* Làm cách nào để tìm được hệ thống use case trong sơ đồ nghiệp vụ
* Yêu cầu
* Sơ đồ khái quát sơ bộ

3.1 Giới thiệu về những yêu cầu nâng cao

Một khi mà nghiệp vụ đã được hiểu và sơ đồ hoá, phần phân tích yêu cầu nghiệp vụ nên được bắt đầu ngay. Mặc dù đã có rất nhiều cách tiếp cận để giới thiệu tổng quan về nghiệpv ụ, sách này vẫn đề cập đến, dựa trên use-case

Use case hệ thống là một tiến trình hệ thống độc lập được đánh giá dựa vào hoạt động của business. Use case hệ thống nâng cao giới thiệu một cách tổng quan phương thức yêu cầu của một hệ thống. Những lời giải thích ở những use cases đó đại diện cho những tính năng không cần phải thực thi, đó là những hạn chế và chất lượng của những chức năng đó. Yêu cầu bổ sung là một yêu cầu không cần thực thi được ứng dụng cho toàn hệ thống, chứ không chỉ những chức năng riêng rẽ. Use-case hệ thống thì hữu ích cho rất nhiều hành động liên quan đến hệ thống development, ví dụ như

* Cách khai báo và kiếm chứng kiến trúc hệ thống. Nói chung, lớp, mối quan hệ và thuộc tính là 1 trong những phần của kiến trúc hệ thống và thu được từ chuỗi
* Tạo ra test-case: Use case có thể được xem là phần cơ bản nhát của hệ thống và là những bài test được chấp nhận (Những bài test được chấp nhận đến từ quan điểm từ user)
* Lập kế hoạch định kì: Mỗi usecase sẽ có độ ưu tiên của nó, và ảnh hưởng đến việc lập trình cũng như quyết định thời gian hoàn thành, như thế thì đôi ngũ phát triển mới có thể đánh giá được làm usecase nào cho từng thời gian định kì được
* Căn cứ tài liệu: Usecase được mô tả là 1 hoạt động bình thường của một hệ thống, cũng như là sự thay đổi hệ thống đại diện cho việc làm cách nào thương thảo với những thứ không được đề cập đến. Những sự miêu tả này là căn cứ chính xác cho việc nói chuyện với khách hàng, vì chúng đã được cấu trúc một cách hoàn chỉnh

Mỗi use case sẽ được đại diện cho một tâp hợp những yêu cầu hệ thống một cách mạch lạc. Thường thì nhiều chức năng thì liên quan đến 1 usecase, nhất là khi nó là một usecase rất phức tạp. Một số chức năng, nói cách khác nó được kết hợp với không chỉ một usecase. Trong một vài trường hợp, nó có thể xảy ra việc một chức năng dựa vào duy nhất một usecase và ngược lại. Chuyện này thường xảy ra trong trường hợp usecase rất đơn giản ví dụ như là báo cáo hoặc quản lý thực thể

Có ít nhất 3 cách tiếp cận cho yêu cầu và use cases

* Tạo ra dánh sách những yêu cầu chức năng, sau đó tạo ra những use case tương tác với chúng
* Tạp ra một tập use-case, sau đó xuất ra tất cả các yêu chức năng từ chúng
* Cân nhắc những use case nào được yêu cầu. Từ những use case chủ yếu (yêu cầu nâng cao), rồi sau đó mở rộng những use case đó dựa theo tập hợp những yêu cầu cụ thể

3.2 Actor hệ thống

Một actor hệ thống là một thực thể của thế giới thực mà có thể tương tác với hệ thống thông qua các usecase. Nhiều actor được phân quyền cụ thể như khách hàng, người soạn thảo, nhân viên bán bàn, nhà điều hành, etc. Actor cũng có thể là hệ thống bên ngoài, có thể hiểu đó là hệ thống không nằm trong phạm vi phát triển

Actor con người hoặc hệ thống bên ngoài tương tác với hệ thống bằng cách gửi và nhận thông tin qua một giao diên. Trong trường hợp là actor con người, thông tin thường được trao đổi thông qua các cổng thông tin như là bàn phím, chuột, hoặc một số thiết bị khác. Những actor này nhận thông tin từ hệ thống thông qua những giao diện như mà màn hình, máy in, hoặc một số thiết bị khác

Giao tiếp với những actor mà là hệ thống bên ngoài thường xảy ra thông qua hệ thống mạng. Trong trường hợp này, những giao diện giao tiếp bao gồm mạng và giao thức của nó

Ý tưởng về các hệ thống bên ngoài như các tác nhân không được nhầm lẫn với các actor nội bộ là các thành phần của hệ thống đang được phát triển. Ví dụ, một hệ thống quản lý cơ sở dữ liệu (DBMS), được sử dụng để thực hiện dữ liệu tồn tại cho hệ thống đang được phát triển, không phải là một actor mà là một phần của kiến ​​trúc bên trong của hệ thống. Các quy tắc sau đây có thể giúp xác định một cách thích hợp các hệ thống bên ngoài có thể là các actor:

* Các tác nhân hệ thống là các hệ thống thông tin hoàn chỉnh, và không chỉ các thư viện của các lớp hoặc các thành phần. Các hệ thống này lưu trữ thông tin của riêng mình, có thể được trao đổi với hệ thống đang được phát triển. Thông tin đó có thể thay đổi một cách độc lập với hệ thống đang được thực hiện.
* Các bên tham gia hệ thống nằm ngoài phạm vi phát triển, nghĩa là nhóm sẽ không nhất thiết phải truy cập vào thiết kế nội bộ của các hệ thống này hoặc khả năng thay đổi nó. Nhóm phải liên lạc với một actor hệ thống sử dụng các định nghĩa của chính actor hệ thống vì chúng không thể sửa đổi.

Một số cách tiếp cận cho rằng có hai loại chủ thể: chính thức và thứ cấp. Các bên tham gia chính là những người mà mục đích mà usecase của họ đang cố gắng thỏa mãn, trong khi các bên thứ hai là các bên chỉ cung cấp một số dịch vụ cho quá trình thảo luận. Ví dụ về các actor phụ là máy in, dịch vụ web hoặc những người phải cung cấp thông tin hoặc xác nhận, ví dụ như thư ký xác nhận khách hàng đã thanh toán bằng tiền mặt. Trong cuốn sách này chúng ta không làm theo sự phân biệt đó: chúng ta chỉ phân biệt các actor là actor hệ thống hay actor con người

3.3 Use case hệ thống

Usecase hệ thống khác với usecase nghiệp vụ (Phần 2.3) ở một số khía cạnh. Ví dụ, các actor nghiệp vụ có thể phải mất hàng ngày hoặc thậm chí hàng tuần để thực hiện một thể hiện usecase business, trong khi usecase hệ thống thường được thực hiện trong một thời gian ngắn, thường là vài phút, với một hoặc một vài actor tương tác với một hệ thống và có được một sự nhất quán Và hoàn thành kết quả cho ít nhất một trong những mục tiêu của họ. Usecase hệ thống cũng phải được thực hiện mà không bị gián đoạn trong khi usecase nghiệp vụ không bị hạn chế về mặt này.

Một sự khác biệt cơ bản giữa use case nghiệp vụ và use case hệ thống là use case nghiệp vụ thường được thực hiện bởi nhiều người, trong khi use case hệ thống được thực hiện bởi một số ít (đôi khi chỉ một) nhân viên. Thực tế là nếu use case hệ thống sẽ được thực hiện bởi nhiều actor, thì họ nên tương tác với hệ thống và đó không phải là tình huống chung. Thông thường, mỗi actor đều truy cập vào hệ thống tại thời điểm thuận tiện, truy cập dữ liệu cần thiết và thực hiện các hành động cần thiết. Điều này dẫn đến một chuỗi usecase hệ thống và không phải usecase đơn lẻ, như được giải thích sau. Tuy nhiên, đôi khi có thể có nhiều actor có thể trực tuyến; Ví dụ như ở siêu thị khi khách hàng mua hàng, người giám sát có thể được triệu tập để thực hiện hành động mà thủ quỹ không được làm (ví dụ như hủy bán). Vấn đề là người giám sát phải có mặt trực tuyến và các actor khác phải đợi cô ấy xuất hiện trước khi tiếp tục.

Mặt khác, các hệ thống tính toán bên ngoài có thể được coi là các tác nhân trực tuyến, vì chúng có liên tục. Ví dụ, có thể giả định rằng một nhà khai thác thẻ tín dụng có thể truy cập trực tuyến liên tục. Hiện tại, khách hàng quyết định thanh toán bằng thẻ tín dụng của mình, nhà khai thác thẻ tín dụng sẽ có mặt.

Một usecase hệ thống cấp cao được đại diện bởi một cái tên bên trong một hình elip. Nó thường liên quan đến một hoặc nhiều actor như trong hình 3.1.

Trong sơ đồ trong Hình 3.1, các hình elip biểu thị usecase hệ thống. Theo mặc định, một actor là một vai trò được thực hiện bởi con người. Các loại actor khác có thể được đại diện bởi việc sử dụng các khuôn mẫu, như thể hiện trong hình 3.1, trong đó stereotype {systemc chỉ ra rằng toán tử thẻ tín dụng là một hệ thống bên ngoài chứ không phải là con người.

HÌNH

Sơ đồ usecase sử dụng là một biểu đồ UML rất phổ biến, nhưng nó cũng thường bị hiểu nhầm. Thông thường, các biểu đồ này hiển thị với hàng chục usecase và một số các mảnh của chúng được đính kèm. Tuy nhiên, trong Inception, điều quan trọng là phải biết những quy trình chính của hệ thống, không cần phải nắm chi tiết. Do đó, sự hiện diện của các mảnh trong biểu đồ, và việc sử dụng bao gồm và mở rộng các mối quan hệ giữa các usecase (đôi khi khám phá ra một phần của cấu trúc bên trong) là không nên. Mối quan tâm về các phân mảnh và các usecase bao gồm hoặc mở rộng những người khác có thể được để lại cho thời điểm mà những quá trình cấp cao được chi tiết bằng các chuỗi trình tự hệ thống (Phần 5.8).

Thông thường, không có đủ thông tin để khám phá ra tất cả các phân mảnh vào thời điểm này. Tại sao chỉ một số các usecase được hiển thị và một số khác không được hiển thị? Đơn giản chỉ cần không thấy gì cả trong sơ đồ trong Inception! Điều này ngăn cản sơ đồ có một số lượng lớn ellip - ses, có thể gây trở ngại cho sự hiểu biết của nó. Do đó, cần có các tiêu chí mạnh để quyết định những usecase nào nên được duy trì trên sơ đồ để tránh, một mặt, một số lượng lớn quá trình chi tiết quá mức, và mặt khác, quá ít quá trình, có thể thiếu các tính năng quan trọng của hệ thống.

Quy tắc này là để xem xét như là một usecase chỉ những quy trình có thể được thực hiện trong riêng rẽ. Các quy trình một phần cần phải được thực hiện trong các quá trình khác không được trình bày trong sơ đồ usecase hệ thống.

Tuy nhiên, ngay cả theo hướng dẫn này, số ca sử dụng trong hệ thống thế giới thực vẫn có thể quá cao, do đó việc xử lý chúng có thể trở nên khó khăn. Để giảm số lượng mà không mất thông tin và độ chính xác có thể sử dụng quy tắc thứ hai. Nó bao gồm các usecase nhóm có liên quan phần nào, đặc biệt nếu có thể được thực hiện nhiều lần, giống như một mẫu. Ví dụ: có thể có bốn usecase chẳng hạn như Tạo sách, Lấy sách, Cập nhật sách và Xóa sách hoặc chỉ một usecase được gọi là Quản lý sách bao gồm bốn quy trình đơn. Đây là một mẫu bởi vì nó có thể được lặp lại cho các khái niệm khác: Quản lý nhà xuất bản có thể được sử dụng thay vì Tạo nhà xuất bản, Truy xuất nhà xuất bản, Nhà xuất bản cập nhật và Xóa nhà xuất bản. Mô hình này được gọi là CRUD, là từ viết tắt của Tạo, Truy xuất, Cập nhật và Xóa.

Các phần phụ tiếp theo sẽ trình bày chi tiết các quy tắc đó và đưa ra các tiêu chí khác để đạt được mức độ chi tiết về usecase tốt nhất.

HÌNH

3.3.1 Session đơn

Một usecase hệ thống tốt phải được thực hiện trong một session duy nhất. 2 Điều này có nghĩa là nó phải bắt đầu và kết thúc mà không bị gián đoạn. Ví dụ, việc đăng ký một lệnh được thực hiện trong một session duy nhất của hệ thống, liên quan đến việc xác định khách hàng, lựa chọn sách, hình ảnh hóa giá cả, thanh toán, lựa chọn địa chỉ, vv. Mỗi khía cạnh này là một yêu cầu chức năng của Hệ thống.

Usecase phải được thực hiện dưới dạng các quy trình đơn lẻ. Các quy trình chỉ có thể xảy ra trong bối cảnh các quá trình khác chỉ là những phân mảnh, chứ không phải là các usecase.

Trong trường hợp của hệ thống Livir, tính thuế là cái gì đó sẽ xảy ra chỉ trong quá trình đặt hàng sách. Giả sử rằng các yêu cầu của hệ thống không xác định rằng nó phải được coi là một quá trình độc lập (mặc dù nó có thể được, nếu các yêu cầu khác nhau). Trong trường hợp này, tính thuế không phải là usecase, và vì thực tế đó, nó không được bao gồm trong biểu đồ. Hình 3.2 minh hoạ tình huống mà bạn phải tránh sử dụng các mảnh trong sơ đồ.

Mặt khác, sách Đặt hàng có thể được coi là usecase vì nó là quy trình bắt đầu và kết thúc được xác định rõ ràng, nó xảy ra trong khoảng thời gian liên tục (không bị gián đoạn) và nó tạo ra kết quả âm thanh (lệnh được đăng ký ).

Trong ví dụ thể hiện trong hình 3.1, có hai kịch bản khả thi có thể được điều tra:

* Hệ thống sẽ chỉ xác nhận đơn đặt hàng nếu thanh toán được thực hiện trực tuyến (không thể lưu giỏ hàng). Trong trường hợp này, quá trình đặt hàng sách và thanh toán chỉ là những phần của một usecase duy nhất có thể được gọi là Mua sách. Nếu đây là yêu cầu thực sự, thì sơ đồ phải được thay đổi và hai usecase được thay bằng một usecase duy nhất có tên Mua sách.
* Hệ thống sẽ đăng ký đơn đặt hàng, nhưng không nhất thiết phải thực hiện thanh toán, nghĩa là giỏ hàng có thể được lưu trữ cho đến khi khách hàng quyết định hoàn thành đơn hàng (có thể thực hiện ngay hoặc vào một thời điểm khác). Đây là trường hợp được biểu diễn trong hình 3.1. Trong trường hợp này, Sắp đặt sách là một usecase và Thanh toán cho sách là usecase khác. Chúng có thể được thực hiện ngay lập tức, hoặc một khoảng thời gian vài ngày, và do đó, chúng nên được coi là usecase độc lập.

Mặt khác, sách không được tự động phân phối khi đơn đặt hàng được thanh toán: một người có trách nhiệm xác minh, theo thời gian, đơn đặt hàng đã được phát hành; Người đó thu thập sách trong kho, và gửi gói. Quá trình phân phối có thể xảy ra ngay sau khi đơn đặt hàng được thanh toán nhưng cũng có thể xảy ra vào ngày tiếp theo (hoặc thậm chí từ sáu đến tám tuần sau). Thực tế này phân loại Giao hàng là một usecase khác. Usecase giao hàng đã mua sách như là tiền đề, nhưng mua không phải là một phần của giao hàng: giao hàng chỉ xảy ra sau khi nó. Do đó, việc mua và phân phối, mặc dù là một phần liên tục và thậm chí là một phần của cùng một usecase nghiệp vụ, phải được coi là các usecase hệ thống khác nhau, như thể hiện trong Hình 3.3 nơi đã giới thiệu usecase Gửi sách.

HÌNH

3.3.2 Sự tương tác

Một usecase cũng phải tương tác, có nghĩa là một actor phải tồn tại để tương tác với hệ thống. Các quy trình nội bộ của hệ thống không phải là các usecase bất kể họ phức tạp đến mức nào. Ở phía bên kia, một truy vấn đơn giản về một số thông tin có thể là một usecase nếu có một actor bắt đầu nó.

Các quy trình nội bộ nói trên là một phần của một usecase (ví dụ như Tính thuế là một phần của Sách Mua) hoặc các yêu cầu bổ sung (ví dụ như Lưu trữ dữ liệu tại cơ sở dữ liệu quan hệ). Trong trường hợp các yêu cầu bổ sung, chúng thường được thực hiện bởi các cơ chế liên quan mà không phụ thuộc vào sự tương tác của người dùng. Trong giai đoạn Khởi động, chúng được ghi lại như các yêu cầu bổ sung (Phần 3.4.8) hoặc các yêu cầu phi chức năng (chú thích về các usecase 3.4.5), do đó rủi ro và nỗ lực mà chúng ngụ ý không được đánh giá thấp.

3.3.3 Kết quả phù hợp

Usecase phải cho kết quả phù hợp, có thể là mục nhập hoàn chỉnh hoặc chuyển đổi trên một mẩu thông tin hoặc đơn giản là một truy vấn mà thông tin liên quan được chuyển cho người dùng. Usecase không thể để thông tin ở trạng thái không phù hợp ở cuối. Ví dụ, việc đăng ký một đơn đặt hàng không thể được ký kết mà không có sự xác định của khách hàng và những cuốn sách mà cô ấy đặt hàng, nếu không thông tin về lệnh sẽ không đầy đủ liên quan đến các quy tắc nghiệp vụ; Cửa hàng sách không thể tiếp tục lại đơn hàng hoặc tính tiền mà không biết ai là khách hàng và sách nào được đặt hàng.

Để quyết định xem usecase có kết quả nhất quán, người ta có thể nghĩ như thế này: chỉ một quá trình hoàn chỉnh là một usecase hệ thống, theo nghĩa người dùng có thể vào máy tính, bật máy, thực hiện quá trình, và Khi kết thúc, tắt máy tính, vì quá trình này hoàn thành và đã đạt được một số mục tiêu nghiệp vụ (một số thông tin liên quan đã được nhận hoặc cập nhật bởi người sử dụng).

Điều này loại trừ từ các đoạn định nghĩa ca sử dụng như Tính thuế, trong trường hợp của ví dụ về hiệu sách, bởi vì những khoản thuế đó chỉ được tính trong quá trình đặt hàng chứ không phải là một quy trình bị cô lập. Điều đó cũng loại trừ các hoạt động như Đăng nhập, bởi vì thực hiện đăng nhập và sau đó tắt máy tính không thể được xem như là một quá trình hoàn chỉnh. Nó có thể chỉ là một phần của một hoặc nhiều usecase (Hình 3.4).

Để tránh đại diện Đăng nhập như là một usecase hoặc thậm chí là một phần của nó nó có thể đơn giản giả định rằng mọi hoạt động của hệ thống chỉ có thể được thực hiện bởi một người sử dụng thường xuyên được xác định và thực hiện để thực hiện nó. Cơ chế này là một phần của thiết kế công nghệ của hệ thống và không cần phải được xem xét trong mô hình usecase cao cấp.

Mặt khác, có thể usecase hoàn chỉnh xảy ra bên trong các usecase khác. Ví dụ, quá trình đăng ký một khách hàng có thể được coi là một usecase hoàn chỉnh. Nhưng quy trình đó cũng có thể xảy ra trong quá trình đặt hàng, đặc biệt nếu đó là lần đầu tiên khách hàng sử dụng hệ thống, hoặc khi nào cần cập nhật dữ liệu (Hình 3.5).

Sự liên kết phụ thuộc giữa các usecase là rập khuôn với {extendc để chỉ ra rằng quá trình đặt hàng sách thỉnh thoảng có thể được mở rộng bằng quá trình đăng ký người dùng. Cả hai quá trình có thể xảy ra độc lập với nhau, nhưng đăng ký khách hàng cũng có thể xảy ra trong quá trình đặt hàng.

Để tránh lạm dụng {extendc trong sơ đồ usecase, điều quan trọng cần lưu ý là chỉ cần duy trì các usecase hoàn chỉnh trong sơ đồ. Những mảnh vỡ như những gì thể hiện trong hình 3.2 và 3.4 phải được tránh, ngay cả khi chúng là các điểm mở rộng của một usecase. Những mảnh vỡ này sẽ được giải quyết thích hợp sau đó, khi các usecase ở mức cao được trình bày chi tiết (Chương 5). Trong thực tế, vào thời điểm này, bất kỳ mối liên hệ giữa các usecase như mở rộng hoặc bao gồm nên được tránh trong biểu đồ vì họ thường không thêm thông tin hữu ích cho giai đoạn khởi động.

3.3.4 Use case thiết yếu

Có thể xác định được hai kiểu viết cho các usecase:

• Usecase thiết yếu, không đề cập đến công nghệ giao diện.

• Các usecase cụ thể (hoặc thực tế) được viết riêng cho một công nghệ giao diện nhất định.

Trong các yêu cầu và phân tích, usecase hệ thống được coi là các yêu cầu chứ không phải là thiết kế. Lỗi thường gặp là bao gồm trong các hành động sử dụng này hoàn toàn liên quan đến công nghệ giao diện (chẳng hạn như Mở cửa sổ chính, In báo cáo, và Đăng nhập). Những người chuẩn bị để xử lý khía cạnh thiết kế sẽ quyết định những hành động đó sau đó, sau khi các yêu cầu được phát hiện.

Ambler (2000) chỉ ra rằng các mô hình thiết yếu linh hoạt hơn, để lại nhiều lựa chọn hơn và dễ dàng thay đổi các công nghệ. Ông cũng nói rằng các mô hình thiết yếu mạnh mẽ hơn các đại diện cụ thể bởi vì chúng có nhiều khả năng vẫn còn hiệu lực khi đối mặt với công nghệ thực hiện thay đổi.

Do đó, các usecase thiết yếu phải được coi là lựa chọn chính xác trong suốt thời gian yêu cầu, mặc dù các thông tin chi tiết về công nghệ hoặc các tùy chọn có thể được chú thích để cho phép quản lý rủi ro và tính toán nỗ lực. Sau này trong cuốn sách này (Phần 5.4.1), sự khác biệt giữa các usecase thiết yếu và bê tông sẽ được giải thích chi tiết hơn.

3.3.5 Tóm tắt

Trong usecase Inception thường ngắn gọn, có nghĩa là chúng được mô tả chỉ bằng tên của họ hoặc, trong một số trường hợp, bằng một hoặc hai câu. Tuy nhiên, đây không phải là cách duy nhất để sử dụng usecase. Sau đó chúng sẽ được mở rộng và có thêm chi tiết về các yêu cầu (Chương 5). Cockburn (2001) xác định ba loại usecase liên quan đến mức độ chi tiết:

• Tóm tắt: Một bản tóm tắt một đoạn của usecase.

• Casual: Viết bằng đơn giản, đoạn văn, phong cách viết hoa. Có thể sẽ thiếu thông tin dự án

Liên quan đến usecase, và có thể sẽ ít nghiêm ngặt hơn trong mô tả của nó so với mô tả đầy đủ

• Trang bị đầy đủ: Mở rộng để bao gồm dòng chảy chính và luồng luân phiên, cũng như các phần khác

Chẳng hạn như điều kiện hậu, điều kiện tiên quyết, các bên liên quan và các biến đổi công nghệ.

Trong cuốn sách này, chúng tôi hy vọng rằng các usecase được xem xét trong giai đoạn Khởi động sẽ ngắn gọn. Điều này có nghĩa là tên của họ thường là đủ để giải thích ý nghĩa của chúng. Tuy nhiên, có thể bổ sung thêm sự khám phá cho các yêu cầu hiểu. Ngoài ra, nếu một số usecase chính phải được mở rộng để xác định những rủi ro liên quan đến sự phức tạp vốn có, điều này cũng có thể chấp nhận được. Thông thường, các usecase đầy đủ hoặc mở rộng sẽ chỉ hữu ích sau khi khởi động, khi các yêu cầu phải được chi tiết đầy đủ.

3.3.6 Ranh giới hệ thống

Một trong những quyết định mà một đội phải thực hiện khi thiết kế usecase hệ thống là nơi đặt ranh giới hệ thống. Hình ảnh, ranh giới chỉ là một hình chữ nhật được đặt trên biểu đồ. Bên trong nó là usecase và bên ngoài là các actor. Trong sơ đồ usecase nghiệp vụ, ranh giới của hệ thống đại diện cho giới hạn của tổ chức (công ty, phòng, vv). Ở đây, nó đại diện cho các giới hạn của một hệ thống tính toán.

Một quyết định được đưa ra trước đó là người lao động phải đưa vào biên giới tự động. Một ví dụ là thư ký giúp khách hàng mua sách. Trong trường hợp đó, khi mô hình các usecase hệ thống Thư Ký biến mất, và usecase được thực hiện bởi chính khách hàng.

Tuy nhiên, nếu đó không phải là hiệu sách ảo, nhưng thường là nơi khách hàng mua hàng tại cửa hàng, nhân viên bán hàng có nên giữ trong sơ đồ sử dụng không? Đối với usecase càng trở nên cần thiết, khuyên rằng chỉ những chủ thể thực sự quan tâm đến usecase mới được giữ trong chương trình. Thư ký của ví dụ chỉ là một proxy của khách hàng; Nhân viên bán hàng không có mục đích cá nhân trong quá trình mua sách. Nó hoạt động đơn giản như một giao diện hệ thống. Cho dù hiệu sách là ảo hay không, usecase thiết yếu phải giống nhau, bởi vì cùng một thông tin sẽ được trao đổi giữa khách hàng và hệ thống; Thư ký chỉ chuyển thông tin khi nhận được nó. Trong trường hợp này, người actor duy nhất sẽ là khách hàng và nhân viên bán hàng thậm chí không nên xuất hiện trong sơ đồ. Bằng cách này, phân tích sẽ tạo ra các usecase công nghệ độc lập.

Ai đó có thể hỏi những điều sau đây: Điều gì sẽ xảy ra nếu nhân viên bán hàng chỉ ra rằng cô ấy đã trợ giúp việc bán hàng sao cho một tỷ lệ phần trăm được trả cho cô ấy? Trong trường hợp đó, những gì chúng ta có là một usecase khác với đề cập đến trong đoạn cuối cùng. Ở đây cả khách hàng và nhân viên bán hàng phải là các actor: khách hàng quan tâm đến sách và thư ký quan tâm đến tỷ lệ phần trăm. Ngoài ra, nếu nhân viên bán hàng có thể thực hiện hành động không được cho phép đối với khách hàng (ví dụ, ghi đè giá sản phẩm), cô ấy phải được coi là một nhân viên và không thể bị xóa khỏi biểu đồ.

3.4 Làm cách nào để tìm usecase hệ thống trong sơ đồ nghiệp vụ?

Để phát hiện usecase hệ thống và các actor, nhóm nghiên cứu có thể xem xét sơ đồ usecase nghiệp vụ với ranh giới tự động hóa (phạm vi tự động hóa).

Thứ nhất, những người thực sự quan tâm đến quá trình được tự động hóa nên được xác định. Đây là những tác nhân nghiệp vụ tương tác với usecase nghiệp vụ trong phạm vi tự động hóa và các nhân viên nghiệp vụ tương tác với các usecase đó nhưng không phải là chính họ trong phạm vi tự động hóa. Trong Hình 2.11, các tác nhân hệ thống sau đây có thể được xác định:

* Khách hàng, nhà xuất bản và thẻ tín dụng điều hành, bởi vì họ là những tác nhân nghiệp vụ tương tác với các usecase sẽ được tự động.
* Quản lý mua lại, bởi vì nó là một nhân viên nghiệp vụ mà sẽ không được tự động hóa, nhưng nó tương tác với một usecase sẽ được.

Mặt khác, những điều sau đây sẽ không được xem là các actor hệ thống:

* Thư ký, bởi vì nó là một nhân viên nghiệp vụ sẽ được tự động hóa.
* Phòng Marketing, bởi vì nó là một nhân viên nghiệp vụ mà không tương tác với bất kỳ usecase

Các nhân viên nghiệp vụ có chức năng sẽ được hệ thống hỗ trợ một phần hoặc toàn bộ có thể là các nguồn thông tin quan trọng bởi vì biết cách thực hiện nhiệm vụ của họ có thể rất quan trọng để hiểu hệ thống phải hoạt động như thế nào. Hơn nữa, việc biết được các nhiệm vụ được thực hiện như thế nào có thể có giá trị vì đôi khi những gì xảy ra trong thực tế không phải là những gì được ghi trong sổ quy trình.

Một nguồn yêu cầu khác là các tác nhân nghiệp vụ sẽ trở thành các tác nhân hệ thống. Tuy nhiên, chúng sẽ không phải lúc nào cũng có sẵn cho yêu cầu tạo ra yêu cầu, và trong trường hợp này, chúng nên được thay thế bằng các chuyên gia về tên miền. Ví dụ, có thể không thể phỏng vấn khách hàng của một hiệu sách mà chưa có. Do đó, một người biết nghiệp vụ và có thể giải thích làm thế nào nó là nghĩa vụ phải làm việc sẽ là một người chủ chốt để có được các yêu cầu phải. Nếu các nhà xuất bản và nhà khai thác thẻ tín dụng cũng không có sẵn, vẫn có thể khảo sát các tài liệu hiện có về các giao thức truyền thông và các dịch vụ được cung cấp, để các yêu cầu liên kết với họ có thể được biết đến. Tuy nhiên, đối với khách hàng, khả năng đó sẽ không khả thi.

Một khi các tác nhân hệ thống đã được xác định từ các sơ đồ usecase nghiệp vụ, sẽ có thể quan sát các biểu đồ hoạt động và trạng thái đã được tạo ra trong quá trình phân tích nghiệp vụ để xác minh các hoạt động được thực hiện bởi các tác nhân hệ thống có thể được coi là các usecase hệ thống.

Bằng cách phân tích các hoạt động của Hình 2.10, chúng ta có thể thấy rằng các hoạt động do Khách hàng thực hiện là:

* Đặt hàng sách.
* Trả tiền sách.

Hoạt động duy nhất của Nhà xuất bản là Gửi sách và hoạt động duy nhất của người vận hành thẻ tín dụng là Xác nhận thanh toán.

Không phải mọi hoạt động của Thư ký ban đầu đều trở thành usecase hệ thống. Như hình 2.7, nhân viên bán hàng được chia thành hai actor: Người quản lý mua lại, sẽ không được tự động, và Clerk, sẽ được tự động hóa. Các hoạt động thực hiện bởi vai trò người quản lý Chuyển đổi có thể trở thành usecase hệ thống. Tuy nhiên, các hoạt động được thực hiện bởi Thư ký nhân viên nghiệp vụ, đang được tự động hoá, trở thành hành động nội bộ của hệ thống. Ví dụ, các hoạt động Đăng ký đặt hàng và Thông báo về khả năng sử dụng và tổng số trong Hình 2.10 không trở thành usecase hệ thống; Chúng chỉ là một phần của usecase sách đặt hàng do khách hàng khởi xướng. Tuy nhiên, Đặt hàng sách từ nhà cung cấp là một usecase hệ thống mà actor là Người quản lý chuyển đổi.

Về các quy trình sẽ được xác định là usecase, người ta vẫn phải xem xét hai mối quan tâm:

* Các hoạt động được kết nối bởi dòng chảy nhất thiết xảy ra ngay lập tức hay chúng có thể xảy ra trong những khoảnh khắc khác nhau không? Trong ví dụ, Đặt hàng sách và Thanh toán cho các hoạt động sách phải xảy ra trong một phiên sử dụng duy nhất của hệ thống hoặc chúng có thể được thực hiện trong những khoảnh khắc khác nhau không? Câu trả lời sẽ phụ thuộc vào cách nghiệp vụ được tổ chức, nghĩa là nó sẽ phụ thuộc vào các quy tắc nghiệp vụ. Nếu công ty quyết định chỉ nhận đơn đặt hàng sau khi thanh toán được thực hiện thì hai hoạt động đó sẽ tạo thành một usecase đơn lẻ. Tuy nhiên, nếu công ty quyết định rằng có thể đăng ký một đơn đặt hàng (lưu một giỏ hàng) và trả tiền cho nó một ngày khác, thì có hai usecase khác nhau.
* Các hoạt động được chọn là hoàn chỉnh và có âm thanh? Ví dụ: Gửi Sách về mô tả của hoạt động do nhà xuất bản thực hiện không? Bằng sự tương tự như quá trình được thực hiện bởi khách hàng, có thể suy ra rằng nhà xuất bản sẽ nhận được một đơn đặt hàng đầu tiên và sau đó nó sẽ gửi các sách mà họ có trong kho. Tuy nhiên, khoản thanh toán trong trường hợp nghiệp vụ-doanh nghiệp thường không được thực hiện ngay lập tức một hóa đơn được tạo ra và trả bởi hiệu sách với một thời hạn xác định trước (ví dụ, bốn tuần sau khi mua).

Tại thời điểm này, cần phải sửa đổi lại sơ đồ hình 2.10. Nhà xuất bản không chỉ đơn giản Gửi sách. Nó sẽ nhận được đơn đặt hàng, gửi hóa đơn thanh toán, và cuối cùng gửi những cuốn sách có sẵn. Các cuốn sách phải được nhận và đăng ký bởi một số nhân viên tại hiệu sách. Nhân viên đó sẽ là một actor hệ thống mới và có thể gọi là Nhân viên tiền gửi.

Một câu hỏi lặp lại là nếu các actor tương ứng với các hồ sơ an ninh. Điều này không nhất thiết phải là trường hợp. Mục đích đằng sau việc xác định và mô hình hoá các tác nhân có liên quan nhiều hơn đến quá trình tìm kiếm và tổ chức các yêu cầu hơn là quá trình cấp quyền truy cập vào hệ thống. Một thiết kế phần mềm tốt sẽ xử lý quyền truy cập một cách năng động, cho phép tạo hồ sơ người dùng, và các nhiệm vụ được liên kết với các hồ sơ khác nhau của người dùng cá nhân. Do khía cạnh đó của hệ thống không phụ thuộc vào miền, nó không đủ để mô hình nó trong phạm vi miền. Nói cách khác, các cá nhân trong usecase không được coi là hồ sơ bảo mật. Những hồ sơ này sẽ được tạo tự động với việc sử dụng cơ chế độc lập miền chung.

Biểu đồ máy trạng trình bày trong Hình 2.15 cũng đặc biệt hữu ích trong các usecase hệ thống phải được phân tích. Mỗi thay đổi trạng thái của một cuốn sách (đối tượng nghiệp vụ chính) phải do ai đó thực hiện và "ai đó" là một actor thực hiện một usecase. Như vậy, từ các chuyển tiếp trong sơ đồ đó, một tập hợp các usecase hệ thống mới được phát hiện như trong Hình 3.6.

Trong hình 3.6, usecase sách Catalog và Order book từ nhà xuất bản được duy trì bên ngoài phạm vi của hệ thống Livir. Điều này là do thực tế là nó đã được phát hiện tại một điểm mà usecases như vậy sẽ được thực hiện trong hệ thống nhà xuất bản, không phải trong hệ thống đang được phát triển. Vì chúng không có trong hệ thống hiệu sách nên chúng có thể được duy trì trong sơ đồ, nhưng bên ngoài ranh giới của hệ thống, bởi vì chúng nằm ngoài phạm vi và sẽ không được thực hiện.

Sau khi đảm bảo rằng nó được hiểu rằng chúng sẽ không được thực hiện, chúng có thể được xóa khỏi biểu đồ, để đơn giản hóa nó và tránh nhầm lẫn với những người có thể không hiểu rằng những gì nằm ngoài ranh giới sẽ không được thực hiện. Trong trường hợp đó, tên của họ có thể được liệt kê trong một danh sách ngoài phạm vi, cho mục đích ghi âm.

Các usecase liên quan đến người quản lý tiền gửi mới là cần thiết để bao gồm hầu hết các bước chuyển đổi được chỉ ra trong sơ đồ máy nhà nước của hình 2.15. Điều thú vị là thông báo rằng một số chuyển tiếp được thực hiện bên ngoài phạm vi của công ty. Ví dụ, khi một khách hàng quyết định trả lại một cuốn sách, điều này không được thực hiện thông qua hệ thống hiệu sách; Cô ấy chỉ đơn giản gửi sách trở lại, thường là qua đường bưu điện. Thông tin về việc trả lại chỉ reachesthe hiệu sách khi nó nhận được cuốn sách. Do đó, usecase được đề cập ở đây là Đăng ký sổ sách trả lại và không phải Trả lại cuốn sách, bởi vì đăng ký được thực hiện bởi người quản lý tiền gửi, chứ không phải bởi khách hàng. Điều này cũng tương tự như việc đăng ký sách đến, được đại diện bởi usecase Nhận sách, bởi vì khi nhà xuất bản gửi sách nó không thể đăng ký trực tiếp vào hệ thống cửa hàng sách (trừ khi cả hai hệ thống được tích hợp). Chỉ có thể đến được nhà sách. Đây là sự khác biệt cơ bản giữa usecase nghiệp vụ và usecase hệ thống: trong usecase nghiệp vụ, quá trình xảy ra trong thế giới thực có thể được mô tả như một phần của usecase, nhưng khi tập trung vào hệ thống, chỉ Các quy trình liên quan đến hệ thống có thể được đại diện.