ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

DƯƠNG THỊ THANH HUYỀN

SINH TỰ ĐỘNG CA KIỂM THỬ TỪ CÁC MÔ HÌNH THỰC THI ĐƯỢC

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

HÀ NỘI – 2017

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

DƯƠNG THỊ THANH HUYỀN

SINH TỰ ĐỘNG CA KIỂM THỬ TỪ CÁC MÔ HÌNH THỰC THI ĐƯỢC

Ngành: Công nghệ thông tin
Chuyên ngành: Kỹ thuật phần mềm
Mã số: 60480103

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

CÁN BỘ HƯỚNG DẪN KHOA HỌC: TS. Đặng Đức Hạnh

HÀ NỘI – 2017

VIETNAM NATIONAL UNIVERSITY, HA NOI UNIVERSITY OF ENGINEERING TECHNOLOGY

DUONG THI THANH HUYEN

AUTOMATED TESTCASE GENERATION FROM EXECUTABLE MODELS

THE MS. THESIS INFORMATION TECHNOLOGY

Supervisor: Dr. DANG DUC HANH

HA NOI-2017

LỜI CẢM ƠN

Đầu tiên, tôi xin gửi lời cảm ơn chân thành và sâu sắc tới thầy Đặng Đức Hạnh – giảng viên bộ môn Công Nghệ Phần Mềm - Người đã trực tiếp hướng dẫn nhiệt tình, giúp đỡ và động viên tôi rất nhiều, góp ý cho tôi những lời khuyên chân thành trong quá trình nghiên cứu để hoàn thành đề tài này.

Tiếp theo, tôi xin chân thành cảm ơn tập thể các thầy, cô giáo Trường Đại học Công Nghệ - Đại học Quốc Gia Hà Nội – những người đã tận tâm truyền đạt những kiến thức quý báu cho tôi trong suốt thời gian học tập.

Cuối cùng, tôi xin gửi lời biết ơn sâu sắc tới gia đình, người thân đã luôn hết lòng giúp đỡ, mang lại cho tôi nguồn động viên tinh thần to lớn và tạo mọi điều kiện thuận lợi cho tôi trong quá trình học tập và hoàn thành luận văn.

Xin trân trọng cảm ơn!

Hà Nội, ngày 13 tháng 10 năm 2017

Học viên

Dương Thị Thanh Huyền

TÓM TẮT

Luận văn trình bày một phương pháp nghiên cứu tự động hóa quá trình sinh ca kiểm thử từ mô hình luồng quy trình nghiệp vụ (BPMN). Hướng nghiên cứu dựa trên lý thuyết kiểm thử dựa trên mô hình. Mục tiêu đề ra là tự động hóa quá trình kiểm thử, nâng cao hiệu quả kiểm thử, tiết kiệm chi phí và thời gian phát triển sản phẩm phần mềm. Phương pháp được đề xuất với nội dung chính như sau: Với đầu vào là mô hình luồng nghiệp vụ BPMN lưu giữ dưới dạng tệp xml, chương trình kiểm thử biến đổi tệp xml bằng cách bóc tách các thông điệp, toán tử và các ràng buộc được đưa vào trong thiết kế. Sau đó thực hiện dò tìm và sinh ca kiểm thử cho các đường đi từ điểm bắt đầu cho tới điểm kết thúc gọi là các đường kiểm thử.

Để kiểm nghiệm mức độ khả thi của phương pháp, một công cụ hỗ trợ đã được cài đặt và thử nghiệm với một số ví dụ đơn giản nhằm minh chứng cho tính đúng đắn và hiệu quả của phương pháp trên. Kết quả thực nghiệm cho thấy hiệu quả của các kịch bản ca kiểm thử là khả thi để áp dụng cho các công ty phát triển phần mềm. Từ các ca kiểm thử được sinh ra có thể áp dụng để kiểm thử tích hợp, kiểm thử hệ thống phần mềm. Hơn nữa, các ca kiểm thử còn có thể áp dụng để kiểm tra tính đúng đắn của các công cụ quản lý quy trình nghiệp vụ.

Từ khóa: Kiểm thử dựa trên mô hình, kiểm thử tự động, mô hình hóa quy trình nghiệp vụ, quản lý quy trình nghiệp vụ.

ABSTRACT

This thesis is researched and proposes a method to auto - generate a set of test cases from the BPMN model based on the model-based testing in order to automate the testing process, increase effectiveness, reduce cost and time of testing. The method contains the following steps. At first, with the input as BPMN model, the program converts xml files by analyzing and dividing the input messages, objects and constraints into fragments. After that, it searches and generates testing paths.

A tool has been implemented and tested with some simple examples in order to study the feasibility of the method. The experimental results have given us the perspective of the tool to apply in automation testing in software companies. From the generated test cases can be applied to the integration test, system testing. In addition, it can be used to test the validity of business process management tools.

Keywords: Model based testing, automated testing, bpmn, bpm.

LỜI CAM ĐOAN

Tôi xin cam đoan rằng những nghiên cứu về sinh tự động ca kiểm thử từ mô hình BPMN được trình bày trong luận văn này dưới sự hướng dẫn của thầy Đặng Đức Hạnh là của tôi. Những gì tôi viết ra không sao chép từ các tài liệu, không sử dụng các kết quả của người khác mà không trích dẫn cụ thể.

Tôi xin cam đoan công cụ kiểm thử tự động tôi trình bày trong luận văn là do tôi tự phát triển, không sao chép mã nguồn của người khác. Nếu sai tôi hoàn toàn chịu trách nhiệm theo quy định của Trường Đại học Công Nghệ - Đại học Quốc Gia Hà Nội.

Hà Nội, ngày tháng năm 2017

Học viên

Dương Thị Thanh Huyền

DANH SÁCH BẢNG BIỂU

DANH SÁCH HÌNH VẼ

BẢNG THUẬT NGỮ VIẾT TẮT

STT	Từ viết tắt	Viết đầy đủ	Ý nghĩa
1	BPMN	Business Process model and notation	Ký pháp và mô hình quy trình nghiệp vụ
2	BPEL	Business Process Execution Language	Ngôn ngữ thực thi luồng quy trình nghiệp vụ
3	OMG standard	Object management group standard	Tiêu chuẩn nhóm quản lý đối tượng
4	KPI	Key Performance Indicators	Các chỉ số hoạt động chính
5	UML	Unified Modeling Language	Ngôn ngữ mô hình hóa thống nhất
6	SUT	Software under test	Phần mềm đang được kiểm thử
7	BPM	Business Process Management	Quản lý quy trình nghiệp vụ
8	BPMI	Business Process Management Initiative	Tổ chức sáng kiến quản lý quy trình nghiệp vụ
9	SDG	Sequence diagram graph	Biểu diễn đồ họa của biểu đồ tuần tự
10	OCL	Object Constraint Language	Ngôn ngữ ràng buộc đối tượng
11	SRS	Software Requirement Specification	Đặc tả yêu cầu phần mềm
12	PIM	Platform-independent model	Độc lập với nền tảng mô hình
13	PSM	Platform-specific model	Phụ thuộc nền tảng mô hình
14	MSDN	Model-Driven Software Engineering	Phát triển phần mềm hướng mô hình
15	M2M	Model-to-Model	Mô hình thành mô hình
16	M2T	Model-to-Text	Mô hình thành văn bản
17	MDE	Model-Driven Engineering	Kỹ thuật ứng dụng hướng mô hình

CHƯƠNG 1: ĐẶT VẤN ĐỀ

Để phù hợp với xu hướng phát triển phần mềm ngày càng cao như hiện nay, các kỹ thuật và phương pháp tự động tạo ra các ca kiểm thử từ mô hình đã được quan tâm ở nhiều nước trên thế giới, nhưng ở Việt Nam kỹ thuật và các phương pháp nghiên cứu lĩnh vực này chưa được áp dụng và phát triển mạnh trong công nghiệp sản xuất phần mềm. Thật vậy, đó là vấn đề cấp bách cần thiết của các công ty phần mềm cũng như của các tổ chức phát triển, triển khai dự án phần mềm.

Kiểm thử phần mềm được tiến hành để cung cấp cho các bên liên quan thông tin về chất lượng của sản phẩm hoặc dịch vụ được kiểm thử. Mục đích chính của kiểm thử là phát hiện ra các lỗi phần mềm để từ đó khắc phục và sửa chữa. Việc kiểm thử không thể khẳng định được rằng các chức năng của sản phẩm đúng trong mọi điều kiện, mà chỉ có thể khẳng định rằng nó hoạt động đúng trong những điều kiện cụ thể. Tùy thuộc vào từng phương pháp, việc kiểm thử có thể được thực hiện bất cứ lúc nào trong quá trình phát triển phần mềm với những kỹ thuật tương ứng.

Quá trình kiểm thử theo các phương pháp truyền thống như là kiểm thử dựa trên đặc tả yêu cầu để tạo ra trường hợp thử nghiệm bằng tay. Ngoài các lợi thế của kỹ năng như kinh nghiệm của kiểm thử viên, quá trình này có nhược điểm là phải mất nhiều nỗ lực, chất lượng của các ca kiểm thử không đồng nhất phụ thuộc vào kinh nghiệm của người kiểm thử. Trong khi đó, việc kiểm thử tự động mà đặc biệt là kiểm thử dựa trên mô hình có những lợi thế giúp giảm chi phí và thời gian, độ bao phủ tốt và giảm lỗi chủ quan, khả năng sử dụng lại cao, sớm phát hiện lỗi, đảm bảo được chất lượng phần mềm.

Có nhiều cách tiếp cận khác nhau để tạo các ca kiểm thử tự động như tạo các ca kiểm thử tự động từ các mô hình như được biểu diễn bằng máy hữu hạn trạng thái, ôtômat, đặc tả đại số, mô hình luồng quy trình nghiệp vụ, biểu đồ trạng thái bằng Unified Modeling Language (UML),…Các mô hình biểu diễn bằng máy hữu hạn trạng thái, ôtômat, đặc tả đại số, biểu đồ trạng thái UML đòi hỏi yêu cầu cao để đạt được đặc tả chính xác hành vi hệ thống. Trong khi mô hình BPMN- Business Process Model and Notation dễ dàng xây dựng, xác minh tính chính xác và rõ ràng thông tin mô tả luồng quy trình nghiệp vụ đối với cả cán bộ phát triển hệ thống cũng như người sử dụng và các bên liên quan. Do đó, trong khuôn khổ luận văn này, tôi lựa chọn tiếp cận một phương pháp kiểm thử tự động từ mô hình luồng quy trình nghiệp vụ BPMN.

Để áp dụng phương pháp kiểm thử kiểm thử dựa trên mô hình đòi hỏi các mô hình phải đặc tả chính xác hành vi của hệ thống. Tuy nhiên, xây dựng mô hình là một công việc khó khăn đòi hỏi nhiều nỗ lực và tiềm ẩn nhiều nguy cơ lỗi. Việc kiểm thử tính đúng đắn cho thiết kế dựa trên mô hình từ chính đặc tả luồng quy trình nghiệp vụ thuận lợi và mang lại nhiều lợi ích hơn. Do các trường hợp kiểm thử được tạo ra trước khi mã nguồn được viết, cho phép các nhà phát triển có thể sử dụng các trường hợp thử nghiệm để kiểm tra chương trình khi họ phát triển mã nguồn. Điều này làm giảm số lần lặp lại giữa phát triển và thử nghiệm, tiếp tục tiết kiệm nguồn lực, tài nguyên. Các trường hợp kiểm thử được tạo trực tiếp từ các yêu cầu của hệ thống có thể được sử dụng để phát hiện các lỗi sớm, giúp làm giảm chi phí.

Kiểm thử dựa trên mô hình thường tạo ra các trường hợp kiểm thử từ mô hình trừu tượng của phần mềm, bao gồm các đặc tả chính thức và mô tả thiết kế. Với mục đích sinh ca kiểm thử trực tiếp từ mô hình luồng quy trình nghiệp vụ, luận văn trình bày các nội dung chính sau:

- ***Chương 1***: Đặt vấn đề
- ***Chương 2***: Giới thiệu tổng quan về mô hình thực thi được, tổng quan về kiểm thử dựa trên mô hình, một số phương pháp kiểm thử dựa trên mô hình. Tập trung nghiên cứu tìm hiểu về mô hình hóa quy trình nghiệp vụ và nghiên cứu cách thức sử dụng ký pháp của BPMN 2.0. Giới thiệu một số công cụ thiết kế và thực thi mô hình BPMN.
- ***Chương 3***: Phát biểu bài toán, đề xuất phương pháp sinh tự động ca kiểm thử từ mô hình BPMN.
- ***Chương 4:*** Mô tả cài đặt và kết quả thực nghiệm triển khai phương pháp đã đề xuất.
- ***Chương 5***: Trình bày tóm tắt kết quả đã đạt được, kết luận, những hạn chế và hướng nghiên cứu phát triển trong tương lai.

CHƯƠNG 2: TỔNG QUAN VỀ MÔ HÌNH HÓA QUY TRÌNH NGHIỆP VỤ VÀ KIỂM THỬ DỰA TRÊN MÔ HÌNH

Chương 2 giới thiệu cơ sở lý thuyết cho luận văn bao gồm: tổng quan về mô hình thực thi được, kiểm thử dựa trên mô hình, giới thiệu một số phương pháp kiểm thử dựa trên mô hình. Tập trung tìm hiểu một mô hình thực thi được – mô hình hóa quy trình nghiệp vụ BPMN, các ràng buộc thiết kế và công cụ quản lý quy trình nghiệp vụ nhằm vận dụng, xây dựng ứng dụng thực nghiệm phục vụ cho kết quả chính của luận văn.

2.1 Giới thiệu

Phương pháp tiếp cận hướng mô hình hóa trong công nghiệp phần mềm có vai trò vô cùng quan trọng không chỉ cho quá trình phát triển phần mềm mà cho cả quá trình kiểm thử phần mềm nhằm tăng tính hiệu quả, đảm bảo chất lượng sản phẩm và tối ưu chi phí. Do đó, các phương pháp sinh ca kiểm thử từ mô hình đã được đề cập trong nhiều nghiên cứu. Tuy nhiên, nghiên cứu sinh ca kiểm thử từ mô hình luồng quy trình nghiệp vụ BPMN chưa được đề cập nhiều. Trong khi BPMN là một mô hình cần thiết trong quá trình phát triển phần mềm để cung cấp cái nhìn tổng quan các nghiệp vụ hệ thống không chỉ cho nhà phát triển sản phẩm mà còn trực quan, dễ hiểu cho khách hàng và các bên liên quan.

Nội dung các phần tiếp theo trong chương sẽ nêu kiến thức tổng quan về kiểm thử dựa trên mô hình, giới thiệu một số phương pháp kiểm thử mô hình, tổng quan về mô hình hóa quy trình nghiệp vụ (BPMN).

2.2 Tổng quan về mô hình thực thi được

2.2.1 Khái niệm mô hình (Model)

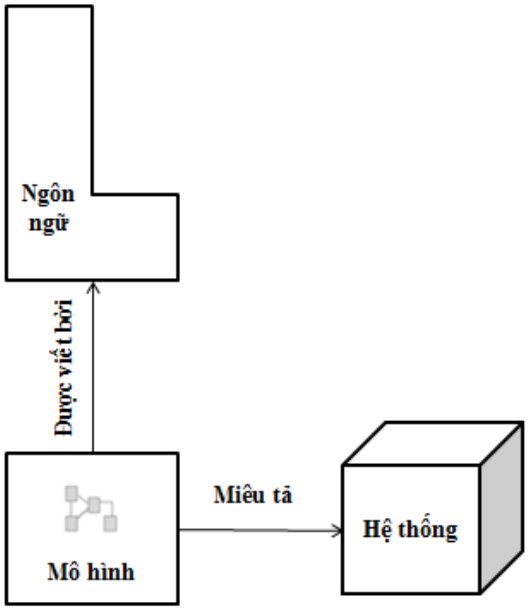
Mô hình là một biểu diễn trừu tượng của cấu trúc, tính năng và hành vi của hệ thống. Mô hình có thể được biểu diễn bằng các ký hiệu đồ họa và diễn tả bằng ngôn ngữ đặc tả miền cụ thể dưới dạng ngôn ngữ hình thức.

Dưới đây là hai định nghĩa về "mô hình" cơ bản của từ điển American Heritage:

- Mô hình là một đối tượng nhỏ được xây dựng để quy mô, mô phỏng chi tiết một đối tượng khác thường là đối tượng lớn hơn [3].
- Mô hình là một sự biểu đồ hóa mô tả chi tiết hệ thống, đồng thời mô tả chi tiết các khía cạnh, các đặc tính của hệ thống [3].

Định nghĩa này thể hiện hai đặc điểm quan trọng của mô hình: Các mô hình phải nhỏ so với kích thước của hệ thống, rằng chúng ta có thể kiểm thử nó mà không mất quá nhiều chi phí, nhưng chúng phải đủ chi tiết để mô tả thực tế và các đặc điểm cần kiểm thử.

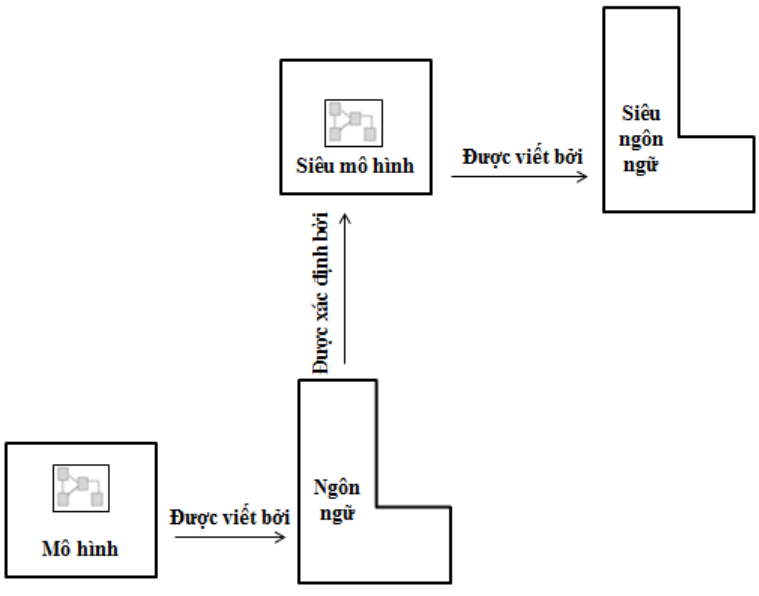
Theo AnnneKe, mô hình được định nghĩa: "Một mô hình là một mô tả (hoặc một phần) của một hệ thống được viết bởi một ngôn ngữ hình thức" [4]. "Ngôn ngữ hình thức là ngôn ngữ với mẫu được xác định rõ ràng và ngữ nghĩa phù hợp với việc biên dịch tự động bởi máy tính" [4].



Hình 2.1: Hình mô tả sự biểu diễn của mô hình.

2.2.2 Khái niệm siêu mô hình (Meta- model)

Meta-model là một mô hình ở mức trừu tượng hơn và sử dụng để biểu diễn mô hình. Meta-model được viết bởi ngôn ngữ gọi là meta-language. Meta-model được biểu diễn như hình sau [4]:

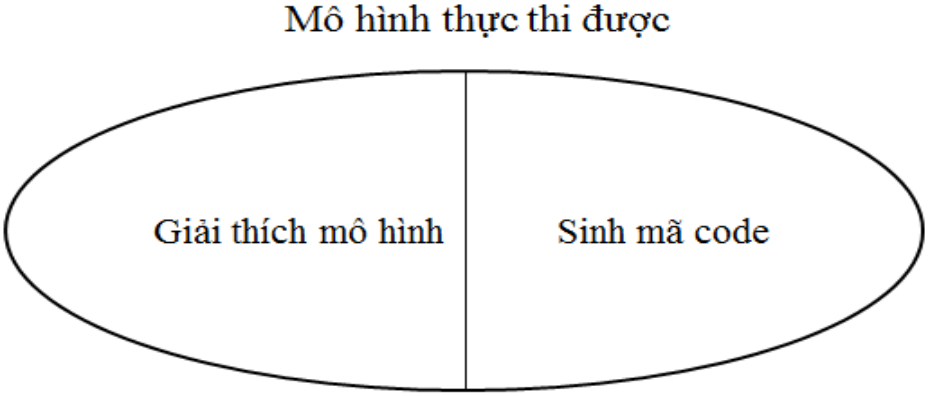


Hình 2.2: Siêu mô hình (Meta-model)

2.2.3 Khái niệm mô hình thực thi được (executable model)

Với xu thế áp dụng kỹ thuật ứng dụng hướng mô hình (MDE), phát triển phần mềm tập trung vào mô hình hóa các thành phần của hệ thống phần mềm dựa trên ngôn ngữ mô hình hóa. Ngôn ngữ mô hình hóa cho phép xác định cấu trúc và hành vi của hệ thống phần mềm một cách chính thức và ở mức trừu tượng cao gần với không gian vấn đề hơn là mức ngôn ngữ lập trình. Ngôn ngữ mô hình thực thi không chỉ cho phép đặc tả các khía cạnh tĩnh của hệ thống mà còn đặc tả các khía cạnh động, tức là hành vi của hệ thống phần mềm thông qua các mô hình thực thi được. Do đó, "một mô hình có thể thực thi được nếu từ đó có thể viết được một chương trình thực thi hoặc chạy mô hình" [10,11].

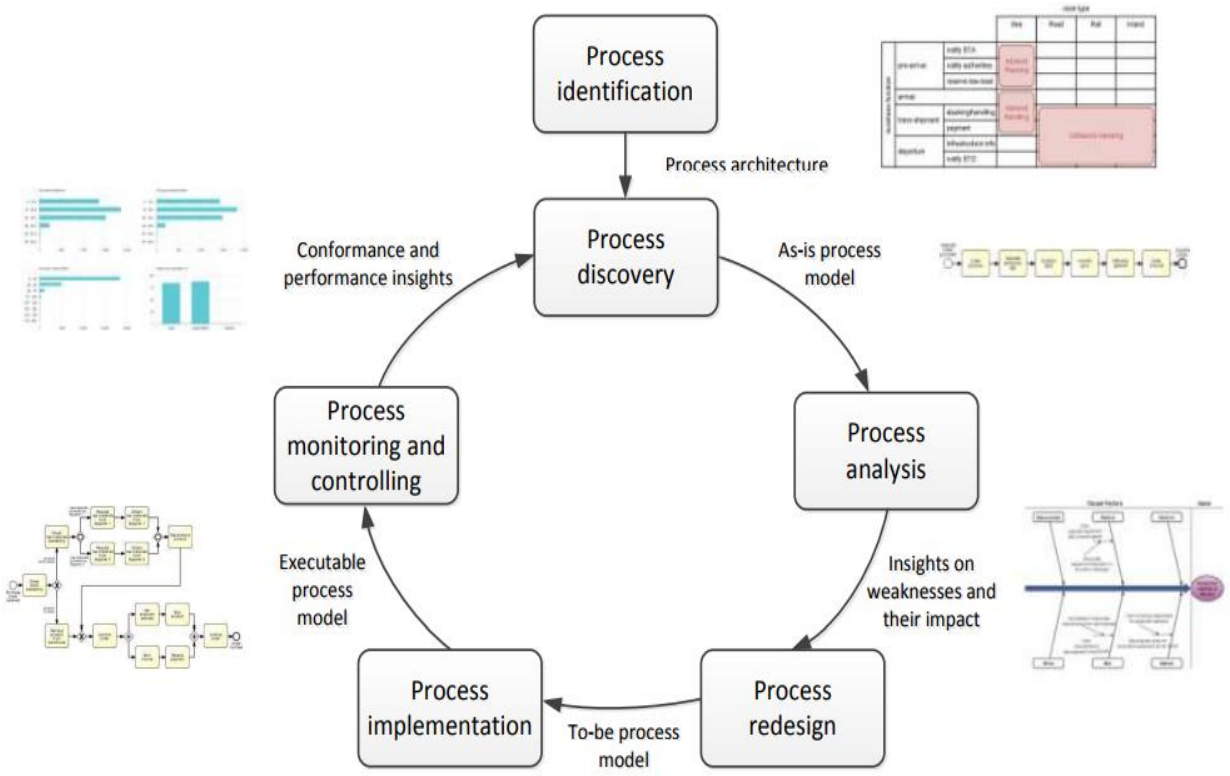
Theo quan điểm trên Jordi Cabot cũng đã xác định: "một mô hình thực thi được là một một mô hình đủ để thực thi" [3]. Theo đó, một mô hình thực thi được khi ngữ nghĩa, hoạt động được xác định định nghĩa đầy đủ. Trong thực tế, khả năng thực thi của mô hình phụ thuộc nhiều vào công cụ thực hiện hơn là bản thân của mô hình (ví dụ một công cụ có thể yêu cầu tính đầy đủ và chi tiết của mô hình hơn trong khi một số công cụ khác có khả năng "lấp đầy khoảng trống" – nghĩa là tự bổ sung những thành phần còn thiếu của mô hình dựa trên các định nghĩa có sẵn của công cụ để thực thi mô hình không đầy đủ đó). Một trong những mô hình thực thi được nổi tiếng nhất là UML được mô tả cụ tả cụ thể trong cuốn sách "Executable UML: A Foundation for Model-Driven Architecture" xuất bản lần đầu tiên vào năm 2002. Dựa trên các định nghĩa được xác định ở đây, bản thân tổ chức OMG đang trong quá trình chuẩn hóa các khái niệm dựa trên mô hình thực thi UML. Tuy nhiên, phiên bản hiện tại của tiêu chuẩn này cũng chưa nêu định nghĩa mô hình thực thi được. Cũng theo Jordi Cabot, việc sinh mã code từ mô hình và giải thích mô hình là hai chiến lược thay thế khác nhau để "implement"- thực hiện những công cụ thực thi:



Hình 2.3: Mối quan hệ giữa mô hình thực thi với sinh mã code và giải thích mô hình [12].

Chiến lược sinh mã code liên quan đến việc sử dụng trình biên dịch mô hình (M2T- Model to Text) để tạo ra mô hình đại diện cấp thấp hơn của mô hình sử dụng các ngôn ngữ và nền tảng lập trình .Thay vào đó, chiến lược diễn giải mô hình dựa trên sự dễ dàng đọc và chạy mô hình.

Trong luận văn này tôi đề cập đến mô hình thực thi được BPMN xuyên suốt tài liệu và từ đó đề xuất phương pháp sinh ca kiểm thử từ BPMN. Đây là mô hình quy trình luồng nghiệp vụ hệ thống với tập các ký pháp hỗ trợ mô tả hành vi của hệ thống . Theo Maccello La Rosa & Marlon Dumas vòng đời quản lý quy trình nghiệp vụ (BPM- Business process model) mô tả như hình sau:



Hình 2.4: Hình mô tả vòng đời quản lý quy trình nghiệp vụ [13].

BPMN là một mô hình mô tả cụ thể hành vi của người dùng và hệ thống đủ chi tiết để có thể thực thi được. Từ BPMN có thể sinh mã chương trình (M2T) thông qua sự hỗ trợ của ngôn ngữ thực thi được BPEL – Business process execution language và sinh các ca kiểm thử cũng như kịch bản kiểm thử tích hợp chức năng, kiểm thử hệ thống. Một thể hiện cụ thể hơn cho việc BPMN là mô hình đủ chi tiết để thực thi được là BPMN có thể thực thi trực tiếp trên công cụ quản lý quy trình nghiệp vụ (BPM). Điều này có nghĩa là, khi có một mô hình BPMN được import vào công cụ BPM, ta có thể thực hiện được luồng nghiệp vụ trên công cụ. Trực quan hóa việc thực thi BPMN trên công cụ Activiti sẽ được trình bày chi tiết trong chương III và chương IV.

2.3 Tổng quan về kiểm thử dựa trên mô hình

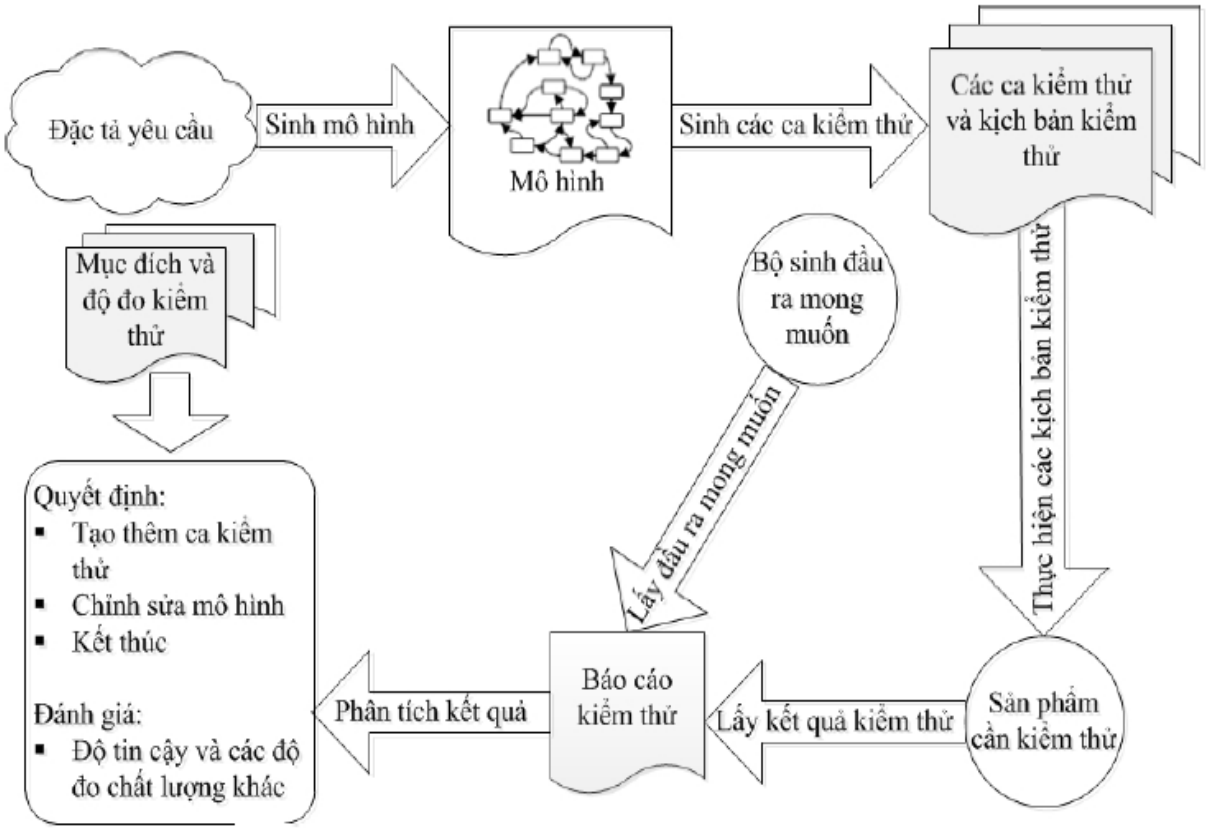
2.3.1 Phương pháp tiếp cận kiểm thử dựa trên mô hình

Có bốn phương pháp chính tiếp cận với kiểm thử dựa trên mô hình như sau:

- ***Sinh ra dữ liệu đầu vào kiểm thử từ một mô hình chính***: đầu vào cơ bản của kiểm thử dựa trên mô hình là các mô hình, từ đó tạo ra các ca kiểm thử bằng cách chọn lựa thông minh một tập hợp con của tập giá trị các trường hợp có khả năng để đưa ra dữ liệu đầu vào kiểm thử.
- ***Sinh ra các ca kiểm thử từ một mô hình môi trường:*** phương pháp này sử dụng một loại mô hình khác, mô hình này sẽ miêu tả môi trường mong muốn của SUT- Software under test. Từ mô hình mô phỏng giả lập này đưa ra các tham số gọi tới SUT. Tuy nhiên, mô hình môi trường không mô hình hóa được toàn bộ hành vi của SUT. Vì vậy nó rất khó để xác định chính xác một kiểm thử là thành công hay thất bại.
- ***Sinh ra các ca kiểm thử với các dự đoán từ một mô hình hành vi***: đưa ra các ca kiểm thử có khả năng thực thi bao gồm các thông tin dự đoán các giá trị đầu ra mong muốn của SUT. Hoặc một vài khâu kiểm tra tự động các giá trị đầu ra thực tế để có thể nhìn thấy nếu chúng là đúng đắn. Điều này khó hơn việc sinh ra dữ liệu kiểm thử đầu vào hoặc kiểm thử dựa trên trình tự gọi tới SUT mà không kiểm tra tới kết quả đầu ra. Để đưa ra kiểm thử với các dự đoán thì người đưa ra các ca kiểm thử phải có đầy đủ thông tin về các hành vi mong đợi của SUT để có thể tiên đoán hoặc kiểm tra các dữ liệu đầu ra của SUT. Một cách khác, với định nghĩa kiểm thử dựa trên mô hình này, mô hình phải mô tả các hành vi mong đợi của SUT, cũng như mối quan hệ giữa chúng, đồng thời mô tả đầu vào và đầu ra cho từng hành vi. Thuận lợi của cách tiếp cận này là nó là phương pháp tiếp cận duy nhất giải quyết được vấn đề kiểm thử dựa trên mô hình bằng việc chọn lựa các giá trị đầu vào và việc đưa ra các trình tự của sự vận hành, việc đưa ra các ca kiểm thử có khả năng thực thi bao gồm thông tin quyết định sau mỗi ca kiểm thử.
- ***Sinh ra các đoạn mã kiểm thử từ các kiểm thử trừu tượng***: sinh ra các ca kiểm thử có thể thực thi bao gồm các thông tin tiên đoán dựa trên mô hình hành vi của SUT. Quá trình sinh ra các ca kiểm thử này bao gồm việc sinh ra dữ liệu kiểm thử và trình tự các phương thức gọi tới kiểm thử tuần tự, sinh ra các dự đoán để kiểm tra kết quả đầu ra của SUT. Đây là một phương pháp tiếp cận hoàn thiện và phức tạp nhất, mang lại hiệu quả tốt nhất. Nó có thể tự động hoàn thiện các tiến trình thiết kế, đưa ra một mô hình hoàn thiện, tái hiện đầy đủ các tuần tự kiểm thử và chuyển đổi thành các kịch bản kiểm thử có thể thực thi [2].

Quá trình kiểm thử dựa trên mô hình được bắt đầu bằng việc xác định yêu cầu của hệ thống từ đó xây dựng mô hình dựa vào các yêu cầu và chức năng của hệ thống. Việc xây dựng mô hình còn phải dựa trên các yếu tố dữ liệu đầu vào và đầu ra. Mô hình này được sử dụng để sinh đầu vào cho các ca kiểm thử. Tiếp đến, chúng ta sẽ sinh giá trị đầu ra mong muốn ứng với mỗi bộ đầu vào. Khi kết thúc bước này, chúng ta đã có các ca kiểm thử. Các kịch bản kiểm thử sẽ được thiết kế và thực thi nhằm phát hiện các lỗi/khiếm khuyết của sản phẩm bằng cách so sánh đầu ra thực tế với đầu ra mong đợi tương ứng của ca kiểm thử. Từ các kết quả kiểm thử, chúng ta sẽ quyết định hành động tiếp theo như sửa đổi mô hình hoặc dừng kiểm thử.

- Sinh mô hình dựa trên các yêu cầu và chức năng của hệ thống.
- Sinh các ca kiểm thử (bộ đầu vào và giá trị đầu ra mong đợi cho mỗi ca kiểm thử).
- Chạy các kịch bản kiểm thử để phát hiện các lỗi/khiếm khuyết của sản phẩm.
- So sánh kết quả đầu ra thực tế với kết quả đầu ra dự kiến.
- Quyết định hành động tiếp theo (sửa đổi mô hình, tạo thêm ca kiểm thử, dừng kiểm thử, đánh giá chất lượng của phần mềm) [1].



Hình 2.5: Quy trình kiểm thử dựa trên mô hình [1]

2.3.2 Thuận lợi và khó khăn của kiểm thử trên mô hình

Trong quá trình phát triển phần mềm, đối với phương pháp kiểm thử thủ công truyền thống thường tốn nhiều thời gian và chi phí. Kiểm thử tự động dựa trên mô hình là một giải pháp hiệu quả góp phần giải quyết vấn đề này.

- Kiểm thử dựa trên mô có các ưu điểm sau:
  - Giảm chi phí và thời gian: Do quá trình kiểm thử hầu hết được thực hiện tự động nên tính hiệu quả của phương pháp này rất cao trong khi thời gian được giảm một cách tối thiểu.
  - Độ bao phủ tốt hơn: Nếu mô hình của hệ thống được xây dựng tốt thì quá trình kiểm thử dựa trên mô hình sinh ra nhiều ca kiểm thử và phát hiện nhiều lỗi. Kiểm thử mô hình cũng cho phép giảm các lỗi chủ quan do người kiểm thử sinh ra trong quá trình kiểm thử sản phẩm.
  - Đầy đủ tài liệu: Mô hinh hệ thống, các đường đi, các ca kiểm thử là các tài liệu quan trọng trong quá trình phát triển phần mềm nói chung và quá trình kiểm thử phần mềm nói riêng. Các tài liệu này cũng giúp cho các kiểm thử viên hiểu hơn về các ca kiểm thử và các kịch bản kiểm thử.
  - Khả năng sử dụng lại cao: Mỗi khi phần mềm bị tiến hóa, chúng ta dễ dạng sinh thêm các ca kiểm thử và kiểm thử lại một cách nhanh chóng và hiệu quả.
  - Hiểu hơn về hệ thống: Kiểm thử dựa trên mô hình giúp người phát triển hiểu hơn về hệ thống cần kiểm thử thông qua việc xây dựng và phân tích mô hình hệ thống.
  - Sớm phát hiện lỗi và sự không ràng buộc trong đặc điểm kỹ thuật và thiết kế vì vậy sẽ tăng thời gian giải quyết vấn đề trong kiểm thử.
  - Tự động tạo và kiểm tra nhằm tránh các ca kiểm thử trùng nhau hoặc không hữu hiệu.
  - Kiểm thử dựa trên mô hình có khả năng đánh giá chất lượng phần mềm.
- Tuy nhiên, kiểm thử dựa trên mô hình không dễ được áp dụng trong thực tế vì một số khó khăn sau:
  - Khó xây dựng mô hình chính xác: Kiểm thử dựa trên mô hình cần có mô hình đặc tả chính xác hành vi của hệ thống. Trong thực tế, việc xây dựng mô hình là rất khó, tốn kém và tiềm ẩn nhiều lỗi.
  - Yêu cầu cao về kiểm thử viên: Do phải xây dựng mô hình của hệ thống vì vậy người kiểm thử phần mềm phải yêu cầu là những người có khả năng phân tích và thiết kế hệ thống. Hơn nữa, người kiểm thử cần có kiến thức tốt về các phương pháp hình thức và đặc tả hình thức, có hiểu biết chi tiết và chính xác về hệ thống.

- Tạo giá trị đầu ra mong đợi cho các ca kiểm thử là một trong những vấn đề khó khăn nhất của kiểm thử dựa trên mô hình.
- Khó khăn trong việc sử dụng các ca kiểm thử được tạo ra từ mô hình: Lập trình viên tiến hành cài đặt hệ thống một cách độc lập nên khi đã cài đặt xong thường khó thực thi các ca kiểm thử được tạo ra từ mô hình vì rất nhiều lí do khác nhau. Thông thường, họ phải tiến hành nghiên cứu mô hình và đặc tả lại các ca kiểm thử mới sử dụng được chúng. Hơn nữa, mô hình hệ thống thường trừu tượng và tổng quát hơn cài đặt của nó. Vấn đề này là một trong những lí do chính của hạn chế này [1].

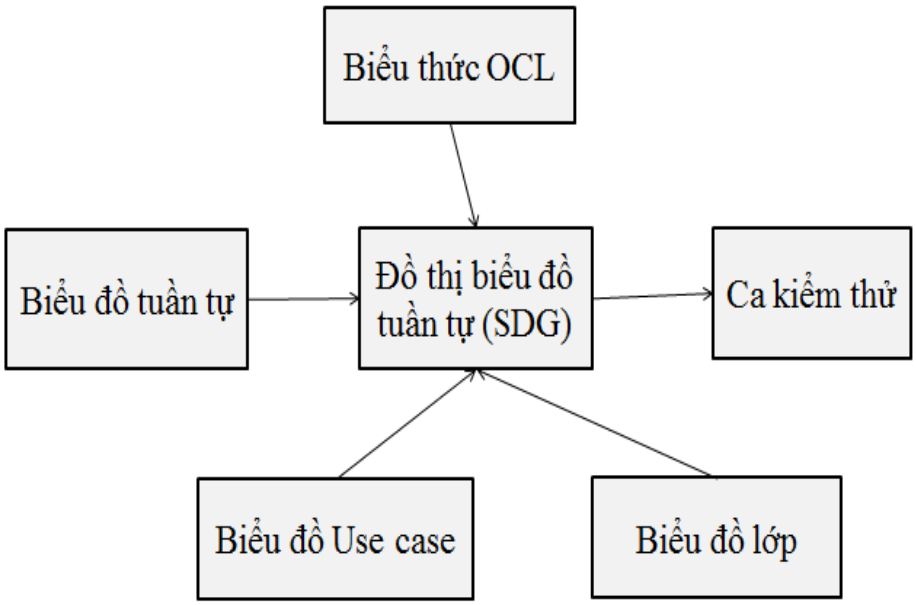
2.4 Một số phương pháp kiểm thử dựa trên mô hình

Để áp dụng phương pháp kiểm thử dựa trên mô hình, yêu cầu cần thiết là xây dựng mô hình đặc tả chính xác hành vi của hệ thống cần kiểm thử. Mô hình được đặc tả bằng một trong các phương pháp hình thức như: máy trạng thái UML, máy hữu hạn trạng thái, biểu đồ trạng thái, văn phạm, bảng quyết định,.. Từ các mô hình trên sẽ sinh các ca kiểm thử. Mỗi ca kiểm thử là một đường đi từ trạng thái bắt đầu đến trạng thái kết thúc của hệ thống. Mỗi ca kiểm thử phải có ít nhất một phép chuyển trạng thái. Trong mục này sẽ giới thiệu tổng quan một số phương pháp phổ biến sinh ca kiểm thử từ mô hình khác không phải là mô hình BPMN.

2.4.1 Sinh tự động ca kiểm thử từ biểu đồ UML và OCL.

Ngôn ngữ ràng buộc đối tượng (Object Constraint Language -OCL): là một ngôn ngữ cho phép mô tả các biểu thức và ràng buộc trên các mô hình hướng đối tượng và các vật thể mô hình hóa đối tượng khác. Trường hợp khác, OCL là liên quan đến điều kiện trước/ sau của một hoạt động. Nó có thể kết hợp các điều kiện OCL khác nhau thông tin điều khiển dòng chảy của một máy trạng thái. Các công thức trong các điều kiện đầu ra được diễn giải để cho phép thực hiện mô hình một cách tượng trưng.

Với biểu đồ ca sử dụng, biểu đồ lớp, biểu đồ tuần tự được biến đổi thành một biểu diễn được gọi là đồ thị biểu đồ tuần tự (Sequence Diagram Graph-SDG). Mỗi nút trong SDG chứa thông tin cần thiết cho việc tạo ra các ca kiểm thử, thông tin này được thu thập từ mẫu các ca kiểm thử, kết hợp với các OCL để tạo ra các ca kiểm thử dựa trên tiêu chí bao phủ toàn bộ các nhánh. Một sơ đồ sơ bộ của cách tiếp cận như thể hiện trong hình:



Hình 2.6: Sinh tự động ca kiểm thử từ biểu đồ UML và biểu thức OCL [5].

2.4.2 Sinh tự động ca kiểm thử từ biểu đồ tuần tự UML

Trong cách tiếp cận này tập trung vào tiêu trí bao phủ đường nhánh. Bộ các ca kiểm thử được tạo ra theo tiêu chí bao phủ đường nhánh hoạt động nhằm mục đích bao phủ các trường hợp lỗi như lỗi đồng bộ hoá, lỗi trong vòng lặp, lỗi trong các điều kiện rẽ nhánh [8].

Các biểu đồ hoạt động cũng được sử dụng để kiểm thử (testing) tính nhất quán giữa mã chương trình và thiết kế [6, 7]. Trong phương pháp kiểm tra hộp xám, các ca kiểm thử được tạo ra từ các mô hình thiết kế mức cao, mô tả cấu trúc và hành vi dự kiến của phần mềm đang được thử nghiệm. Đề xuất cho cách tiếp cận này được tạo ra các ca kiểm thử từ một biểu đồ hoạt động bao gồm ba bước sau:

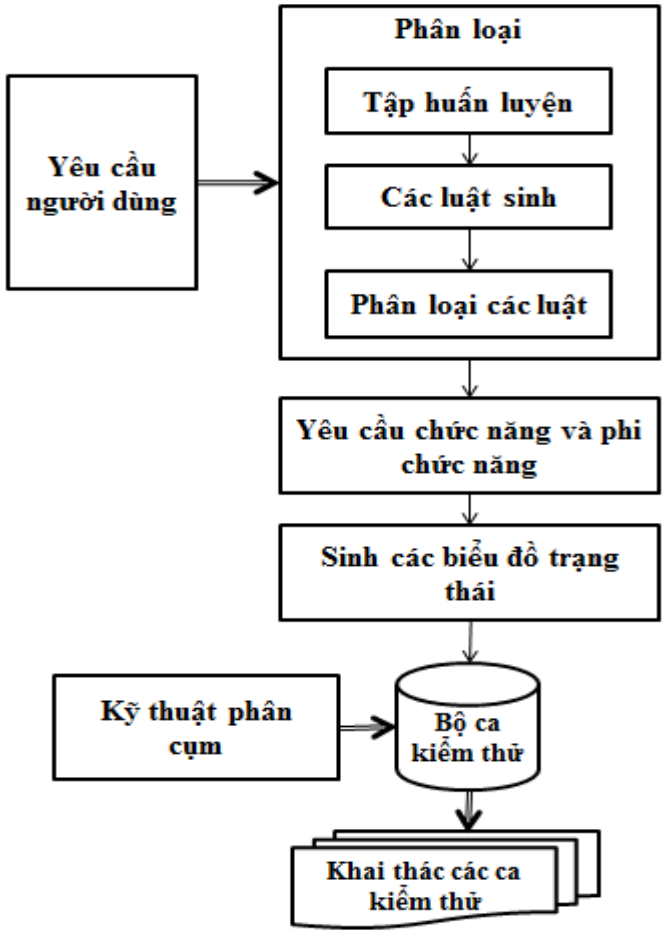
- ***Bước 1***: Bổ sung sơ đồ hoạt động với thông tin kiểm tra cần thiết: trạng thái đầu ra của sơ đồ hoạt động này sẽ là trạng thái đầu vào của sơ đồ hoạt động tiếp theo. Cũng giống như vậy, các sơ đồ lớp hoạt động lần lượt được kết nối với nhau vào một biểu đồ chung.
- ***Bước 2***: Chuyển đổi sơ đồ hoạt động thành một đồ thị hoạt động: Một đồ thị hoạt động là một đồ thị trực tiếp, trong đó mỗi nút trong biểu đồ hoạt động biểu diễn một cấu trúc (nút ban đầu, nút cuối cùng luồng, nút quyết định, nút phân nhánh, nút kết nối, nút kết hợp…), và mỗi cạnh biểu đồ hoạt động thể hiện dòng chảy trong sơ đồ hoạt động [8].
- ***Bước 3***: Sinh các trường hợp thử nghiệm từ biểu đồ hoạt động theo tiêu chuẩn bao phủ các đường nhánh.

2.4.3 Khai thác đáng tin cậy các trường hợp kiểm thử tự động từ đặc tả yêu cầu phần mềm.

Đây là một cách tiếp cận mới để sinh tự động tạo các ca kiểm thử từ đặc tả yêu cầu phần mềm - Software Requirement Specification (SRS). Để thực hiện được điều này thì đầu tiên là chuyển đổi một mô hình SRS chi tiết sang mô hình UML, thứ hai là tạo ra các ca kiểm thử từ mô hình UML và cuối cùng khai thác các ca kiểm thử.

Lợi thế là các ca kiểm thử có thể được tự động tạo ra từ các biểu đồ trạng thái này. Và nó cũng thể hiện một phương pháp để giảm bộ ca kiểm thử bằng cách sử dụng các phương pháp khai thác. Các thuật toán khai thác dữ liệu có thể được áp dụng ở các mức trừu tượng khác nhau và giúp người dùng khám phá nhiều mẫu ca kiểm thử có ý nghĩa hơn. Khai thác dữ liệu sẽ tạo ra các mẫu ca kiểm thử từ cơ sở dữ liệu đã tồn tại. Cách tiếp cận này như sau ba bước chính:

- Bước 1: Tạo ra các quy tắc phân loại.
- Bước 2: Tạo các trường hợp thử nghiệm từ máy trạng thái UML.
- Bước 3: Cuối cùng các kỹ thuật khai thác dữ liệu được áp dụng cho các ca kiểm thử được tạo ra để giảm kích thước bộ thử nghiệm hơn nữa [9].



Hình 2.7: Khai thác đáng tin cậy các trường hợp kiểm thử tự động từ SRS [9].

2.5 Tổng quan về mô hình hóa quy trình nghiệp vụ BPMN

2.5.1 Tổng quan về mô hình hóa quy trình nghiệp vụ

- ***Lịch sử và khái niệm***

Thuyết quản lý quy trình nghiệp vụ có nguồn gốc từ các học thuyết chất lượng những năm 70. Vào năm 1977, một mô tả quy trình đầu tiên quan trọng của nhà kinh tế Adam Smith trong ví dụ nổi tiếng của ông về một nhà máy sản xuất pin, từ đó Adam Smith đã đưa ra ý tưởng về sự chuyên môn hóa lao động.

Vào đầu thế kỷ XX, kỹ sư người Mỹ là Frederick Winslow Taylor đã tập trung vào việc chuẩn hóa các quy trình, đào tạo có hệ thống và xác định rõ vai trò của quản lý và nhân viên thông qua các nguyên tắc được trình bày thông qua cuốn sách "Principles of Scientific Management".

Tiếp theo học thuyết nổi tiếng của Taylor, nhà sáng lập hãng Ford Motor là Henry Ford đã mở rộng khái niệm chuyên môn hóa lao động và đưa ra quy định về trình tự để hoàn thành công việc. Khái niệm này cho phép quá trình sản xuất hàng loạt và đưa ra được các thông số kỹ thuật của quá trình sản xuất.

Đến khoảng cuối thế kỷ XX, chuyên gia quản lý tập trung Perter Drucker đã tập trung vào việc đơn giản hóa và phân cấp các quá trình, dẫn đến khái niệm về gia công phần mềm (outsourcing).

Qua quá trình trên, quy trình nghiệp vụ được hiểu là một tập hợp các hoạt động có liên quan hoặc có cấu trúc được thực hiện dưới sự kết hợp các hành vi của con người, hệ thống, thông tin và những yếu tố khác để tạo ra kết quả theo mục tiêu định trước. Với tốc độ phát triển khoa học công nghệ áp dụng vào quá trình sản xuất ngày càng nhanh. Quản lý quy trình nghiệp vụ trở thành thách thức cho các tổ chức nhằm đạt được hiệu suất và khả năng cạnh tranh.

- ***Quản lý quy trình nghiệp vụ***

Quản lý quy trình nghiệp vụ (BPM) là một phương pháp được thiết kế để kiểm soát việc thực hiện quy trình nghiệp vụ nhằm cải thiện các quy trình nghiệp vụ thông qua sự kết hợp của công nghệ và nghiệp vụ. Quản lý quy trình nghiệp vụ là một mô hình làm việc kết hợp giữa các bên liên quan gồm: bộ phận kinh doanh, nghiệp vụ và bộ phận công nghệ thông tin cùng nỗ lực để làm cho các quy trình nghiệp vụ hiệu quả và tối ưu hơn.

Quản lý quy trình nghiệp vụ mang lại giá trị cho doanh nghiệp thông qua cải thiện năng suất tốt hơn, hiệu quả nhân viên cao hơn, dịch vụ khách hàng tốt hơn dẫn tới giảm chi phí và tăng lợi nhuận. Tất cả lợi ích này là kết quả trực tiếp từ việc cải thiện quy trình làm việc. Quy trình nghiệp vụ là một hoạt động hoặc một tập các hoạt động nhằm đạt được một mục tiêu cụ thể của tổ chức. Quản lý

quy trình nghiệp vụ (BPM) là một cách tiếp cận có hệ thống để cải tiến các quy trình đó.

- ***Mô hình hóa quy trình nghiệp vụ***

Quy trình nghiệp vụ thường được mô tả trực quan bằng sơ đồ luồng cho thấy chuỗi các hoạt động hoặc nhiệm vụ với một số điểm chuẩn hoặc điểm quyết định nhất định. Một số cách biểu diễn mô hình hóa luồng nghiệp vụ:

- Quy trình nghiệp vụ tuần tự: Các quy trình tuần tự được thể hiện trên một tài liệu với một điểm bắt đầu và điểm kết thúc rõ ràng. Theo quy trình này, một tổ chức sẽ thực hiện một loạt các hành động để hoàn thành nhiệm vụ trong điều kiện thời gian ràng buộc.
- Quy trình nghiệp vụ theo trạng thái: Quy trình nghiệp vụ theo trạng thái không có điểm bắt đầu và điểm kết thúc nghiêm ngặt. Các quy trình này có thể hoàn thành ở bất kỳ giai đoạn nào tùy thuộc vào sự thay đổi của luồng công việc. Ngoài ra, điển hình cho luồng quy trình điều khiển theo trạng thái là sự lặp lại có tính chất chu kỳ trên cùng một bước trong quy trình.
- Quy trình nghiệp vụ song song: các hoạt động được thực hiện song song đồng thời. Đối với luồng quy trình này các hoạt động trên tất cả chi nhánh phải được hoàn thành trước khi bước tiếp theo trong quá trình kinh doanh có thể bắt đầu.

2.5.2 Mô hình hóa quy trình nghiệp vụ với BPMN

Ký pháp và mô hình quy trình nghiệp vụ (BPMN) là một tiêu chuẩn mô hình hóa ở dạng đồ họa để xác định các quy trình nghiệp vụ trong một sơ đồ quy trình nghiệp vụ. Tổ chức sáng kiến quản lý quy trình nghiệp vụ (BPMI) đã phát triển BPMN, được duy trì bởi nhóm quản lý đối tượng (OMG) kể từ khi hai tổ chức được sáp nhập vào năm 2005. Phiên bản đầu tiên BPMN v1.0 được phát hành năm 2004. Phiên bản gần đây nhất là BPMN v2.0.2 được OMG công bố tháng 12/2013 với nhiều thay đổi và mở rộng hơn so với phiên bản cũ.

Ký pháp và mô hình hóa quy trình nghiệp vụ BPMN là một tiêu chuẩn cho các mô hình quy trình nghiệp vụ cung cấp một ký hiệu đồ hoạ cho việc thể hiện các quy trình nghiệp vụ trong một sơ đồ quy trình nghiệp vụ.

Mục tiêu của BPMN là hỗ trợ quản lý quy trình nghiệp vụ, cho cả đối tượng xử lý kỹ thuật và đối tượng người dùng nghiệp vụ bằng cách cung cấp một tập ký hiệu trực quan cho người dùng, nhưng có thể biểu diễn các ngữ nghĩa quy trình phức tạp. Đặc tả BPMN cũng cung cấp một ánh xạ giữa ký hiệu đồ hoạ và các cấu trúc cơ bản của các ngôn ngữ thực thi được, đặc biệt là ngôn ngữ thực thi quy trình nghiệp vụ (BPEL).

2.5.3 Các phần tử (element) của BPMN

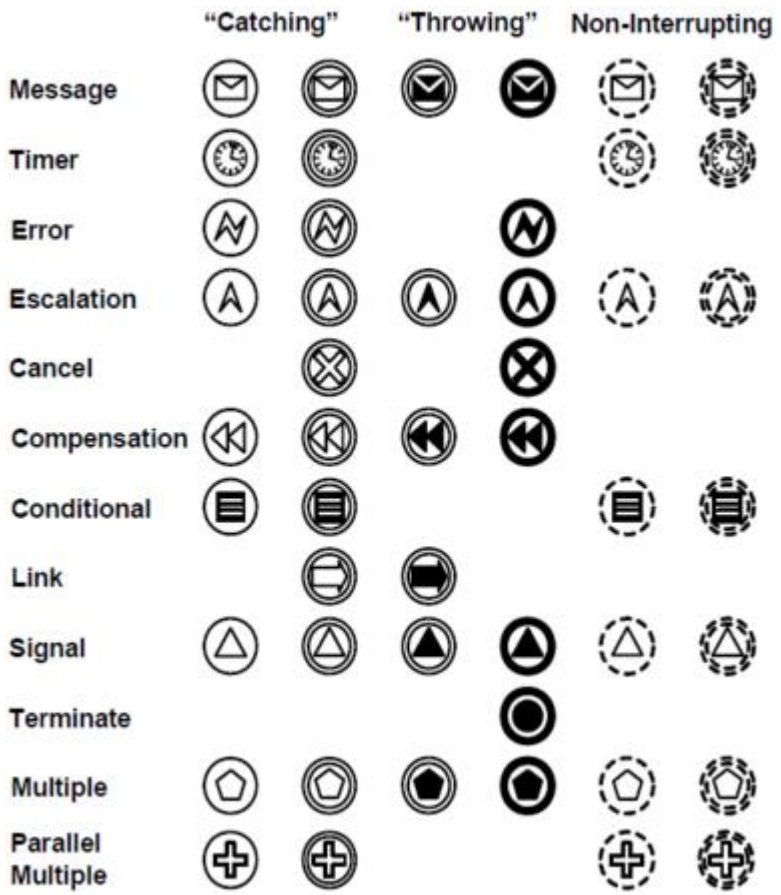
Các phần tử của BPMN được phân thành 5 loại cơ bản sau:

2.5.3.1 Flow Object

Là các yếu tố đồ họa chính định nghĩa hành vi của quy trình nghiệp vụ. Có ba đối tượng luồng gồm: event, activity, gateway.

- ***Event:*** Một sự kiện là nguyên nhân, tác động đòi hỏi hoặc cho phép một phản ứng xảy ra trong quy trình. Những sự kiện ảnh hưởng đến luồng xử lý quy trình thường có nguyên nhân (kích hoạt) và tác động (kết quả). Có 03 loại event dựa trên sự tác động của chúng lên flow: start, intermediate và end.

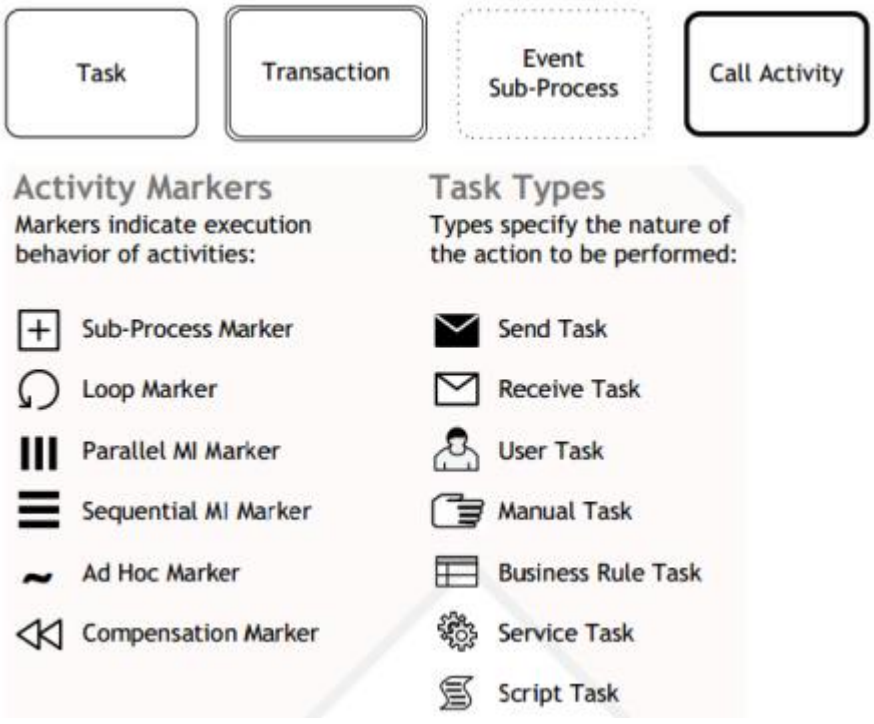
- Start Event: sự kiện bắt đầu một quy trình
- End Event: sự kiện kết thúc của một quy trình
- Intermediate Event: sự kiện xảy ra trong quy trình có tác động đến quy trình. Các intermediate event trong BPMN2.0



Hình 2.8: Các ký pháp của các kiểu Intermediate Events trong BPMN [14]

- ***Activities:*** Là công việc được thực hiện trong quy trình nghiệp vụ. Một hoạt động có thể là nguyên tử hoặc phi nguyên tử. Các kiểu activities gồm: task, transaction, sub-process và call activity

- Task: là một đơn vị công việc cơ bản nhất được thực hiện, một task được sử dụng khi công việc trong quy trình không thể chia nhỏ đến mức chi tiết hơn. Khi task được đánh dấu bởi ký hiệu [+] nó khởi tạo một quy trình con, một hoạt động có thể được làm mịn.
- Transaction: là một tập hợp các hoạt động có tính chất logic, có thể cùng để thực hiện theo một quy định giao thức giao dịch.
- Sub process: Quy trình con là một activity mà các chi tiết nội bộ đã được mô hình hóa bằng cách sử dục cac activities, gateways, events và sequece flows. Một sub process là một đối tượng đồ họa trong một quy trình, nhưng cũng có thể là một quy trình cấp thấp hơn. Các quy trình con xác định một phạm vi ngữ cảnh có thể được sử dụng cho khả năng hiển thị thuộc tính, phạm vi giao dịch, để xử lý trường hợp ngoại lệ, sự kiện hoặc bồi thường.
- Call Activity: Hành động đóng gói tổng thể định nghĩa các tác vụ quy trình tái sử dụng được đánh dấu bởi ký hiệu [+]



Hình 2.9: Ký pháp các kiểu Activity trong BPMN [14]

- ***Gateways***: được sử dụng để kiểm soát sự phân kỳ và hội tụ của chuỗi hoạt động. Tại đây các hoạt động sẽ được kết hợp hoặc phân nhánh thành các luồng xử lý trong quy trình.

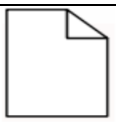
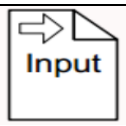
Bảng 2.1: Bảng danh sách các kiểu Gatewway trong BPMN

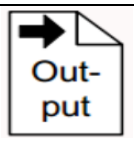
Kiểu gateway	Ý nghĩa	Ký pháp
Inclusive Gateway	Được sử dụng để tạo ra các hướng đi khác nhau tại cùng một cổng.	
Exclusive Gateway	Được sử dụng tạo các luồng thay thế trong quy trình và duy nhất một luồng được thực hiện.	
Parallel Gateway	Được sử dụng để tạo các luồng song song mà không cần đánh giá bất kỳ điều kiện nào.	
Complex Gateway	Được sử dụng để mô hình hóa hành vi đồng bộ phức tạp.	
Event-based Gateway	Điều kiện xác định luồng thực hiện của quy trình dựa trên một sự kiện được đánh giá.	
Parallel Event-based Gateway	Hai quá trình song song được bắt đầu dựa trên một sự kiện, nhưng không có đanh giá của sự kiện.	
Exlusive Event-based Gateway	Một sự kiện đang được đánh giá để xác định đường đi nào sẽ bị loại trừ lẫn nhau.	

2.5.3.2 Data

Đối tượng Data cung cấp tài nguyên cần thiết cho quy trình, thể hiện qua một đối tượng đơn lẻ hoặc một tập đối tượng. Data biểu diễn với bốn phần tử: Data object, Data input Data Output, Data Store.

Bảng 2.2: Bảng danh sách các kiểu Data trong BPMN

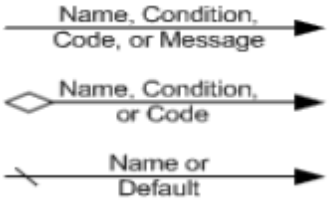
Kiểu data	Ý nghĩa	Ký pháp
Data object	Biểu diễn luồng thông tin trong tiến trình như tài liệu nghiệp vụ hoặc thư điện tử.	
Data input	Là một kiểu tham số đầu vào cho các quy trình, tiến trình.	

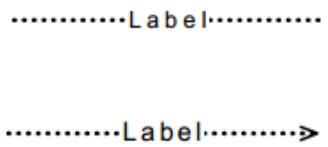
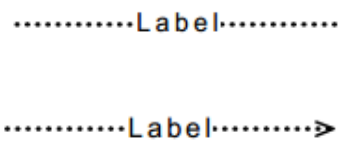
Kiểu data	Ý nghĩa	Ký pháp
Data output	Là một kiểu tham số đầu ra, là kết quả của quy tình, tiến trình.	
Data store	Là nơi lưu trữ dữ liệu nơi mà tiến trình có thể đọc, ghi dữ liệu như cơ sở dữ liệu.	

2.5.3.3 Connection Object

Các connection object có vai trò liên kết các thành phần trong luồng quy trình nghiệp vụ. Connection object có các loại liên kết cơ bản để kết nối các đối tượng với nhau hoặc với thông tin khác gồm: Sequence flow, Message flow, Asociation và data association.

Bảng 2.3: Bảng danh sách các connection object cơ bản trong BPMN

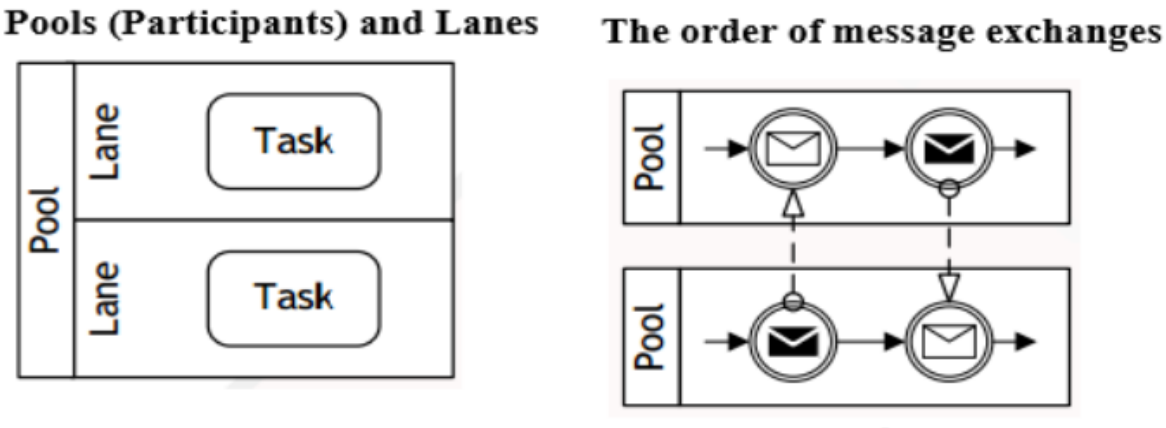
Kiểu Connection	Ý nghĩa	Ký pháp
Sequence Flow	Được sử dụng để thể hiện thứ tự các hoạt động sẽ được thực hiện trong quy trình. Một sequence flow được thể hiện bởi một đường thẳng rắn/liên tục (solid line) với một đầu mũi tên rắn (solid arrowhead) và được sử dụng để hiển thị theo thứ tự (the sequence) các hoạt động sẽ được thực hiện trong một quy trình Gồm 03 loại: sequence flow, normal/default flow, condition flow - Normal follow: Đề cập đến những đường đi Sequence Flow mà không bắt đầu từ các intermediate Event gắn liền với biên của các hoạt động. - Condition Flow: có một điều kiện chỉ định xem luồng xử lý đó có	

Kiểu Connection	Ý nghĩa	Ký pháp
	được thực hiện hay không - Exception Flow: Xảy ra bên ngoài luồng xử lý thông thường của quy trình dựa trên một intermidiaate event gắn liền với biên của activity trong suốt hiệu suất của quy trình	
Message flow	Biểu diễn luồng thông tin trao đổi giữa các thành phần trong quy trình nghiệp vụ. Một Message Flow được thể hiện bởi một đường ngang nhiều nét gạch đứt (dashed line) và được sử dụng để hiển thị luồng thông điệp của hai người tham gia quy trình riêng biệt (các thực thể hoặc các vai trò nghiệp vụ) đó là gửi và nhận chúng.	
Association	Dùng để liên kết giữa Flow Node và Artifacts, hiển thị đầu vào/đầu ra của các hoạt động. Một Association được thể hiện bởi một đường ngang nhiều chấm (dotted line) với một đầu mũi tên đóng, được sử để kết hợp dữ liệu, văn bản, và các Artifacts khác với Flow Objects.	
Data Association	Được sử dụng để di chuyển dữ liệu giữa Data objects, Properties, và đầu vào/đầu ra của các activities, quy trình và global tasks. Mục đích thu thập dữ liệu từ data object hoặc đàu vào dữ liệu tiến trình là dữ liệu đầu vào cho các hoạt động và sau đó đẩy các giá trị đầu ra từ việc thực hiện hoạt động trở lại data object hoặc đầu ra dữ liệu quá trình.	

2.5.3.4 Swimlanes

Có hai cách thức để nhóm các phần tử mô hình hóa chính thông qua Swimlanes là Pool và Lane. Trong đó:

- ***Pool:*** là biểu diễn đồ họa của một thành phần tham gia trong một Collaboration và một hình chứa tập các hoạt động từ các pool khác nhau. Một pool có thể có chi tiết nội bộ trong một quy trình được thực thi.
- ***Lane***: là một phân vùng thuộc một Process (đôi khi thuộc một Pool). Một lane đôi khi là thuộc một pool, là một phân vùng của pool và sẽ được mở rộng theo toàn bộ chiều dài của pool, hoặc theo chiều dọc hoặc chiều ngang. Các lane được sử dụng để tổ chức và phân loại các hoạt động.

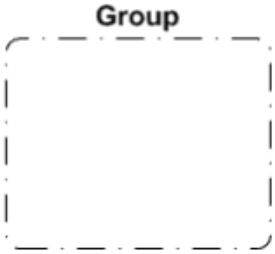


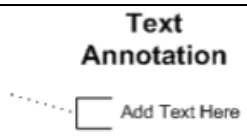
Hình 2.10: Các thành phần trong Swim Lane và Group [14]

2.5.3.5 Artifacts

Được sử dụng để cung cấp thông tin bổ sung về mô hình/quy trình. Có hai artifact tiêu chuẩn nhưng những nhà mô hình hóa hay công cụ mô hình hóa có thể tự do thêm các Artifact khi cần thiết. Hai artifact tiêu biểu gồm: Group và Text Annotation.

Bảng 2.4: Bảng danh sách các Artifacts trong BPMN

Kiểu artifact	Ý nghĩa	Ký pháp
Group	Là một nhóm các ký hiệu thành phần đồ họa cùng loại, không phân biệt trình tự thực hiện cũng như các luồng tuần tự trong một group, tên phân loại được biểu diễn trong biểu đồ như là nhãn của group.	

Kiểu artifact	Ý nghĩa	Ký pháp
	Group được thể hiện bởi một hình chữ nhật tròn góc được vẽ bởi một đường nhiều gạch nối. Nhóm này có thể sử dụng phân tích hoặc làm tài liệu, mà không ảnh hưởng đến sequence flow.	
Annotation	Các Annotation là 1 cơ chế cho Modeler để cung cấp thông tin văn bản bổ sung cho người đọc 1 sơ đồ BPMN	

2.5.4 Các mô hình thành phần của BPMN

Mô hình hóa quy trình nghiệp vụ được sử dụng để truyền tải một lượng lớn các thông tin một cách trực quan đến các bên liên quan. BPMN được thiết kế bao gồm nhiều kiểu mô hình hóa và cho phép việc tạo ra các quy trình nghiệp vụ điểm-điểm. Có 3 kiểu mô hình thành phần cơ bản trong mô hình BPMN điểm-điểm:

- ***Processes hay Orchestration***: bao gồm các quy trình nội bộ và quy trình bên ngoài có sự tương tác, kết nối với nhau.
  - Quy trình nghiệp vụ nội bộ là những quy trình nội bộ của một tổ chức cụ thể. Những quy trình này có thể được gọi chung là luồng công việc hay quy trình BPM. Một từ đồng nghĩa thường được sử dụng trong các dịch vụ Web là điều phối (Orchestration) các dịch vụ. Có 2 loại quy trình nghiệp vụ riêng là: Quy trình riêng không thể thực thi (phục vụ mục đích tài liệu hóa hành vi của quy trình) và quy trình nghiệp vụ riêng có thể thực thi (l phục vụ mục đích được thực thi theo ngữ nghĩa xác định).
  - Quy trình công khai thể hiện các tương tác giữa một quy trình riêng với một quy trình khác hoặc với một thành phần tham gia. Quy trình công khai chỉ những hoạt động được sử dụng để giao tiếp với một thành phần tham gia khác . Tất cả các hoạt động nội bộ của quy trình riêng đều không được biểu diễn trong quy trình công khai. Do vậy, quy trình công khai hiển thị cho bên ngoài biết luồng thông điệp (Message Flow) và thứ tự của luồng thông điệp đó để phục vụ tương tác với chính quy trình.
- ***Collaborations***: mô hình cộng tác mô tả các tương tác giữa hai hay nhiều thực thể nghiệp vụ. Một mô hình cộng tác thường chứa hai hoặc nhiều Pool đại diện cho các thự thể tham gia tương tác với nhau. Mỗi pool có thể có hoặc không

chứa quy trình bên trong. Thông tin trao đổi được thể hiện bởi một luồng thông điệp kết nối giữa hai Pool.

- ***Choreography***: Là mô hình điều phối theo trình tự định sẵn với kết quả mong đợi, là một định nghĩa về hành vi được mong đợi, về cơ bản là một bản thủ tục giữa các thành phần tham gia tương tác. Trong khi một mô hình quy trình thông thường tồn tại trong một Pool thì một mô hình Choreography tồn tại giữa các Pool. Choreography giống như một quy trình nghiệp vụ riêng do nó bao gồm nhiều hoạt động, sự kiện và cổng (Gateway). Tuy nhiên, điểm khác biệt của một Choreography ở chỗ các hoạt động là những tương tác thể hiện một tập (một hoặc nhiều) trao đổi thông điệp liên quan đến hai hay nhiều thành phần tham gia. Ngoài ra, điểm khác biệt nữa so với một quy trình thông thường là ở Choreography không có thành phần điều khiển trung tâm, không có thực thể chịu trách nhiệm cũng như không có người quan sát như Quy trình.

2.5.5 Các điều kiện ràng buộc thiết kế BPMN

Để đảm bảo dữ liệu thiết kế được đưa vào là phù hợp với chương trình kiểm thử Thì yêu cầu đầu tiên là phải đảm bảo chất lượng, tính đúng đắn của mô hình nghiệp vụ đầu vào (BPMN) theo các tập luật và ràng buộc sau:

- Ràng buộc cho một số lớp khái niệm Event
  - StartEvent không có luồng xử lý đầu vào
  - EndEvent không có luồng xử lý đầu ra
  - IntermediateEvent nhất định phải có luồng xử lý đầu ra hay phải là nguồn của một luồng xử lý nào đó
    Trong lưu đồ quy trình tối thiểu phải có một ký hiệu sự kiện khởi tạo và một ký hiệu sự kiện kết thúc luồng quy trình.
  - Mỗi ký hiệu mô tả hoạt động cần có ít nhất một luồng vào và một luồng ra
- Ràng buộc cho các lớp khái niệm Gateway
  - InclusiveGateway và ExclusiveGateway có luồng mặc định
  - InclusiveGateway có tối thiểu một luồng xử lý đầu vào
  - ExclusiveGateway có tối thiểu một luồng xử lý đầu ra
  - Luồng xử lý đầu ra của ParallelGateway không phải là luồng điều kiện
  - Một điểm phân nhánh/ hợp nhánh khi dùng để phân nhánh thì cần ít nhất một luồng vào và hai luồng ra, khi dùng để hợp nhánh thì cần ít nhất hai luồng vào và một luồng ra.
  - Các thành phần hoạt động, sự kiện, điểm phân nhánh/ hợp nhánh phải nằm trên ít nhất một luồng bắt đầu từ một sự kiện khởi tạo và kết thúc bởi một sự kiện kết thúc.

- Tránh các mẫu thiết kế sau trong BPMN: Đây là một số mẫu phổ biến có tính chất nhập nhằng, gây khó hiểu đối với người sử dụng.
  - Mẫu bế tắc: dựa theo tính chất đặc tả BPMN cho phần tử Gateway, quy trình sẽ không thể được hoàn thành nếu mô hình được thiết kế đầu vào của một phần tử Parallel Gateway là đầu ra của phần tử Exclusive Gateway.
  - Mẫu nhiều điểm kết thúc: ngược lại với mẫu bế tắc, một mô hình mà đầu ra của Parallel Gateway là đầu vào của phần tử Exclusive Gateway.
  - Mẫu thiết kế dựa vào tài liệu đặc tả có thể dễ dàng hiểu được thứ tự thực hiện tác vụ tuy nhiên nếu từ một tác vụ luồng dữ liệu đồng thời đi đến hai tác vụ thì người sử dụng không thể biết được trình tự thực hiện các tác vụ.
  - Mẫu thiết kế có vòng lặp: do các nghiệp vụ có thể được xử lý và có sự sửa đổi trong quy trình nên việc tác vụ có thể được thực hiện nhiều lần do đó vòng lặp trong quy trình là không thể tránh khỏi. Vì vậy việc giảm thiểu vòng lặp xảy ra, ảnh hưởng đến quy trình chúng ta cần đưa ra một số điều kiện ràng buộc cho các vòng lặp.

2.5.6 Công cụ thiết kế và thực thi mô hình BPMN

Hiện nay có rất nhiều tổ chức đã phát triển công cụ hỗ trợ thiết kế mô hình quy trình nghiệp vụ. Đến thời điểm tháng 9/2017 trên website chính thức của BPMN thống kê có 65 tools hỗ trợ thiết kế BPMN bao gồm cả công cụ Open source, Free và bản thương mại. Trong phần này sẽ giới thiệu 03 công cụ quản lý quy trình nghiệp vụ phổ biến là: Visio, Bizagi modeler và Activiti.

2.5.6.1 Công cụ MS Visio

MS Visio là một phần mềm văn phòng đã quen thuộc với nhiều người, nhất là những người làm quy trình. Đa số mọi người đều quen việc sử dụng biểu đồ Flow Chart để lưu đồ hóa quy trình. Ngày nay, phương pháp BPM đang ngày càng được đánh giá cao và áp dụng rộng rãi tại nhiều doanh nghiệp. Có nhiều phần mềm ra đời để hỗ trợ việc lưu đồ hóa quy trình theo chuẩn BPMN, và Visio cũng không nằm ngoài xu thế khi hỗ trợ lưu đồ hóa theo chuẩn này. Phiên bản Visio 2010 hỗ trợ BPMN 1.2. Và từ phiên bản Visio 2013 đã hỗ trợ BPMN 2.0.2 hiện đang được dùng phổ biến nhất.

- Thiết kế BPMN: Công cụ MS Visio đã dựng sẵn Template về BPMN cho người dùng dễ dàng sử dụng. Khi mở BPMN Template, Visio sẽ hiển thị tập hợp 16 ký hiệu cơ bản của BPMN ở phía bên trái, và bên phải là màn hình để vẽ lưu đồ. Khi muốn thêm một ký hiệu vào lưu đồ, ta chỉ click vào ký hiệu đó và kéo thả vào màn hình vẽ lưu đồ.

- Kiểm tra lưu đồ theo các quy tắc của BPMN: MS Visio cho phép kiểm tra lại lưu đồ đã vẽ đã đúng theo các quy tắc của BPMN thông qua tính năng Check Diagram. Trường hợp có lỗi thì công cụ hiển thị các lỗi bên dưới màn hình lưu đồ, người dùng có thể xem chi tiết vào từng thông báo lỗi và hướng xử lý.
- Tạo quy trình con từ một nhóm các Task có sẵn: Visio có chức năng cho phép ta tạo một quy trình con từ một nhóm các thành phần có sẵn. Khi chọn một tập hợp các hoạt động trong quy trình chon chức năng tạo quy trình con thì các thành phần trong Group sẽ được thay thế bằng một ký hiệu của Sub-process (quy trình con).

2.5.6.2 Công cụ Bizagi

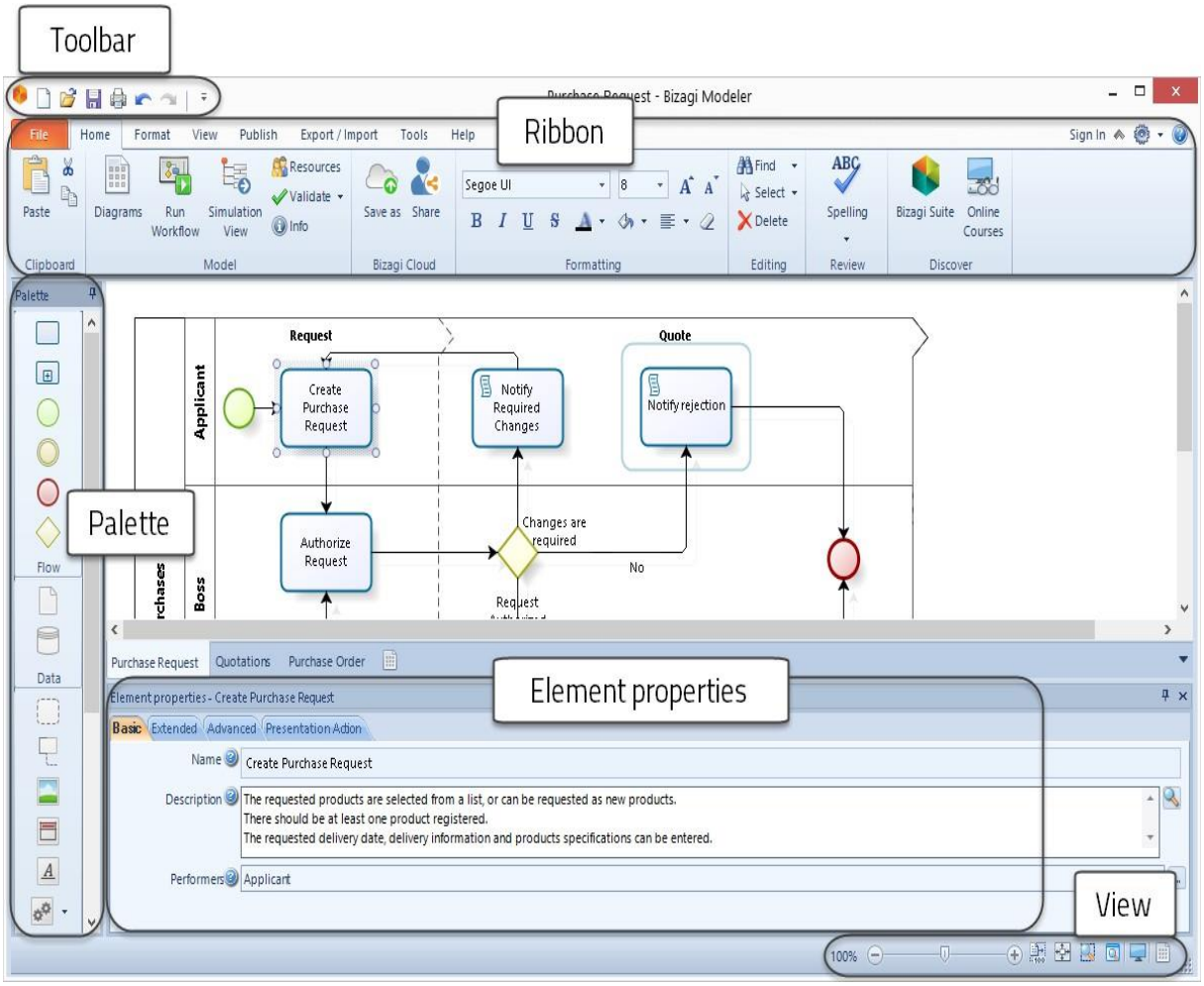
Bizagi BPM Suite là một bộ gồm 3 sản phẩm: Bizagi Modeler, Bizagi Studio, Bizagi Engine. Trong đó:

- **Bizagi *Modeler*** là một công cụ miễn phí dùng để vẽ lưu đồ và tài liệu hóa quy trình được phát triển bởi công ty Bizagi. Công cụ này cho phép chúng ta vẽ các lưu đồ và tài liệu hóa quy trình của doanh nghiệp một cách trực quan với định dạng tiêu chuẩn BPMN. Bizagi Modelr có giao diện thân thiện, gần gũi với người dùng như các phần mềm trong bộ Office của Microsoft. Một model có thể chứa một hoặc nhiều lưu đồ. Một file được hiểu là một model, và có thể lưu ở 2 định dạng là .bpm (để làm việc cá nhân) hoặc .bpmc (để làm việc trong nhóm hợp tác). Bizagi Modeler cho phép xuất bản tài liệu quy trình ở các định dạng Word, PDF, SharePoint hoặc Wiki với chất lượng cao. Các quy trình cũng có thể dễ dàng import từ hoặc export sang Visio hoặc XML và một số công cụ khác.

- ***Bizagi Studio và Bizagi Engine*** là công cụ để tự động hóa mô hình và chuyển mô hình sang một hệ thống có thể thực thi quy trình. Bizagi Studio cho phép chúng ta nhập mọi thông tin cần thiết cho việc thực thi quy trình như: thời gian tiêu chuẩn, chi phí, giao diện người dùng, quy tắc. Các thông tin này được lưu lại như một model trong một database và được sử dụng khi chạy bởi Bizagi Engine khi thực thi quy trình thông qua một cổng làm việc cho người dùng cuối.

- ***Giao diện người dùng***

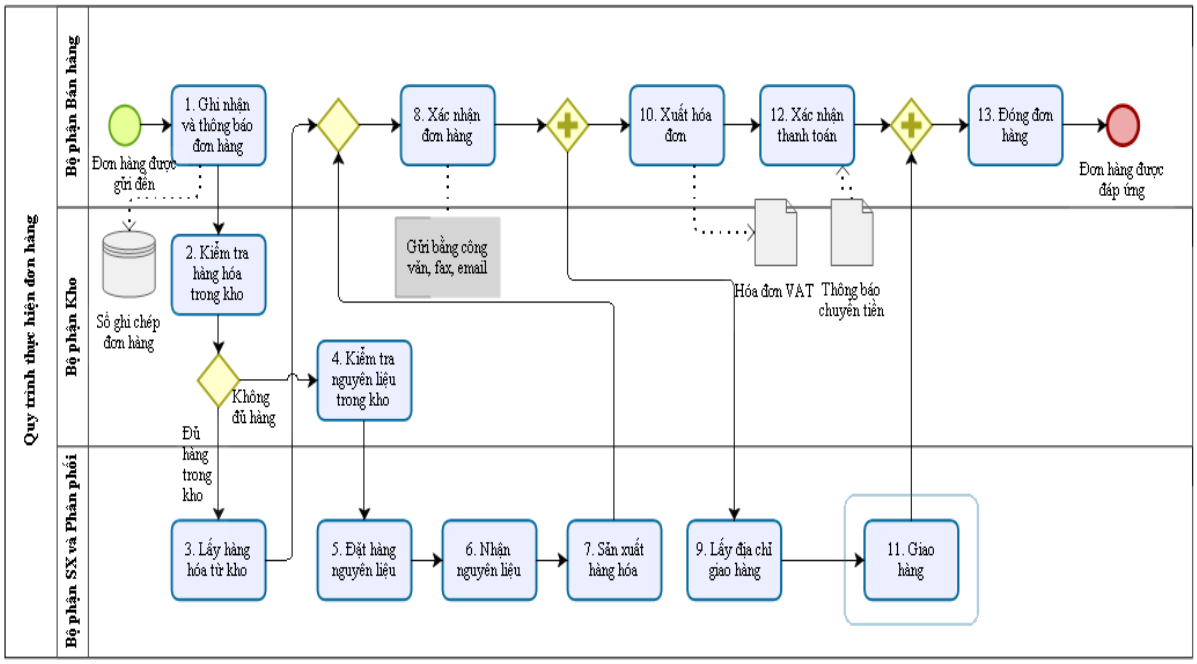
Bizagi Modeler có giao diện người dùng rất đơn giản, dễ dàng sử dụng và rất trực quan, quen thuộc. Nó có 5 thành phần chính là: Toolbar, Ribbon, Palette, Element Properties và View.



Hình 2.11: Giao diện người dùng trong Bizagi Modeler

- ***Một số tính năng của Bizagi Modeler***
  - Nhóm một số các thành phần trong lưu đồ lại với nhau bằng cách kéo biểu tượng Group (nhóm) từ palette đặt vào vị trí mong muốn, kéo hoặc thay đổi kích cỡ group theo ý.
  - Ta có thể thêm các chú thích bằng cách dùng Annotation , hay các đoạn mô tả bằng cách dùng Formatted Text để bổ sung thêm thông tin cho lưu đồ.
  - Bizagi cho phép bổ sung thêm các thuộc tính bất kỳ cho các thành phần của lưu đồ.
  - Nếu gắn một Event (sự kiện) vào một Task hay Sub-process thì sẽ cần vẽ thêm luồng ngoại lệ khi gặp Event đó.
  - Tab Basic cho phép ta sửa tên, viết miêu tả tóm tắt và chọn người thực hiện cho hoạt động đó từ danh sách người thực hiện đã được định nghĩa.

Ví dụ minh họa quy trình thực hiện đơn hàng thiết kế trên Bizagi:

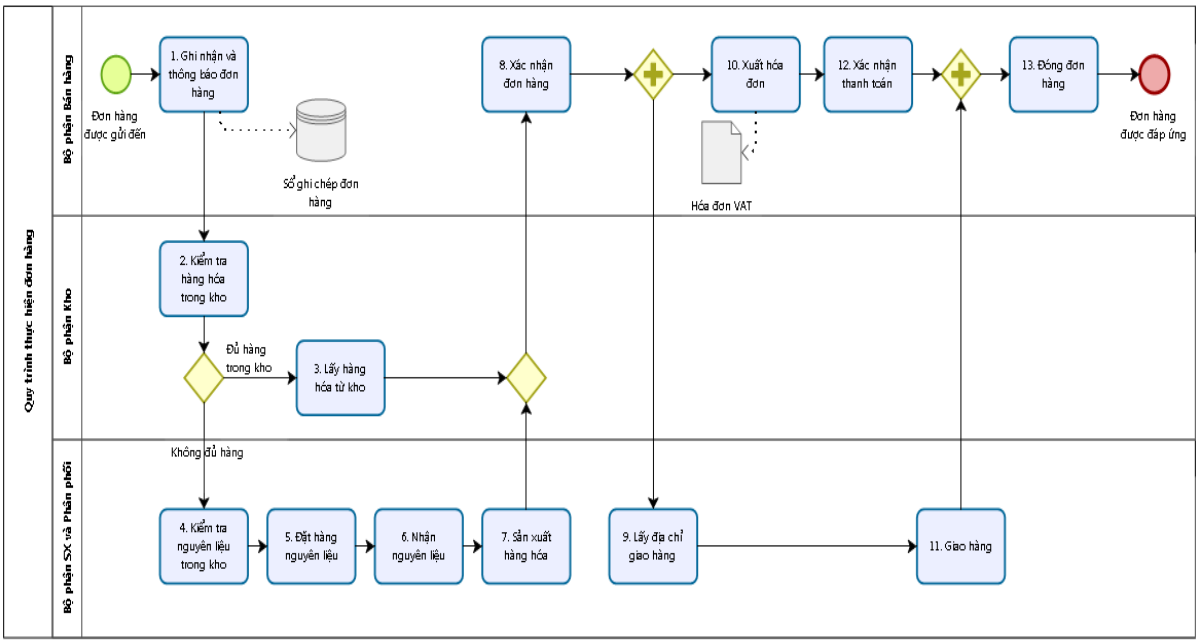


Hình 2.12: Quy trình thực hiện đơn hàng thiết kế trên Bizagi

- ***Tinh chỉnh và kiểm tra lưu đồ:*** Sau khi đã vẽ xong luồng kiểm soát quy trình cần tiến hành tinh chỉnh lưu đồ để đảm bảo tính chính xác của trình tự thực hiện, tuân thủ đúng theo quy tắc của ngôn ngữ BPMN và tính mỹ quan, dễ đọc dễ theo dõi. Có thể thực hiện một số bước nhỏ như sau:

- Sử dụng tính năng kiểm tra lưu đồ của công cụ để kiểm tra, phát hiện các lỗi. Nếu có thì xem lỗi chi tiết ở phần nào và thực hiện chỉnh sửa cho đến khi hết lỗi.
- Kiểm tra xem lưu đồ được vẽ đầy đủ các hoạt động chưa, thứ tự thực hiện có theo đúng luồng tuần tự, vẽ ở đúng các Lane, được đặt tên hay chưa.
- Kiểm tra các ký hiệu mô tả các thành phần đã được vẽ đúng chưa, đã kết nối với nhau đúng chưa. Các thành phần cùng loại có được đặt tên theo một cách thức giống nhau chưa.
- Xóa bỏ các thành phần thừa nếu có. Xóa bỏ các ký tự bổ sung nếu nhiều quá và làm rối lưu đồ.
- Có thể thay đổi vị trí của các thành phần để hạn chế các mũi tên cắt nhau, đảm bảo tính dễ nhìn, dễ theo dõi và mỹ quan của lưu đồ.
- Kiểm tra lại các điểm phân nhánh/ hợp nhánh đã sử dụng đủ, đúng loại điểm phân nhánh/ hợp nhánh chưa.

Ví dụ mô hình quy trình thực hiện đơn hàng sau khi đã tinh chỉnh:



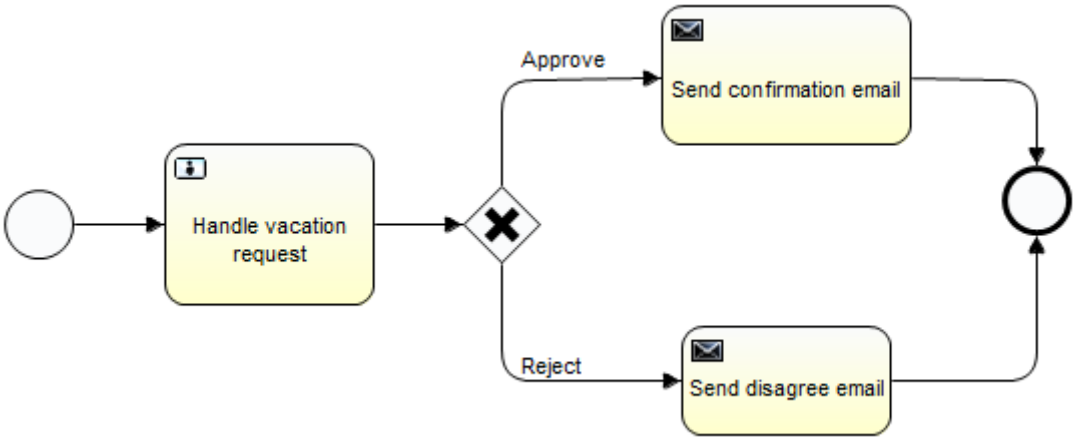
Hình 2.13: Mô hình quy trình thực hiện đơn hàng sau khi tinh chỉnh

2.5.6.3 Công cụ Activiti

Activiti design là một plugin của eclipse, công cụ hỗ trợ thiết kế một cách trực quan, cung cấp các đối tượng, thành phần được định nghĩa theo tiêu chuẩn BPMN. Trong activiti người sử dụng có thể dễ dàng thiết kế, thay đổi linh hoạt mô hình quy trình nghiệp vụ dựa trên việc dễ dàng thay đổi kiểu cho các đối tượng thành phần cũng như tạo các phần tử mới, liên kết các phần tử thành quy trình hoàn trình, các đối tượng đều được thể hiện trực quan, dễ hiểu. Với bộ công cụ của activiti người dung có thể sử dụng các chức năng chính như:

- Activiti Modeler powered by Signavio: dùng để vẽ, mô hình hóa các mô hình nghiệp vụ (các kí pháp sử dụng BPMN 2.0)
- Activiti Cycle: dùng để triển khai các mô hình nghiệp vụ để các mô hình nghiệp vụ có thể thực thi một các tự động.
- Activiti Explorer: dùng để thực thi tự động các quy trình nghiệp vụ đã được mô hình hóa.
- Activiti Probe: dùng để quản lý cơ sở dữ liệu, các mô hình hóa nghiệp vụ
- Activiti Administrator: dùng để quản lý các người sử dụng, các nhóm sử dụng trong hệ thống Activiti

Sau đây là một ví dụ đơn giản của mô hình BPMN được thiết kế và thực thi trên activiti.



Hình 2.14: Ví dụ về mô hình BPMN được thiết kế bởi Activiti

```
<process id="myProcess" name="My process" isExecutable="true">
  <startEvent id="startevent1" name="Start">
    <extensionElements>
      <activiti:formProperty id="day" name="Number of days" type="string"
variable="day" required="true"></activiti:formProperty>
      <activiti:formProperty id="firstday" name="First day of holiday (dd-MM-yyyy)"
type="date" variable="firstday" required="true"></activiti:formProperty>
      <activiti:formProperty id="reason" name="Reason" type="string"
variable="reason" required="true"></activiti:formProperty>
    </extensionElements>
  </startEvent>
  <userTask id="usertask2" name="Handle vacation request"
activiti:assignee="gonzo">
    <documentation>kermit would like to take $day of vaction (Reason:
$reason)</documentation>
    <extensionElements>
      <activiti:formProperty id="handle" name="Do you approve this request?"
type="enum" variable="handle" required="true">
        <activiti:value id="OK" name="Approve"></activiti:value>
        <activiti:value id="NOK" name="Reject"></activiti:value>
      </activiti:formProperty>
      <activiti:formProperty id="motivation" name="Motivation" type="string"
variable="motivation"></activiti:formProperty>
    </extensionElements>
  </userTask>
  <exclusiveGateway id="exclusivegateway1" name="Exclusive
Gateway"></exclusiveGateway>
  <sequenceFlow id="flow2" sourceRef="usertask2"
```

```
targetRef="exclusivegateway1"></sequenceFlow>
   <sequenceFlow id="flow3" name="Approve" sourceRef="exclusivegateway1"
targetRef="mailtask1">
     <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${handle=='OK'}]]></conditionExpressi
on>
   </sequenceFlow>
   <sequenceFlow id="flow4" name="Reject" sourceRef="exclusivegateway1"
targetRef="mailtask2">
     <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${handle=='NOK'}]]></conditionExpres
sion>
   </sequenceFlow>
   <endEvent id="endevent1" name="End"></endEvent>
   <sequenceFlow id="flow5" sourceRef="mailtask1"
targetRef="endevent1"></sequenceFlow>
   <sequenceFlow id="flow6" sourceRef="mailtask2"
targetRef="endevent1"></sequenceFlow>
   <serviceTask id="mailtask1" name="Send confirmation email"
activiti:type="mail">
     <extensionElements>
       <activiti:field name="to">
         <activiti:string><![CDATA[lienmini1510@gmail.com]]></activiti:string>
       </activiti:field>
       <activiti:field name="from">
         <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
       </activiti:field>
       <activiti:field name="subject">
         <activiti:string><![CDATA[Letter of acceptance]]></activiti:string>
       </activiti:field>
       <activiti:field name="html">
         <activiti:string><![CDATA[Agree with your vacation
request]]></activiti:string>
       </activiti:field>
     </extensionElements>
   </serviceTask>
   <serviceTask id="mailtask2" name="Send disagree email" activiti:type="mail">
     <extensionElements>
       <activiti:field name="to">
         <activiti:string><![CDATA[lienmini1510@gmail.com]]></activiti:string>
```

```
      </activiti:field>
      <activiti:field name="from">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
      </activiti:field>
      <activiti:field name="subject">
        <activiti:string><![CDATA[Disagree with your vacation
request]]></activiti:string>
      </activiti:field>
      <activiti:field name="html">
        <activiti:string><![CDATA[We have a important meeting
!]]></activiti:string>
      </activiti:field>
    </extensionElements>
  </serviceTask>
  <sequenceFlow id="flow7" sourceRef="startevent1"
targetRef="usertask2"></sequenceFlow>
 </process>
```

Hình 2.15: Dạng xml của mô hình BPMN

Khi quy trình được mô hình hóa sẽ tự động sinh file dạng *.bpmn20.xml , file này là đầu vào để thực thi tự động mô hình bằng công cụ Activiti Explorer.

2.6 Tổng kết chương

Nội dung chương 2 đã khái quát về mô hình thực thi được, giới thiệu một số phương pháp kiểm thử dựa trên mô hình. Đồng thời cũng cung cấp các kiến thức nền tảng cơ bản về mô hình BPMN, các ký pháp cơ bản để có thể thiết kế được mô hình này.

Việc sinh các ca kiểm thử từ mô hình là phương pháp phổ biến và có tính ứng dụng cao, tuy nhiên lại chưa có nhiều nghiên cứu về phương pháp sinh ca kiểm thử từ mô hình BPMN. Vậy với mô hình một mô hình thực thi được BPMN, với nhiều ưu điểm trong việc dễ dàng thiết kế , kiểm tra tính đúng đắn và tính thiết thực cao của mô hình BPMN thì phương pháp tiếp cận để sinh ca kiểm thử từ mô hình này như nào? Chúng ta hãy cùng tìm hiểu câu trả lời cho vấn đề này trong chương tiếp theo (Chương 3).

CHƯƠNG 3: PHƯƠNG PHÁP SINH CA KIỂM THỬ TỪ MÔ HÌNH BPMN

Nội dung chương 3 tập trung vào việc phát biểu bài toán và đề xuất phương pháp sinh kịch bản kiểm thử từ mô hình BPMN.

3.1 Giới thiệu

Kiểm thử dựa trên mô hình là một kỹ thuật kiểm thử hộp đen, dựa trên phương pháp ứng dụng các mô hình thiết kế vào kiểm thử phần mềm. Mô hình thiết kế là đại diện cho các hành vi mong muốn của một hệ thống cần kiểm thử, kiểm thử dựa trên mô hình đại diện cho một chiến lược thử nghiệm hay một môi trường kiểm thử.

Một mô hình đặc tả hệ thống thường là một bản tóm tắt, trình bày một phần hành vi mong muốn của hệ thống. Chúng được biểu diễn bằng máy hữu hạn trạng thái, ôtômat, đặc tả đại số, biểu đồ trạng thái bằng UML, v.v. Các ca kiểm thử có thể được sinh ra từ mô hình theo nhiều cách khác nhau. Trường hợp kiểm thử dựa trên một mô hình gọi là kiểm thử chức năng có cùng mức độ trừu tượng như mô hình. Những trường hợp kiểm thử này được gọi là một bộ kiểm thử trừu tượng. Một bộ kiểm thử trừu tượng không thể khẳng định hoàn toàn được tính đúng đắn của hệ thống. Vì hệ thống trên thiết kế cũng có một mức sai trừu tượng. Một bộ kiểm thử thực thi cần phải được bắt nguồn từ một bộ thử nghiệm trừu tượng tương ứng. Các bộ kiểm thử thực thi phụ thuộc trực tiếp với hệ thống kiểm thử. Điều này đạt được bằng cách ánh xạ các ca kiểm thử trừu tượng với các các ca kiểm thử cụ thể phù hợp để thực thi. Trong một số môi trường kiểm thử dựa trên mô hình, mô hình có đủ thông tin để tạo ra dãy ca kiểm thử thực thi trực tiếp. BPMN là môt mô hình có đầy đủ thông tin để có thể sinh trực tiếp các kịch bản ca kiểm thử. Trong các phần tiếp theo của chương sẽ trình bày bài toán, phương pháp tiếp cận để sinh ca kiểm thử tụ động từ mô hình BPMN.

3.2 Phát biểu bài toán

Để phù hợp với xu hướng phát triển phần mềm hiện nay, các kỹ thuật và phương pháp tự động tạo ra các ca kiểm thử từ mô hình ngày càng được quan tâm nhằm nâng cao tính hiệu và tiết kiệm chi phí. Có nhiều hướng tiếp cận các mô hình khác nhau để sinh các kịch bản kiểm thử từ mô hình. Tuy nhiên, việc xây dựng mô hình là một công việc khó khăn phức tạp và đòi hỏi tính chính xác cao như các mô hình ôtômat, máy hữu hạn trạng thái, đặc tả đại số,…Trong phạm vi luận văn này, tôi lựa chọn đầu vào là mô hình BPMN được thiết kế trên công cụ Activiti để sinh đầu ra là các kịch bản ca kiểm thử. Do BPMN là mô

hình thực thi được có thể dễ dàng xây dựng và kiểm tra tính đúng đắn của mô hình, đồng thời mang lại nhiều giá trị thực tế trong nghiệp phát triển phần mềm.

- ***BPMN là một mô hình mô tả cụ thể hành vi của người dùng và hệ thống đủ chi tiết để có thể thực thi được:***

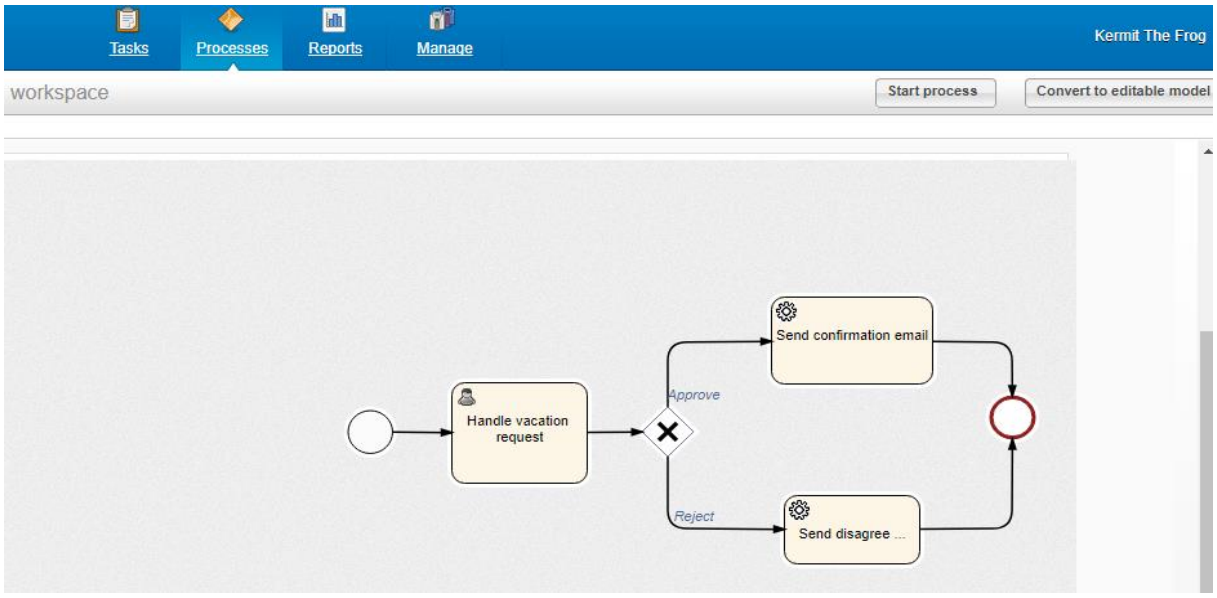
- Từ BPMN có thể sinh mã chương trình (M2T) thông qua sự hỗ trợ của ngôn ngữ thực thi được BPEL – Business process execution language và sinh các ca kiểm thử cũng như kịch bản kiểm thử tích hợp chức năng, kiểm thử hệ thống.
- Một thể hiện cụ thể hơn cho việc BPMN là mô hình đủ chi tiết để thực thi được là BPMN có thể thực thi trực tiếp trên công cụ quản lý quy trình nghiệp vụ (BPM) như Bizagi, Activiti. Điều này có nghĩa là, khi có một mô hình BPMN được import vào công cụ BPM, ta có thể thực hiện được luồng nghiệp vụ trên công cụ.

Ví dụ thể hiện mô hình BPMN thực thi được trên công cụ quản lý quy trình nghiệp vụ Activiti.

Mô tả mô hình: Đây là luồng quy trình xin nghỉ phép: người xin nghỉ nhập thông tin số ngày nghỉ (number of days), ngày bắt đầu nghỉ (first day of holiday), lí do xin nghỉ (Reason) như hình 3.3 . Sau đó, tài khoản có quyền phê duyệt vào xử lý yêu cầu có thể đồng ý/không đồng ý (Approve/Reject) yêu cầu như hình 3.4 hoặc hình 3.6. Lúc này người xin nghỉ sẽ nhận được mail thông báo phê duyệt hoặc từ chối yêu cầu phụ thuộc vào lựa chọn của người phê duyệt.

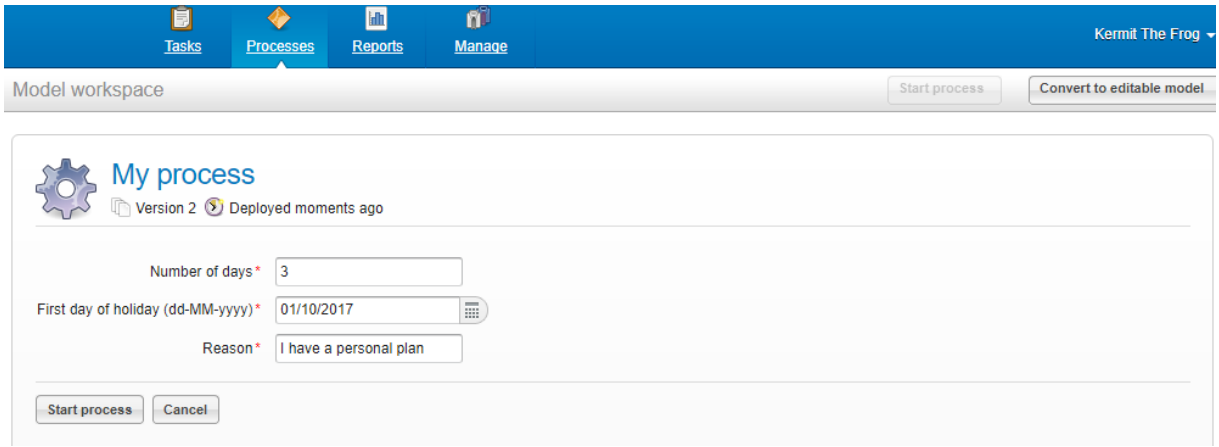
Thực thi mô hình: Từ mô hình đầu vào trên có thể thực thi được trên công cụ Activiti Designer như sau:

Bước 1: Đăng nhập với tài khoản của người xin nghỉ và import mô hình BPMN lên công cụ quản lý quy trình nghiệp vụ Activiti:



Hình 3.1: Mô hình yêu cầu kỳ nghỉ được import lên công cụ Activiti Design

Bước 2: Ấn chọn nút "Start process" và nhập các thông tin đăng ký xin nghỉ:

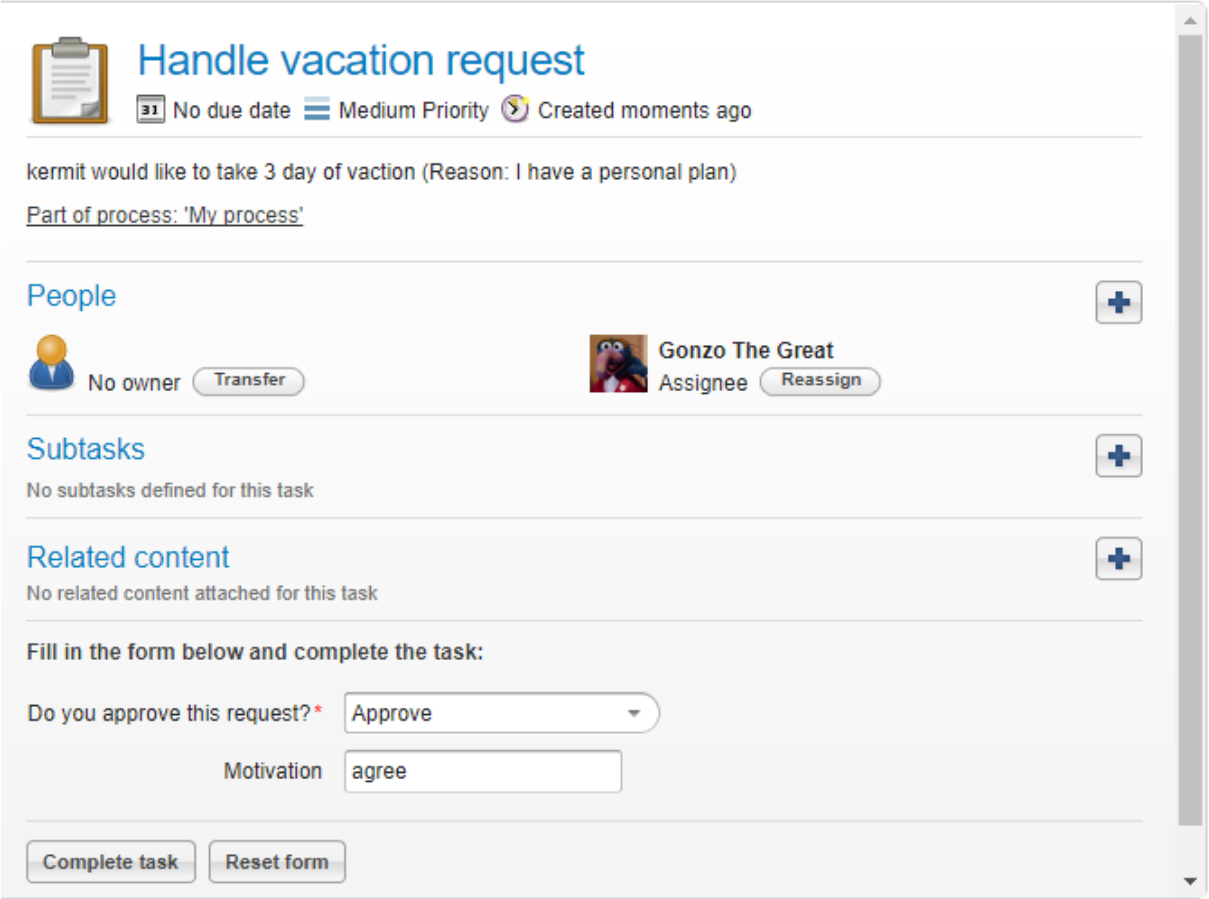


Hình 3.2: Màn hình nhập thông tin đăng ký nghỉ

Bước 3: Đăng nhập với tài khoản của người có quyền xử lý yêu cầu xin nghỉ, tài khoản gonzo với địa chỉ mail là duonghuyen0805@gmail.com để xử lý yêu cầu:

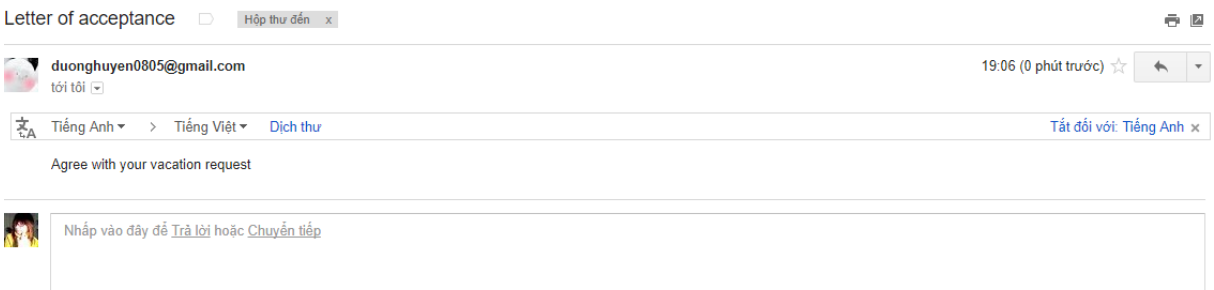
- ***Trường hợp 1:*** yêu cầu xin nghỉ được phê duyệt

Tài khoản có quyền xử lý yêu cầu chọn thông tin trên form như sau:



Hình 3.3: Màn hình nhập thông tin đồng ý yêu cầu xin nghỉ

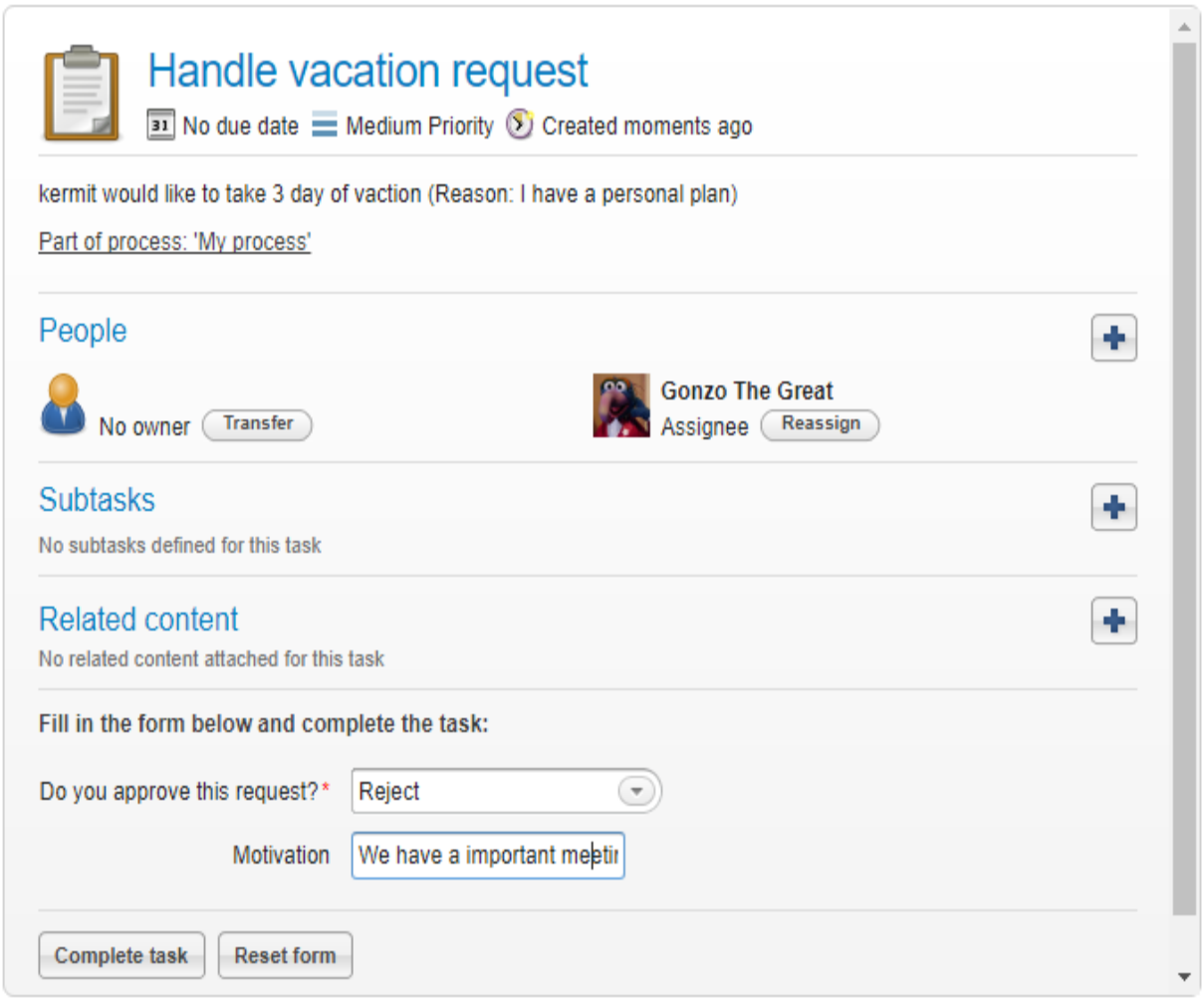
Khi đó, tài khoản xin nghỉ (lienmini1510@gmail.com) sẽ nhận được email thông báo:



Hình 3.4: Thông báo yêu cầu xin nghỉ được phê duyệt

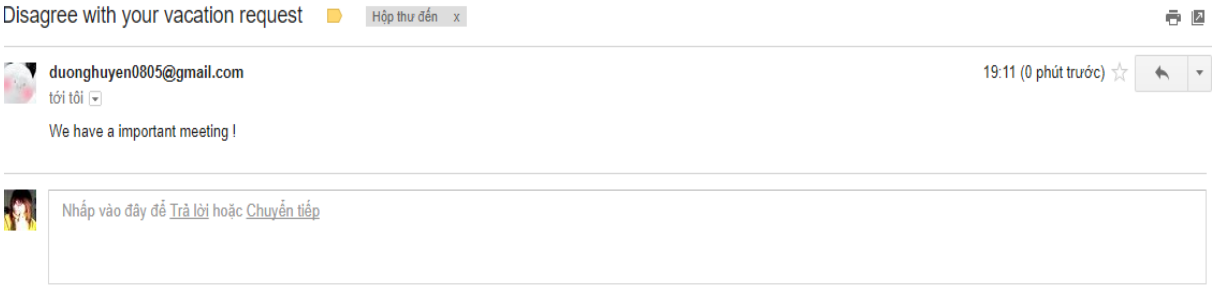
- ***Trường hợp 2:*** yêu cầu xin nghỉ không được phê duyệt

Tài khoản có quyền xử lý yêu cầu chọn thông tin trên form như sau:



Hình 3.5: Màn hình thông tin từ chối yêu cầu xin nghỉ

Khi đó, tài khoản xin nghỉ (lienmini1510@gmail.com) sẽ nhận được email thông báo:



Hình 3.6: Thông báo yêu cầu xin nghỉ không được phê duyệt

- ***Lợi ích của việc sinh ca kiểm thử từ mô hình BPMN:***
  - Tạo cơ sở cho sinh ca kiểm thử cho hệ thống phần mềm, các ca kiểm thử này áp dụng tốt nhất cho giai đoạn kiểm thử tích hợp và kiểm thử hệ thống.
  - Tạo ca sử dụng cho các phần mềm quản lý quy trình nghiệp vụ như Activiti, Bizagi,…

3.3 Thuật toán sinh kịch bản ca kiểm thử từ mô hình BPMN

3.3.1 Ý tưởng cơ bản

Mô hình BPMN biểu diễn thiết kế theo trình tự các bước trong luồng quy trình nghiệp vụ. Bên trong mỗi mô hình BPMN gồm nhiều thành phần và các thông tin đính kèm. Mỗi thành phần, cấu trúc có vai trò khác nhau trong ca sử dụng. Mô hình BPMN khi thiết kế có thể biểu diễn dưới hai dạng: dạng biểu đồ (diagram) và dạng xml biểu diễn cấu trúc mô hình, thông tin thuộc tính chi tiết của từng thành phần. Để có thể sinh kịch bản kiểm thử từ mô hình BPMN trước tiên cần phân tích dữ liệu đầu vào là mô hình BPMN được lưu dưới dạng file xml thành các đối tượng trong java để chương trình có thể xử lý. Dựa trên dữ liệu đã được biến đổi từ đó đưa về mô hình có dạng logic như đồ thị dòng điều khiển. Sau đó duyệt toàn bộ các nhánh trên đồ thị dạng logic luồng điều khiển có đường đi từ điểm bắt đầu đến điểm kết thúc, mỗi nhánh tương ứng với các kịch bản ca kiểm thử được sinh từ mô hình.

3.3.2 Chuyển đổi mô hình BPMN sang dạng CFG

Đồ thị luồng điều khiển (CFG- Control Flow Graph) là một đồ thị có hướng trong đó có các nốt (node) và các cạnh thể hiện cho luồng điều khiển. Một CFG luôn có một điểm đầu vào và một điểm đầu ra. Luồng quy trình nghiệp vụ được thiết kế theo trình tự hoạt động bao gồm nhiều thành phần và các thông tin đính kèm. Để có thể duyệt mô hình BPMN để tạo ra các kịch bản ca kiểm thử chúng ta phải liệt kê tất cả các thành phần, ký hiệu bên trong luồng quy trình nghiệp vụ BPMN và dùng thuật toán để biến đổi tương ứng cho từng thành phần ký hiệu đó. Đầu ra của bước này là đối tượng BpmnModel.

Các nhóm đối tượng trên mô hình BPMN ban đầu khi biến đổi sẽ được phân thành 2 loại đối tượng chính trong BpmnModel:

- FlowNode: Bao gồm các đối tượng Event, Activities và Gateways trong nhóm Flow Object như: StartEvent, EndEvent, UserTask, ServiceTask, Gateway, ... trong mô hình BPMN.
- Connect Object: là các đối tượng có vai trò liên kết các thành phần trong luồng quy trình nghiệp vụ. Các loại liên kết cơ bản để kết nối các đối tượng với nhau hoặc với thông tin khác, cụ thể gồm: sequence flow, message flow, association, data association.

Chương trình thực hiện duyệt từng thẻ trong file xml và lưu vào các đối tượng tương ứng. Các đối tượng có thuộc tính chung là id, name, document. Với mỗi loại node có thuộc tính riêng cho từng node. Tuy nhiên:

- Các đối tượng trong nhóm FlowNode có thuộc tính chung là List<IncomingFlow> và List <OutgoingFlow>.

- Các đối tượng trong nhóm Connect Object có thuộc tính chung là SourceRef (Node nguồn) và TargetRef (Node đích).

Thuật toán sinh đồ thị CFG

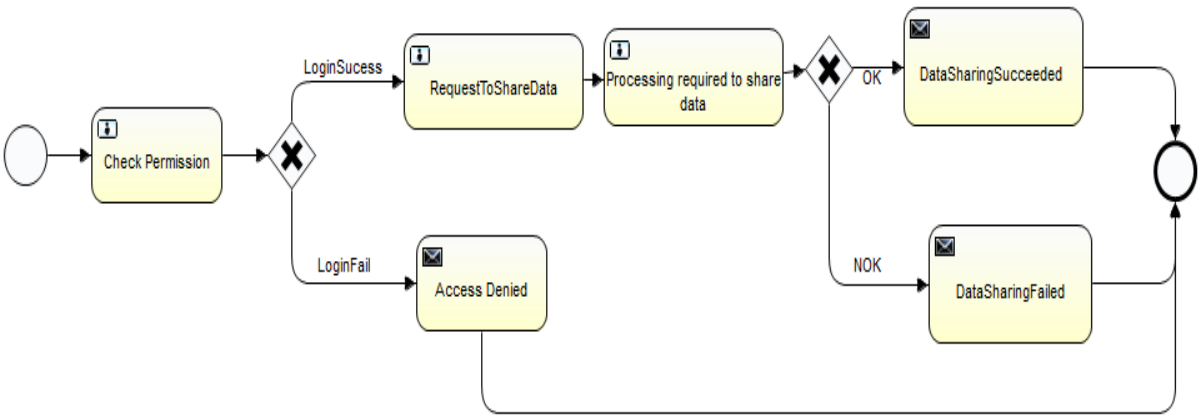
Đầu vào: mô hình luồng quy trình nghiệp vụ A (tệp xml)

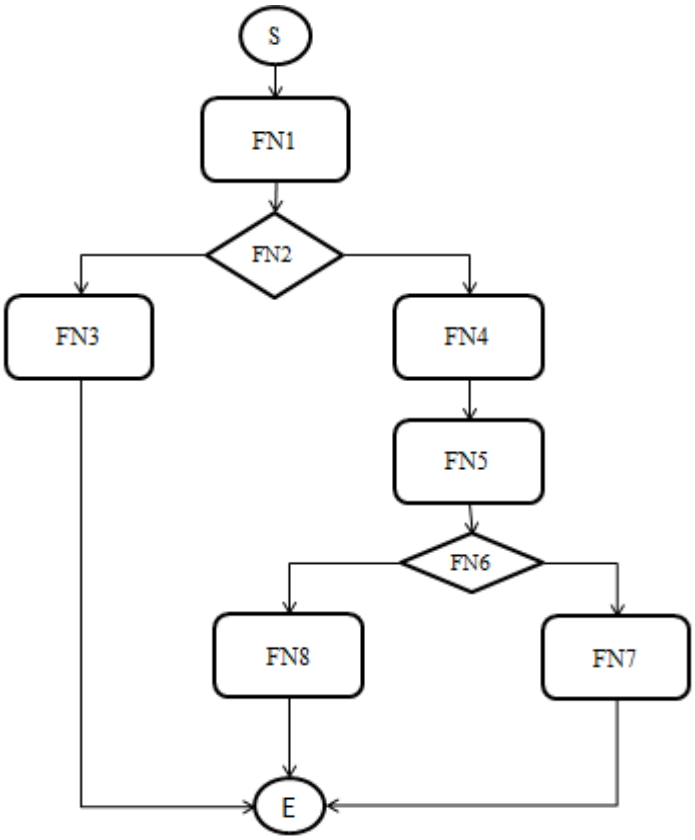
Đầu ra: Đồ thị G:(FN,CO, S, E) với FN là tập các nốt, S là nốt khởi tạo, E là nốt kết thúc và CO là tập các cạnh trong đồ thị CFG.

$CO = \{(x, y) | x, y \in FN \cup End\}$

1. create intitial node *S*;
2. create empty *bpmnModel*;
3. create *P* pointer at start element of business process model and notation A in xml file;
4. repeat
5. *P* read each element of A to add to *bpmnModel*
6. move *P* to next element;
7. until *P* meet element process end of xml file
8. return G;

Ví dụ bài toán chia sẻ data (Sharedata) được mô hình hóa như sau:





Hình 3.7: Đồ thị CFG cho bài toán chia sẻ data

3.3.3 Thuật toán sinh kịch bản ca kiểm thử

Thuật toán sinh kịch bản kiểm thử từ bpmnModel

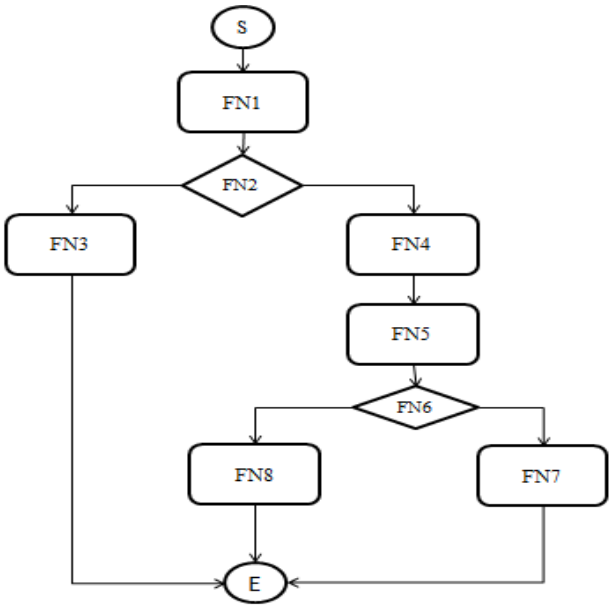
Đầu vào: Đối tượng bpmnModel (kết quả thu được sau khi biến đổi tệp đầu vào xml thành CFG)

Đầu ra: List<TestPath> là tập tất cả các đường kiểm thử

1. Create List <TestPath>; //*khởi tạo danh sách đường kiểm thử ban đầu là rỗng*
2. List<Gateway> = getAllGatewayFromBpmn(bpmnModel);
3. Create List<FileTmp>;
4. Create MapLine; //*ma trận chứa tổ hợp các đường kiểm thử*
5. Create HashMap<Integer, List<FlowNode> mapTestPath;
6. For each path in MapLine
7. For each outgoing in gateway
8. TestPath = removeSequenceFlow();
9. List<fileTmp> +=List<fileTmp>.put(TestPath);
10. End for.
11. End for.
12. For (file in List<FileTmp>;

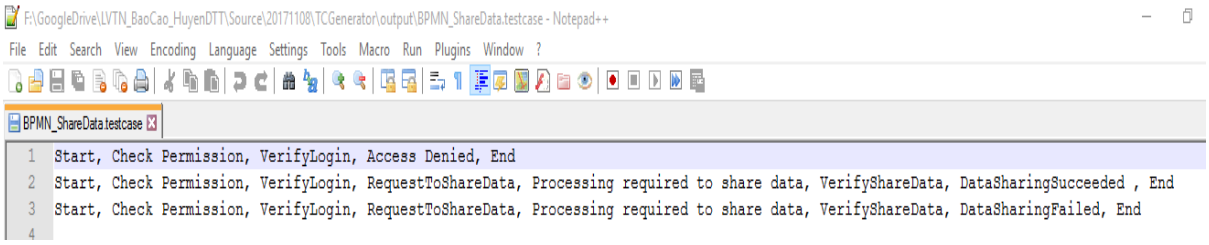
```
13.         if (file is Loop)
14.         {
15.         For FlowNode in List<FlowNode>
16.         List<FlowNode> += List<FlowNode> + List<Node is not Loop>
17.         Find targetNodeLoop;
18.         List<FlowNode>.add(targetNodeLoop);
19.         List<SequenceFlow> outGoings =
    targetNodeLoop.getOutgoingFlow();
20.         if (outGoings < 2)
21.                 List<SequenceFlow> = List<SequenceFlow> + outgoing
22.          else
23.                 For (SequenceFlow flow : outGoings)
24.                     NextNode = flow.getTargetFlows();
25.                     Testpath = List<FlowNode> + genTestCase(NextNode);
26.                       if (NotExist(TestPath, MapTestPath))
27.                                 MapTestPath.put<testPath>;
28.                       End for.
29.          }
30.          else
31.                {
32.                 Create List<FlowNode>;
33.                 Create ougoing;
34.                 List<FlowNode> += List<FlowNode>.add (StartEvent);
35.                 ConnectObject outGoing = StartEvent.getOutgoingFlow();
36.                 Repeat
37.                         NextNode = outGoing.getTargetFlows();
38.                         List<FlowNode>.add(NextNode);
39.                 until NextNode is EndEvent;
40.                 testPath = List<FlowNode>;
41.                  if (isNotExist (testPath,List<TestPath>))
42.                                 List<TestPath>.put(TestPath);
43.                     }
44. End for.
45. Return List<TestPath>;
```

Từ biểu đồ logic dạng CFG của bài toán chia sẻ data, sau khi áp dụng thuật toán sinh kịch bản ca kiểm thử ta được biểu diễn đầu ra như sau:



STT	Các nhánh tương ứng trên đồ thị
1	S → FN1 → FN2 → FN3 → E
2	S→ FN1 → FN2 → FN4 → FN5 → FN6 → FN7 → E
3	S→ FN1 → FN2 → FN4 → FN5 → FN6 → FN8 → E

File kết quả các kịch bản ca kiểm thử tương ứng với các nhánh trên đồ thị:



Hình 3.8: Kịch bản ca kiểm thử cho bài toán chia sẻ data

3.4 Tổng kết chương

Nội dung trong chương 3 đã giới thiệu khái quát về phương pháp sinh ca kiểm thử từ mô hình, từ đó tập trung vào phương pháp sinh ca kiểm thử từ mô hình BPMN. Trong đó, có mô tả bài toán và từ đó đưa hướng xử lý thông qua thuật toán sinh ca kiểm, đồng thời cũng có ví dụ trực quan sơ bộ về đầu ra của bài toán thực nghiệm. Trong chương tiếp theo (Chương 4), chúng ta sẽ cùng tìm hiểu chi tiết hơn về môi trường cài đặt và chương trình ứng dụng thực nghiệm.

CHƯƠNG 4: CÀI ĐẶT & THỰC NGHIỆM

Chương 4 tập trung giới thiệu môi trường cài đặt chương trình thực nghiệm, kết quả thực nghiệm được trình bày thông qua hai bài toán ví dụ, cuối cùng là phần ý nghĩa thực nghiệm từ chương trình đã xây dựng.

4.1 Môi trường cài đặt

Chương trình sinh kịch bản kiểm thử từ mô hình BPMN được xây dựng trên công cụ và môi trường sau:

- Công cụ phát triển
  - NetBeans IDE verion 8.2
  - Eclipse Java verion Oxygen
  - Apache-tomcat-8.5.23
  - Activiti-5.22.0
- Môi trường cài đặt:
  - Hệ điều hành Windows 10
  - Hệ điều hành Windows 8
  - Hệ điều hành Windows 7
- Môi trường lập trình: Java SE Development Kit 8
- Yêu cầu phần cứng:
  - Bộ vi xử lý: 1 GHz. 32-bit (x86) hoặc 64-bit (x64)
  - RAM: 1 GB (32-bit) hoặc 2 GB (x64)
  - Ổ đĩa cứng: trống 50MB
  - Độ phân giải màn hình: 800 x 600 hoặc cao hơn

4.2 Kết quả thực nghiệm

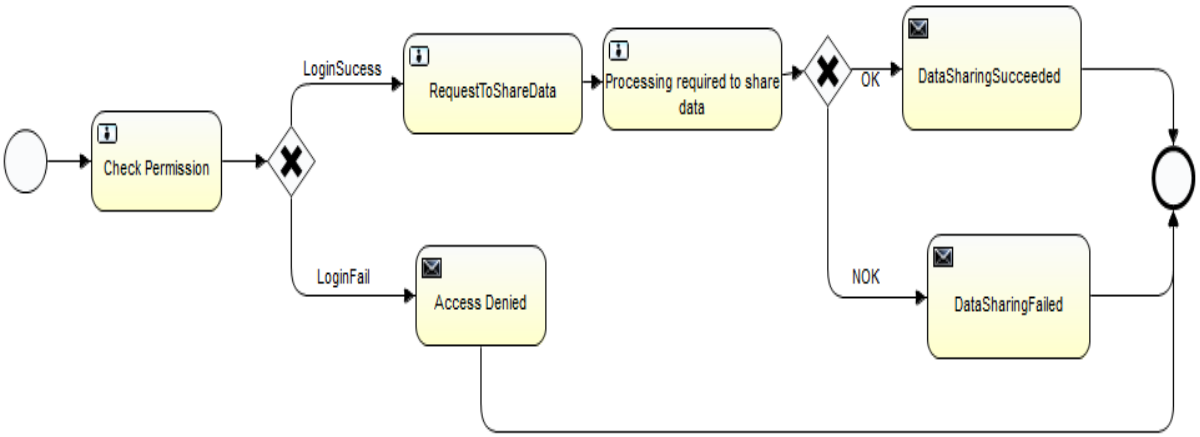
Để kiểm tra tính đúng đắn, tính khả thi và khả năng áp dụng thực tế quá trình kiểm thử phần mềm trong các công ty/tổ chức phát triển phần mềm, sau đây tôi xin trình bày 02 bài toán minh họa kết quả thực nghiệm. Đầu vào của mỗi bài toán là một mô hình BPMN biểu diễn luồng quy trình nghiệp vụ. Thông qua chương trình cài đặt sẽ sinh ra đầu ra là một file định dạng *.txt chứa tập các kịch bản kiểm thử. Các kịch bản ca kiểm thử này sau đó được áp dụng để kiểm thử công cụ quản lý quy trình nghiệp vụ Activiti bằng cách thực thi mô hình BPMN trên Activiti.

- ***Bài toán 01:*** (luồng nghiệp vụ đa nhánh)

Mô tả yêu cầu bài toán chia sẻ dữ liệu (Share data) : Tài khoản yêu cầu chia sẻ dữ liệu nhập thông tin đăng nhập ứng dụng. Thông tin đăng nhập sẽ được xác minh để xác định quyền truy cập của tài khoản yêu cầu. Nếu thông tin không xác thực thì tài khoản yêu cầu sẽ nhận được thông từ chối đăng nhập (Access

Denied) và kết thúc giao dịch. Ngược lại, tài khoản yêu cầu sẽ nhập thông tin yêu cầu chia sẻ dung lượng đến một tài khoản khác. Yêu cầu sẽ được kiểm tra, nếu thỏa mãn các điều kiện ràng buộc nghiệp vụ thì thực hiện yêu cầu chia sẻ dữ liệu, ngược lại thì kết thúc giao dịch.

Đầu vào: là file xml biểu diễn luồng nghiệp vụ sau:



Hình 4.1: Mô hình BPMN của yêu cầu chia sẻ data

```
<process id="myProcess" name="My process" isExecutable="true">
  <startEvent id="Start" name="Start">
    <documentation>Start</documentation>
    <extensionElements>
      <activiti:formProperty id="username" name="Username" type="string"
variable="username" required="true"></activiti:formProperty>
      <activiti:formProperty id="password" name="Password" type="string"
variable="password" required="true"></activiti:formProperty>
    </extensionElements>
  </startEvent>
  <userTask id="usertask1" name="Check Permission" activiti:assignee="gonzo">
    <extensionElements>
      <activiti:formProperty id="checkPermission" name="Check Permission"
type="enum" variable="checkPermission" required="true">
        <activiti:value id="true" name="Login Success"></activiti:value>
        <activiti:value id="fail" name="Login Fail"></activiti:value>
      </activiti:formProperty>
    </extensionElements>
  </userTask>
  <sequenceFlow id="flow1" sourceRef="Start"
targetRef="usertask1"></sequenceFlow>
  <exclusiveGateway id="VerifyLogin" name="VerifyLogin" activiti:async="true">
    <documentation>VerifyLogin</documentation>
  </exclusiveGateway>
  <sequenceFlow id="flow2" sourceRef="usertask1"
targetRef="VerifyLogin"></sequenceFlow>
```

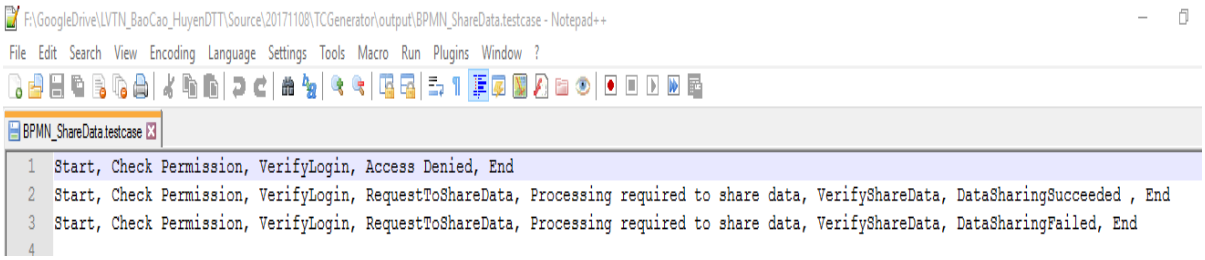
```
    <sequenceFlow id="LoginFail" name="LoginFail" sourceRef="VerifyLogin"
targetRef="mailtask3">
      <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${checkPermission=='fail'}]]></conditio
nExpression>
    </sequenceFlow>
    <sequenceFlow id="LoginSucess" name="LoginSucess" sourceRef="VerifyLogin"
targetRef="usertask2">
      <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${checkPermission=='true'}]]></conditi
onExpression>
    </sequenceFlow>
    <exclusiveGateway id="VerifyShareData"
name="VerifyShareData"></exclusiveGateway>
    <sequenceFlow id="OK" name="OK" sourceRef="VerifyShareData"
targetRef="mailtask1">
      <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${handle=='OK'}]]></conditionExpressi
on>
    </sequenceFlow>
    <endEvent id="End" name="End"></endEvent>
    <sequenceFlow id="flow7" sourceRef="mailtask3"
targetRef="End"></sequenceFlow>
    <sequenceFlow id="flow8" sourceRef="mailtask1"
targetRef="End"></sequenceFlow>
    <sequenceFlow id="NOK" name="NOK" sourceRef="VerifyShareData"
targetRef="mailtask2">
      <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${handle=='NOK'}]]></conditionExpres
sion>
    </sequenceFlow>
    <userTask id="usertask2" name="RequestToShareData"
activiti:assignee="kermit">
      <extensionElements>
        <activiti:formProperty id="data" name="Enter the data you want to share"
type="string" variable="data" required="true"></activiti:formProperty>
        <activiti:formProperty id="receiver" name="Receiver" type="string"
variable="receiver" required="true"></activiti:formProperty>
      </extensionElements>
    </userTask>
    <serviceTask id="mailtask1" name="DataSharingSucceeded "
activiti:type="mail">
      <extensionElements>
        <activiti:field name="to">
          <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
        </activiti:field>
        <activiti:field name="from">
          <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
```

```
      </activiti:field>
      <activiti:field name="subject">
        <activiti:string><![CDATA[Data Sharing Succeeded]]></activiti:string>
      </activiti:field>
      <activiti:field name="html">
        <activiti:string><![CDATA[Successful sharing request!]]></activiti:string>
      </activiti:field>
    </extensionElements>
  </serviceTask>
  <serviceTask id="mailtask2" name="DataSharingFailed" activiti:type="mail">
    <extensionElements>
      <activiti:field name="html">
        <activiti:string><![CDATA[Data sharing request failed!]]></activiti:string>
      </activiti:field>
      <activiti:field name="to">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
      </activiti:field>
      <activiti:field name="from">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
      </activiti:field>
      <activiti:field name="subject">
        <activiti:string><![CDATA[Data Sharing Failed]]></activiti:string>
      </activiti:field>
    </extensionElements>
  </serviceTask>
  <sequenceFlow id="flow9" sourceRef="mailtask2"
targetRef="End"></sequenceFlow>
  <userTask id="usertask3" name="Processing required to share data"
activiti:assignee="gonzo">
    <extensionElements>
      <activiti:formProperty id="handle" name="Processing required to share data"
type="enum" variable="handle" required="true">
        <activiti:value id="OK" name="Success"></activiti:value>
        <activiti:value id="NOK" name="Fail"></activiti:value>
      </activiti:formProperty>
    </extensionElements>
  </userTask>
  <sequenceFlow id="flow10" sourceRef="usertask2"
targetRef="usertask3"></sequenceFlow>
  <sequenceFlow id="flow11" sourceRef="usertask3"
targetRef="VerifyShareData"></sequenceFlow>
  <serviceTask id="mailtask3" name="Access Denied" activiti:type="mail">
    <extensionElements>
      <activiti:field name="to">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
      </activiti:field>
      <activiti:field name="from">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
```

```
      </activiti:field>
      <activiti:field name="subject">
        <activiti:string><![CDATA[Access Denied]]></activiti:string>
      </activiti:field>
      <activiti:field name="html">
        <activiti:string><![CDATA[Invalid username/password !]]></activiti:string>
      </activiti:field>
    </extensionElements>
  </serviceTask>
</process>
```

Hình 4.2: Biểu diễn dạng xml mô hình BPMN của yêu cầu chia sẻ data

Đầu ra: Các đường path bao phủ các nhánh như sau



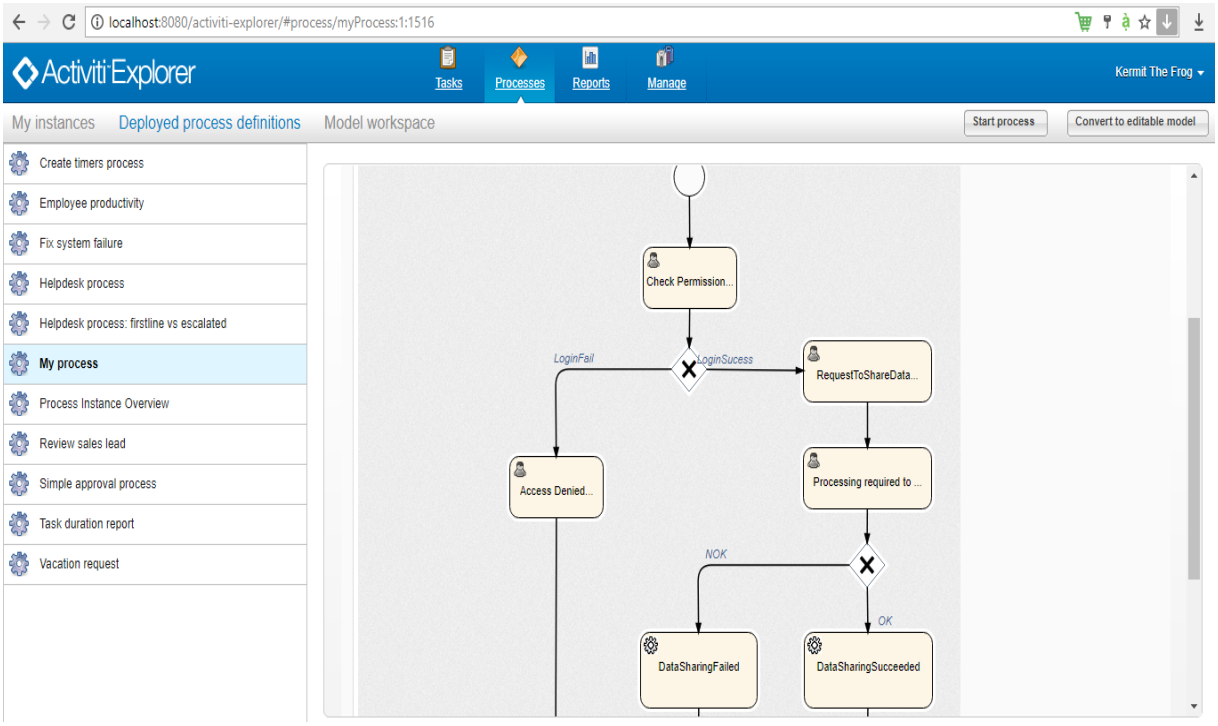
Hình 4.3: Kịch bản ca kiểm thử của yêu cầu chia sẻ data

Cụ thể gồm các kịch bản kiểm thử sau:

STT	Kịch bản ca kiểm thử
1	Start, Check Permission, VerifyLogin, Access Denied, End
2	Start, Check Permission, VerifyLogin, RequestToShareData, Processing required to share data, VerifyShareData, DataSharingSucceeded , End
3	Start, Check Permission, VerifyLogin, RequestToShareData, Processing required to share data, VerifyShareData, DataSharingFailed, End

- ***Thực hiện các kịch bản ca kiểm thử của mô hình chia sẻ dữ liệu trên công cụ Active-explorer.***

Khi import đầu vào là file bpmn model lên công cụ activit-explorer, giao diện chương trình sau khi import thành công như sau:



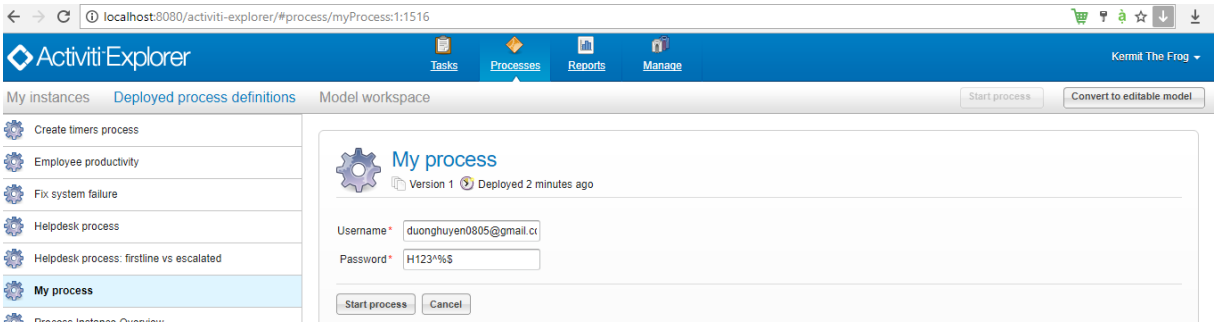
Hình 4.4: Mô hình BPMN "Share data" được import lên công cụ activiti

- ***Kịch bản ca kiểm thử đầu tiên:*** Thông tin đăng nhập không xác thực nên tài khoản yêu câu nhận được thông báo từ chối đăng nhập (Access Denied)

Kịch bản ca kiểm thử
Bước 1: Start, Bước 2: Check Permission, Bước 3: VerifyLogin, Bước 4: Access Denied, Bước 5: End

Bước 1: Start

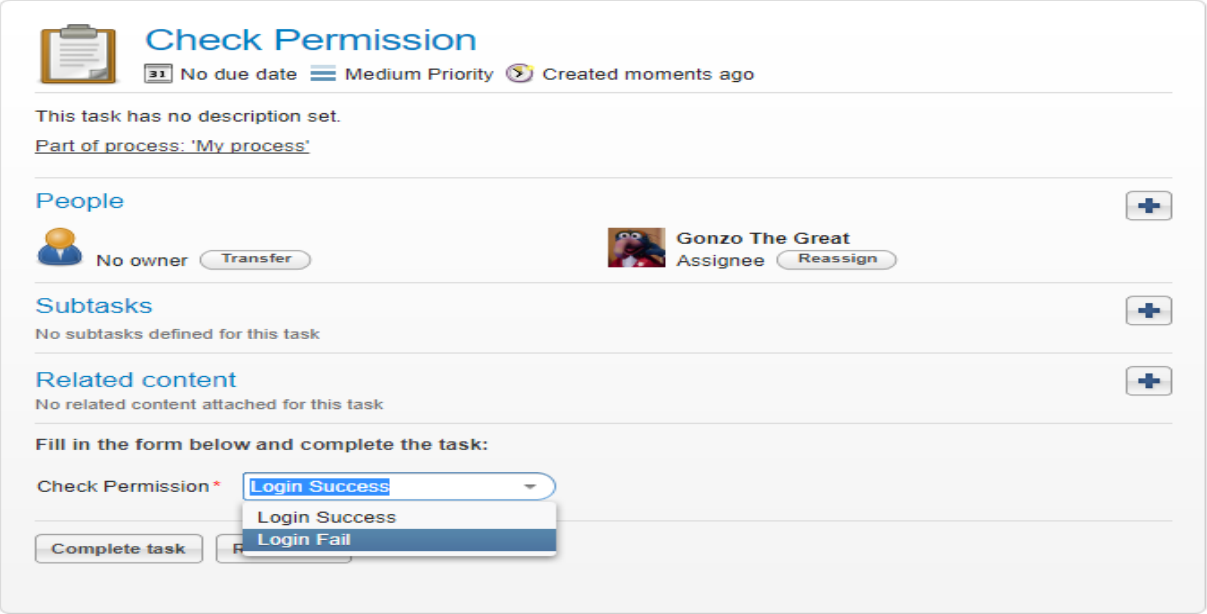
- Nhập thông tin username/password và ấn chọn "Start process" để bắt đầu luồng nghiệp vụ



Hình 4.5: Bước start trong kịch bản ca kiểm thử thứ nhất

Bước 2: Check Permission

- Đăng nhập với account "gonzo". Sau đó, trên form hiển thị cho bước "Check Permission" thực hiện không cho phép truy cập.



Hình 4.6: Bước Check Permission trong kịch bản ca kiểm thử thứ nhất

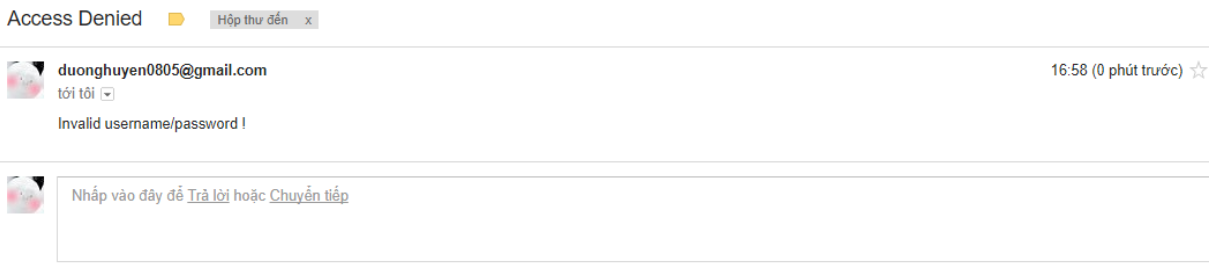
Bước 3: VerifyLogin

- Tại gateway "VerifyLogin" kiểm tra dữ liệu từ bước 2 là để xác định user có quyền đăng nhập thành công hay thất bại. Trong trường hợp này user đăng nhập thất bại nên hiển thị thông báo qua email

Bước 4: Access Denied

- Login với acc Kermit để kiểm tra, các task trong danh sách công việc đã hoàn thành

- Một thông báo được gửi đến địa chỉ mail của acc yêu cầu chia sẻ dữ liệu để thông báo đăng nhập thất bại. *(Trong luồng nghiệp vụ này đang cấu hình người gửi và người nhận/người gửi đều là địa chỉ mail: duonghuyen0805@gmail.com)*



Hình 4.7: Thông báo access denied trong kịch bản ca kiểm thử thứ nhất

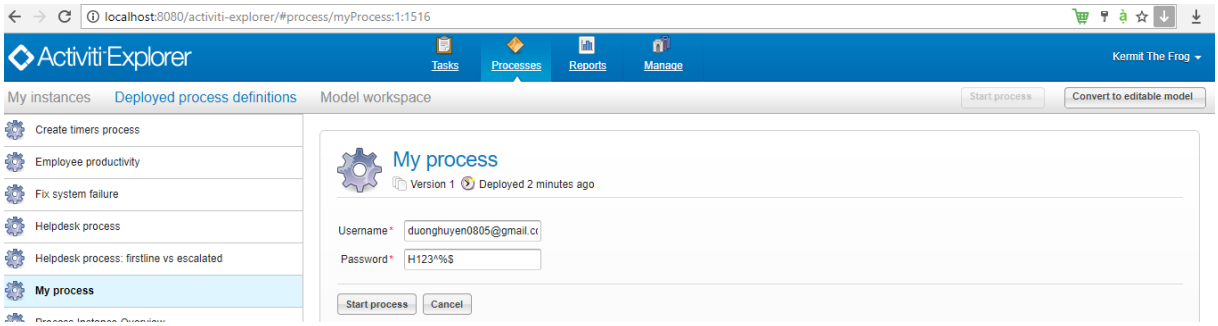
Bước 4: End.

- ***Kịch bản ca kiểm thử thứ 2:*** Yêu cầu chia sẻ dữ liệu thành công

Kịch bản ca kiểm thử
Bước 1: Start, Bước 2: Check Permission, Bước 3: VerifyLogin, Bước 4: RequestToShareData, Bước 5: Processing required to share data, Bước 6: VerifyShareData, Bước 7: DataSharingSucceeded, Bước 8: End

Bước 1: Start

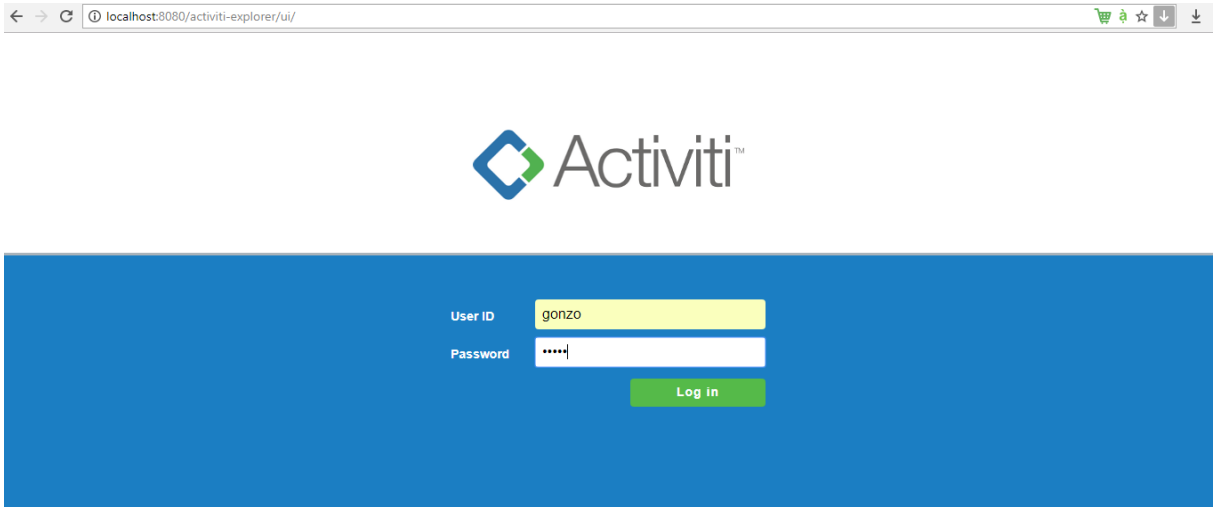
- Nhập thông tin username/password và ấn chọn "Start process" để bắt đầu luồng nghiệp vụ



Hình 4.8: Bước Start process trong kịch bản ca kiểm thử thứ nhất

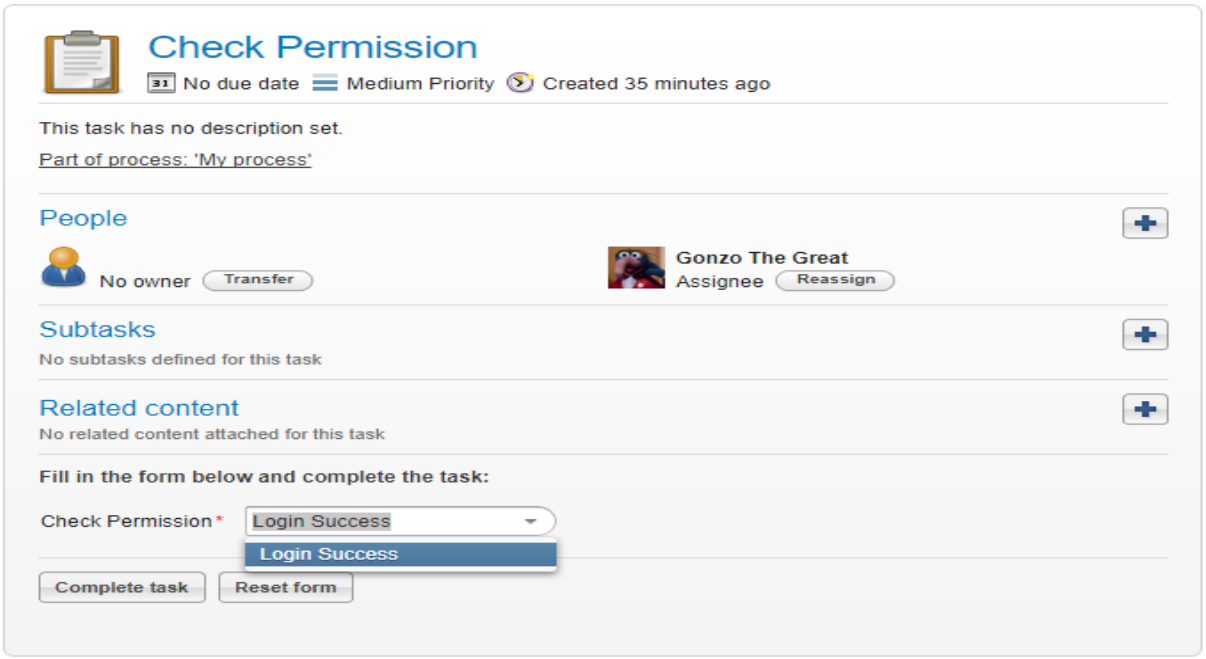
Bước 2: Check Permission

- Đăng nhập với account "gonzo"



Hình 4.9: Bước đăng nhập của kịch bản ca kiểm thử thứ hai

- Form hiển thị cho bước "Check Permission". Tại bước này thực hiện cho phép truy cập



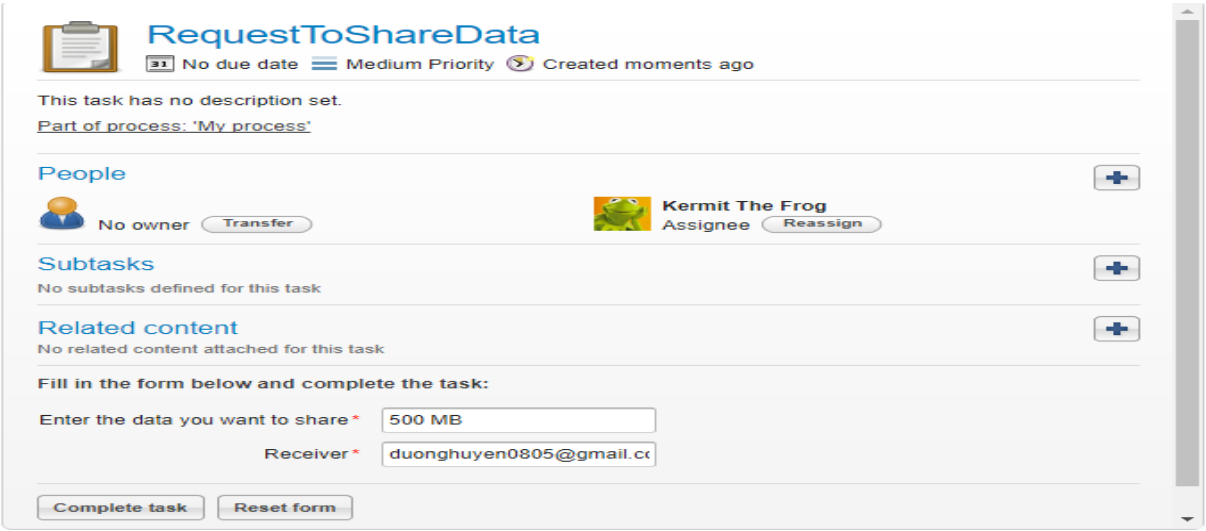
Hình 4.10: Bước Check Permission của kịch bản ca kiểm thử thứ hai

Bước 3: VerifyLogin

- Tại gateway "VerifyLogin" kiểm tra dữ liệu từ bước 2 là để xác định user có quyền đăng nhập thành công hay thất bại. Trong trường hợp này user đăng nhập thành công nên chuyển sang bước tiếp theo.

Bước 4: RequestToShareData

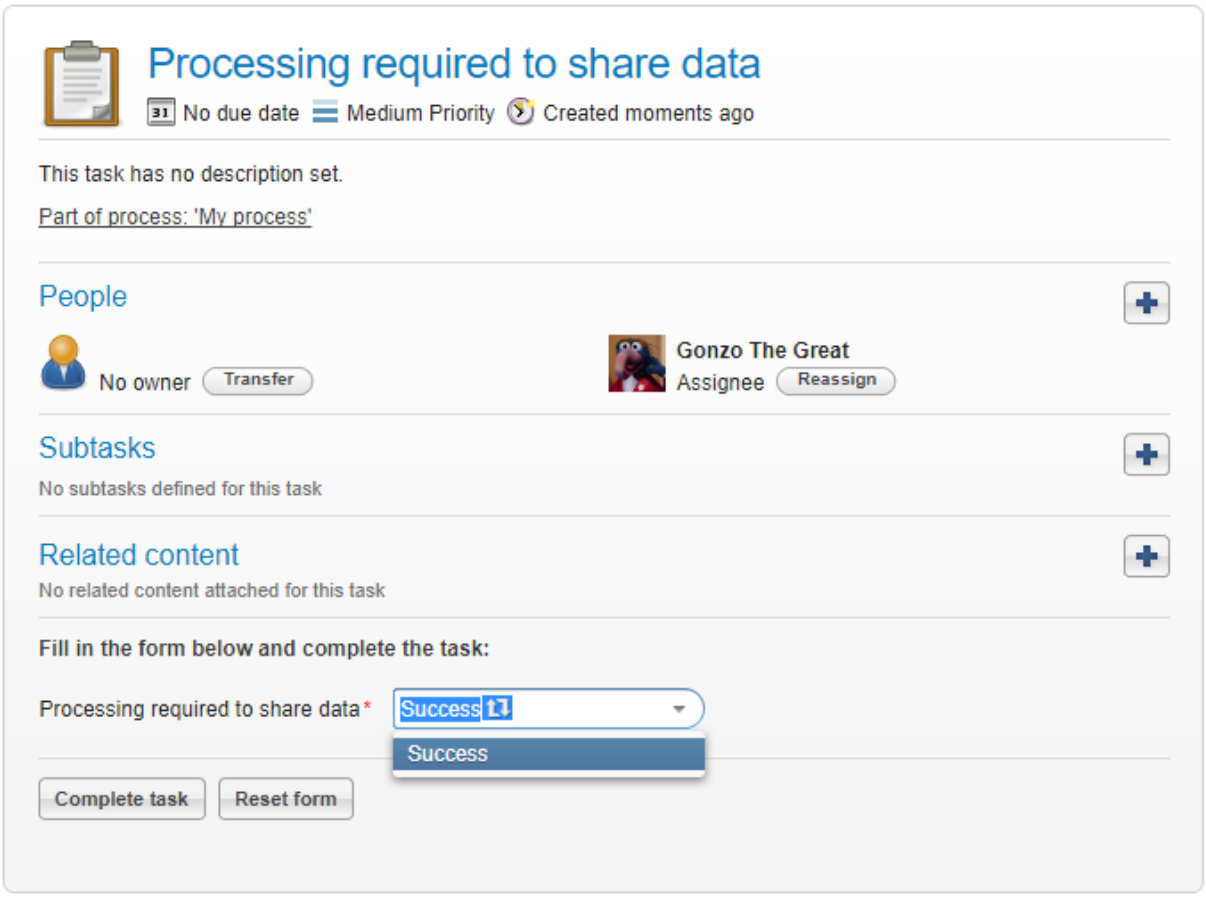
- User Kermit thực hiện nhập dữ liệu chia sẻ và đối tượng được chia sẻ tương ứng, sau đó chọn "complete task"



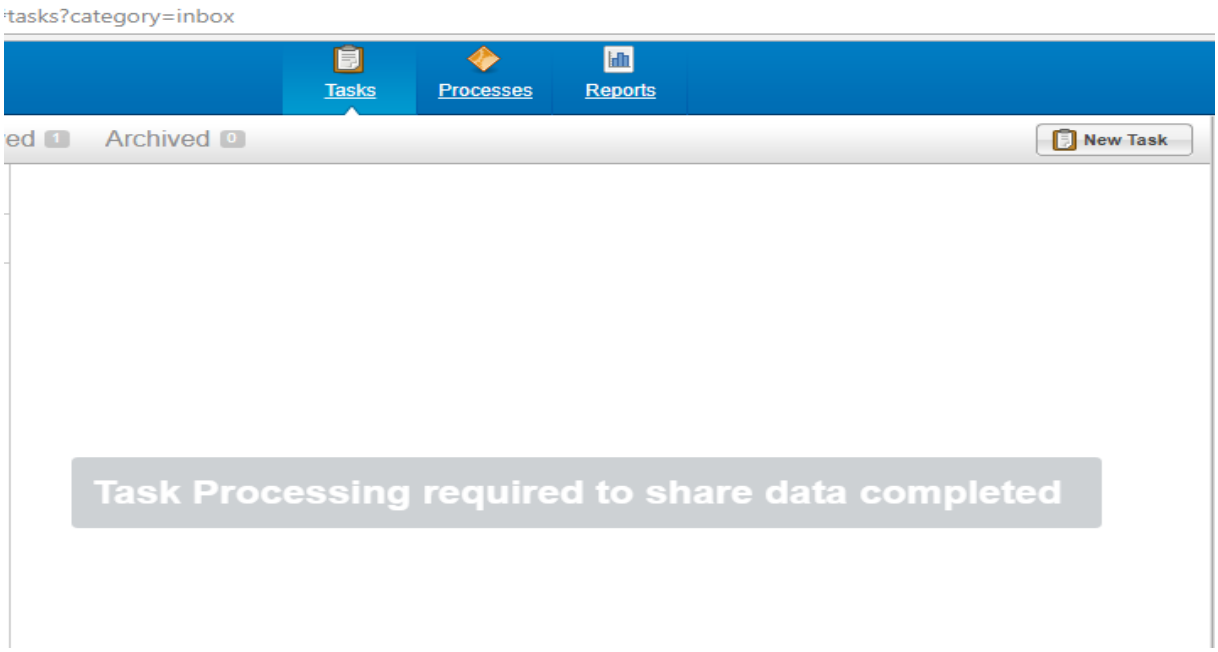
Hình 4.11: Bước RequestToShareData của kịch bản kiểm thử thứ hai

Bước 5: Processing required to share data

- Kiểm tra các điều kiện và cho phép chia sẻ dữ liệu thành công



Hình 4.12: Bước xử lý yêu cầu chia sẻ dữ liệu của kịch bản kiểm thử thứ hai



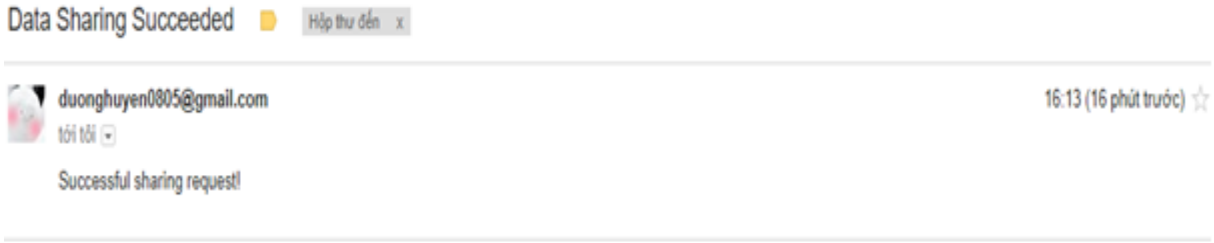
Hình 4.13: Hoàn thành yêu cầu chia sẻ dữ liệu của kịch bản kiểm thử thứ hai

Bước 6: VerifyShareData

- Tại gateway "VerifyShareData" kiểm tra dữ liệu từ bước 5 là để xác định yêu cầu chia sẻ dữ liệu của user có thành công hay không. Trong trường hợp này yêu cầu được thực hiện thành công nên chuyển sang bước tiếp theo.

Bước 7: DataSharingSucceeded

- Login với acc Kermit để kiểm tra, các task trong danh sách công việc đã hoàn thành
- Một email được gửi đến địa chỉ mail của acc yêu cầu chia sẻ dữ liệu (có thể cấu hình gửi thêm đến cả email của người nhận) để thông báo yêu cầu chia sẻ dữ liệu thành công.



Hình 4.14: Email thông báo của kịch bản ca kiểm thử thứ hai

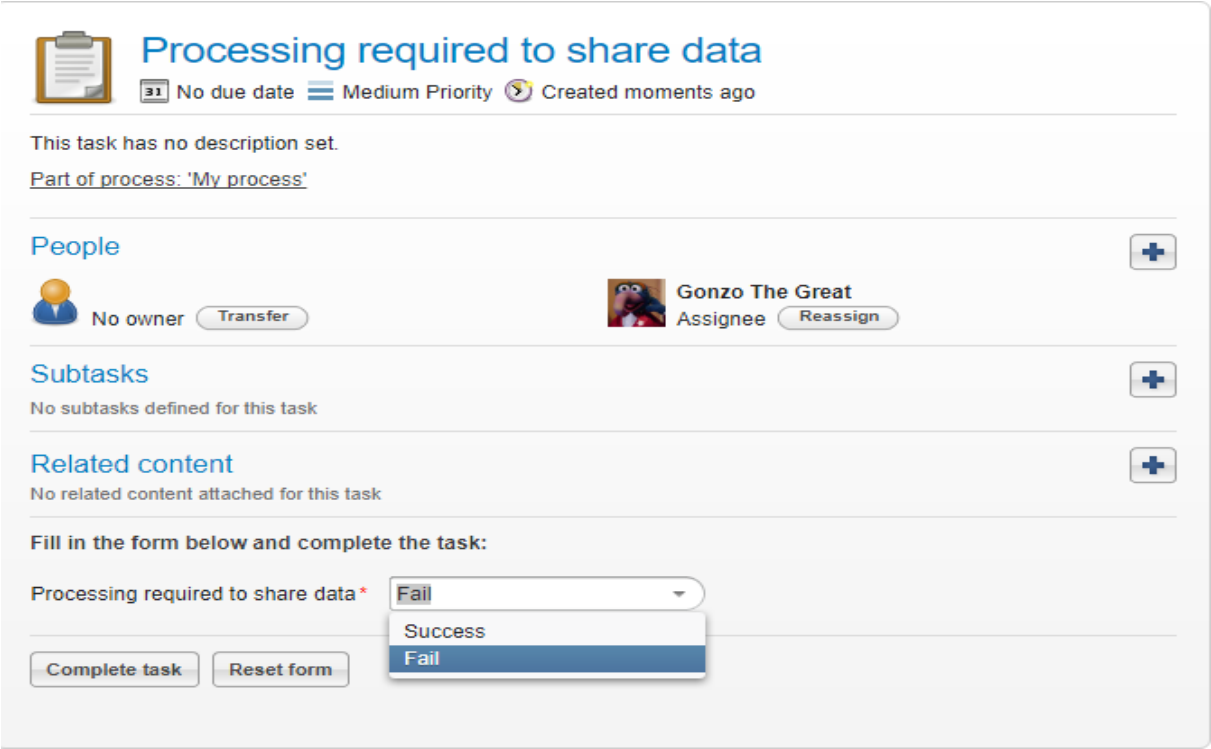
Bước 8: End.

- ***Với ca kiểm thử thứ 3:*** Yêu cầu chia sẻ dữ liệu không thành công

Kịch bản ca kiểm thử
Bước 1: Start, Bước 2: Check Permission, Bước 3: VerifyLogin, Bước 4: RequestToShareData, Bước 5: Processing required to share data, Bước 6: VerifyShareData, Bước 7: DataSharingFailed, Bước 8: End

Các bước từ 1 đến bước 4 thực hiện tương tự như với ca kiểm thử thứ hai.

Bước 5: Tại bước xử lý yêu cầu chia sẻ dữ liệu thay vì chọn "Success" như ca kiểm thử trên, chúng ta chọn "Fail"



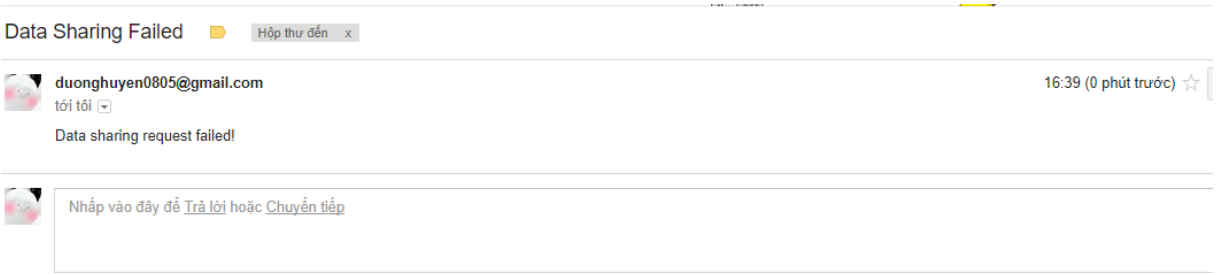
Hình 4.15: Bước xử lý yêu cầu chia sẻ dữ liệu của kịch bản kiểm thử thứ ba

Bước 6: VerifyShareData

- Tại gateway "VerifyShareData" kiểm tra dữ liệu từ bước 5 là để xác định yêu cầu chia sẻ dữ liệu của user có thành công hay không. Trong trường hợp này yêu cầu được thực hiện thành công nên chuyển sang bước tiếp theo.

Bước 7: DataSharingSucceeded

- Login với acc Kermit để kiểm tra, các task trong danh sách công việc đã hoàn thành
- Một email được gửi đến địa chỉ mail của acc yêu cầu chia sẻ dữ liệu (có thể cấu hình gửi thêm đến cả email của người nhận) để thông báo yêu cầu chia sẻ dữ liệu không thành công.



Hình 4.16: Email thông báo của kịch bản ca kiểm thử thứ ba.

Bước 8. End (kết thúc luồng nghiệp vụ)

Bài toán 2: (luồng nghiệp vụ đa nhánh và có vòng lặp)

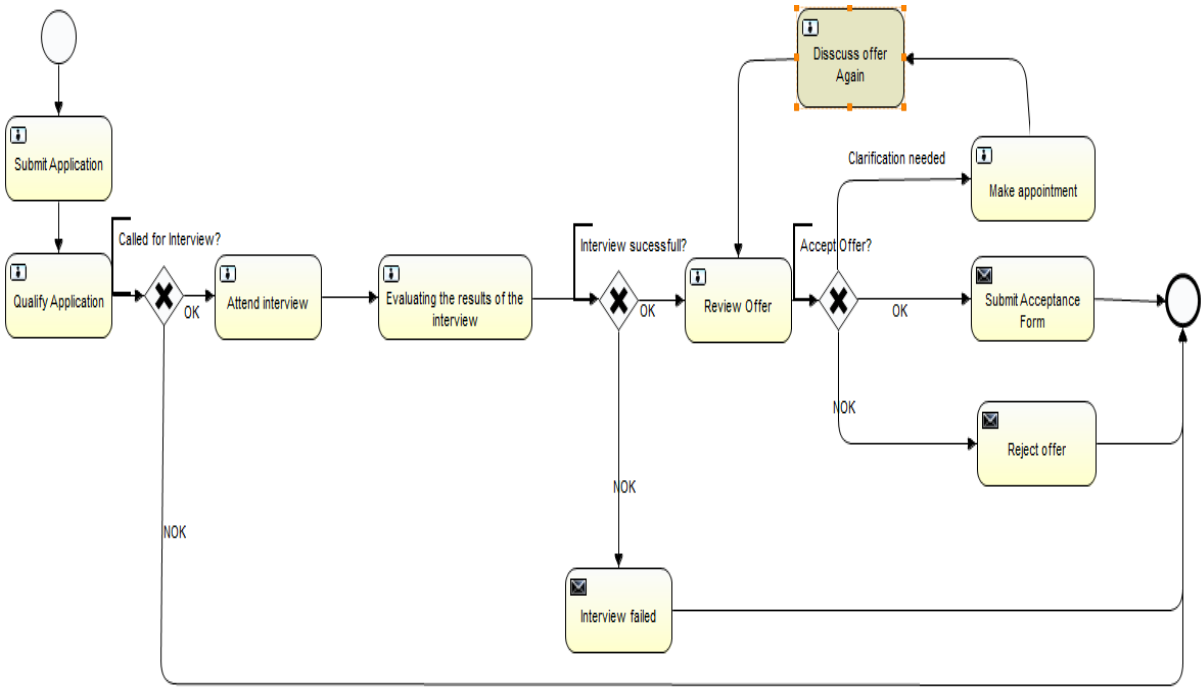
Mô tả luồng nghiệp vụ quy trình xin việc (apply for job): ứng viên thực hiện tạo hồ sơ xin việc *(Create Application*) và gửi hồ sơ xin việc *(Submit Application)* đến nhà tuyển dụng. Nhà tuyển dụng đánh giá hồ sơ xin việc và quyết định có mời ứng viên tham gia phỏng vấn tuyển dụng hay không? Nếu hồ sơ không đạt yêu cầu nhà tuyển dụng không mời ứng viên tham gia phỏng vấn thì kết thúc luồng nghiệp vụ. Ngược lại, ứng viên tham gia phỏng vấn (*Attend Interview)*. Sau buổi phỏng vấn, nhà tuyển dụng thực hiện đánh giá kết quả *(Evaluating the results of the interview).*

Nếu kết quả nếu phỏng vấn không thành công nhà tuyển dụng gửi thông báo từ chối (*Send Rejection)* tới ứng viên và kết thúc luồng nghiệp vụ.

Nếu kết quả phỏng vấn thành công, nghĩa là ứng viên đáp ứng được yêu cầu của nhà tuyển dụng, lúc này ứng viên sẽ thực hiện xem xét mức lương mà nhà tuyển dụng đề nghị (*Review Offer):*

- Nếu ứng viên không đồng ý mức lương nhà tuyển dụng đề nghị thì sẽ gửi thông báo từ chối (*Reject Offer)* tới nhà tuyển dụng và kết thúc luồng nghiệp vụ như mô tả trong hình 4.24 và hình 4.25 bên dưới.
- Nếu ứng viên đồng ý với mức lương nhà tuyển dụng đề nghị thì gửi thông báo chấp nhận *(Submit Acceptance Form)* tới ứng viên và kết thúc luồng nghiệp vụ.
- Trường hợp ứng viên cần trao đổi lại về mức lương đề nghị, nhà tuyển dụng sẽ đặt lịch hẹn (*Make appointment)* với ứng viên và thảo luận lại về mức lương (*Disscuss offer again)* sau đó sẽ quy lại quá trình xem xét mức lương mà ứng viên yêu cầu (*Review Offer).*

Đầu vào: Mô hình BPMN của luồng quy trình xin việc



Hình 4.17: Mô hình BPMN của luồng quy trình xin việc (apply for job).

Thể hiện dạng xml:

```
<process id="myProcess" name="My process" isExecutable="true">
  <startEvent id="startevent1" name="Start"></startEvent>
  <userTask id="SubmitApplication" name="Submit Application"
activiti:assignee="kermit">
    <documentation>Steps by step to apply for a job: Enter your email address-&gt;
Attach a job application form -&gt; Click the completed application
button.</documentation>
    <extensionElements>
      <activiti:formProperty id="mail" name="Input the email address" type="string"
variable="mail" required="true"></activiti:formProperty>
    </extensionElements>
  </userTask>
  <sequenceFlow id="flow1" sourceRef="startevent1"
targetRef="SubmitApplication"></sequenceFlow>
  <userTask id="QualifyApplication" name="Qualify Application"
activiti:assignee="gonzo">
    <documentation>After the candidate submits the application, the recruiter will
review the application is successful or not? If not: Send mail rejected application and
end process flow. If successful: Notice to Candidate PV time and move on to the next
step: Candidates interviewed</documentation>
    <extensionElements>
      <activiti:formProperty id="qualify" name="Qualify Application?" type="enum"
variable="qualify" required="true">
        <activiti:value id="OK" name="eligible"></activiti:value>
        <activiti:value id="NOK" name="ineligible"></activiti:value>
      </activiti:formProperty>
```

```
    </extensionElements>
  </userTask>
  <sequenceFlow id="flow2" sourceRef="SubmitApplication"
targetRef="QualifyApplication"></sequenceFlow>
  <exclusiveGateway id="CalledForInterview" name="Called for
Interview?"></exclusiveGateway>
  <sequenceFlow id="flow3" sourceRef="QualifyApplication"
targetRef="CalledForInterview"></sequenceFlow>
  <userTask id="AttendInterview" name="Attend interview"
activiti:assignee="kermit"></userTask>
  <sequenceFlow id="CalledInterview_OK" name="OK"
sourceRef="CalledForInterview" targetRef="AttendInterview">
    <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${qualify=='OK'}]]></conditionExpressi
on>
  </sequenceFlow>
  <exclusiveGateway id="Interview" name="Interview
sucessfull?"></exclusiveGateway>
  <userTask id="ReviewOffer" name="Review Offer" activiti:assignee="kermit">
    <documentation>Candidates consider the proposal to sign labor contracts of
employer</documentation>
    <extensionElements>
      <activiti:formProperty id="offer" name="Review Offer" type="enum"
variable="offer" required="true">
        <activiti:value id="OK" name="Agree"></activiti:value>
        <activiti:value id="NOK" name="Disagree"></activiti:value>
        <activiti:value id="Clarify" name="Need clarify"></activiti:value>
      </activiti:formProperty>
    </extensionElements>
  </userTask>
  <sequenceFlow id="InterviewSucessfull" name="OK" sourceRef="Interview"
targetRef="ReviewOffer">
    <conditionExpression xsi:type="tFormalExpression"><![CDATA[${evaluate
=='OK'}]]></conditionExpression>
  </sequenceFlow>
  <endEvent id="endevent1" name="End"></endEvent>
  <sequenceFlow id="flow8" sourceRef="SubmitAcceptancesForm"
targetRef="endevent1"></sequenceFlow>
  <exclusiveGateway id="Consider" name="Accept Offer?"></exclusiveGateway>
  <sequenceFlow id="flow9" sourceRef="ReviewOffer"
targetRef="Consider"></sequenceFlow>
  <sequenceFlow id="ConsiderNOK" name="NOK" sourceRef="Consider"
targetRef="mailtask2">
    <conditionExpression xsi:type="tFormalExpression"><![CDATA[${offer
=='NOK'}]]></conditionExpression>
  </sequenceFlow>
  <sequenceFlow id="flow12" sourceRef="mailtask2"
targetRef="endevent1"></sequenceFlow>
```

```
   <userTask id="MakeAppointment" name="Make appointment"
activiti:assignee="gonzo">
     <extensionElements>
       <activiti:formProperty id="time" name="Time" type="date" variable="time"
required="true"></activiti:formProperty>
       <activiti:formProperty id="location" name="Location" type="string"
variable="location" required="true"></activiti:formProperty>
     </extensionElements>
   </userTask>
   <sequenceFlow id="Clarification" name="Clarification needed"
sourceRef="Consider" targetRef="MakeAppointment">
     <conditionExpression xsi:type="tFormalExpression"><![CDATA[${offer
=='Clarify'}]]></conditionExpression>
   </sequenceFlow>
   <userTask id="DisscussOffer" name="Disscuss offer Again"
activiti:assignee="kermit"></userTask>
   <sequenceFlow id="flow14" sourceRef="MakeAppointment"
targetRef="DisscussOffer"></sequenceFlow>
   <sequenceFlow id="InterviewFail" name="NOK" sourceRef="Interview"
targetRef="mailtask1">
     <conditionExpression xsi:type="tFormalExpression"><![CDATA[${evaluate
=='NOK'}]]></conditionExpression>
   </sequenceFlow>
   <serviceTask id="mailtask1" name="Interview failed" activiti:type="mail">
     <extensionElements>
       <activiti:field name="to">
         <activiti:expression><![CDATA[${mail}]]></activiti:expression>
       </activiti:field>
       <activiti:field name="from">
         <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
       </activiti:field>
       <activiti:field name="subject">
         <activiti:string><![CDATA[Interview failed]]></activiti:string>
       </activiti:field>
       <activiti:field name="html">
         <activiti:string><![CDATA[Dear Sir/Madam,
We are sorry that you did not fit my current job vacancies.
We have other opportunities to cooperate with you.
Thanks & Best reagards,
Huyen Duong]]></activiti:string>
       </activiti:field>
     </extensionElements>
   </serviceTask>
   <serviceTask id="SubmitAcceptancesForm" name="Submit Acceptance Form"
activiti:type="mail">
     <extensionElements>
       <activiti:field name="from">
         <activiti:expression><![CDATA[${mail}]]></activiti:expression>
```

```
      </activiti:field>
      <activiti:field name="to">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
      </activiti:field>
      <activiti:field name="subject">
        <activiti:string><![CDATA[Accept offer]]></activiti:string>
      </activiti:field>
      <activiti:field name="html">
        <activiti:string><![CDATA[Submit acceptance form]]></activiti:string>
      </activiti:field>
    </extensionElements>
  </serviceTask>
  <sequenceFlow id="flow26" sourceRef="mailtask1"
targetRef="endevent1"></sequenceFlow>
  <sequenceFlow id="CalledInterview_NOK" name="NOK"
sourceRef="CalledForInterview" targetRef="endevent1">
    <conditionExpression
xsi:type="tFormalExpression"><![CDATA[${qualify=='NOK'}]]></conditionExpres
sion>
  </sequenceFlow>
  <sequenceFlow id="ConsiderOK" name="OK" sourceRef="Consider"
targetRef="SubmitAcceptancesForm">
    <conditionExpression xsi:type="tFormalExpression"><![CDATA[${offer
=='OK'}]]></conditionExpression>
  </sequenceFlow>
  <sequenceFlow id="flow28" sourceRef="DisscussOffer"
targetRef="ReviewOffer"></sequenceFlow>
  <userTask id="evaluating" name="Evaluating the results of the interview"
activiti:assignee="gonzo">
    <documentation>After the interview, the interviewer evaluates the interview
results: Pass / Fail</documentation>
    <extensionElements>
      <activiti:formProperty id="evaluate" name="Evaluate the results of the
Interview" type="enum" variable="evaluate" required="true">
        <activiti:value id="OK" name="Pass"></activiti:value>
        <activiti:value id="NOK" name="Fail"></activiti:value>
      </activiti:formProperty>
    </extensionElements>
  </userTask>
  <sequenceFlow id="flow29" sourceRef="AttendInterview"
targetRef="evaluating"></sequenceFlow>
  <sequenceFlow id="flow30" sourceRef="evaluating"
targetRef="Interview"></sequenceFlow>
  <serviceTask id="mailtask2" name="Reject offer" activiti:type="mail">
    <extensionElements>
      <activiti:field name="to">
        <activiti:string><![CDATA[duonghuyen0805@gmail.com]]></activiti:string>
      </activiti:field>
```

```
      <activiti:field name="from">
        <activiti:expression><![CDATA[${mail}]]></activiti:expression>
      </activiti:field>
      <activiti:field name="subject">
        <activiti:string><![CDATA[Reject offer]]></activiti:string>
      </activiti:field>
      <activiti:field name="html">
        <activiti:string><![CDATA[Reject offer]]></activiti:string>
      </activiti:field>
    </extensionElements>
  </serviceTask>
  <textAnnotation id="Notation1_CalledForInterView">
    <text>Called for Interview?</text>
  </textAnnotation>
  <textAnnotation id="Notation2_InterviewSuccessfull">
    <text>Interview sucessfull?</text>
  </textAnnotation>
  <textAnnotation id="Notation3_AcceptOffer">
    <text>Accept Offer?</text>
  </textAnnotation>
  <association id="association1" sourceRef="usertask1"
targetRef="Notation2_InterviewSuccessfull"></association>
 </process>
```

Hình 4.18: Mô hình BPMN dạng xml của luồng quy trình xin việc

Kết quả đầu ra: dạng file *.txt

ApplyForJob_Loop.testcase

```
1  Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview,
2                                                                                    Interview sucessfull?, Review Offer, Accept Offer?, Reject offer, End
3  Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview,
4                                                                                 Interview sucessfull?, Review Offer, Accept Offer?, Submit Acceptance Form, End
5  Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview,
6                                                                                                  Interview sucessfull?, Interview failed, End
7  Start, Submit Application, Qualify Application, Called for Interview?, End
8  Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview,
9                          Interview sucessfull?, Review Offer, Accept Offer?, Make appointment, Disscuss offer Again, Review Offer, Accept Offer?, Reject offer, End
10 Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview,
11                 Interview sucessfull?, Review Offer, Accept Offer?, Make appointment, Disscuss offer Again, Review Offer, Accept Offer?, Submit Acceptance Form, End
```

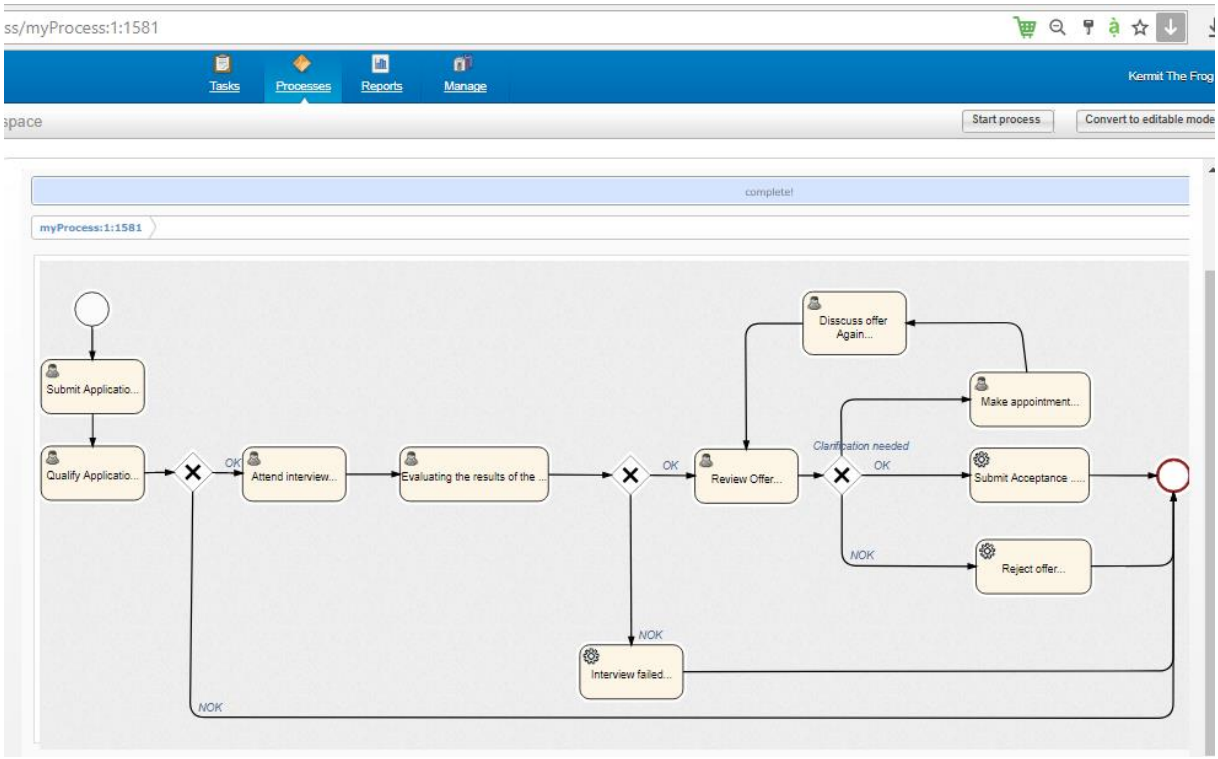
Hình 4.19: Kịch bản ca kiểm thử của luồng nghiệp vụ xin việc

Cụ thể gồm các kịch bản kiểm thử sau:

STT	Kịch bản kiểm thử
1	Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview, Interview sucessfull?, Review Offer, Accept Offer?, Reject offer, End
2	Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview, Interview

STT	Kịch bản kiểm thử
	sucessfull?, Review Offer, Accept Offer?, Submit Acceptance Form, End
3	Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview, Interview sucessfull?, Interview failed, End
4	Start, Submit Application, Qualify Application, Called for Interview?, End
5	Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview, Interview sucessfull?, Review Offer, Accept Offer?, Make appointment, Disscuss offer Again, Review Offer, Accept Offer?, Reject offer, End
6	Start, Submit Application, Qualify Application, Called for Interview?, Attend interview, Evaluating the results of the interview, Interview sucessfull?, Review Offer, Accept Offer?, Make appointment, Disscuss offer Again, Review Offer, Accept Offer?, Submit Acceptance Form, End

- ***Thực thi các ca kịch bản kiểm thử trên Activiti- explorer. Khi import mô hình quy trình xin việc lên công cụ activiti sẽ có giao diện như sau:***



Hình 4.20: Import bpmn lên công cụ activiti

- ***Kịch bản ca kiểm thử thứ nhất:*** ứng viên phỏng vấn thành công nhưng từ chối đề nghị từ nhà tuyển dụng.

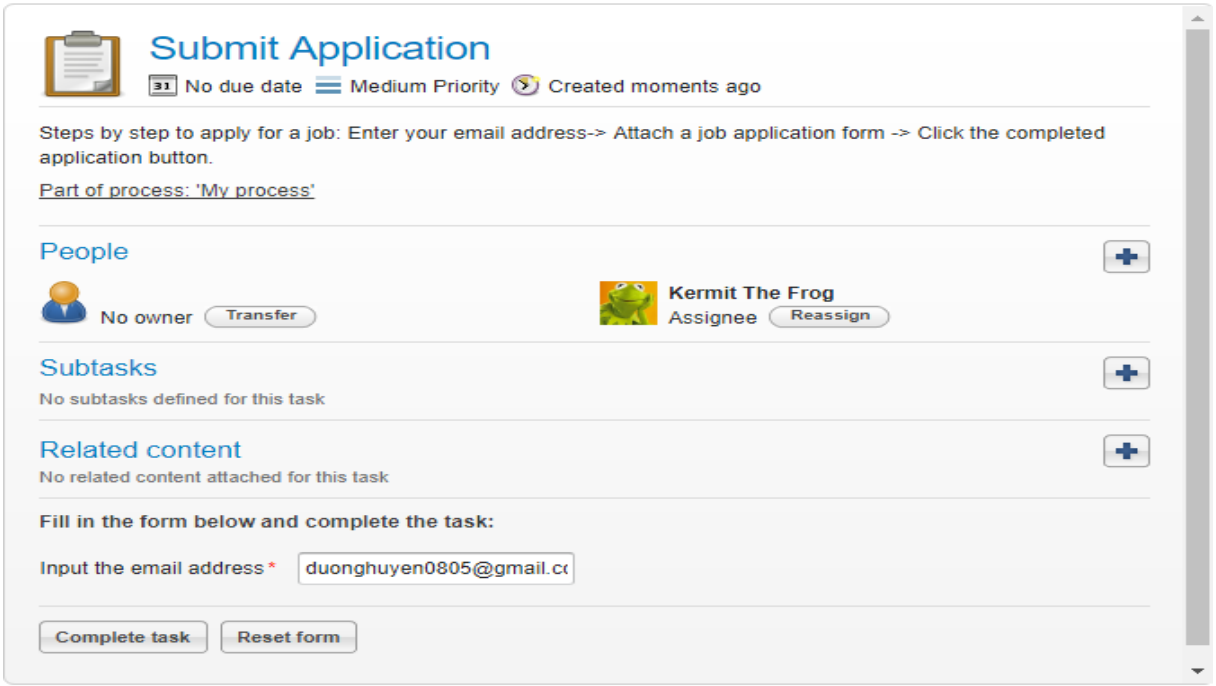
Kịch bản kiểm thử
Bước 1: Start, Bước 2: Submit Application, Bước 3: Qualify Application, Bước 4: Called for Interview? Bước 5: Attend interview, Bước 6: Evaluating the results of the interview, Bước 7: Interview sucessfull? Bước 8: Review Offer, Bước 9: Accept Offer? Bước 10: Reject offer, Bước 11: End.

Bước 1: Start

- Đăng nhập với tài khoản của ứng viên – tài khoản Kermit.
- Thực hiện import bpmn model lên activity
- Click chọn button Start process

Bước 2: Submit Application

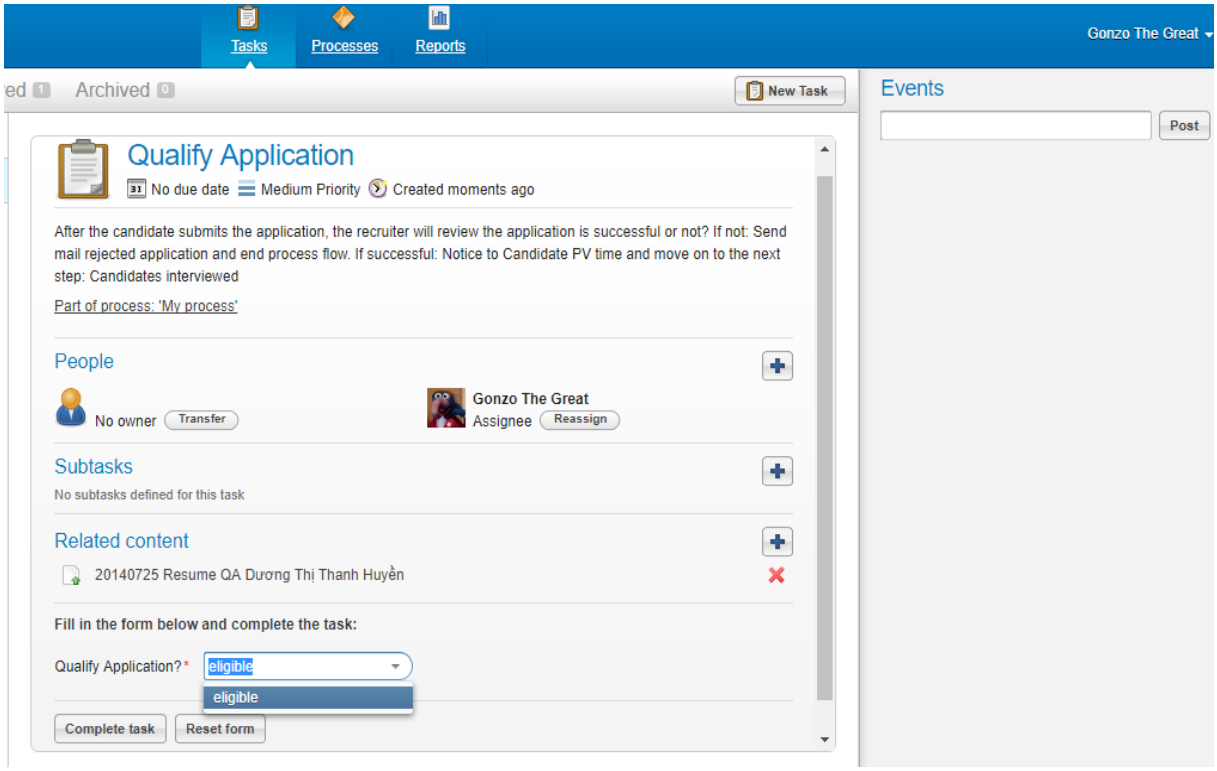
- Nhập địa chỉ email
- Đính kèm file thông tin đơn xin việc chi tiết
- Ấn chọn nút completed task để gửi đơn xin việc



Hình 4.21: Bước Submit application của kịch bản ca kiểm thử thứ nhất

Bước 3: Qualify Application

- Đăng nhập với tài khoản nhà tuyển dụng (Gonzo) để xét duyệt đơn xin việc có đạt yêu cầu hay không.
- Sau đó chọn đủ điều kiện (eligible) và ấn chọn nút "Complete task".



Hình 4.22: Bước Qualify Application của kịch bản ca kiểm thử thứ nhất

Bước 4: Called for Interview?

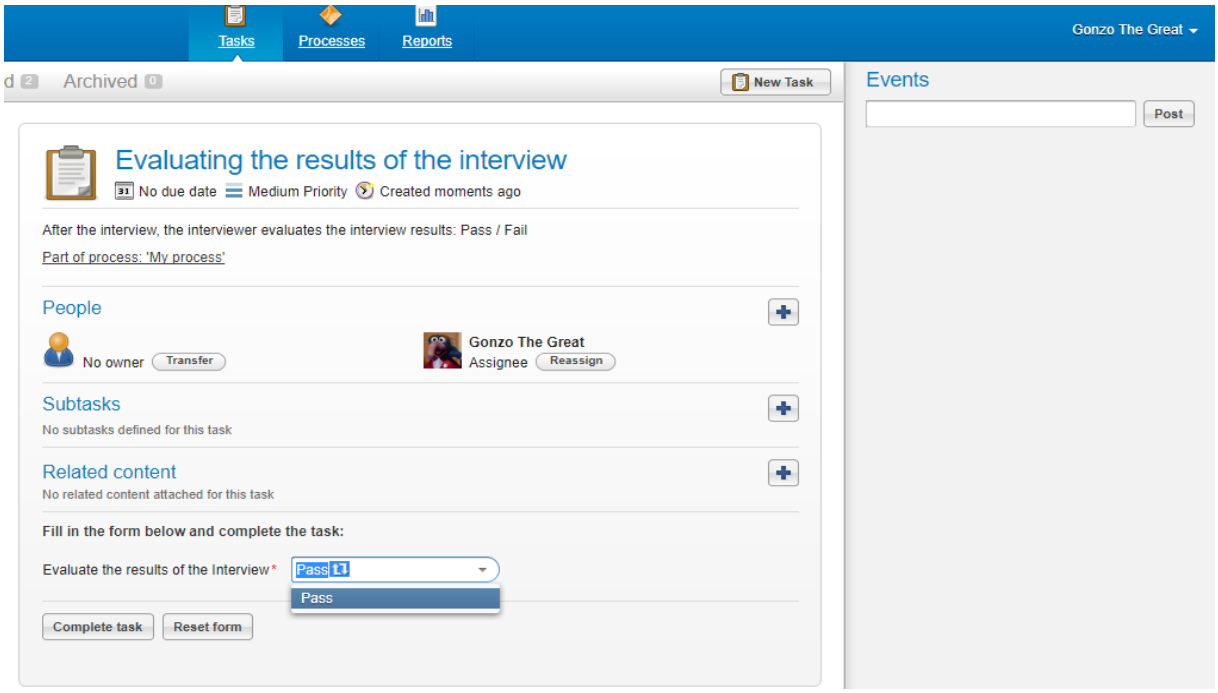
- Tại gateway "Called for Interview?" kiểm tra dữ liệu từ bước 3 để xác định đơn xin việc có đủ điều kiện hay không. Trong trường hợp này dữ liệu tại bước 3 đã chọn đơn xin việc đủ điều kiện nên sẽ chuyển sang bước tiếp theo tham gia phỏng vấn (attend interview)

Bước 5: Attend interview

- Thực hiện đăng nhập với tài khoản người tuyển dung (kermit) và ấn chọn nút "Complete task" để hoàn thành buổi phỏng vấn

Bước 6: Evaluating the results of the interview

- Đăng nhập với tài khoản của nhà tuyển dụng (gonzo) để đánh giá kết quả phỏng vấn.
- Chọn "Pass" để đánh giá kết quả phỏng vấn của ứng viên đạt yêu cầu nhà tuyển dụng



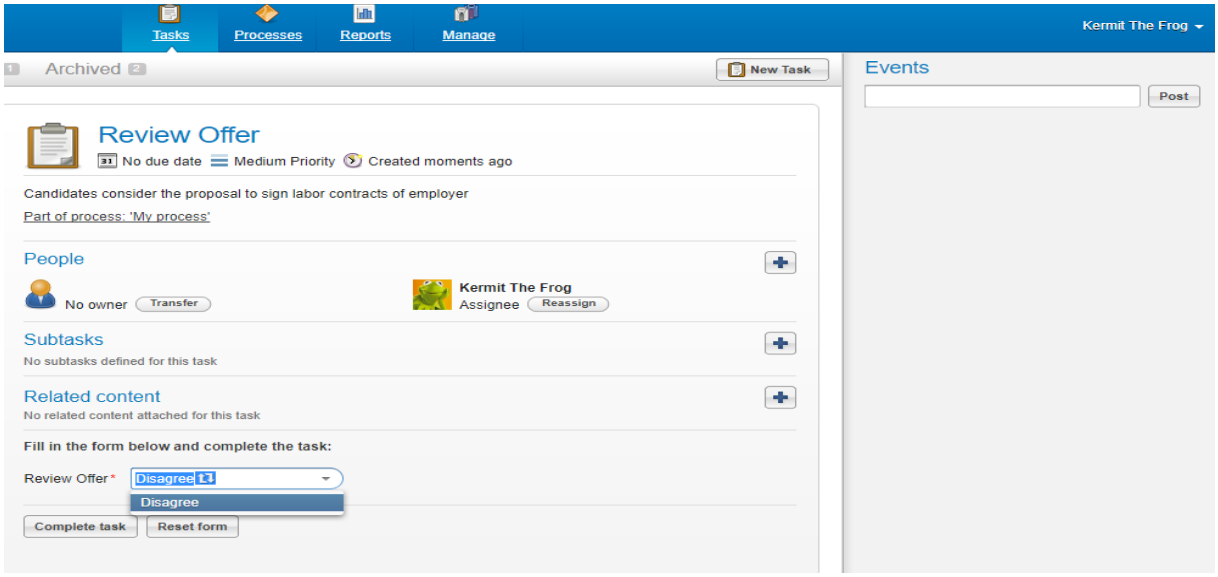
Hình 4.23: Bước đánh giá kết quả phỏng vấn của kịch bản kiểm thử thứ nhất

Bước 7: Interview sucessfull?

- Tại gateway "Interview sucessfull?" kiểm tra dữ liệu từ bước 6 để xác định kết quả phỏng vấn có đạt hay không? Trong trường hợp này dữ liệu tại bước 6 đã chọn kết quả phỏng vấn đạt yêu cầu nên sẽ chuyển sang bước tiếp theo: ứng viên cân nhắc đề nghị ký hợp đồng lao động từ nhà tuyển dụng (*Review Offer)*

Bước 8: Review Offer

- Ứng viên cân nhắc đề nghị của nhà tuyển dụng. Tại bước này ứng viên chọn không đồng ý (disagree)



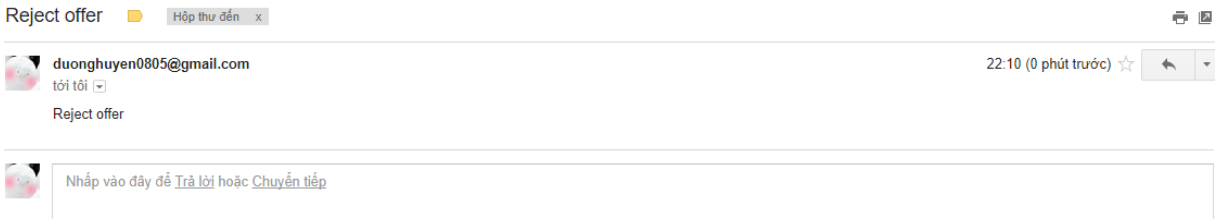
Hình 4.24: Bước review offer của kịch bản ca kiểm thử thứ nhất

Bước 9: Accept Offer?

- Tại gateway "Accept Offer" kiểm tra dữ liệu từ bước 8 là để xác định theo trong mô hình. Trong trường hợp này , tại bước 8 ứng viên đã chọn từ chối đề nghị (Reject offer) của nhà tuyển dụng .

Bước 10: Reject offer,

- Sau khi ứng viên từ chối đề nghị của nhà tuyển dụng, một email được gửi đến nhà tuyển dụng:



Hình 4.25: Mail từ chối đề nghị của kịch bản ca kiểm thử thứ nhất

Bước 11: End.

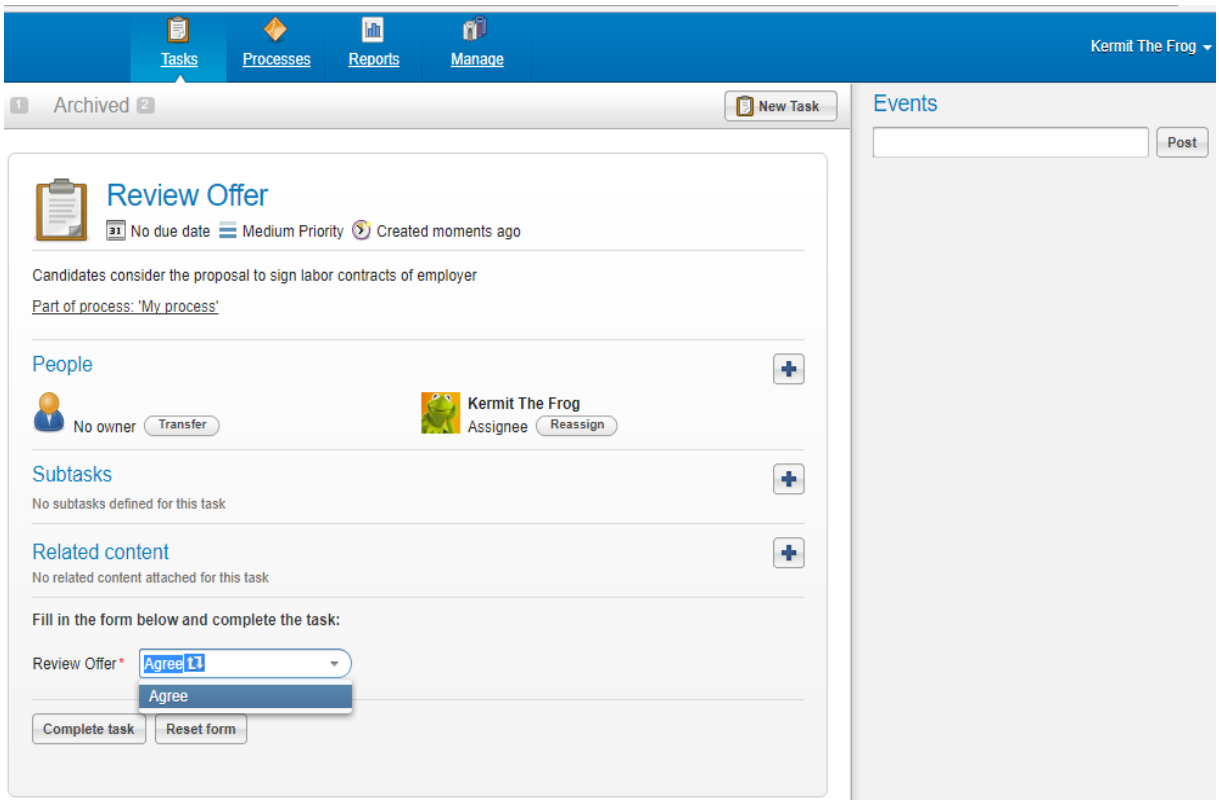
- ***Kịch bản ca kiểm thử thứ hai:*** ứng viên phỏng vấn thành công và đồng ý với đề nghị của nhà tuyển dụng.

Kịch bản kiểm thử
Bước 1: Start, Bước 2: Submit Application, Bước 3: Qualify Application, Bước 4: Called for Interview? Bước 5: Attend interview, Bước 6: Evaluating the results of the interview, Bước 7: Interview sucessfull? Bước 8: Review Offer, Bước 9: Accept Offer? Bước 10: Submit Acceptance Form, Bước 11: End.

Các bước từ 1 đến 7 thực hiện giống như ca kiểm thử thứ nhất.

Bước 8: Review Offer

- Ứng viên cân nhắc đề nghị của nhà tuyển dụng. Tại bước này ứng viên chọn đồng ý (Agree)



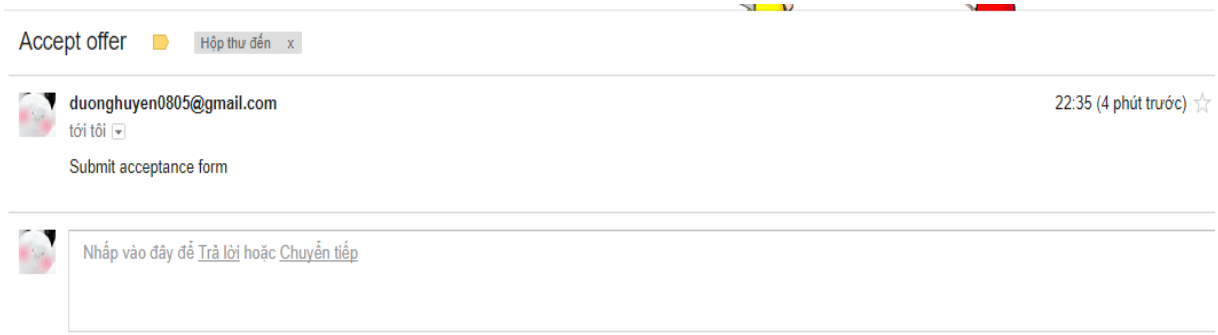
Hình 4.26: Bước Review Offer của kịch bản ca kiểm thử thứ hai

Bước 9: Accept Offer?

- Tại gateway "Accept Offer" kiểm tra dữ liệu từ bước 8 là để xác định theo trong mô hình. Trong trường hợp này , tại bước 8 ứng viên đã chọn đồng ý với đề nghị (Agree) của nhà tuyển dụng .

Bước 10: Submit Acceptance Form

- Sau khi ứng viên đồng ý đề nghị của nhà tuyển dụng, một email được gửi đến nhà tuyển dụng:



Hình 4.27: Email accept offer của kịch bản kiểm thử thứ hai

Bước 11: End.

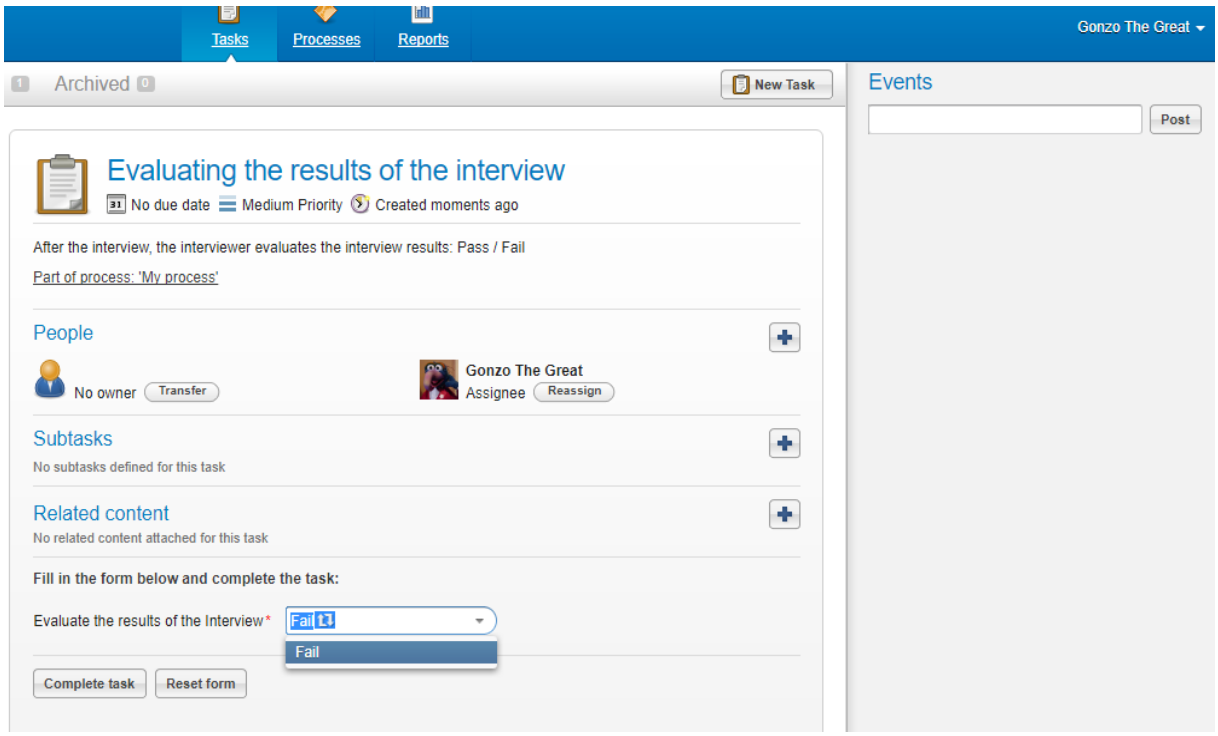
- **Kịch bản ca kiểm thử thứ ba:** ứng viên phỏng vấn không thành công

Kịch bản kiểm thử
Bước 1: Start, Bước 2: Submit Application, Bước 3: Qualify Application, Bước 4: Called for Interview? Bước 5: Attend interview, Bước 6: Evaluating the results of the interview, Bước 7: Interview sucessfull? Bước 8: Interview failed, Bước 9: End.

Các bước từ 1 đến 5 thực hiện theo đúng kịch bản ca kiểm thử đầu tiên.

Bước 6: Evaluating the results of the interview

- Đăng nhập với tài khoản của nhà tuyển dụng (gonzo) để đánh giá kết quả phỏng vấn.
- Chọn "Fail" để đánh giá kết quả phỏng vấn của ứng viên không đạt yêu cầu nhà tuyển dụng



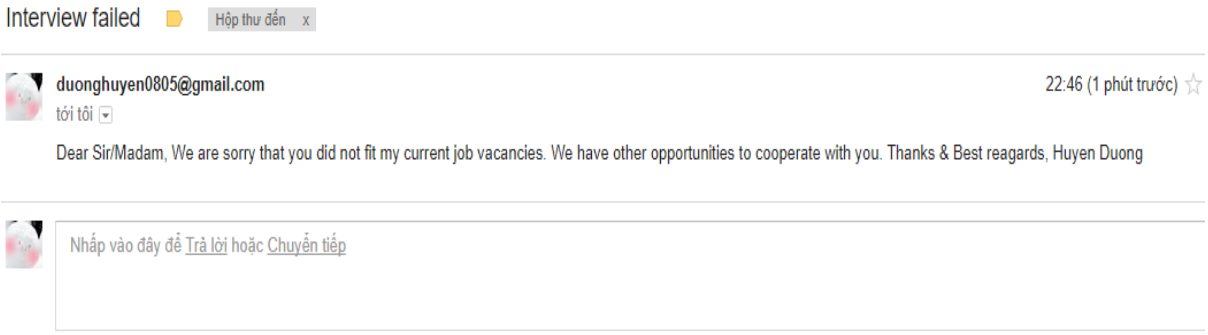
Hình 4.28: Bước đánh giá kết quả phỏng vấn của kịch bản kiểm thử thứ ba

Bước 7: Interview sucessfull?

- Tại gateway "Interview sucessfull?" kiểm tra dữ liệu từ bước 6 để xác định kết quả phỏng vấn có đạt hay không? Trong trường hợp này dữ liệu tại bước 6 đã chọn kết quả phỏng vấn không đạt yêu cầu.

Bước 8: Interview failed

- Một email được gửi tới ứng viên



Hình 4.29: Email thông báo của kịch bản ca kiểm thử thứ ba

Bước 9: End.

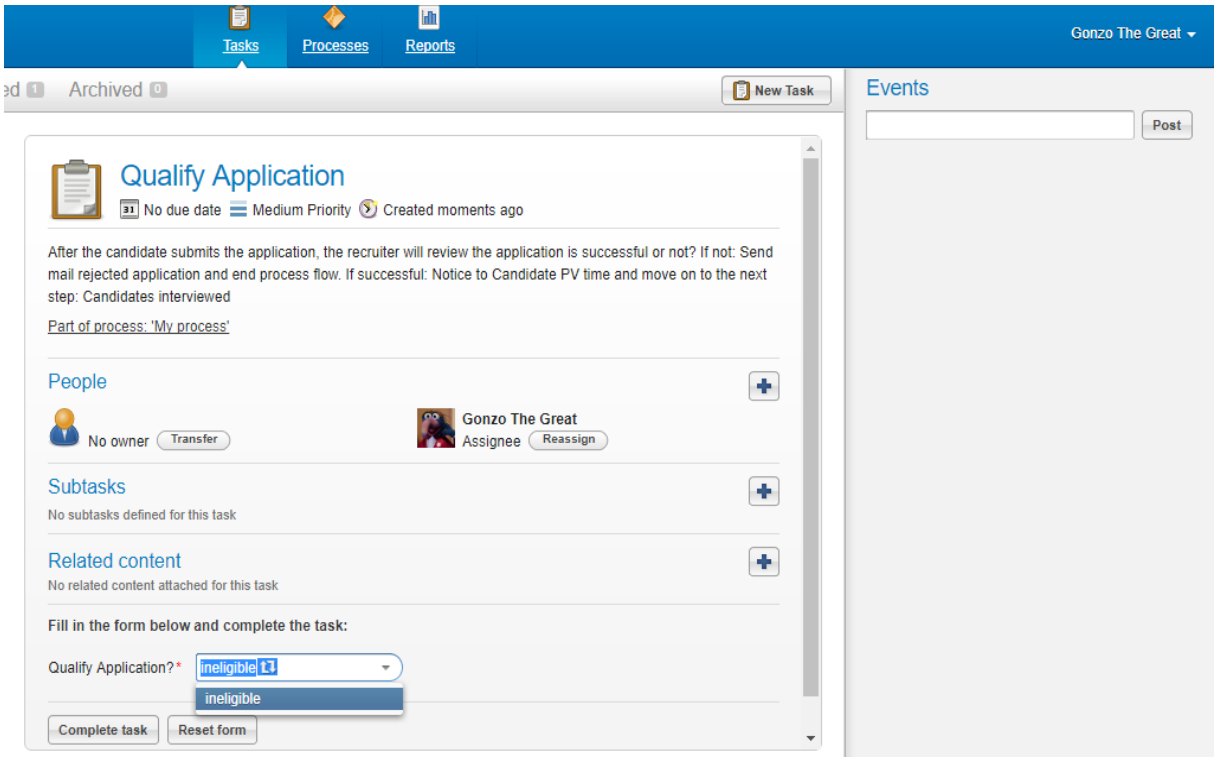
- ***Kịch bản ca kiểm thử thứ tư:*** hồ sơ ứng tuyển của ứng viên không đạt

Kịch bản kiểm thử
Bước 1: Start, Bước 2: Submit Application, Bước 3: Qualify Application, Bước 4: Called for Interview? Bước 5: End.

Các bước 1 và 2 thực hiện theo kịch bản ca kiểm thử đầu tiên.

Bước 3: Qualify Application

- Đăng nhập với tài khoản nhà tuyển dụng (*Gonzo*) để xét duyệt đơn xin việc có đạt yêu cầu hay không.
- Sau đó chọn không đủ điều kiện (*ineligible*) và ấn chọn nút "*Complete task".*



Hình 4.30: Bước Qualify Application của kịch bản ca kiểm thử thứ tư

Bước 4: Called for Interview?

- Tại gateway "Called for Interview?" kiểm tra dữ liệu từ bước 3 để xác định đơn xin việc có đủ điều kiện hay không. Trong trường hợp này dữ liệu tại bước 3 đã chọn đơn xin việc không đủ điều kiện nên sẽ chuyển sang bước tiếp theo kết thúc luồng nghiệp vụ.

Bước 5: End.

- ***Kịch bản ca kiểm thử thứ năm*:** ứng viên phỏng vấn thành công, sau khi nhà tuyển dụng đưa đề nghị ký hợp đồng thì ứng viên cần làm rõ một vài điều kiện trong hợp đồng, nhà tuyển dụng thực hiện đặt lịch và trao đổi lại với ứng viên. Sau đó, ứng viên xem xét kỹ đề nghị và từ chối đề nghị của nhà tuyển dụng.

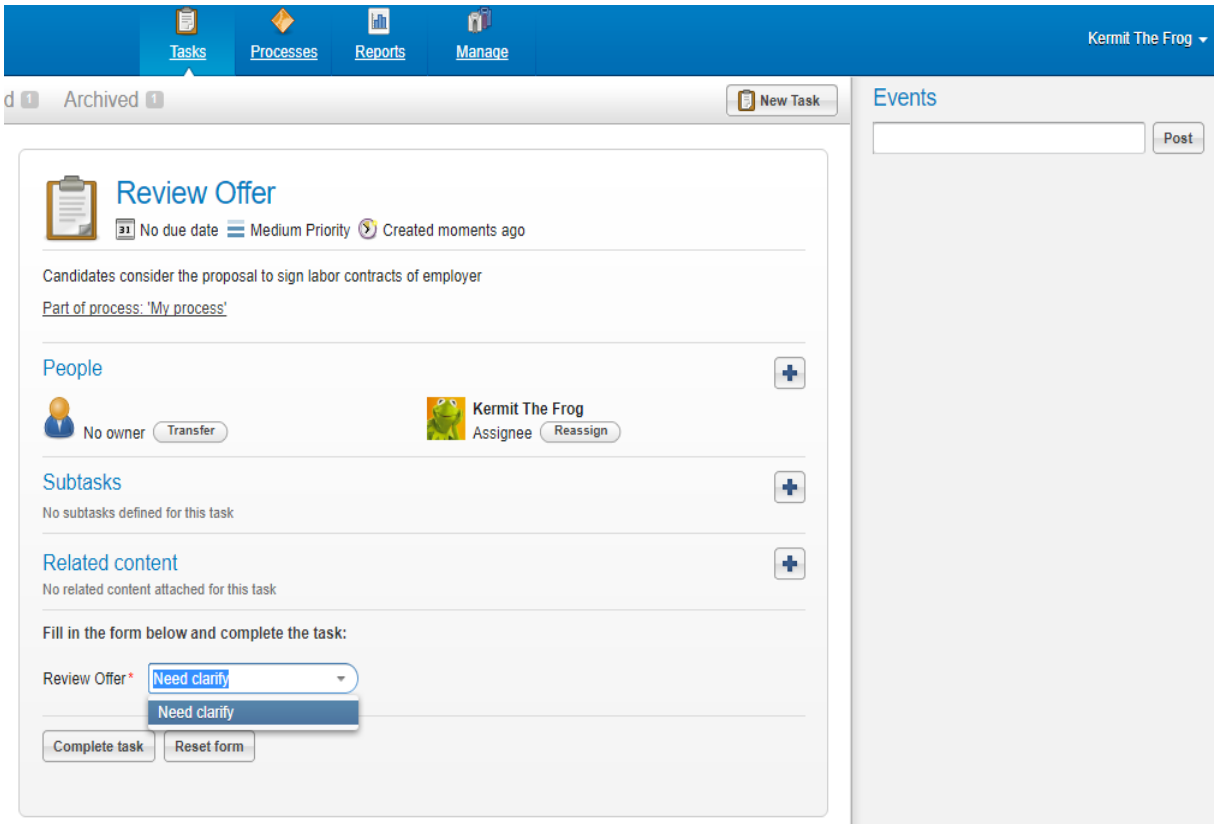
Kịch bản kiểm thử
Bước 1: Start, Bước 2: Submit Application, Bước 3: Qualify Application, Bước 4: Called for Interview? Bước 5: Attend interview, Bước 6: Evaluating the results of the interview, Bước 7: Interview sucessfull? Bước 8: Review Offer,

Kịch bản kiểm thử
Bước 9: Accept Offer? Bước 10: Make appointment, Bước 11: Disscuss offer Again, Bước 12: Review Offer Bước 13: Accept Offer? Bước 14: Reject offer Bước 15: End

Từ bước 1 đến bước 8 thực hiện tương tự theo kịch bản ca kiểm thử đầu tiên.

Bước 9: Accept Offer?

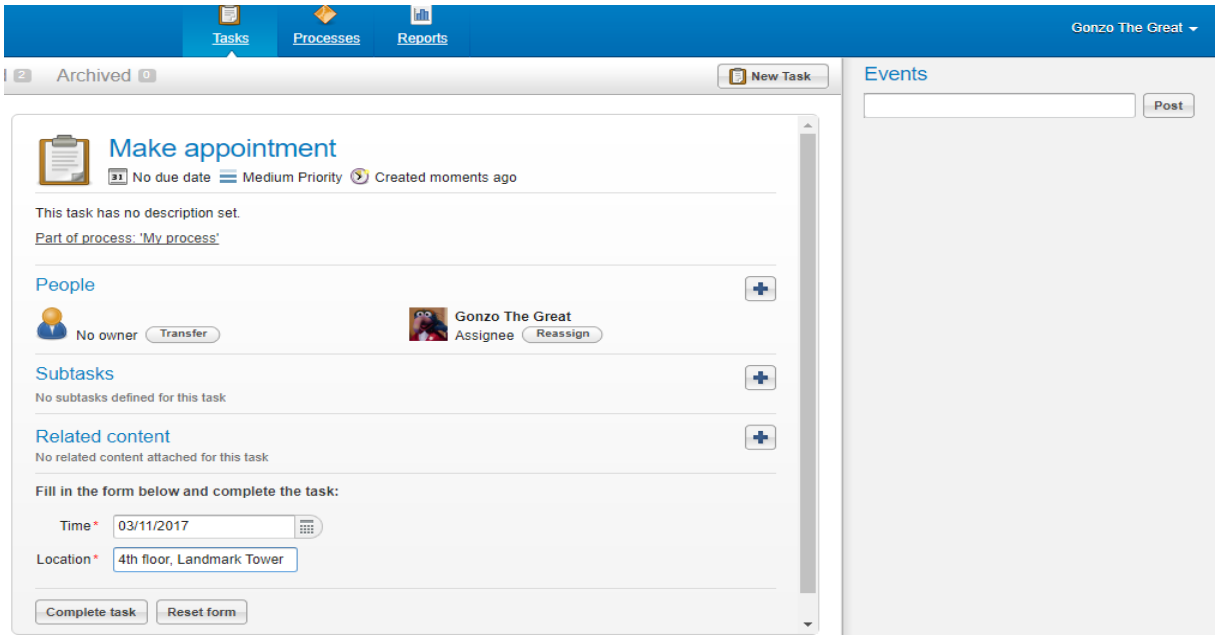
- Đăng nhập với tài khoản ứng viên (Kermit)
- Xem xét đề nghị của nhà tuyển dụng, ứng viên chưa rõ yêu cầu nhà tuyển dụng cần trao đổi thêm nên chọn "Need clarify"



Hình 4.31: Bước Reviw Offer của kịch bản ca kiểm thử thứ tư

Bước 10: Make appointment

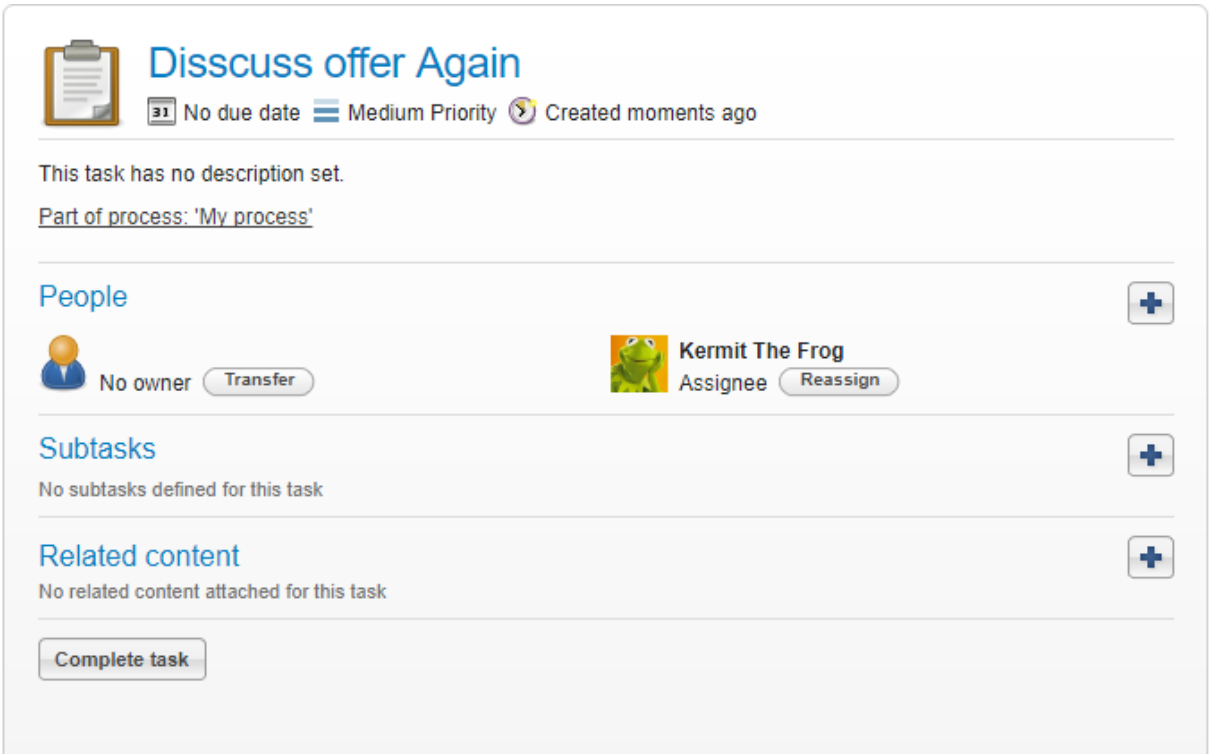
Nhà tuyển dụng đặt lịch hẹn thảo luận với ứng viên



Hình 4.32: Bước Make appointment của kịch bản ca kiểm thử thứ tư

Bước 11: Disscuss offer Again

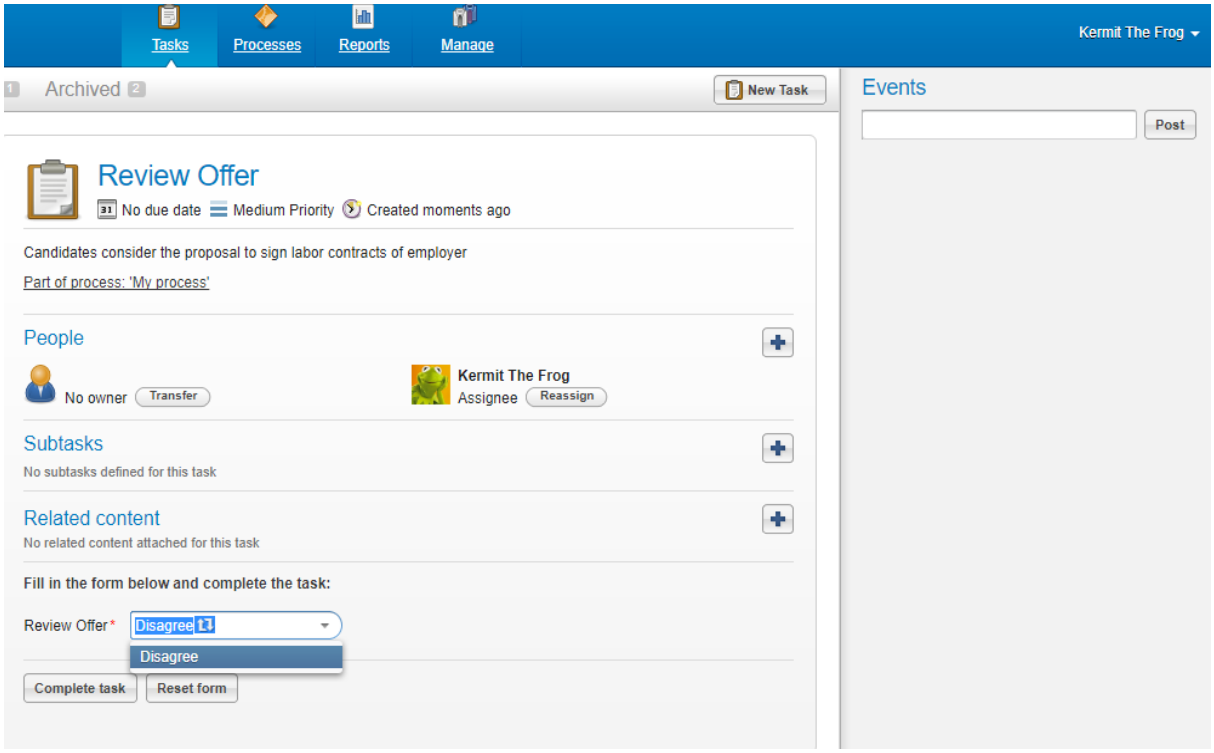
Ứng viên và nhà tuyển dụng tiến hành thảo luận lại các điều kiện để ký hợp đồng



Hình 4.33: Bước discuss offer again của kịch bản ca kiểm thử thứ tư

Bước 12: Review Offer

- Ứng viên cân nhắc đề nghị của nhà tuyển dụng. Tại bước này ứng viên chọn không đồng ý (disagree)



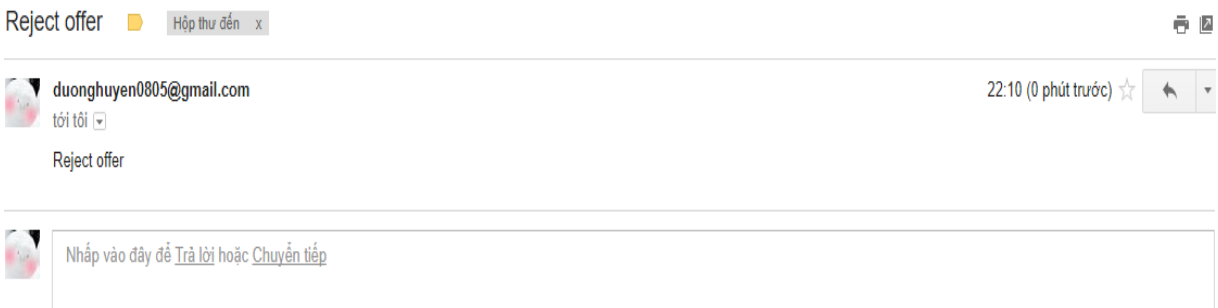
Hình 4.34: Bước Review Offer của kịch bản ca kiểm thử thứ tư

Bước 13: Accept Offer?

- Tại gateway "Accept Offer" kiểm tra dữ liệu từ bước 12 là để xác định theo trong mô hình. Trong trường hợp này , tại bước 12 ứng viên đã chọn từ chối đề nghị (Reject offer) của nhà tuyển dụng .

Bước 14: Reject offer,

- Sau khi ứng viên từ chối đề nghị của nhà tuyển dụng, một email được gửi đến nhà tuyển dụng:



Hình 4.35: Email thông báo của kịch bản ca kiểm thử thứ tư

Bước 15: End.

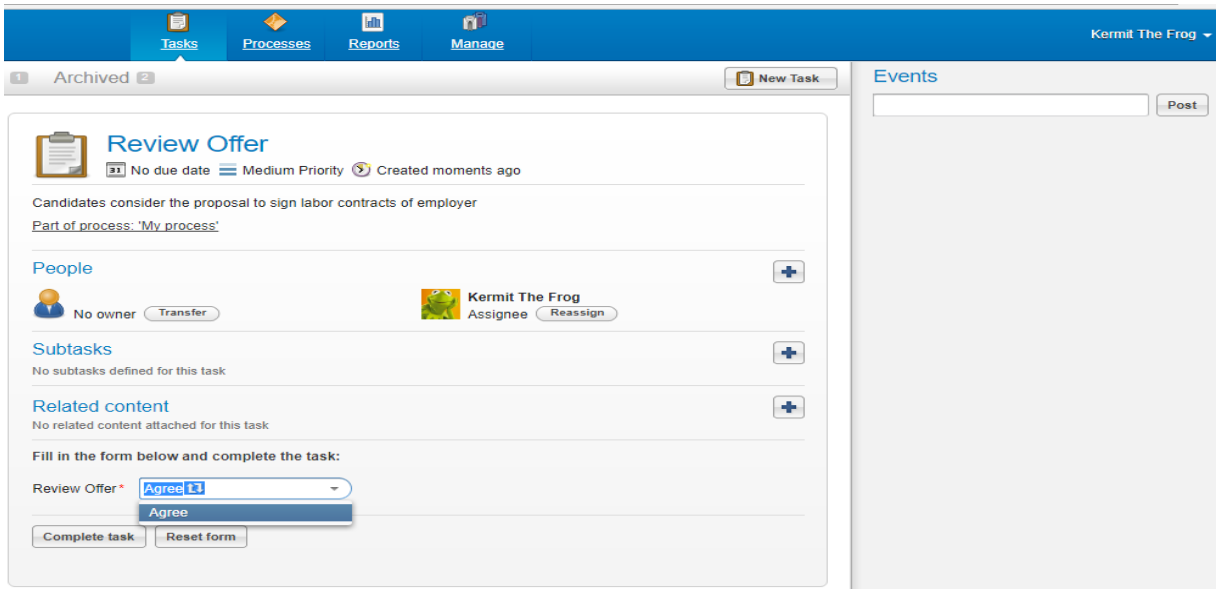
- **Kịch bản ca kiểm thử thứ sáu:** ứng viên phỏng vấn thành công, sau khi nhà tuyển dụng đưa đề nghị ký hợp đồng thì ứng viên cần làm rõ một vài điều kiện trong hợp đồng, nhà tuyển dụng thực hiện đặt lịch và trao đổi lại với ứng viên. Sau đó, ứng viên xem xét kỹ đề nghị và từ chối đề nghị của nhà tuyển dụng.

Kịch bản kiểm thử
Bước 1: Start, Bước 2: Submit Application, Bước 3: Qualify Application, Bước 4: Called for Interview? Bước 5: Attend interview, Bước 6: Evaluating the results of the interview, Bước 7: Interview sucessfull? Bước 8: Review Offer, Bước 9: Accept Offer? Bước 10: Make appointment, Bước 11: Disscuss offer Again, Bước 12: Review Offer Bước 13: Accept Offer? Bước 14: Submit Acceptance Form Bước 15: End

Từ bước 1 đến bước 11 thực hiện tương tự theo kịch bản ca kiểm thử thứ 5.

Bước 12: Review Offer

Ứng viên cân nhắc đề nghị của nhà tuyển dụng. Tại bước này ứng viên chọn đồng ý (Agree)



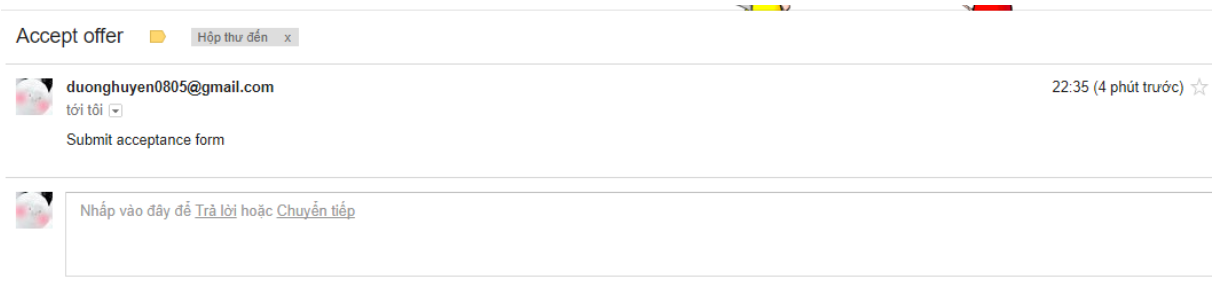
Hình 4.36: Bước Review offer của kịch bản ca kiểm thử thứ năm

Bước 13: Accept Offer?

- Tại gateway “Accept Offer” kiểm tra dữ liệu từ bước 12 là để xác định theo trong mô hình. Trong trường hợp này , tại bước 12 ứng viên đã chọn đồng ý với đề nghị (Agree) của nhà tuyển dụng .

Bước 14: Submit Acceptance Form

- Sau khi ứng viên đồng ý đề nghị của nhà tuyển dụng, một email được gửi đến nhà tuyển dụng:



Hình 4.37: Email thông báo của kịch bản ca kiểm thử thứ năm

Bước 15: End.

4.3 Ý nghĩa thực nghiệm

Kết quả thực nghiệm sinh thành công các kịch bản ca kiểm thử từ mô hình BPMN cho thấy tính khả thi của việc áp dụng phương pháp sinh kịch bản kiểm thử tự động từ mô hình luồng quy trình nghiệp vụ trong phát triển phần mềm. Điều này giúp giảm chi phí về thời gian và nguồn lực, mang lại nhiều hiệu quả cho quá trình kiểm thử. Ngoài ra, do đầu vào của chương trình là biểu đồ luồng quy trình nghiệp vụ trực quan nên các kiểm thử viên có thể thuận lợi quan sát việc thực thi các ca kịch bản kiểm thử. Các ca kịch bản kiểm thử được sinh ra có thể áp dụng để kiểm thử giai đoạn tích hợp chức năng, kiểm thử hệ thống, kiểm thử chấp nhận trong quá trình phát triển phần mềm. Ngoài ra, còn có thể áp dụng để kiểm thử các công cụ quản lý quy trình nghiệp vụ. Tuy công cụ còn hạn chế ở việc chưa sinh được dữ liệu đầu vào cho ca kiểm thử nhưng sẽ là tiền đề cho việc phát triển, mở rộng nhằm hoàn thiện chu trình kiểm thử tự động khép kín dựa trên mô hình luồng quy trình nghiệp vụ.

CHƯƠNG 5: KẾT LUẬN

Sau khi nghiên cứu cơ sở lý thuyết và thực hiện luận văn này, tôi đã thu được nhiều kiến thức mới và có hướng nhìn rõ ràng và cụ thể hơn về phương pháp sinh ca kiểm thử tự động từ các mô hình thực thi được, đặc biệt từ một mô hình thực thi được cụ thể là mô hình BPMN.

Luận văn tập trung nghiên cứu phương pháp sinh các kịch bản kiểm thử từ mô hình hóa quy trình nghiệp vụ thông qua ngôn ngữ mô hình hóa BPMN, từ đó đề xuất phương pháp áp dụng kiểm thử tự động dựa trên mô hình BPMN trong công nghiệp phần mềm.

Qua quá trình nghiên cứu, thực nghiệm, luận văn đạt được các kết quả chính sau:

- Tìm hiểu tổng quan về kiểm thử dựa trên mô hình. Trình bày một số phương pháp kiểm thử dựa trên mô hình, những thuận lợi và khó khi áp dụng phương pháp này trong kiểm thử tự động.
- Cung cấp cái nhìn tổng quan về bài toán mô hình hóa quy trình nghiệp vụ và ngôn ngữ mô hình hóa BPMN. Đồng thời giới thiệu các công cụ quản lý quy trình nghiệp vụ có thể hỗ trợ thiết kế và thực thi mô hình BPMN, trực quan mô hình BPMN được thiết kế và thực thi trên công cụ Activiti.
- Luận văn đóng góp phương pháp sinh kịch bản ca kiểm thử từ mô hình BPMN dựa trên việc duyệt tất cả các nhánh nghiệp vụ của mô hình BPMN và sinh ra các ca kịch bản kiểm thử tương ứng. Kết quả đầu ra có thể được áp dụng cho quá trình kiểm thử tích hợp, kiểm thử hệ thống và kiểm thử chấp nhận trong phát triển phần mềm. Ngoài ra, kết quả đầu ra cũng được áp dụng để kiểm thử tính đúng đắn của các công cụ quản lý quy trình nghiệp vụ Activiti.

Việc áp dụng ngôn ngữ mô hình hóa quy trình BPMN mang lại nhiều hiệu quả lợi ích trong việc xây dựng, thực hiện và quản lý quy trình của các cơ quan tổ chức. Đặc biệt hỗ trợ tốt cho quá trình sinh ca kiểm thử tự động từ mô hình. Tuy nhiên trong quá trình làm luận văn này, tôi nhận thấy bên cạnh những ưu điểm của BPMN thì áp dụng mô hình này khi sinh ca kiểm thử hiện cũng có hạn chế là chưa sinh được dữ liệu đầu vào tự động cho ca kiểm thử, do BPMN là mô hình quy trình nghiệp vụ tổng quát nên hiện chưa có ràng buộc cụ thể chi tiết về mặt dữ liệu có thể sinh dữ liệu đầu vào cho ca kiểm thử.

Để có thể có tính ứng dụng cao hơn trong thực tế, tôi đưa ra một số đề xuất và định hướng nghiên cứu tiếp theo như sau:

- Tiếp tục hoàn thiện chương trình sinh ca kiểm thử từ ngôn ngữ mô hình hóa BPMN. Do quỹ thời gian có hạn, chương trình chưa thật sự hoàn hảo ở việc chưa sinh được dữ liệu kiểm thử đầu vào tự động cho ca kiểm thử. Hướng nghiên cứu tiếp theo là tìm hiểu và áp dụng các tiền điều kiện và hậu điều kiện cho các lớp đối tượng mở rộng của mô hình BPMN, từ đó có thể áp dụng các ngôn ngữ ràng buộc đối tượng như OCL trong việc sinh dữ liệu đầu vào cho các ca kiểm thử.
- Nghiên cứu và xây dựng phương pháp sinh ca kiểm thử cho các công cụ quản lý quy trình nghiệp vụ. Phạm vi của luận văn này mới áp dụng sinh ca kiểm thử cho mô hình BPMN được thiết kế và thực thi trên công cụ quản lý quy trình nghiệp vụ Activiti. Từ đó áp dụng kiểm thử tính đúng đắn của công cụ quản lý quy trình nghiệp vụ Activiti. Định hướng tiếp theo tôi sẽ tập trung tìm hiểu và mở rộng phạm vi áp dụng của chương trình ứng dụng cho các công cụ quản lý quy trình nghiệp vụ khác.

TÀI LIỆU THAM KHẢO

Tiếng Việt:

[1] Phạm Ngọc Hùng, Trương Anh Hoàng, Đặng Văn Hưng (2014), "*Giáo trình kiểm thử phần mềm"*, Nhà xuất bản giáo dục Việt Nam.

[2] Nguyễn Văn Hòa, "*Phương pháp sinh dữ liệu kiểm thử tự động từ các biểu đồ tuần tự UML, biểu đồ lớp và ràng buộc OCL*", Luận văn thạc sĩ Đại học Công Nghệ- Đại học Quốc Gia Hà Nội.

Tiếng Anh:

[3] Mark Utting, Bruno Legeard. (2007), *Practical Model – Based Testing: A Tools Approach*, Elsevier Inc.

[4] AnneKe K., Jos W., Wim B. (2003), *MDA Explained: The Model Driven Architecture: Practice and Promise*, Addison Wesley, United States.

[5] Vinaya Sawant and Ketan Shah.Automatic (2011). *Generation of Test Cases from UML Models. International Conference on Technology Systems and Management (ICTSM)*. International Jounal of Computer Appilication,.

[6] W. Linzhang, Y. Jiesong, Y. Xiaofeng, H. Jun, L. Xuandong, and Z. Guoliang. (2004), *Generating test cases from UML activity diagram based on gray-box metho*d. In 11th Asia-Pacific Software Engineering Conference (APSEC04), pp. 284-291.

[7] C. Mingsong, Q. Xiaokang, and L. Xuandong. (2006), *Automatic test case generation for UML activity diagrams.* In 2006 international workshop on Automation of software test, pp. 2-8.

[8] Debasish Kundu and Debaisis Samanta.(2008), *Journal of Object Technology*. In Chair of Software Engineering.

[9] Lilly Raamesh1 and G. V. Uma. (2010) *Reliable Mining of Automatically Generated Test Cases from Software Requirements Specification*. International Journal of Computer Science Issues.

[10] Paul C. Jorgensen. (2009) Modeling Software Behavior: A Craftsman's Approach. Auerbach Publications.

[11] Dipl.-Ing. Tanja Mayerhofer, BSc. (2014), *Defining Executable Modeling Languages with fUML*

[12] Marco Brambilla, Jordi Cabot, Manuel Wimmer (2012), *Model –Driven Software Engineering in Practice, Austria.*

[13] Marlon Dumas, Marcello La Rosa, Jan Mendling, Hajo A. Reijers (2013), *Fundamentals of Business Process Management*

Nguồn tham khảo khác:

[14] BPMN Poster, http://www.bpmn.org/, 2017-09-01