

Lời cảm ơn

Trước hết, chúng em xin chân thành cảm ơn Khoa Công Nghệ Thông Tin, trường Đại Học Khoa Học Tự Nhiên, TpHCM đã tạo điều kiện cho chúng em thực hiện đề tài tốt nghiệp này.

Chúng em cũng xin gửi lời tri ân chân thành đến Thầy Lâm Quang Vũ và Cô Trần Hạnh Nhi, những người đã tận tình hướng dẫn, chỉ bảo chúng em trong suốt thời gian thực hiện đề tài. Trong quá trình làm việc, nhờ có sự hướng dẫn nhiệt tình của Thầy Lâm Quang Vũ và cách gợi mở vấn đề rất sâu sắc của Cô Trần Hạnh Nhi, chúng em có thể nắm bắt và tìm hiểu vấn đề một cách nhanh chóng.

Chúng con vô cùng cảm ơn Cha, Mẹ đã luôn luôn động viên, nuôi dạy chúng con, cũng như tạo mọi điều kiện thuận lợi về vật chất và tinh thần để chúng con hoàn thành luận văn này.

Chúng em cũng xin chân thành cảm ơn quý Thầy Cô trong Khoa Công Nghệ Thông Tin đã tận tình giảng dạy, trang bị cho chúng em những kiến thức cần thiết trong suốt quá trình học tập tại trường để chúng em có thể hoàn tất luận văn này.

Sinh viên thực hiện

Võ Hữu Phúc – 06HC132

Đào Anh Vũ – 06HC228

Lời nói đầu

Hiện nay, quản lý quy trình công việc nói chung và quản lý hồ sơ công văn nói riêng là nhu cầu cần thiết của những nhà quản lý. Giai đoạn trước năm 2003, hệ thống quản lý quy trình nghiệp vụ còn ở dạng sơ khai, nhà quản lý chủ yếu sử dụng phương pháp truyền thống để thực hiện việc gửi nhận hồ sơ, quản lý phản hồi từ người dùng thông qua phương pháp truyền thống như gọi điện thoại, email, SMS, ...

Tuy nhiên, với nhu cầu ngày càng cao, quỹ thời gian ngày càng hạn hẹp, nhu cầu quản lý tiến độ và trạng thái hiện tại của quy trình công việc lại trở nên hết sức cần thiết. Phương pháp quản lý truyền thống cũng bộc lộ nhiều hạn chế như không phản ánh kịp thời tình trạng của quy trình nghiệp vụ, tốn nhiều thời gian phản hồi từ người gửi đến người nhận và ngược lại, tình trạng chứng thực trên tài liệu theo cách truyền thống diễn ra khá chậm, phải tốn thời gian di chuyển hồ sơ tài liệu qua lại giữa các phòng ban, tốn chi phí cho việc chuyển đổi từ tài liệu dạng giấy in sang dạng số và ngược lại phải tốn chi phí in ấn.

Chính vì những hạn chế trên đây của hệ thống quản lý quy trình công việc truyền thống mà nhóm đã thực hiện nghiên cứu giải pháp nguồn mở cho workflow để quản lý hồ sơ công văn hiệu quả hơn, giải quyết một phần những hạn chế nêu trên và mở ra hướng phát triển cho một hệ thống hoàn thiện hơn.

Trường Đại Học Khoa Học Tự Nhiên
Khoa Công Nghệ Thông Tin

ĐỀ CƯƠNG CHI TIẾT KHÓA LUẬN TN

Tên Đề Tài: <i>Nghiên Cứu Giải Pháp Nguồn Mở Cho Workflow Quản Lý Hồ Sơ Công Văn</i>
Giáo viên hướng dẫn: Thầy Lâm Quang Vũ, Cô Trần Hạnh Nhi
Thời gian thực hiện: Từ ngày 11-07-2008 đến ngày 20-03-2009
Sinh viên thực hiện: Võ Hữu Phúc – 06HC132 Đào Anh Vũ – 06HC228
Loại đề tài: Tìm hiểu công nghệ và xây dựng ứng dụng

Nội Dung Đề Tài:

Nội dung đề tài: nghiên cứu giải pháp nguồn mở cho Workflow quản lý hồ sơ công văn

Yêu cầu:

- Thử nghiệm triển khai và có thể mở rộng một môi trường

workflow để hỗ trợ việc định nghĩa, quản lý và thực thi các quy trình nghiệp vụ liên quan đến quản lý hồ sơ công văn.

- Môi trường workflow cơ sở phải là mã nguồn mở
- Các quy trình nghiệp vụ sử dụng để kiểm chứng ứng dụng phải là các quy trình thực tế được áp dụng tại Khoa CNTT/ trường ĐHKHTN.

Phương pháp thực hiện:

1. Tìm hiểu các hệ thống workflow và sự hỗ trợ các quy trình nghiệp vụ.
2. Phân tích tính chất các quy trình quản lý hồ sơ công văn để đề ra các yêu cầu về mô hình hoá và vận hành các quy trình nghiệp vụ dạng này.
3. Xác định các tiêu chí cần có đối với workflow engine hỗ trợ mô hình hoá và vận hành các quy trình quản lý hồ sơ công văn.
4. Dựa trên các tiêu chí đặt ra, chọn lựa một workflow engine để làm cơ sở cho ứng dụng sẽ phát triển.
5. Nghiên cứu workflow engine được lựa chọn để nhận xét và đánh giá khả năng triển khai ứng dụng từ workflow engine này.
6. Đề xuất các sửa đổi, cải tiến, mở rộng cần thiết đối với workflow engine lựa chọn.
7. Xây dựng ứng dụng hoàn chỉnh từ workflow engine lựa chọn, kiểm chứng ứng dụng với các quy trình nghiệp vụ thực tế.

Kết quả đạt được:

- Báo cáo kết quả nghiên cứu về mặt lý thuyết.
- Triển khai được một ứng dụng nguồn mở cho workflow quản lý hồ sơ công văn.

Kế Hoạch Thực Hiện:

Tên công việc	Ngày bắt đầu	Ngày kết thúc	Người thực hiện
1. Phân tích yêu cầu và mục tiêu của luận văn	01-08-2008	07-08-2008	Võ Hữu Phúc
2. Tìm hiểu Business Process và workflow system	08-08-2008	15-08-2008	Đào Anh Vũ Võ Hữu Phúc
3. Khảo sát & phân tích loại business process được hỗ trợ bởi các engine hiện có	16-08-2008	31-08-2008	Đào Anh Vũ
4. Lựa chọn một engine và tiến hành thử nghiệm			
4.1. Xây dựng tiêu chí để chọn lựa workflow engine & chọn wfmOpen.	01-09-2008	05-09-2008	Đào Anh Vũ Võ Hữu Phúc

4.2. Tiến hành nghiên cứu cấu trúc wfmOpen	06-09-2008	17-09-2008	Đào Anh Vũ
4.3. Cài đặt thử nghiệm	18-09-2008	20-09-2008	Đào Anh Vũ
4.4. Triển khai business process đơn giản trên wfmOpen	06-09-2008	15-09-2008	Võ Hữu Phúc
4.5. Vẽ lại mô hình kiến trúc của wfmOpen	16-09-2008	18-09-2008	Võ Hữu Phúc
4.6. Nghiên cứu ngôn ngữ XPDL	19-09-2008	22-09-2008	Đào Anh Vũ Võ Hữu Phúc
5. Nghiên cứu workflow patterns	23-09-2008	30-09-2008	Võ Hữu Phúc
6. Nghiên cứu và so sánh tính năng của wfmOpen và jBPM			
6.1. Nghiên cứu các tính năng của jBPM	25-09-2008	27-09-2008	Đào Anh Vũ
6.2. Tiến hành xây dựng ứng dụng mẫu trên jBPM	28-09-2008	02-10-2008	Đào Anh Vũ
6.3. Lập bảng đánh giá các tính năng của jBPM và wfmOpen → lựa chọn jBPM	03-10-2008	06-10-2008	Võ Hữu Phúc
7. Tiến hành nhận xét về các engine so	07-10-	14-10-	Võ Hữu Phúc

với các tiêu chí đặt ra	2008	2008	
8. Triển khai và cải tiến	15-10- 2008	31-01- 2008	Đào Anh Vũ Võ Hữu Phúc
9. Thực hiện viết báo cáo kết quả và nội dung trình bày trước hội đồng	01-02- 2008	20-02- 2008	Đào Anh Vũ Võ Hữu Phúc
Xác nhận của GVHD	Ngày 06 tháng 10 năm 2008 SV Thực hiện Võ Hữu Phúc Đào Anh Vũ		

Mục Lục

ĐỀ CƯƠNG CHI TIẾT KHÓA LUẬN TN	3
Chương 1. Tổng quan.....	13
1.1 Giới thiệu business process & workflow	13
1.1.1 Giới thiệu về quy trình nghiệp vụ (business process):.....	13
1.1.2 Giới thiệu luồng công việc (workflow):	14
1.1.3 Các thành phần của một hệ thống workflow:	14
1.1.4 Luồng xử lý của hệ thống quản lý workflow (Workflow management system):	16
1.1.5 Mối tương quan giữa luồng công việc và quy trình nghiệp vụ:.....	18
1.2 Phát biểu bài toán:.....	18
1.3 Mục tiêu của luận văn:.....	19
1.3.1 Nghiên cứu phần mềm triển khai luồng công việc:	19
1.3.2 Ứng dụng kết quả nghiên cứu workflow để quản lý hồ sơ công văn:	20
Chương 2. - Ngôn ngữ mô hình hóa quy trình nghiệp vụ	21
2.1 Ngôn ngữ mô hình hóa:	21
2.2 Các ngôn ngữ mô hình hóa quy trình nghiệp vụ:	22
2.3 Ngôn ngữ được jBPM sử dụng:.....	23
Chương 3. - Nghiên cứu jBPM-jPDL workflow engine:	25
3.1 Giới thiệu jPDL và jBPM engine:	25
3.1.1 Sơ lược về quá trình khảo sát và lựa chọn jBPM:	25
3.1.2 Giới thiệu jPDL:	26
3.1.3 Process graph:	31
3.1.4 Giới thiệu jBPM-engine:.....	35
3.1.5 Vai trò của workflow pattern với thiết kế luồng công việc:	37
3.2 Hiện trạng của hệ thống:	44
3.2.1 Mô tả tổng quát:.....	44
3.3 Chức năng mới:.....	47
3.3.1 Xây dựng document trên jPDL-editor:	47
3.3.2 Document management system:	51
3.3.3 Vấn đề security và tích hợp với LDAP	66
Chương 4. - Thử nghiệm và cài đặt:	73
4.1 Quy trình tổng quát:	73
4.1.1 Đặc tả:	73
4.1.2 Phân tích quy trình:.....	74
4.2 Quy trình thông báo tin tức:.....	74
4.2.1 Thiết kế quy trình:	74
4.2.2 Đặc tả quy trình:	74
4.2.3 Triển khai quy trình: demo	75
4.3 Quy trình tính lương:	75
4.3.1 Thiết kế quy trình:	75
4.3.2 Đặc tả quy trình:	76
4.4 Quy trình phân công giảng dạy:.....	77
4.4.1 Thiết kế quy trình:	77
4.4.2 Đặc tả quy trình:	78
4.4.3 Triển khai quy trình:demo	80
4.5 Quy trình xét học bổng:	80

4.5.1	Thiết kế qui trình:	80
4.5.2	Đặc tả qui trình:	81
4.5.3	Triển khai qui trình: demo	83
4.6	Qui trình đăng ký đi dã ngoại:	83
4.6.1	Thiết kế qui trình:	83
4.6.2	Đặc tả qui trình:	83
4.6.3	Triển khai qui trình: demo	84
4.7	Phân tích và đánh giá kết quả thực hiện:	84
4.7.1	Kết quả đã đạt được:	84
Chương 5. - Tổng kết.....		87
5.1	Nhận xét:	87
5.1.1	Vấn đề khảo sát và lựa chọn hệ thống workflow:.....	87
5.1.2	Vấn đề tích hợp nhiều hệ thống khác nhau (Editor, LDAP, MySQL, CA):	87
5.1.3	Vấn đề thiếu tài liệu đặc tả chi tiết về core engine cũng như các phần khác:..	88
5.1.4	Vấn đề thiếu tài liệu đặc tả editor:	89
5.1.5	Vấn đề tiêu chuẩn của việc thêm document node:.....	89
5.2	Hướng mở rộng:.....	90
5.2.1	Tích hợp thêm với hệ thống Chứng thực tài liệu và chứng thực người dùng:..	90
5.2.2	Xây dựng hệ thống hỗ trợ tạo qui trình dạng ad-hoc:	90
5.2.3	Nghiên cứu vấn đề triển khai qui trình lúc runtime:	91

Danh mục hình

Hình 1-1 Sơ đồ kiến trúc workflow	15
Hình 1-2 Qui trình thực thi workflow	17
Hình 2-1 Sơ lược các ngôn ngữ mô hình hóa hiện tại	22
Hình 3-1 Đặc tả jPDL	26
Hình 3-2 Kiến trúc jBPM.....	28
Hình 3-3 Giao diện định nghĩa luồng công việc	30
Hình 3-4 Kiến trúc jPDL.....	31
Hình 3-5 Các node trong process editor.....	31
Hình 3-6 Kiến trúc jBPM.....	35
Hình 3-7 Kiến trúc BPEL.....	36
Hình 3-8 Các gói trong jPDL source	47
Hình 3-9 Document Node	47
Hình 3-10 Thuộc tính trạng thái của document	48
Hình 3-11 Đặc tả trạng thái tài liệu dạng XML	48
Hình 3-12 Đặc tả thông tin document node trong xsd	49
Hình 3-13 Mã nguồn xsd cần điều chỉnh	49
Hình 3-14 Assign trạng thái document trong process.....	50
Hình 3-15 Thể hiện document với task trong xml	50
Hình 3-16 Bổ sung thêm element trong xsd	51
Hình 3-17 Sơ đồ use-case cho phân hệ quản trị tài liệu.....	54
Hình 3-18 Sơ đồ lớp phân hệ quản trị tài liệu.....	58
Hình 3-19 Sơ đồ quan hệ CSDL phân hệ quản trị tài liệu	60
Hình 3-20 Sơ đồ trạng thái của tài liệu	65
Hình 3-21 Sơ đồ luồng xử lý đăng nhập theo JAAS	69
Hình 3-22 Sơ đồ lớp phân hệ chứng thực và cấp quyền người dùng.....	71
Hình 4-1 Qui trình thông báo tin tức.....	75
Hình 4-2 Qui trình tính lương	77
Hình 4-3 Qui trình đăng ký giảng dạy	80

Hình 4-4 Qui trình đăng ký học bổng	82
Hình 4-5 Qui trình đăng ký dã ngoại	84

Danh mục Bảng

Bảng 1-1 So sánh luồng công việc và qui trình nghiệp vụ	18
Bảng 3-1 So sánh Super state và process state	33
Bảng 3-2 Control flow patterns.....	40
Bảng 3-3 resource patterns.....	42
Bảng 3-4 Data patterns.....	44
Bảng 3-5 Danh sách các actor cho phân hệ DMS.....	54
Bảng 3-6 Danh sách các use-case phân hệ DMS	55
Bảng 3-7 Danh sách các lớp/quan hệ cho phân hệ DMS.....	59
Bảng 3-8 Bảng DMS_FILESYSTEM.....	60
Bảng 3-9 Bảng DMS_WORKFLOWDOCSET.....	61
Bảng 3-10 Bảng DMS_DOCUMENT	62
Bảng 3-11 Bảng DMS_STATE	63
Bảng 3-12 Bảng DMS_FILEINFO	63
Bảng 3-13 Bảng DMS_TASKDOCUMENT.....	64
Bảng 3-14 Danh sách lớp/ quan hệ phân hệ chứng thực và phân quyền	72
Bảng 4-1 Bảng trạng thái document ThôngBao	75
Bảng 4-2 Bảng thống kê giờ dạy.....	76
Bảng 4-3 Bảng thanh toán lương	77
Bảng 4-4 Bảng thông báo.....	77
Bảng 4-5 Bảng danh sách môn học.....	78
Bảng 4-6 Danh sách đăng ký giảng dạy.....	79
Bảng 4-7 Danh sách đăng ký thiết bị	79
Bảng 4-8 Bảng yêu cầu học bổng	81
Bảng 4-9 Bảng thông tin học bổng	81
Bảng 4-10 Bảng danh sách đăng ký học bổng	82
Bảng 4-11 Bảng thông báo dã ngoại.....	83
Bảng 4-12 Bảng trạng thái danh sách đăng ký dã ngoại.....	83

Chương 1. Tổng quan

Chương này sẽ trình bày tổng quan về khái niệm qui trình nghiệp vụ (business process) và luồng công việc (workflow). Trong đó, qui trình nghiệp vụ (business process) được phát biểu bằng lời một qui trình nghiệp vụ cụ thể trong thực tế. Sau đó, dùng một ngôn ngữ đặc tả qui trình cụ thể (jPDL) để mô tả lại qui trình nghiệp vụ (business process) đó thành dạng thực thi được trên hệ thống. Qui trình được đặc tả thành dạng thực thi được gọi là luồng công việc (workflow).

1.1 Giới thiệu business process & workflow

- Trước hết chúng ta cùng tìm hiểu các khái niệm về business process và workflow.

1.1.1 Giới thiệu về qui trình nghiệp vụ (business process):

- [2] Qui trình nghiệp vụ là một thuật ngữ được sử dụng để mô tả các phương thức (procedures) chính yếu của một tổ chức, bao gồm:
 - Nhiều bước thực hiện.
 - Nhiều người thực hiện.
 - Một lượng lớn tài nguyên được sử dụng.
- [2] Qui trình nghiệp vụ là quá trình lặp lại những hành động ít nhiều tuân theo những qui tắc nào đó. Một qui trình nghiệp vụ hướng mục đích và liên quan trực tiếp đến việc sáng tạo có giá trị hướng thị trường của doanh nghiệp. Thực thi qui trình nghiệp vụ đòi hỏi việc sử dụng những tài nguyên khan hiếm.
- Trong một tổ chức, có nhiều nhân tố góp phần làm tăng mức độ phức tạp của qui trình nghiệp vụ như:
 - Các tiến trình không được ghi nhận thành tài liệu đầy đủ.
 - Các qui tắc ràng buộc không được đảm bảo.
 - Nhân viên thiếu thông tin.

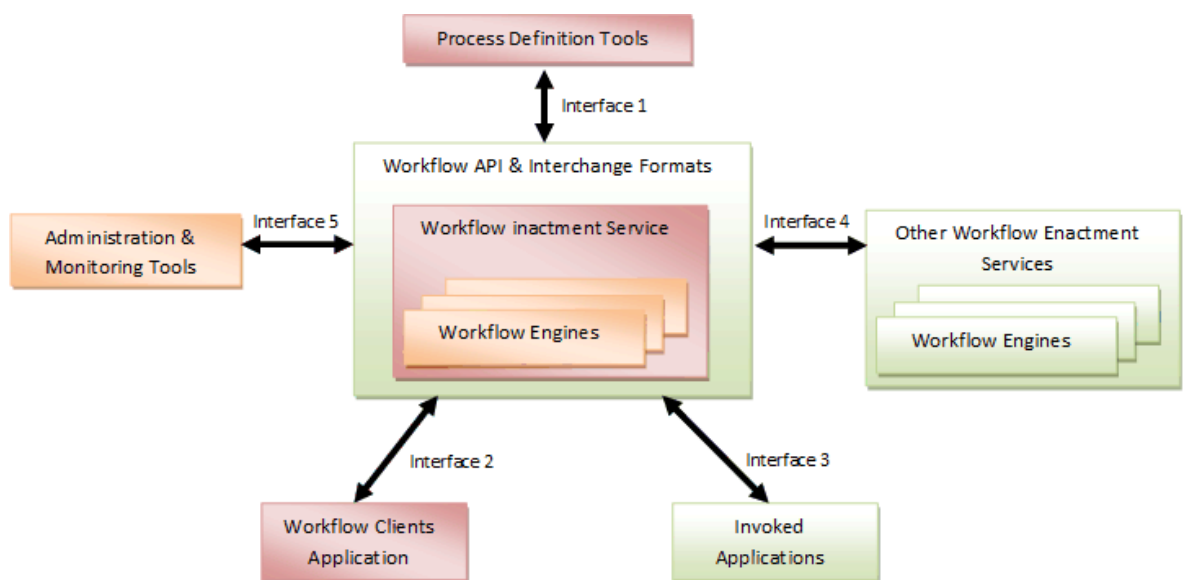
- Tổ chức, doanh nghiệp thiết công cụ hỗ trợ việc giám sát.
 - Các bước thực hiện, người thực hiện và tài nguyên được sử dụng phối hợp không đồng bộ.
- [2] Hệ thống quản lý luồng công việc giải quyết các vấn đề này bằng cách tự động hóa các khía cạnh của một business process như: *Ai cần phải làm gì, khi nào, và bằng công cụ gì.*
 - Tóm lại, qui trình nghiệp vụ là nội dung đặc tả bằng lời một qui trình nghiệp vụ nào đó trong thực tế.

1.1.2 Giới thiệu luồng công việc (workflow):

- Luồng công việc (workflow) là phần đặc tả lại qui trình nghiệp vụ (business process) bằng cách sử dụng một ngôn ngữ đặc tả qui trình cụ thể (ở đây là jPDL) để thể hiện lại qui trình nghiệp vụ (business process) được phát biểu bằng lời.
- Theo định nghĩa của tổ chức WfMC, luồng công việc là qui trình nghiệp vụ tự động toàn bộ hay từng phần, trong đó các tài liệu, thông tin và tác vụ được chuyển từ người tham gia qui trình đến một người khác bằng hành động theo những ràng buộc nào đó.
- Hệ quản trị luồng công việc là một phần mềm nhận thông tin truyền vào là mô tả hình thức của qui trình nghiệp vụ và duy trì trạng thái của quá trình thực thi tiến trình bằng cách chuyển giao các hoạt động giữa người dùng và ứng dụng.
- Luồng công việc là sự trừu tượng hóa qui trình nghiệp vụ tập trung vào luồng đối tượng và tài liệu điện tử. Hành động của người dùng và các quyết định trong ngữ cảnh của qui trình nghiệp vụ thông thường được loại bỏ hoặc thêm vào dựa trên mối liên hệ tương tác với ứng dụng trên hệ thống.

1.1.3 Các thành phần của một hệ thống workflow:

- Dưới đây là phần giới thiệu các thành phần của một hệ thống workflow tổng quát. Trong đó thành phần chính yếu là workflow engine. Đây là thành phần cơ bản mà mọi tổ chức workflow đều có, tùy theo đặc trưng ngôn ngữ đặc tả qui trình được sử dụng, mà mức độ hỗ trợ của workflow engine sẽ khác nhau. Với những mã nguồn workflow engine được công bố, các thành phần khác ngoài workflow engine có thể được thêm vào hoặc hoàn toàn không có. Trong trường hợp này là mã nguồn jBPM workflow engine, nhóm phải xây dựng thêm những thành phần khác để giải quyết yêu cầu của bài toán đặt ra.
- Kiến trúc workflow:



Hình 1-1 Sơ đồ kiến trúc workflow

- Interface 1 đặc tả việc trao đổi luồng công việc giữa công cụ mô hình hóa ngoại vi và WMS. Các công cụ ngoại vi bao gồm editor để định nghĩa luồng công việc đồ họa hoặc dạng văn bản. Những công cụ mô hình hóa quá trình nghiệp vụ

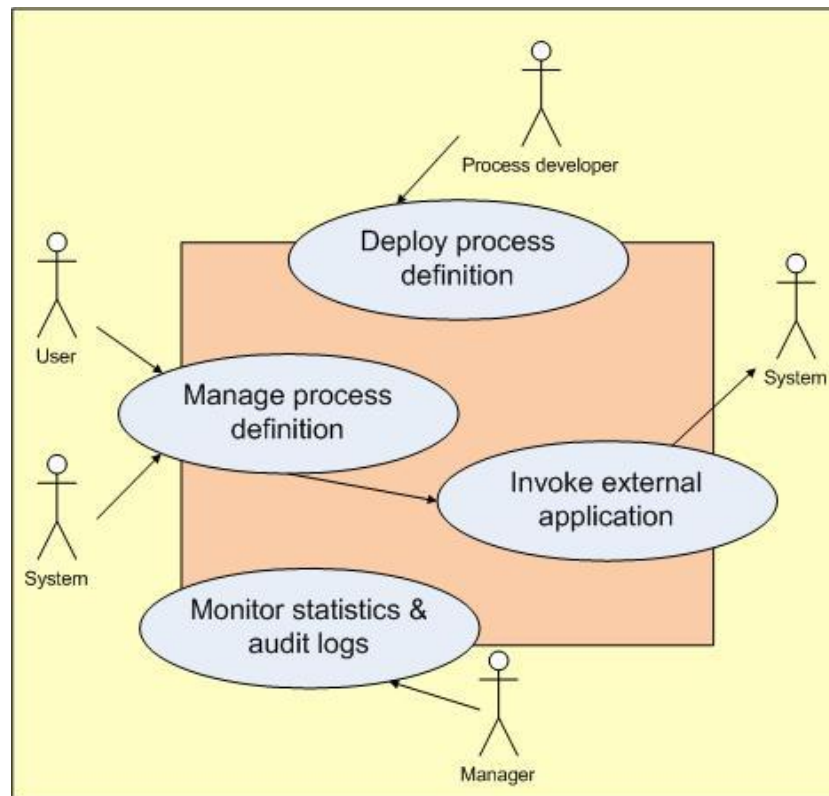
tổng quát hỗ trợ chuẩn của WfMC cũng có thể được dùng để đặc tả luồng công việc. [Holl95, p. 20].

- Interface 2 mô tả quá trình liên lạc giữa workflow engine và ứng dụng luồng công việc phía người dùng. [Holl95, pp. 31].
- Interface 3 để phục vụ nhu cầu tích hợp với hệ thống ngoại vi [Holl95, pp. 35].
- Interface 4 để tích hợp với các hệ thống quản lý luồng công việc khác. Việc đặc tả bao gồm việc triệu gọi từ xa, trao đổi và đồng bộ hóa dữ liệu giữa nhiều hệ thống quản lý luồng công việc khác nhau. [Jung01, p. 126] [Holl95, pp. 41].
- Interface 5 mô tả quá trình liên lạc giữa các dịch vụ được kích hoạt và các công cụ quản lý và giám sát [Jung01, p. 126].

1.1.4 Luồng xử lý của hệ thống quản lý workflow (Workflow management system):

- Hệ thống quản lý workflow bắt đầu bằng việc người dùng định nghĩa luồng công việc dựa theo đặc tả của qui trình nghiệp vụ. Sau khi đặc tả xong, luồng công việc được đưa vào hệ thống workflow để có thể thực thi. Người dùng có thể tạo được nhiều thể hiện từ một workflow được triển khai trên hệ thống. Bên cạnh đó, người dùng có thể khai thác chức năng quản lý qui trình, chức năng quản lý người dùng đơn giản mà hệ thống cung cấp sẵn.
- Hệ thống quản lý luồng công việc là hệ thống bao gồm thông tin đặc tả qui trình nghiệp vụ và duy trì trạng thái thực thi tiến trình bằng cách chuyển giao những hành động thực thi qui trình giữa người tham gia và ứng dụng.
- Trước hết, hệ thống quản lý luồng công việc sẽ nhận thông tin đặc tả về qui trình bao gồm sơ đồ các trạng thái và các hoạt động. Hệ thống quản lý luồng công việc sẽ thực hiện việc duy trì trạng thái

của quá trình thực thi để người dùng có thể giám sát những thay đổi trong hệ thống cũng như sự chuyển đổi trạng thái của các hoạt động.



Hình 1-2 Quy trình thực thi workflow

- **Quá trình định nghĩa:** cho phép người tạo qui trình có thể triển khai qui trình đó. Trong trường hợp này, ‘người phát triển qui trình’ là sự kết hợp giữa người phân tích nghiệp vụ và lập trình viên.
- **Quá trình thực thi:** cho phép người dùng và hệ thống thực hiện việc tạo một thể hiện qui trình. Thể hiện của qui trình (process instance) là một qui trình được định sẵn được thực thi trong hệ thống. Luồng quản lý của qui trình là quá trình mô tả trạng thái hệ thống (state machine). Hai phương thức chính yếu của việc thực thi qui trình là khởi động và kết thúc qui trình.
- **Quá trình thực thi ứng dụng:** quá trình này đóng góp vào khả năng tương tác giữ hệ thống quản lý luồng công việc với những hệ

thống ngoại vi. Khi người dùng hoặc hệ thống quản lý quá trình thực thi thể hiện, những sự kiện được tạo ra. Định nghĩa qui trình xác định một phân luận lý của logic chương trình phải được thực thi thông qua sự kiện. Logic chương trình sẽ thực hiện việc giao tiếp với hệ thống ngoại vi hoặc nội vi của tổ chức.

- **Quá trình giám sát:** người quản lý có thể tăng cường khả năng quản lý toàn bộ hệ thống bên trong thông qua thống kê và ghi nhận quá trình thực thi của tiến trình.

1.1.5 Mối tương quan giữa luồng công việc và qui trình nghiệp vụ:

Mô hình hóa qui trình nghiệp vụ	Quản lý luồng công việc
<ul style="list-style-type: none"> * Bao gồm thao tác qui trình * Ở mức độ tư duy của doanh nghiệp. * Qui trình nghiệp vụ có thể liên quan đến nhiều nguồn tài nguyên khác (không chỉ riêng tin học). 	<ul style="list-style-type: none"> * Tập trung xử lý qui trình/nghiệp vụ dạng số. * Nhấn mạnh đến việc ứng dụng công nghệ thông tin. * Luồng công việc là qui trình nghiệp vụ kết hợp với ứng dụng công nghệ thông tin.

Bảng 1-1 So sánh luồng công việc và qui trình nghiệp vụ

1.2 Phát biểu bài toán:

- Quản lý hồ sơ công văn là công việc liên quan đến nhiều người thực hiện, trong đó tài liệu được chuyển đến, chuyển đi và ký duyệt ... Mỗi một qui trình nghiệp vụ quản lý hành chánh liên quan đến một hoặc nhiều tài liệu, việc quản lý thủ công những hồ sơ tài liệu này mất rất nhiều thời gian, chi phí và hàng loạt những khó khăn khác như: lưu trữ, giám sát theo dõi, cũng như khó khăn trong việc quản lý tài nguyên.
- Tính kịp thời của việc tiếp nhận công văn cũng là một vấn đề mà qui trình quản lý thông thường gặp phải, khi tiếp nhận công văn có thời gian yêu cầu

phản hồi ngắn, trong khi đó công văn phải di chuyển qua nhiều cấp khác nhau trong tổ chức và sau đó phải phản hồi theo trình tự ngược lại.

1.3 Mục tiêu của luận văn:

1.3.1 Nghiên cứu phần mềm triển khai luồng công việc:

- Trước hết, nhóm sẽ xác định mục tiêu của đề tài và các tiêu chí cần đạt của một hệ thống workflow. Từ những tiêu chí đó, thực hiện khảo sát sơ lược các hệ thống workflow engine nguồn mở phù hợp với tiêu chí đề ra.
- Trong quá trình nghiên cứu, nhóm đã đặt ra những tiêu chí để lựa chọn workflow engine. Từ những tiêu chí đó, nhóm đã khảo sát và nghiên cứu nhiều engine khác nhau trong hệ thống mã nguồn mở để có thể lựa chọn được jBPM.
- Hiện tại có nhiều phần mềm nguồn mở để quản lý và triển khai workflow. Sơ lược có các phần mềm sau:
 - [jBpm](#) – là engine quản lý luồng công việc dễ mở rộng và linh động. Quy trình nghiệp vụ được biểu thị dưới dạng ngôn ngữ súc tích, đơn giản nhưng rất mạnh mẽ, được sử dụng làm thông tin vào của hệ thống thời gian thực jBPM. Bên cạnh đó, tính dễ tích hợp với các hệ thống khác, đặc biệt là ứng dụng dành cho doanh nghiệp.
 - [wfmOpen](#) – đây là engine triển khai luồng công việc xây dựng trên nền J2EE được đề xuất bởi tổ chức Workflow Management Coalition (WfMC) và tổ chức Object Management Group (OMG). Luồng công việc được đặc tả bằng ngôn ngữ XML Process Definition Language (XPDL), ngôn ngữ này cũng được đề xuất bởi tổ chức WfMC.
 - [OpenWFE](#) – là engine mã nguồn mở quản lý luồng công việc viết bằng ngôn ngữ Java. Ngôn ngữ định nghĩa luồng

công việc được hỗ trợ là Lisp dialect, thể hiện thông qua đặc tả file xml.

- [Enhydra Shark](#) – hoàn toàn dựa theo tiêu chuẩn của WfMC và OMG, sử dụng XPDL như là ngôn ngữ định nghĩa chính. Sử dụng Enhydra DODS cho việc lưu trữ thông tin của qui trình và các hoạt động.
- [Taverna](#), [Freeflow](#).
- Bên cạnh những phần mềm nguồn mở, có nhiều phần mềm thương mại cũng tham gia vào việc hỗ trợ quản lý workflow, gồm các phần mềm như sau:
 - [Bea's WLI](#), [Carnot](#), [Dralasoft](#), [Filenet](#), [Fujitsu's i-Flow](#)
 - [IBM's holosofx tool](#), [Intalio](#)

1.3.2 Ứng dụng kết quả nghiên cứu workflow để quản lý hồ sơ công văn:

- Thực hiện khảo sát và lựa chọn core engine nguồn mở đáp ứng nhu cầu quản lý luồng công việc để thực hiện nghiên cứu trong số các phần mềm nguồn mở mà nhóm đã tìm kiếm.
- Dựa vào mô tả bài toán thực tế ở trên, chúng ta sẽ thực hiện nghiên cứu workflow để giải quyết vấn đề quản lý hồ sơ công văn.

2 - Ngôn ngữ mô hình hóa quy trình nghiệp vụ

Chương này sẽ trình bày tổng quan về các ngôn ngữ mô hình hóa hiện tại trong cả hệ thống phần mềm nguồn mở và hệ thống phần mềm thương mại. Trong đó nhấn mạnh đến ngôn ngữ mô hình hóa jPDL, đây là ngôn ngữ sẽ được sử dụng để đặc tả quy trình nghiệp vụ trong luận văn này.

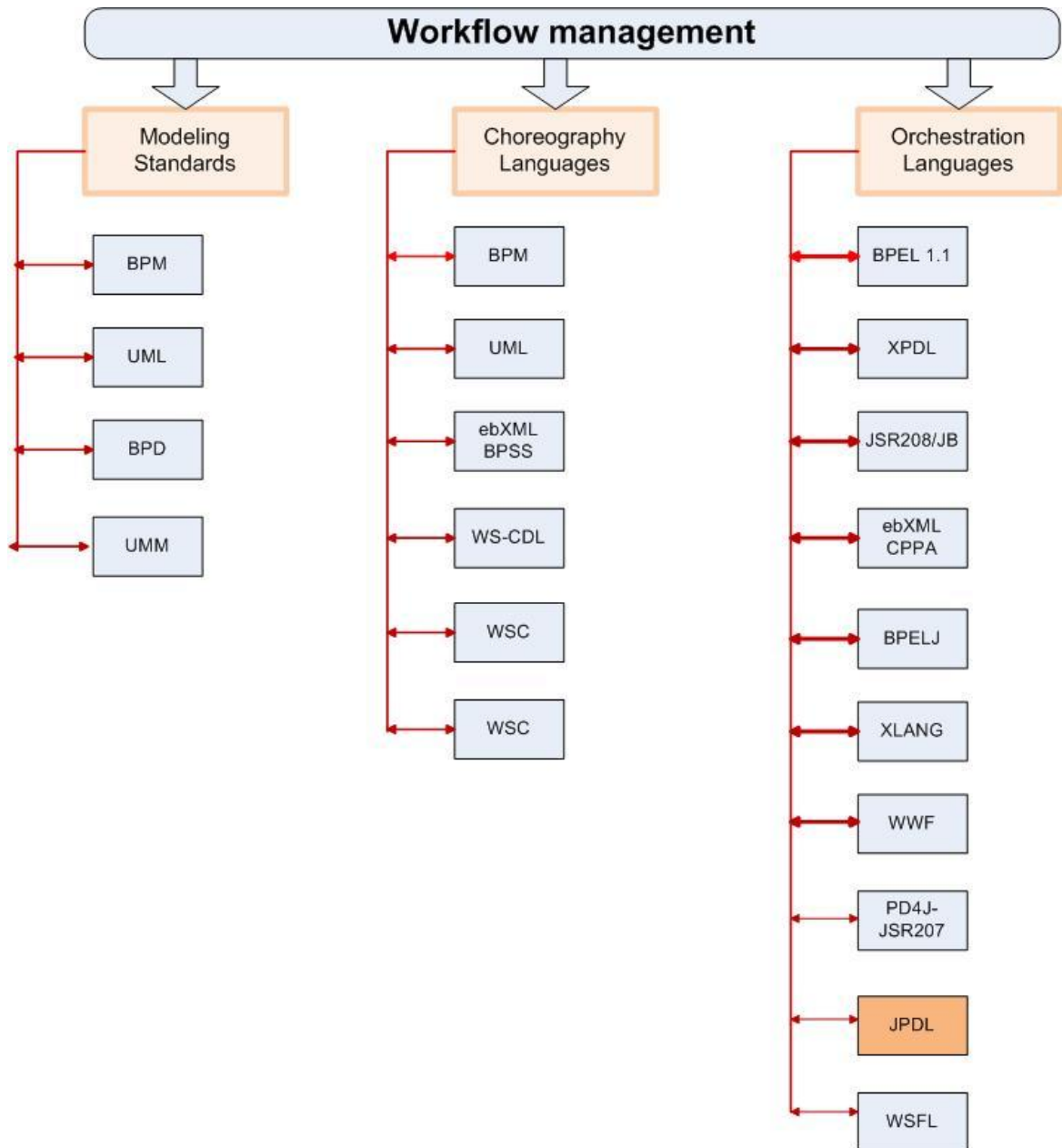
2.1 Ngôn ngữ mô hình hóa:

- Ngôn ngữ mô hình hóa quy trình nghiệp vụ là ngôn ngữ dùng để đặc tả lại quy trình nghiệp vụ (business process) trong thực tế mà workflow engine hỗ trợ để thực thi quy trình được đặc tả dựa theo ngôn ngữ đó.
- Để có thể thực thi một hệ thống quản lý luồng công việc, trước hết cần phải đặc tả được quy trình nghiệp vụ theo ngôn ngữ mà workflow engine hỗ trợ.
- Mỗi hệ thống quản lý luồng công việc có thể hỗ trợ một hoặc nhiều ngôn ngữ đặc tả hình thức khác nhau. Ngôn ngữ đặc tả luồng công việc thường được đặc tả dưới dạng ký hiệu đồ họa. Những ngôn ngữ thường được sử dụng gồm:
 - Ngôn ngữ đặc tả luồng công việc sẵn có:
 - **jPDL**: đây là ngôn ngữ sẽ được sử dụng để đặc tả quy trình nghiệp vụ trong thực tế.
 - [XPDL](#), [YAWL](#), [SCUFL](#).
 - Ngôn ngữ đặc tả quy trình dựa trên web service:
 - [BPEL](#), [BPML](#), [WSFL](#), [XLANG](#), [Wf-XML](#), [SWSL](#).
 - Dạng định nghĩa công việc
- Ngoài ra, có thể xây dựng ứng dụng luồng công việc bằng cách sử dụng ngôn ngữ lập trình bằng cách kết hợp hàm thư viện sẵn có với các interface, ví dụ như:

- Windows Workflow Foundation.
- Workflow OSID.

2.2 Các ngôn ngữ mô hình hóa quy trình nghiệp vụ:

- Giới thiệu sơ lược về các ngôn ngữ dùng để mô hình hóa nghiệp vụ hiện tại.



Hình 2-1 Sơ lược các ngôn ngữ mô hình hóa hiện tại

- Các ngôn ngữ chuẩn hiện tại không hỗ trợ biểu diễn document trong qui trình thực thi.
- Bên cạnh đó, chức năng quản trị tài liệu cũng không được mô tả trong hệ thống. Người dùng phải tự xây dựng thêm chức năng quản lý document cũng như xây dựng thêm ký hiệu để biểu diễn document trong qui trình.
- Những ngôn ngữ trên tuy có hỗ trợ định nghĩa người dùng thực hiện tác vụ, nhưng phần hỗ trợ này không nâng cao tính tái sử dụng của chương trình. Do người dùng phải xác định cụ thể người được phân công, nên khi áp dụng qui trình cho một bộ phận hành chính khác sẽ phải cấu hình lại toàn bộ (từ mức định nghĩa cho đến mức xử lý).
- Ngoài ra, việc phân công tác vụ cho nhóm cũng không được hỗ trợ rõ rệt, người dùng phải tự khảo sát và tìm hiểu hệ thống workflow engine để có thể phân công cùng một tác vụ cho nhiều người tại cùng một thời điểm.

2.3 Ngôn ngữ được jBPM sử dụng:

- Hệ thống jBPM sử dụng hai ngôn ngữ để đặc tả qui trình nghiệp vụ là BPEL và jPDL.
- Trong đó, ngôn ngữ BPEL là ngôn ngữ dùng để đặc tả qui trình nghiệp vụ nhưng sử dụng web service. Một ngôn ngữ khác được jBPM sử dụng là ngôn ngữ jPDL.
- Ngôn ngữ được sử dụng cho hệ thống workflow jBPM là jPDL. Đây là ngôn ngữ được phát triển độc lập bởi jBPM. Ưu điểm của ngôn ngữ này là khả năng mở rộng cao. Người dùng có thể đặc tả thêm những ký hiệu khác để phục vụ cho yêu cầu của bài toán cụ thể.
- Trong số các ngôn ngữ dùng để đặc tả qui trình, có những ngôn ngữ do tổ chức WfMC (Workflow Management Coalition), OMG (Object Management Group), những tổ chức này đưa ra nhiều loại ngôn ngữ đặc tả qui trình khác nhau (XPDL, BPEL, XLANG, ...). Dựa trên các ngôn ngữ chuẩn của các tổ

chức này, những hệ thống workflow engine khác nhau sẽ được tạo ra để có thể quản lý quy trình dựa trên một ngôn ngữ đặc tả cụ thể nào đó.

- Những ngôn ngữ chuẩn do các tổ chức tạo ra có nhược điểm là khả năng mở rộng không cao. Từ đó dẫn đến việc không đáp ứng được yêu cầu của bài toán đặt ra, đó là nhu cầu biểu diễn document trong quy trình.

- **BPEL (Web Service Business Process Execution Language – WS-BPEL):** ngôn ngữ này được chuẩn hóa bởi tổ chức WfMC (Workflow management coalition). Ngôn ngữ đặc tả này được sử dụng để tương tác với web service. Do được chuẩn hóa và sử dụng rộng rãi trên nhiều hệ thống workflow engine khác nhau nên việc mở rộng khả năng đặc tả document sẽ bị hạn chế.
- **jPDL (jBPM process definition language):** đây là ngôn ngữ được sử dụng trong luận văn này. jPDL được tạo ra bởi nhóm phát triển hệ thống jBPM dựa trên ngôn ngữ đặc tả quy trình XPD. Trong đó, khả năng mở rộng của ngôn ngữ này rất lớn. Tuy chưa được các tổ chức chuẩn hóa quy trình thừa nhận chính thức, nhưng tầm hưởng và mức độ cải tiến của ngôn ngữ này rất to lớn.

3 - Nghiên cứu jBPM-jPDL workflow engine:

Chương này đi sâu chi tiết vào hệ thống quản lý workflow jBPM mà nhóm đã cải tiến và xây dựng thêm. Trong đó, phần hiện trạng sẽ mô tả chi tiết trạng thái cũng như khả năng đáp ứng ban đầu của hệ thống so với yêu cầu của bài toán đặt ra. Sau đó sẽ là phần nội dung cải tiến, những chức năng được xây dựng thêm để đáp ứng yêu cầu của bài toán đặt ra. Phần mở rộng sẽ mô tả hướng phát triển của hệ thống, những tính năng cần phải xây dựng thêm để đáp ứng yêu cầu mở rộng của bài toán.

3.1 Giới thiệu jPDL và jBPM engine:

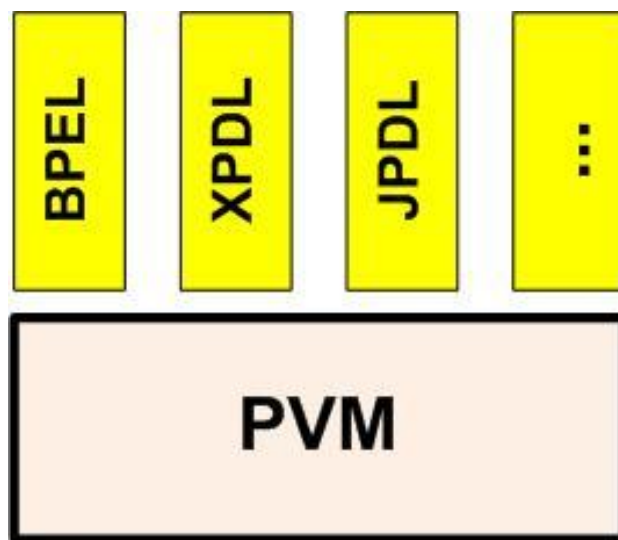
3.1.1 Sơ lược về quá trình khảo sát và lựa chọn jBPM:

- Trước hết, nhóm tìm hiểu yêu cầu đáp ứng của một hệ thống quản lý workflow. Sau khi đã có những tiêu chí cụ thể mà một workflow engine cần phải đáp ứng, nhóm thực hiện việc khảo sát những workflow engine trong hệ thống mã nguồn mở.
- Do các workflow engine mã nguồn mở không có nhiều tài liệu đặc tả, nên để khảo sát được chính xác, nhóm phải triển khai thử hệ thống để có thể kết luận được workflow engine này có đáp ứng được tiêu chí đã đề ra hay không.
- Trước khi làm việc với hệ thống jBPM, nhóm đã làm việc thử với hệ thống workflow engine wfOpen. Sau khi khảo sát, nhóm đã nhận thấy hệ thống jBPM đáp ứng được yêu cầu quan trọng mà các workflow engine khác không đáp ứng được trong quá trình nhóm thực hiện khảo sát. Đó là:
 - ❖ **Nhiều thể hiện:** jBPM có khả năng đáp ứng nhu cầu tạo ra nhiều thể hiện của tác vụ nào đó trong quy trình.
 - ❖ **Ngôn ngữ đặc tả quy trình:** do jBPM hỗ trợ ngôn ngữ đặc tả quy trình jPDL, đây là ngôn ngữ có khả năng mở

rộng cao nên nhóm có thể bổ sung thêm đặc tả document.

- ❖ **Cộng đồng hỗ trợ:** jBPM chạy trên nền J2EE, workflow engine này thực thi trên hệ máy chủ JBoss vốn được sử dụng rộng rãi và phổ biến, cộng đồng sử dụng jBPM rất lớn nên những khó khăn trong quá trình triển khai và ứng dụng một phần được giải quyết thông qua trao đổi thảo luận với cộng đồng jBoss-jBPM.
- ❖ **Mã nguồn:** trong cộng đồng nguồn mở, phần mã nguồn của jBPM được cung cấp khá đầy đủ để người dùng có thể nghiên cứu được. Tuy nhiên, phiên bản của mã nguồn công bố bao giờ cũng cũ hơn và có nhiều lỗi hơn so với phiên bản được đóng gói.

3.1.2 Giới thiệu jPDL:



Hình 3-1 Đặc tả jPDL

- jPDL là ngôn ngữ xây dựng dựa trên một framework chung, ngôn ngữ qui trình trực quan để biểu diễn qui trình nghiệp vụ trong đó có task, wait-state cũng như giao tiếp đồng bộ, timer, action tự động,

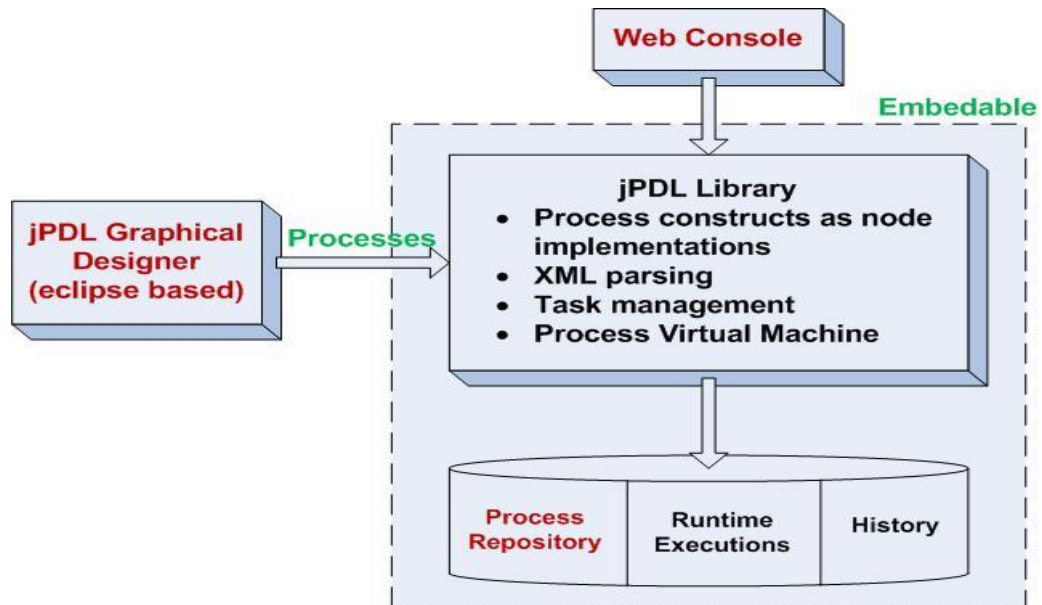
... Để kết hợp những cách biểu diễn với nhau, jPDL có cơ chế quản lý luồng có khả năng mở rộng cao và mạnh mẽ nhất.

- jPDL phụ thuộc ít và được dùng dễ dàng như thư viện của java.
- jPDL có thể cấu hình với bất kỳ loại cơ sở dữ liệu nào và có thể triển khai trên mọi server.

3.1.2.1 Tổng quan:

- jPDL là ngôn ngữ thực thi qui trình với mô hình cấu trúc tốt, có tiềm năng, tích hợp java thuần túy và cung cấp công cụ quản lý task vụ mạnh mẽ.
- jPDL hỗ trợ việc cộng tác giữa người phân tích nghiệp vụ và lập trình viên. Trước hết, jPDL dựa trên đồ thị, điều này cho phép người dùng tự do hơn trong thiết kế cấu trúc khối. Tiếp theo là event listener cho phép lập trình viên gắn kèm những phản xử lý nghiệp vụ kỹ thuật ẩn trong những qui trình nghiệp vụ mà không làm thay đổi thể hiện đồ họa của qui trình. Bằng cách đó, việc chuyển đổi từ mô hình phân tích sang mô hình thực thi sẽ dễ dàng hơn.
- jPDL có công cụ hỗ trợ thiết kế đồ họa để quản lý tập tin định nghĩa qui trình. Công cụ thiết kế hỗ trợ cả trình soạn thảo XML dạng đồ họa và dạng văn bản, hai dạng này luôn được cập nhật đồng thời với nhau. Cú pháp XML của file jPDL rất gọn nhẹ và dễ đọc.
- jPDL có thể dễ dàng nhúng vào bất kỳ ứng dụng java nào. Ngôn ngữ có khả năng triển khai như một dịch vụ độc lập hoặc như một thư viện của java.
- Thành phần nòng cốt của luồng công việc và quản lý qui trình nghiệp vụ được đóng gói dưới dạng thư viện java đơn

giản. Thư viện này bao gồm dịch vụ quản lý và thực thi qui trình trong cơ sở dữ liệu jPDL.



Hình 3-2 Kiến trúc jBPM

3.1.2.2 Công cụ đồ họa thiết kế qui trình bằng ngôn ngữ jPDL:

- jPDL cũng bao gồm công cụ thiết kế đồ họa. Trình thiết kế này là công cụ đồ họa để quản lý qui trình nghiệp vụ dưới dạng plugin của eclipse.
- Tính năng quan trọng nhất của trình thiết kế đồ họa là nó bao gồm hỗ trợ cả phân tích nghiệp vụ lẫn phát triển kỹ thuật nghiệp vụ. Tính năng này cho phép chuyển đổi dễ dàng từ mô hình hóa nghiệp vụ sang cài đặt thực tế.

3.1.2.3 Ứng dụng web jBPM:

- Ứng dụng web jBPM phục vụ cho ba mục đích chính. Trước tiên, nó phục vụ như là trung tâm xử lý cho mọi tương tác với tác vụ đang được thực thi phát sinh bởi tín hiệu thực thi qui trình. Mục đích thứ hai là quản trị và giám sát hệ thống

dạng console, cho phép thanh tra và thao tác với thể hiện đang thực thi. Cuối cùng là chức năng giám sát hoạt động nghiệp vụ.

3.1.2.4 Thư viện lõi của jBPM:

- Thành phần cốt lõi của JBoss jBPM là thư viện java J2SE thuần túy để định nghĩa việc quản lý quy trình nghiệp vụ và môi trường dạng runtime cho thể hiện của thực thi quy trình.
- JBoss jBPM là thư viện java, được sử dụng trong bất kỳ môi trường java nào như ứng dụng web, ứng dụng swing, EJB, webservice, ... Thư viện jBPM có thể được đóng gói và sử dụng như stateless session EJB. Điều này cho phép triển khai dạng cụm (clustered deployment) và tương thích cao với mọi dạng thức tương tác.

3.1.2.5 Thành phần chứng thực JBoss jBPM:

- JBoss jBPM có thể tích hợp với bất kỳ thư mục nào chứa thông tin của tổ chức và người dùng.
- JBoss jBPM cũng có khả năng tích hợp với các hệ thống quản lý chứng thực khác như LDAP, Active Directory.
- Hiện tại, jBPM sử dụng cơ chế quản lý chứng thực dựa trên tập tin cấu hình bền vững.

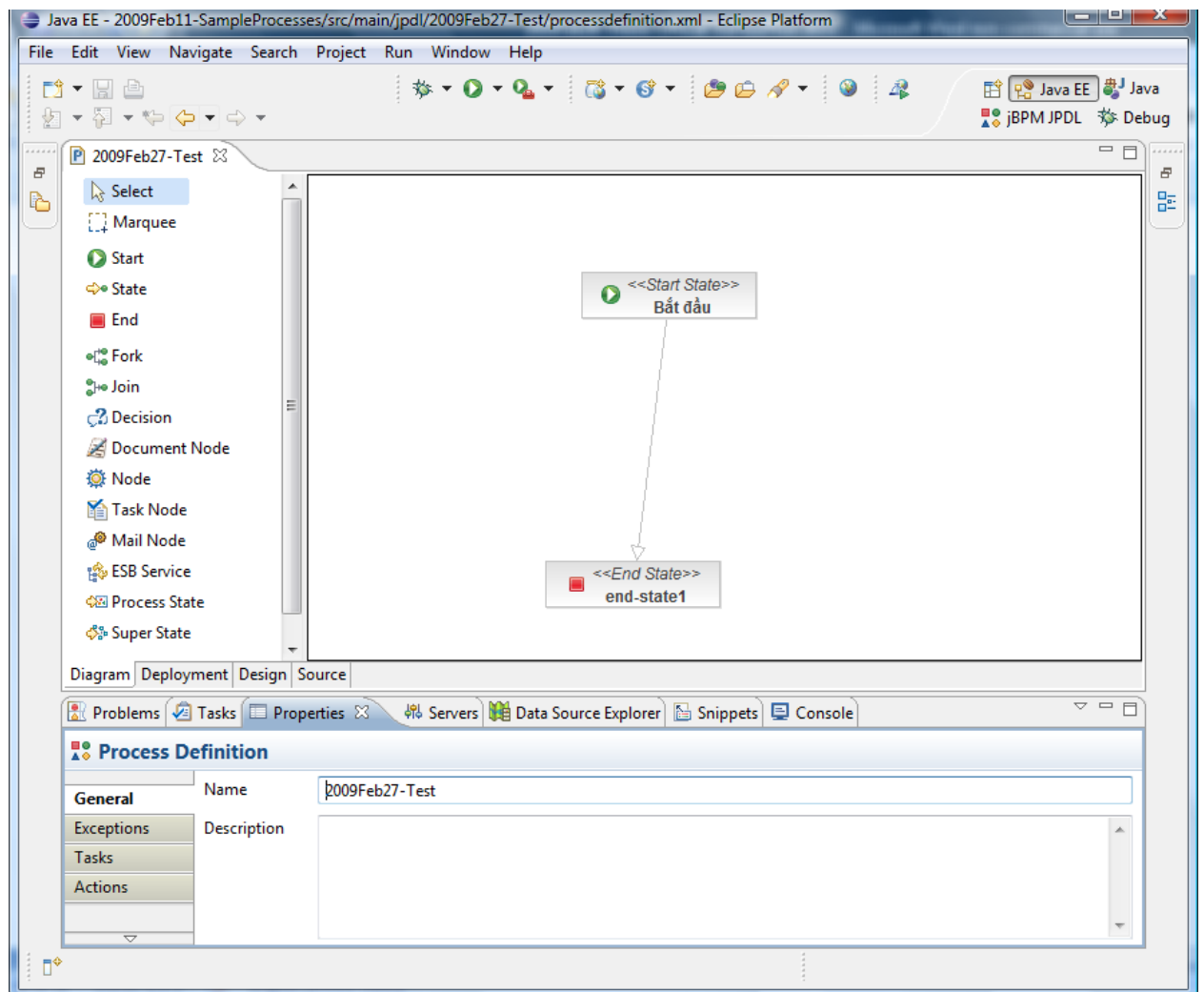
3.1.2.6 The JBoss jBPM Job Executor:

- Là thành phần phục vụ cho việc giám sát và thực thi công việc trong môi trường java chuẩn. Các công việc được dùng cho timers và thông tin đồng bộ. Trong môi trường tổ chức kinh doanh, JMS và EJB Timer Service được dùng cho mục đích này. Một cách tổng quát, trình thực thi công việc được

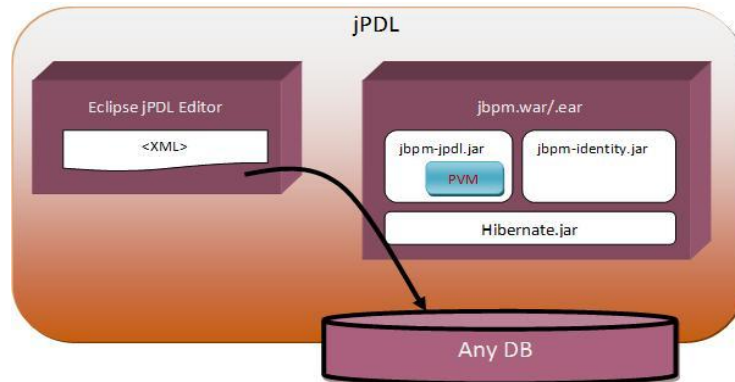
dùng trong môi trường mà không có sự tồn tại của cả JMS và EJB.

3.1.2.7 Giới thiệu jPDL Editor:

- jPDL editor được phát triển dưới dạng plugins để có thể dễ dàng tích hợp trên hệ thống eclipse. Trong phần này, nhóm phát triển sử dụng Eclipse Ganymede để thực hiện việc minh họa cũng như diễn giải các ký hiệu được sử dụng để phát triển workflow.



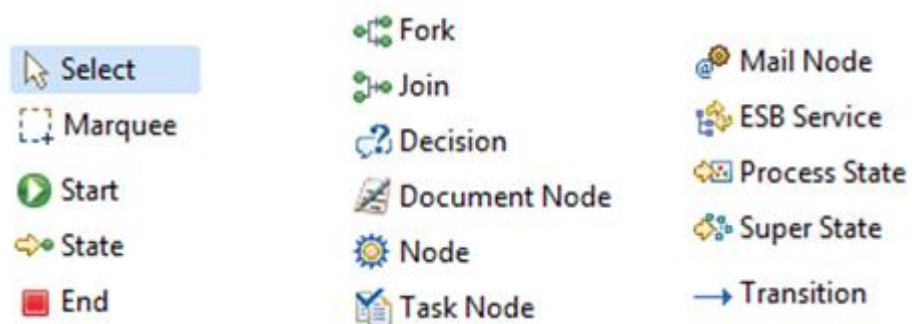
Hình 3-3 Giao diện định nghĩa luồng công việc



Hình 3-4 Kiến trúc jPDL

- Xin vui lòng tham khảo phần phụ lục để biết chi tiết cách cài đặt plugin vào eclipse.

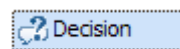
3.1.3 Process graph:



Hình 3-5 Các node trong process editor

- Để mô hình hóa qui trình nghiệp vụ, ta sử dụng hệ thống các ký hiệu (notation) trong thư việc jpdl.
- Mỗi ký hiệu sẽ được sử dụng riêng biệt trong từng tình huống khác nhau.

3.1.3.1 Decision Node:



- Dùng để đưa ra nhiều lựa chọn, có thể sử dụng với n lựa chọn khác nhau.

- Kết quả trả về của là tên của transition được chọn.
- Lựa chọn được thực thi từ code, bằng cách kế thừa lớp Decision Handler.
- Có thể tạo nhiều leaving transition cho decision node.

3.1.3.2 Process State:

- Process state là trạng thái được gắn kèm trong một process definition khác.
- Khi luồng công việc thực hiện đến process state, một process instance của process con được tạo ra và gắn vào luồng công việc.
- Luồng thực thi của super process sẽ chuyển sang trạng thái chờ cho đến khi process con hoàn tất.
- Sau khi process con hoàn tất, qui trình chính sẽ được tiếp tục.

3.1.3.3 Super State:

- Là một nhóm các node được gộp chung lại với nhau. Superstate có thể lồng nhau.
- Bao gồm các thuộc tính con như:
 - ❖ Superstate transitions: transition từ qui trình chính có thể trở trực tiếp vào super state. Các node ở bên ngoài cũng có thể trở trực tiếp vào bên trong của super state. Ngoài ra, superstate cũng có thể tự trở đến nó.
 - ❖ Superstate events: bao gồm superstate-enter và superstate-leave. Các event này sẽ được khởi tạo theo trình tự thông qua các transition từ qui trình chính.

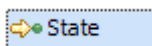
3.1.3.4 So sánh SUPER STATE vs PROCESS STATE:

Super state	Process state
-------------	---------------

- Dùng để nhóm các node lại với nhau.	- Dùng để khai báo sub-process.
- Không được đặt start-state & end-state vào giữa	

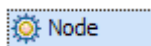
Bảng 3-1 So sánh Super state và process state

3.1.3.5 Node State:



- State là một node có trạng thái chờ.
- Điểm khác biệt so với task node là state không có task instance được tạo ra trong task list. Điều này hữu ích nếu qui trình phải chờ phản hồi của hệ thống ngoại vi.

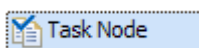
3.1.3.6 Node:



- Node được sử dụng trong trường hợp người dùng cần thiết kế hàm xử lý riêng trong node. Trong trường hợp này, người dùng sẽ sử dụng code để tạo ra các hàm xử lý cụ thể với từng qui trình nghiệp vụ trong thực tế.
- Node được kích hoạt thực hiện khi tín hiệu thực thi lan truyền đến node. Các hàm xử lý kèm theo có thể thao tác đến toàn bộ qui trình xử lý của luồng công việc.
- Node có bao gồm code kèm theo thường phục vụ cho những qui trình nghiệp vụ quan trọng, do vậy node được thiết kế để người dùng có thể nhìn thấy được trên editor.
- Hai chức năng chính của node là:
 - ❖ Thực thi mã nguồn java.
 - ❖ Có khả năng lan truyền thông tin kích hoạt qui trình.

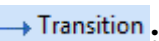
- Mỗi node có nhiều lựa chọn để lan truyền thông tin kích hoạt khác nhau như:
 - ❖ Không lan truyền kích hoạt.
 - ❖ Kích hoạt thông qua leaving transition.
 - ❖ Tạo luồng kích hoạt mới.
 - ❖ Kế thúc luồng kích hoạt.

3.1.3.7 Task-Node:



- Dùng để thực thi một hoặc nhiều task, task node được tương tác bởi người dùng.
- Với những quy trình nghiệp vụ đòi hỏi phải có thao tác của người dùng thì task-node được lựa chọn.
- Cơ chế hoạt động của một task node như sau:
 - ❖ Tín hiệu kích hoạt được gửi đến.
 - ❖ Task instance được tạo ra.
 - ❖ Khi này task-node sẽ chuyển sang trạng thái chờ nhận phản hồi từ người dùng.
 - ❖ Sau khi người dùng phản hồi lại, tín hiệu kích hoạt được tạo ra hoặc một tín hiệu thông tin được khởi tạo bởi hệ thống để cho biết người dùng đã thực hiện xong.

3.1.3.8 Transitions



- Là thành phần dùng để liên kết các node trong luồng công việc. Transition node gắn với node bắt đầu và node kết thúc.
- Mỗi transition có tên, hầu hết các công cụ của jBPM phụ thuộc vào tính duy nhất của tên transition.
- Trong trường hợp có nhiều transition trùng tên nhau, mặc định, hệ thống sẽ lấy tên transition đầu tiên trong danh sách

transition lúc bắt đầu triển khai qui trình. Ngoài ra cũng có thể can thiệp bằng code để cho phép lựa chọn transition trùng tên.

3.1.4 Giới thiệu jBPM-engine:

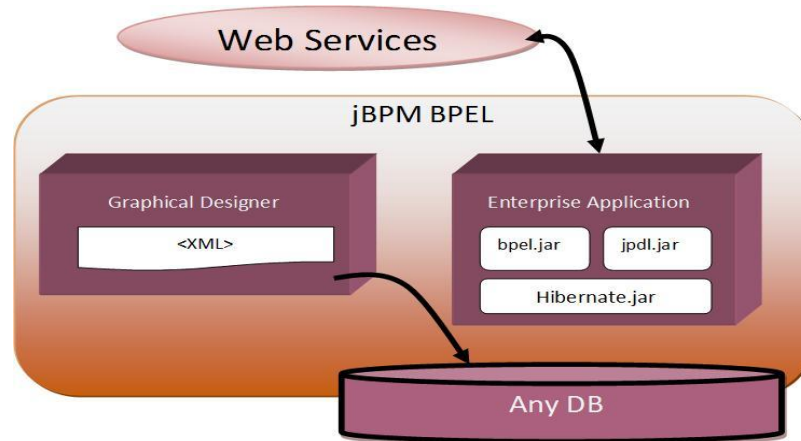
3.1.4.1 Tổng quát:



Hình 3-6 Kiến trúc jBPM

3.1.4.2 Thành phần trong jBPM:

- jPDL là ngôn ngữ qui trình với các lớp java thuần túy và khả năng quản lý những task phức tạp.
- Một trong những tính năng nổi bật của jPDL là nó có khả năng mở rộng ngôn ngữ bằng cách thêm vào node mới. Để đáp ứng nhu cầu người sử dụng, jPDL cũng cho phép thêm vào thể hiện đồ họa của các node mở rộng này. Tất cả các node được xây dựng trong jPDL plugin sẽ sử dụng các extension point trong khi cấu hình plugin.
- BPEL là ngôn ngữ hướng dịch vụ. Trong BPEL, người dùng có thể tạo thêm dịch vụ được dùng như một hàm khác của service khác.



Hình 3-7 Kiến trúc BPEL

- SEAM pageflow là ngôn ngữ cho phép định nghĩa việc di chuyển giữa các trang trong ứng dụng web SEAM một cách trực quan. Tuy không được tách biệt thành một dự án riêng, nhưng pageflow là ngôn ngữ qui trình quan trọng hỗ trợ bởi jBPM. SEAM đề xuất sử dụng jBPM Pageflow như là cơ chế chính để di chuyển giữa các trang web, trong đó, có hỗ trợ tạo thêm những pageflow một cách trực quan bằng cách sử dụng bộ công cụ hỗ trợ giao diện đồ họa (Graphical Process Designer).
- The Process Virtual Machine (PVM): PVM quản lý phần công việc chung nhất như việc quản lý trạng thái và thực thi đồ thị. Các ngôn ngữ qui trình mở rộng thêm cho PVM, cho phép các ngôn ngữ này có thể sử dụng cùng một kỹ thuật. PVM là một thư viện java đơn giản để xây dựng và thực thi đồ thị. Ngôn ngữ qui trình là tập các kiểu hoạt động được tạo ra bằng cách sử dụng tính năng cung cấp bởi PVM.
- The Graphical Process Designer (GPD) là tập các plugin của eclipse xây dựng trên nền tảng Web Tools Platform (WTP) và Graphical Editing Framework (GEF). GPD dùng cho nhiều mục đích khác nhau. Trước tiên là hỗ trợ tạo mới và

chỉnh sửa qui trình nghiệp vụ jPDL và Seam pageflows dạng đồ họa. Mục đích khác nữa là cho phép người mở rộng bổ sung thêm node mới và thậm chí cho phép xây dựng lại một hệ thống ký hiệu định nghĩa dựa trên đồ thị hoàn toàn mới.

- Identity: Thông thường, việc quản lý người dùng, nhóm và các quyền được biết đến như là quản lý định danh. jBPM hiện tại đã bao gồm thành phần quản lý định danh có thể thay thế dễ dàng bởi thành phần tùy biến bởi người sử dụng. Thành phần chứng thực này sẽ sớm được tách thành một dự án riêng độc lập với jBPM.
- Task management: tác vụ và việc quản lý tác vụ là phần quan trọng nhất trong hầu hết các ngôn ngữ qui trình. Thành phần quản lý tác vụ sẽ được tách rời thành một dự án khác trong tương lai. Điều này giúp tạo ra nhiều sự khác biệt hơn giữa ngôn ngữ qui trình như jPDL và XPD.
- Enterprise: thành phần doanh nghiệp hiện đang được xây dựng, thành phần này sẽ bao gồm các enterprise bean có thể dùng cho ứng dụng server như: JBoss AS.

3.1.5 Vai trò của workflow pattern với thiết kế luồng công việc:

3.1.5.1 Giới thiệu workflow pattern:

- Năm 1999, workflow patterns khởi đầu được cộng tác nghiên cứu bởi Eindhoven University of Technology (bởi giáo sư [Professor Wil van der Aalst](#)) và Queensland University of Technology (bởi giáo sư [Professor Arthur ter Hofstede](#)).
- Mục tiêu của việc cộng tác là tạo tập khái niệm ban đầu cho kỹ thuật qui trình. Cụ thể hơn, công trình nghiên cứu cung cấp những thử nghiệm trên nhiều mẫu khác nhau (control flow, data, resource và exception handling) cần được hỗ trợ

bởi ngôn ngữ workflow hoặc ngôn ngữ mô hình hóa quy trình nghiệp vụ.

- Kết quả của những thử nghiệm có thể được dùng để đánh giá mức độ tương hợp của từng ngôn ngữ quy trình cụ thể hoặc hệ thống workflow trong một dự án cụ thể, đánh giá được ưu, khuyết điểm dựa trên nhiều cách tiếp cận đặc tả quy trình, yêu cầu nghiệp vụ cụ thể cần triển khai trong hệ thống thông tin dựa trên quy trình cụ thể và là nền tảng cho sự phát triển của các công cụ và ngôn ngữ.

3.1.5.2 Đánh giá mức độ hỗ trợ của hệ thống jBPM với OpenWFE và EnhydraShark:

- Trong các bảng so sánh dưới đây, mỗi pattern sẽ được so sánh lần lượt với 3 hệ thống là jBPM, OpenWFE và EnhydraShark.
- Với mỗi workflow pattern, ký hiệu +, - và +/- có ý nghĩa như sau:
 - ❖ +: trong trường hợp hệ thống hỗ trợ trực tiếp mẫu workflow tương ứng.
 - ❖ -: trong trường hợp hệ thống không hỗ trợ, xem xét tương đương với những sơ đồ có lồng code hoặc dạng sơ đồ spaghetti.
 - ❖ +/-: được sử dụng trong trường hợp hệ thống không hỗ trợ trực tiếp.
- Control-Flow Patterns:

Pattern	Product		
	jBPM	OpenWFE	Enhydra Shark

Sequence	+	+	+
Parallel Split	+	+	+
Synchronization	+	+	+
Exclusive Choice	+	+	+
Simple Merge	+	+	+
Multi-Choice	-	+/-	+
Structured Synchronizing Merge	-	-	-
Multi-Merge	+	-	-
Structured Discriminator	-	+	-
Arbitrary Cycles	+	+	+
Implicit Termination	+	+	+
Multiple Instances without Synchronization	+	+	+
Multiple Instances with a Priori Design-Time Knowledge	-	+	-
Multiple Instances with a Priori Run-Time Knowledge	-	+	-
Multiple Instances without a Priori Run-Time Knowledge	-	-	-
Deferred Choice	+	-	-
Interleaved Parallel Routing	-	+/-	-
Milestone	-	-	-
Cancel Activity	+	-	-
Cancel Case	-	+/-	+
Structured Loop	-	+	-
Recursion	-	+	+
Transient Trigger	+	+	-
Persistent Trigger	-	-	-

Cancel Region	-	-	-
Cancel Multiple Instance Activity	-	-	-
Complete Multiple Instance Activity	-	-	-
Blocking Discriminator	-	-	-
Cancelling Discriminator	-	+	-
Structured Partial Join	-	+	-
Blocking Partial Join	-	-	-
Cancelling Partial Join	-	+	-
Generalised AND-Join	+	-	-
Static Partial Join for Multiple Instances	-	+	-
Cancelling Partial Join for Multiple Instances	-	+	-
Dynamic Partial Join for Multiple Instances	-	-	-
Local Synchronizing Merge	-	+/-	-
General Synchronizing Merge	-	-	-
Critical Section	-	-	-
Interleaved Routing	-	+	-
Thread Merge	+/-	-	-
Thread Split	+/-	-	-
Explicit Termination	-	-	-

Bảng 3-2 Control flow patterns

- Resource Patterns:

Pattern	Product		
	jBPM	OpenWFE	Enhydra Shark
Direct Allocation	+	-	+
Role-Based Allocation	-	+	+

Deferred Allocation	+	+	+
Authorisation	-	-	-
Seperation of Duties	-	-	-
Case Handling	-	-	-
Retain Familiar	+	-	-
Capability Based Allocation	-	-	-
History Based Allocation	-	-	-
Organisational Allocation	-	-	-
Automatic Execution	+	+	+
Distribution by Offer - Single Resource	-	-	+
Distribution by Offer - Multiple Resources	-	+	+
Distribution by Allocation - Single Resource	+	-	-
Random Allocation	-	-	-
Round Robin Allocation	-	-	-
Shortest Queue	-	-	-
Early Distribution	-	-	-
Distribution on Enablement	+	+	+
Late Distribution	-	-	-
Resource-Initiated Allocation	-	-	-
Resource-Initiated Execution - Allocated Work Item	+	-	-
Resource-Initiated Execution - Offered Work Item	-	+	+
System Determined Work Queue Content	-	-	-
Resource-Determined Work Queue Content	-	-	-
Selection Autonomy	+	+	+
Delegation	-	-	-

Escalation	-	+	-
Deallocation	-	+	+
Stateful Reallocation	-	+	-
Stateless Reallocation	-	-	-
Suspension/Resumption	+	-	-
Skip	-	-	-
Redo	-	+/-	-
Pre-Do	-	-	-
Commencement on Creation	-	-	-
Commencement on Allocation	-	-	-
Piled Execution	-	-	-
Chained Execution	-	-	-
Configurable Unallocated Work Item Visibility	-	+/-	-
Configurable Allocated Work Item Visibility	-	+/-	-
Simultaneous Execution	-	-	-
Additional Resources	-	-	-

Bảng 3-3 resource patterns

- Data Patterns:

Pattern	Product		
	jBPM	OpenWFE	Enhydra Shark
Task Data	+/-	-	+/-
Block Data	-	+	+
Scope Data	-	+/-	-
Multiple Instance Data	-	+	+
Case Data	+	+	+

Folder Data	-	-	-
Workflow Data	-	+	-
Environment Data	+/-	+	+/-
Task to Task	+	+	+
Block Task to SubWorkflow Decomposition	-	+	+
SubWorkflow Decomposition to Block Task	-	+	+
To Multiple Instance Task	-	-	-
From Multiple Instance Task	-	-	-
Case to Case	+/-	+/-	+/-
Task to Environment - Push-Oriented	+/-	+	+/-
Environment to Task - Pull-Oriented	+/-	+	+/-
Environment to Task - Push-Oriented	-	-	-
Task to Environment - Pull-Oriented	-	-	-
Case to Environment - Push-Oriented	-	-	-
Environment to Case - Pull-Oriented	-	-	-
Environment to Case - Push-Oriented	-	-	-
Case to Environment - Pull-Oriented	-	-	-
Workflow to Environment - Push-Oriented	-	-	-
Environment to Workflow - Pull-Oriented	-	-	-
Environment to Workflow - Push-Oriented	-	-	-
Workflow to Environment - Pull-Oriented	-	-	-
Data Transfer by Value - Incoming	-	-	+/-
Data Transfer by Value - Outgoing	-	-	+/-
Data Transfer - Copy In/Copy Out	+	+	+
Data Transfer by Reference - Unlocked	-	-	-
Data Transfer by Reference - With Lock	-	+	-

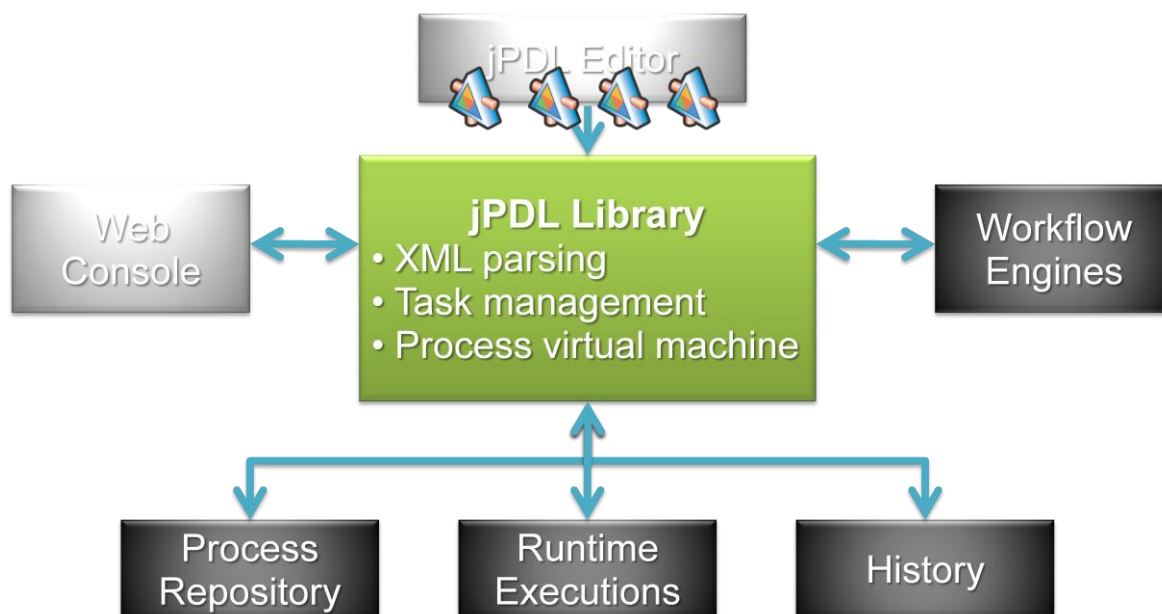
Data Transformation - Input	+	+	+
Data Transformation - Output	+	+	+
Task Precondition - Data Existence	-	+	-
Task Precondition - Data Value	-	+	-
Task Postcondition - Data Existence	-	-	-
Task Postcondition - Data Value	-	-	+/-
Event-Based Task Trigger	-	-	-
Data-Based Task Trigger	-	-	-
Data-Based Routing	+/-	+/-	+

Bảng 3-4 Data patterns

3.2 Hiện trạng của hệ thống:

3.2.1 Mô tả tổng quát:

3.2.1.1 Kiến trúc hệ thống ban đầu:



○ jPDL editor:

- ❖ Ban đầu, hệ thống chỉ có jPDL editor ở dạng sơ khai, không hỗ trợ đặc tả document trong qui trình.

- ❖ Hệ thống chỉ có phần đặc tả các bước của qui trình, trong đó, người dùng trong mỗi bước sẽ được chỉ định cụ thể và không thay đổi được.

- Web console:

- ❖ Hệ thống giao diện người dùng ban đầu chỉ được xây dựng với các chức năng cơ bản như quản lý các qui trình, quản lý người dùng, quản lý phân quyền người dùng.
- ❖ Hệ thống giao diện không có chức năng ánh xạ người dùng tổng quát (lúc thiết kế qui trình) với người dùng thực tế khi qui trình được triển khai.

- Process repository:

- ❖ Hệ thống ban đầu chỉ phục vụ lưu trữ thông tin của qui trình được tạo triển khai. Mỗi khi qui trình được thực thi thì một thể hiện sẽ được tạo ra và có thể thực thi đồng thời, mang tính độc lập cao. Vì vậy có thể thực thi cùng lúc nhiều thể hiện khác nhau với cùng một qui trình được triển khai.

3.2.1.2 Yêu cầu của bài toán và khả năng đáp ứng của hệ thống:

- Biểu diễn document trong workflow:

- ❖ Hệ thống ban đầu không hỗ trợ đặc tả document trong ngôn ngữ qui trình jPDL (jPDL process language).

- Quản lý document:

- ❖ Do không hỗ trợ đặc tả document trong qui trình nên các chức năng quản lý document cũng không được xây dựng trong hệ thống.
- Đặc tả người dùng tổng quát:
 - ❖ Hệ thống không hỗ trợ đặc tả người dùng tổng quát khi xây dựng workflow mà chỉ hỗ trợ chỉ định người dùng cụ thể trong qui trình mà người dùng đó đã tồn tại trong cơ sở dữ liệu HyperSonic ban đầu (HyperSonic Data base).
 - ❖ Do vậy, khi gán một tác vụ tại một bước nào đó trong qui trình thì phải gán cho một user cụ thể. Đây là điểm hạn chế lớn của hệ thống ban đầu.
 - ❖ Tính tái sử dụng của một qui trình sẽ không cao.
- Đặc tả tài liệu tổng quát:
 - ❖ Nhằm tăng tính tái sử dụng của hệ thống, nhóm đã phát triển phần đặc tả tài liệu tổng quát để có thể tái sử dụng qui trình.
- Tích hợp với hệ thống tài khoản sẵn có:
 - ❖ Hệ thống ban đầu chỉ hỗ trợ tích hợp tài khoản với hệ thống người dùng có sẵn trong cơ sở dữ liệu HyperSonic nên tính tương thích không cao.
 - ❖ Nhược điểm của hệ thống ban đầu là người dùng phải xây dựng hệ thống tài khoản truy cập ban đầu. Hệ thống tài khoản này hoàn toàn tách biệt với hệ thống LDAP, Active Directory trong hệ thống có sẵn.
- Lưu trữ với cơ sở dữ liệu MySQL:

❖ Hệ thống ban đầu không hỗ trợ sử dụng cơ sở dữ liệu MySQL mà chỉ hỗ trợ cơ sở dữ liệu HyperSonic.

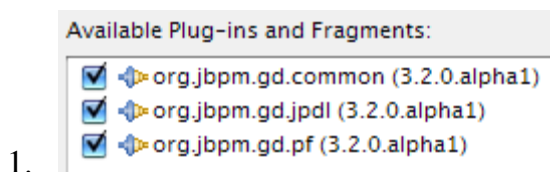
○ Sử dụng tài khoản trên LDAP để đăng nhập vào hệ thống:

❖ Trong khoa đã sử dụng những hệ thống lưu trữ người dùng khác để đăng nhập chung cho nhiều hệ thống khác nhau, cụ thể là sử dụng hệ thống tài khoản LDAP (Light-weight directory access protocol).

3.3 Chức năng mới:

3.3.1 Xây dựng document trên jPDL-editor:

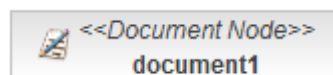
- Bộ jPDL được tạo thành từ 3 gói



Hình 3-8 Các gói trong jPDL source

- Trong đó, 2 gói chứa mã nguồn thư viện là: org.jbpm.gd.common & org.jbpm.gd.pf. Gói org.jbpm.gd.jpdl chứa toàn bộ mã nguồn của document node, trong đó lớp document sẽ sử dụng những hàm thư viện khai báo trong 2 gói trên.

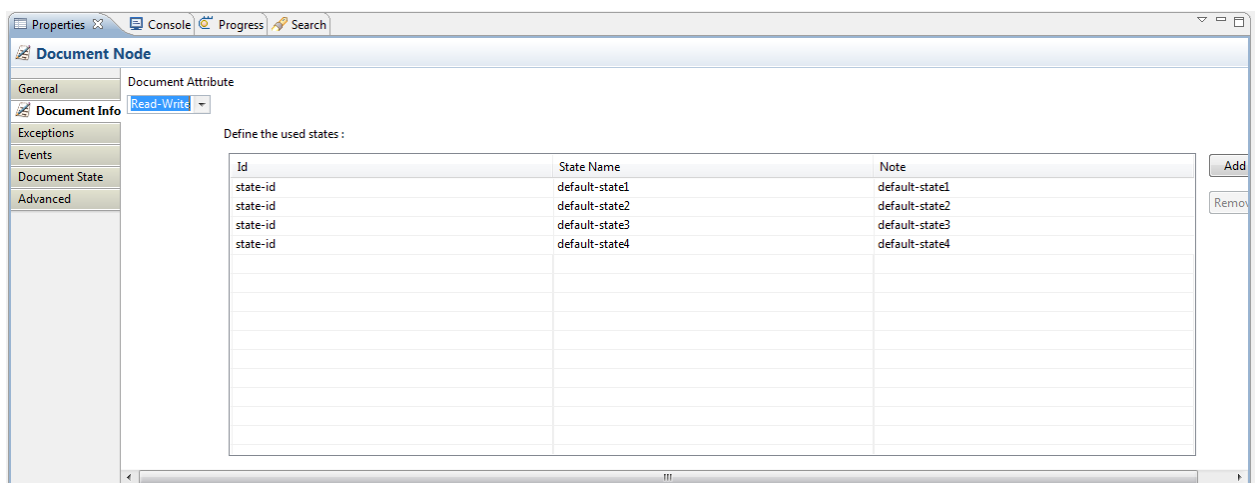
3.3.1.1 Document node:



Hình 3-9 Document Node

- Đặc tả luồng xử lý:

- Người dùng thêm document node vào luồng công việc, số lượng document node được thêm vào tương ứng với từng loại quy trình nghiệp vụ cụ thể.
- Với mỗi document node, người dùng nhập vào thông tin cần thiết như :
 - ❖ Tên document: tên của document được sử dụng mang tính trừu tượng, không nhất thiết phải giống tên file thực tế.
 - ❖ Các state của document: mỗi state của document sẽ được biểu diễn cụ thể trong bảng.



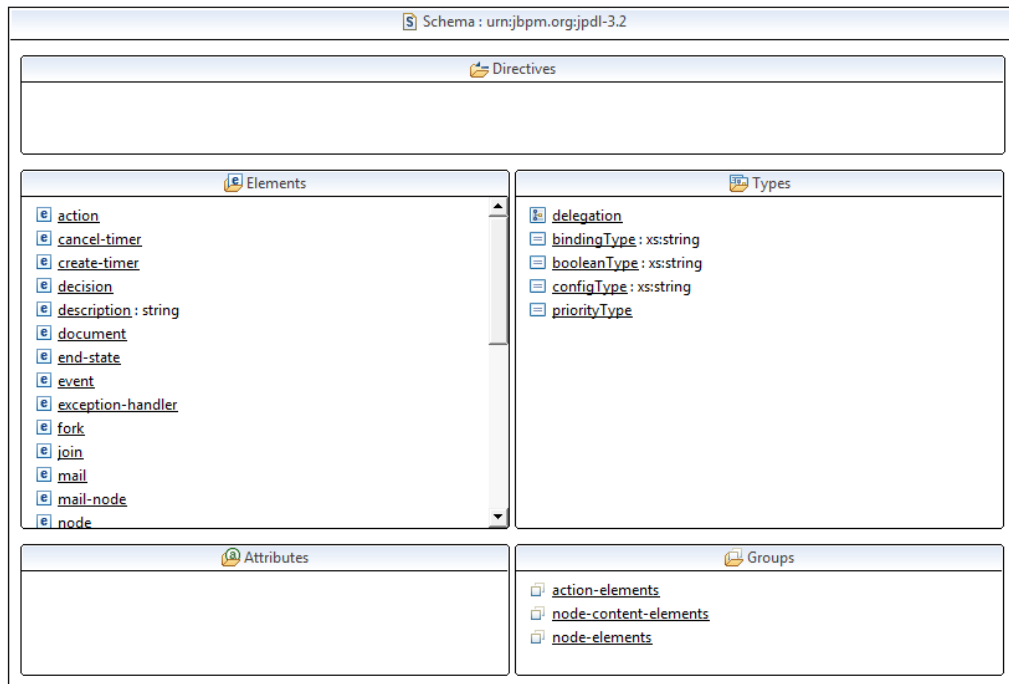
Hình 3-10 Thuộc tính trạng thái của document

- ❖ Bên cạnh đó, biểu diễn bên file xml cũng được cập nhật đồng bộ tương ứng với những thay đổi trong hệ thống.

```
<document name="document1" documentAttribute="Read-Write">
  <document-state stateId="state-id" stateName="default-state1" stateNote="default-state1"></document-state>
  <document-state stateId="state-id" stateName="default-state2" stateNote="default-state2"></document-state>
  <document-state stateId="state-id" stateName="default-state3" stateNote="default-state3"></document-state>
  <document-state stateId="state-id" stateName="default-state4" stateNote="default-state4"></document-state>
</document>
```

Hình 3-11 Đặc tả trạng thái tài liệu dạng XML

- Bổ sung thông tin node mới trong file xsd:



Hình 3-12 Đặc tả thông tin document node trong xsd

- Sau khi thực hiện các bước trên để xây dựng và khai báo các lớp java. Bước cuối cùng là bổ sung thông tin về các element cũng như attribute được sử dụng trong node mới được tạo ra.

```

370 <!-- DOCUMENT STATE-->
371 <!-- #### -->
372 <xs:element name="document">
373   <xs:complexType>
374     <xs:choice minOccurs="0" maxOccurs="unbounded">
375       <xs:element name="document-state">
376         <xs:complexType>
377           <xs:attribute name="stateId" type="xs:string" />
378           <xs:attribute name="stateName" type="xs:string"/>
379           <xs:attribute name="stateNote" type="xs:string"/>
380         </xs:complexType>
381       </xs:element>
382     </xs:choice>
383     <xs:attribute name="name" type="xs:string" />
384     <xs:attribute name="description" type="xs:string" />
385   </xs:complexType>
386 </xs:element>

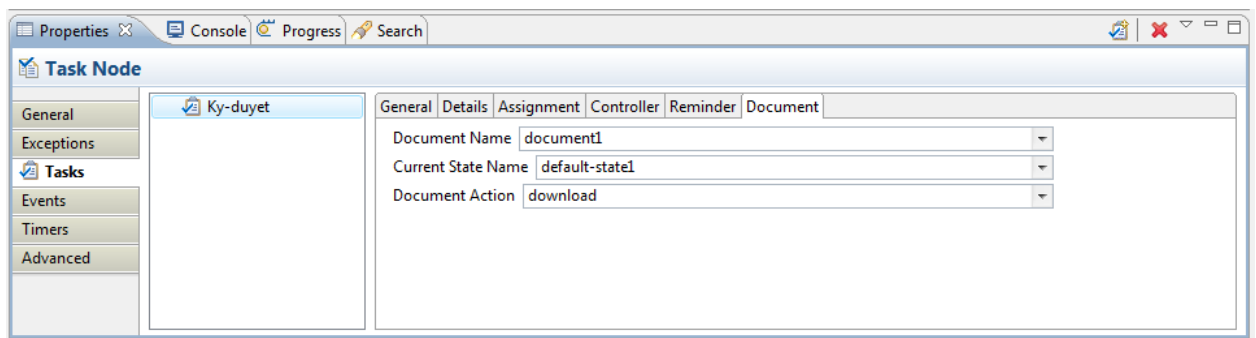
```

Hình 3-13 Mã nguồn xsd cần điều chỉnh

3.3.1.2 Document property tab:

- Đặc tả luồng xử lý:

- Sau khi thêm document node vào editor, người dùng sẽ tạo những task có liên kết với document. Thông tin document được đặt trong document property tab, bao gồm thông tin tất cả các document node được khai báo trong editor.
- Với mỗi document được chọn, thông tin tương ứng của từng document cũng được load lên cho người thiết kế lựa chọn.
- Ứng với mỗi bước trong luồng công việc, document được sử dụng kèm theo sẽ được chỉ định trạng thái cụ thể lúc đó và hành động mà người dùng có thể thao tác trên document đó.



Hình 3-14 Assign trạng thái document trong process

- Thông tin được cập nhật trong file xml tương ứng như sau:

```
<task>
  <document documentId="boney-id" documentName="document1" currentStateId="boney-id" currentStateName="default-state1" documentAction="upload">
  </document>
</task>
```

Hình 3-15 Thẻ hiện document với task trong xml

- Bổ sung thông tin node mới trong file xsd:

```

334 <!-- TASK -->
335 <!-- ##### -->
336 <xs:element name="task">
337   <xs:complexType>
338     <xs:choice minOccurs="0" maxOccurs="unbounded">
339       <xs:element ref="description" />
340       <xs:element ref="assignment"/>
341       <xs:element ref="controller"/>
342       <xs:element ref="event"/>
343       <xs:element ref="timer"/>
344     <xs:element name="reminder">
345       <xs:complexType>
346         <xs:attribute name="duedate" type="xs:string" use="required" />
347         <xs:attribute name="repeat" type="xs:string" />
348       </xs:complexType>
349     </xs:element>
350     <xs:element name="document">
351       <xs:complexType>
352         <xs:attribute name="name" type="xs:string" use="required" />
353         <xs:attribute name="documentName" type="xs:string" use="required" />
354         <xs:attribute name="documentState" type="xs:string" />
355         <xs:attribute name="documentAction" type="xs:string" />
356       </xs:complexType>
357     </xs:element>
358   </xs:choice>
359   <xs:attribute name="name" type="xs:string" />
360   <xs:attribute name="blocking" type="booleanType" default="false"/>
361   <xs:attribute name="signalling" type="booleanType" default="true"/>
362   <xs:attribute name="description" type="xs:string" />
363   <xs:attribute name="duedate" type="xs:string" />
364   <xs:attribute name="swimlane" type="xs:string" />
365   <xs:attribute name="priority" type="priorityType" default="normal" />
366   <xs:attribute name="notify" type="booleanType" default="false"/>

```

Hình 3-16 Bổ sung thêm element trong xsd

3.3.2 Document management system:

3.3.2.1 Giới thiệu:

- Hệ quản trị tài liệu là một hệ thống cho phép người dùng quản lý tài liệu điện tử ở dạng các tập tin. Những tài liệu này có thể là các văn bản, hình ảnh hoặc bảng tính. Thao tác quản lý bao gồm việc cung cấp một nơi chứa tài liệu, quản lý các phiên bản của một tài liệu, các phương thức đưa tài liệu vào cũng như lấy tài liệu về từ DMS. Một số hệ thống DMS còn có thêm các chức năng nâng cao như mã hóa và phân quyền người dùng truy cập vào các tài liệu, quản lý sự chuyển đổi trạng thái của các tài liệu, cung cấp một cơ chế tìm kiếm nâng cao các tài liệu trong hệ thống.
- Một hệ thống dòng công việc sẽ không có ý nghĩa nhiều nếu người dùng không thể thao tác với các tài liệu. Thông

thường, công việc quản lý nội dung của một hệ thống dòng công việc được thực hiện bởi một DMS gắn kèm theo hoặc bản thân hệ thống dòng công việc cũng phải là một DMS.

3.3.2.2 Các chức năng chính của một hệ thống DMS:

- Check in / check out:
 - ❖ Để đảm bảo tính nhất quán của một tài liệu, DMS cần phải quản lý việc check in và check out của người dùng trên tài liệu đó. Thông thường công việc này được thực hiện bằng cách tạo ra phiên bản mới của tài liệu khi có một sự thay đổi đối với nó.
- Quản lý phiên bản – Versioning:
 - ❖ Chức năng này nhằm cung cấp cho người dùng khả năng tham khảo lại một tài liệu ở các phiên bản trước đó. Các thông tin cần phải quản lý đối với mỗi phiên bản, ngoài các thông tin về bản thân tài liệu gốc còn bao gồm: người tạo version mới, thời gian được tạo ra version mới, ghi chú trên phiên bản mới. Đối với các DMS quản lý tài liệu theo trạng thái thì các phiên bản mới được tạo ra theo một biểu đồ trạng thái (statechart) được định nghĩa trước.
- Chức năng tìm kiếm:
 - ❖ Chức năng này cung cấp cho người dùng khả năng tìm kiếm một tài liệu hoặc một phiên bản của tài liệu theo các điều kiện khác nhau.

3.3.2.3 Xây dựng DMS cho engine workflow JBPM:

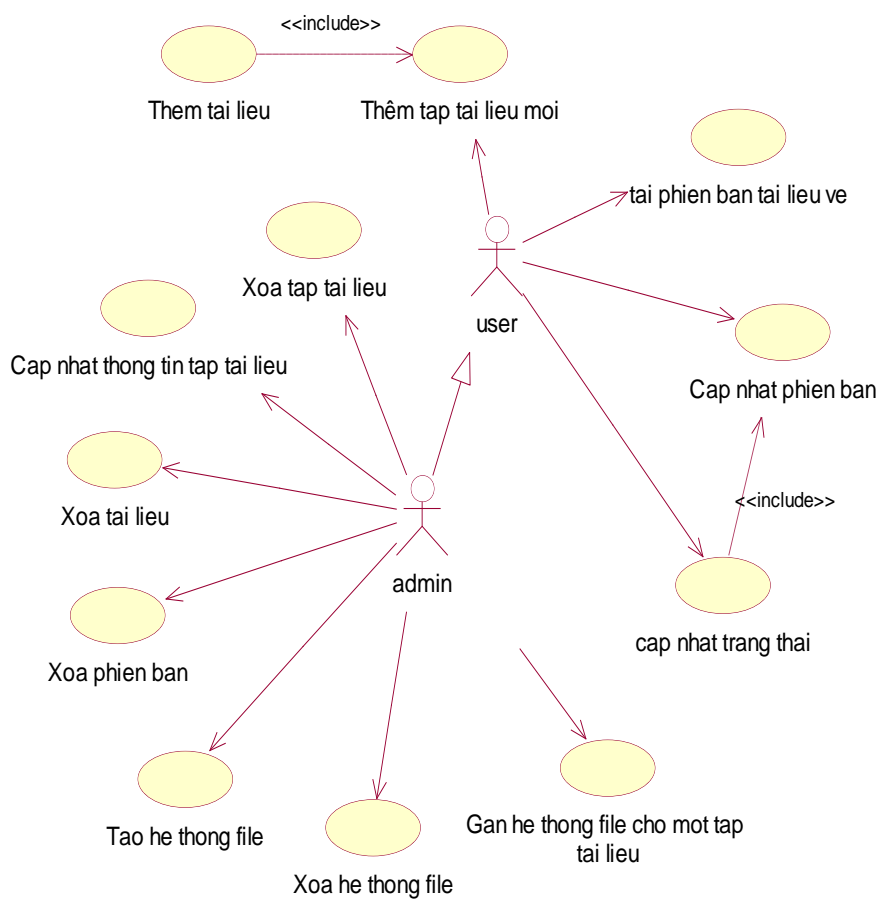
- Phân tích:

❖ Yêu cầu đặt ra đối với DMS được xây dựng bao gồm các phần sau:

- Cung cấp cho người dùng một nơi lưu trữ tài liệu.
- Cho phép người dùng download và upload tài liệu thông qua giao diện web.
- Cho phép người dùng định nghĩa trước một tập các tài liệu làm việc trong một quy trình.
- Cho phép người dùng định nghĩa các trạng thái có thể có của một document.
- Quản lý các tài liệu thông qua các trạng thái, mỗi một trạng thái tương ứng với một version của tài liệu.
- Cho phép người dùng tìm kiếm và tham khảo các phiên bản trước của tài liệu.

❖ Vì jBPM được xây dựng trên nền tảng Java Enterprise Application, kết nối cơ sở dữ liệu MySQL thông qua Hibernate nên DMS cũng sẽ được phát triển trên cùng nền tảng và công nghệ kết nối cơ sở dữ liệu, Eclipse được chọn làm môi trường phát triển tích hợp.

○ Sơ đồ use-case:



Hình 3-17 Sơ đồ use-case cho phân hệ quản trị tài liệu

❖ Danh sách các Actor:

STT	Tên Actor	Mô tả
1	User	Người sử dụng
2	Administrator	Quản trị hệ thống, kế thừa từ Người sử dụng.

Bảng 3-5 Danh sách các actor cho phân hệ DMS

STT	Tên use-case	Mô tả
1	Thêm tập tài liệu	Thêm một tập tài liệu mới vào DMS

2	Xóa tập tài liệu	Xóa một tập tài liệu cùng với tất cả tài liệu thuộc tập tài liệu đó ra khỏi hệ thống.
3	Cập nhật tập tài liệu	Cập nhật thông tin về tên, hệ thống file của một tập tài liệu.
4	Thêm tài liệu mới	Thêm tài liệu mới vào một tập tài liệu, người dùng không trực tiếp thực hiện use-case này.
5	Xóa tài liệu	Xóa một tài liệu ra khỏi hệ thống.
6	Tạo hệ thống file	Tạo một hệ thống file để lưu trữ các tài liệu.
7	Cập nhật hệ thống file	Cập nhật thông tin hệ thống file.
8	Xóa hệ thống file	Xóa hệ thống file ra khỏi hệ thống.
9	Gán hệ thống file cho một tập tài liệu.	Một tập tài liệu cần phải có nơi lưu trữ. Use-case này gán một hệ thống file cho một tập tài liệu.
10	Tải về phiên bản	Tải về một phiên bản của một tài liệu nào đó.
11	Cập nhật một tài liệu.	Cập nhật một tài liệu
12	Xóa phiên bản	Xóa một phiên bản của một tài liệu.

Bảng 3-6 Danh sách các use-case phân hệ DMS

- Đặc tả một số Use-Case chính:
 - Trong quá trình đặc tả, nếu điều kiện thực hiện, điều kiện kết thúc, điểm mở rộng nếu có thì sẽ được đưa thêm vào. Những điều kiện không có sẽ được lược bỏ.

- Đặc tả use-case “Thêm tập tài liệu”:
 - Tóm tắt:
 - ❖ Một quy trình nghiệp vụ thường có một tập tài liệu làm việc đi kèm, use-case này thực hiện việc thêm tập tài liệu cho quy trình. Việc thêm một tập tài liệu mới được thực hiện cùng với việc triển khai một quy trình nghiệp vụ mới.
 - Đặc tả dòng sự kiện:
 - ❖ Dòng sự kiện chính:
 - Use-case bắt đầu khi người dùng triển khai một quy trình nghiệp vụ mới.
 - Hệ thống đọc thông tin tên của quy trình để gán cho tên của tập tài liệu mới.
 - Hệ thống đọc thông tin của từng tài liệu được mô tả trong tập tin mô tả nghiệp vụ và lưu thông tin này cùng với thông tin của tập tài liệu.
 - Hệ thống lưu toàn bộ thông tin của tập tài liệu cùng với thông tin của từng tài liệu vào cơ sở dữ liệu.
 - Hệ thống kết thúc quá trình tạo tập tài liệu mới.
 - ❖ Dòng sự kiện khác
 - Có nhiều hơn một tài liệu trong tập tin mô tả có cùng tên và định dạng, hệ thống sẽ báo lỗi và dừng quá trình triển khai.
 - ❖ Yêu cầu đặc biệt

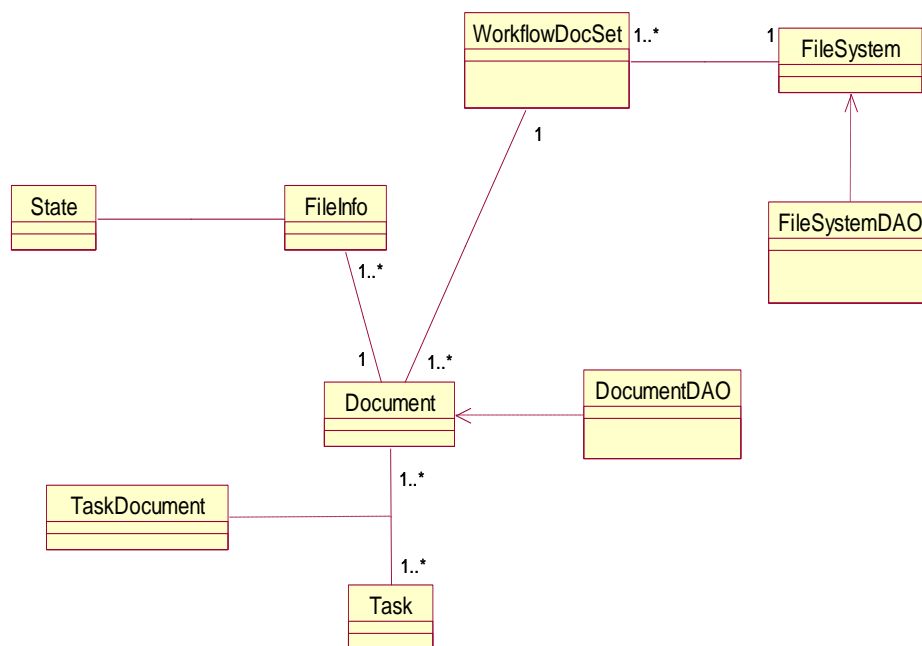
- Tập tin mô tả phải được viết đúng định dạng đã được quy định trước (Xem quy định tại mục phụ lục).
- Đặc tả use-case “Thêm tài liệu”:
 - Tóm tắt:
 - ❖ Việc thêm một tài liệu mới được thực hiện cùng với việc thêm mới một tập tài liệu tại thời điểm triển khai một quy trình nghiệp vụ.
 - Dòng sự kiện:
 - ❖ Dòng sự kiện chính:
 - Use-case bắt đầu khi người dùng triển khai một quy trình nghiệp vụ mới sau khi đã ghi nhận thông tin về tập tài liệu.
 - Hệ thống đọc thông tin từng tài liệu được mô tả trong tập tin định nghĩa quy trình.
 - Với mỗi một tập thông tin về một tài liệu được mô tả, hệ thống tạo một đối tượng tài liệu ảo.
 - Hệ thống lưu tất cả các đối tượng tài liệu ảo đó xuống cơ sở dữ liệu.
 - Hệ thống kết thúc quá trình tạo tài liệu mới.
 - ❖ Dòng sự kiện khác:
 - Có nhiều hơn một tài liệu trong tập tin mô tả có cùng tên và định dạng, hệ thống sẽ báo lỗi và dừng quá trình triển khai.

- Nếu có một tài liệu được mô tả trong tập tin định nghĩa quy trình nghiệp vụ bị thiếu thông tin về tên thì hệ thống sẽ báo lỗi và dừng quá trình triển khai.

❖ Yêu cầu đặc biệt:

- Tập tin mô tả phải được viết đúng định dạng đã được quy định trước

❖ Sơ đồ lớp mức phân tích:



Hình 3-18 Sơ đồ lớp phân hệ quản trị tài liệu

- Bảng danh sách các lớp đối tượng và quan hệ:

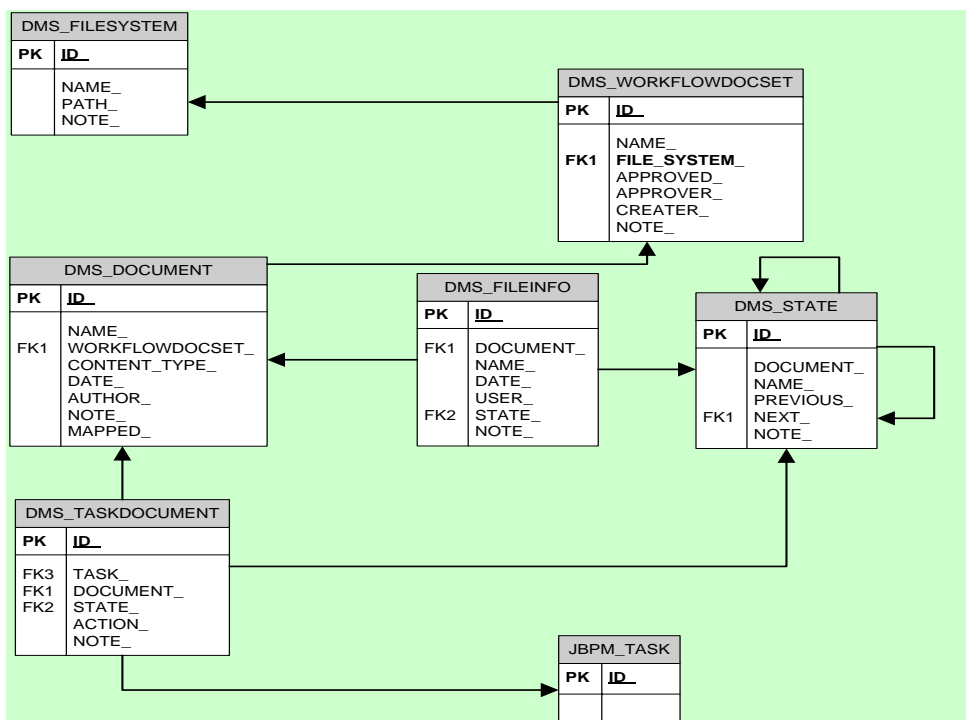
STT	Tên lớp/Quan hệ	Loại	Ý nghĩa/ Ghi chú
1	FileSystem	Entity	Lớp này dùng để thể hiện các hệ thống file.
2	WorkflowDocSet	Entity	Lớp này thể hiện một tập

			các tài liệu thuộc về một qui trình nghiệp vụ.
3	Document	Entity	Lớp này là thể hiện của tài liệu trong DMS.
4	State	Entity	Lớp này thể hiện trạng thái của tài liệu.
5	FileInfo	Lớp quan hệ	Lớp thể hiện cho một phiên bản cụ thể của một tài liệu. Một phiên bản của một tài liệu phải ở một trạng thái nào đó.
6	TaskDocument	Lớp quan hệ	Lớp này là thể hiện mối quan hệ giữa một task và một tài liệu.

Bảng 3-7 Danh sách các lớp/quan hệ cho phân hệ DMS

- Thiết kế:

- Cơ sở dữ liệu:



Hình 3-19 Sơ đồ quan hệ CSDL phân hệ quản trị tài liệu

○ Bảng DMS_FILESYSTEM

STT	Tên Trường	Kiểu dữ liệu	Dung lượng	Khóa chính	Khóa ngoại	Cho phép null	Ghi chú
1	ID	Bigint	8	Yes		No	Mã định danh của quan hệ, mã này tự động tăng
2	NAME_	Varchar	200	No		No	
3	PATH_	Varchar	255	No		No	Đường dẫn vật lý trên máy chủ
4	NOTE_	Text	255	No		Yes	

Bảng 3-8 Bảng DMS_FILESYSTEM

○ Bảng DMS_WORKFLOWDOCSET

STT	Tên Trường	Kiểu dữ liệu	Dung	Khóa	Khóa	Cho	Ghi chú
-----	------------	--------------	------	------	------	-----	---------

		liệu	lượng	chính	ngoại	phép null	
1	ID	Bigint	8	Yes		No	Mã định danh của quan hệ
2	NAME_	Varchar	200	No		No	
3	FILES_YSTEM_	bigint	8	No		Yes	Khóa ngoại tham chiếu đến bảng DMS_FILES YSTEM
4	NOTE_	Text	255	No		Yes	

Bảng 3-9 Bảng DMS_WORKFLOWDOCSET

○ Bảng DMS_DOCUMENT

STT	Tên Trường	Kiểu dữ liệu	Dung lượng	Khó a chính h	Khó a ngo ại	Ch o phé p nul l	Ghi chú
1	ID	Bigint	8	Yes		No	Mã định danh của quan hệ, tự động tăng.
2	NAME_	Varcha r	200	No		No	
3	WORKFLOWDOCSET —	bigint	8	No	Yes	Yes	Khóa ngoại tham chiếu đến bảng DMS_WORKFLOWDOCSET

4	CONTENT_TYPE	Varchar	100			Yes	Cho biết kiểu của tài liệu.
5	MAPPED_	Boolean	1				True: tài liệu đã được map Ngược lại: chưa được map
6	DATE_	DateTime	4				Ngày giờ upload tài liệu
7	AUTHOR_	Varchar	100				Tên người upload
8	NOTE_	Text	255	No		Yes	

Bảng 3-10 Bảng DMS_DOCUMENT

○ Bảng DMS_STATE

STT	Tên Trường	Kiểu dữ liệu	Dung lượng	Khóa chính	Khóa ngoại	Cho phép null	Ghi chú
1	ID	Bigint	8	Yes		No	Mã định danh của quan hệ
2	NAME_	Varchar	200	No		No	
3	DOCUMENT_	bigint	8	No	Yes	No	Khóa ngoại tham chiếu đến bảng DMS_DOCUMENT
4	PREVIOUS_	Bigint	8		Yes	Yes	Khóa ngoại tham chiếu đến chính bảng DMS_STATE để cho biết trạng thái trước đó là gì

5	NEXT_	Bigint	8		Yes	Yes	Khóa ngoại tham chiếu đến chính bảng DMS_STATE để cho biết trạng thái sau đó là gì
6	NOTE_	Text	255	No		Yes	

Bảng 3-11 Bảng DMS_STATE

○ Bảng DMS_FILEINFO

STT	Tên Trường	Kiểu dữ liệu	Dung lượng	Khóa chính	Khóa ngoại	Cho phép null	Ghi chú
1	ID	Bigint	8	Yes		No	Mã định danh của quan hệ
2	NAME_	Varchar	200	No		No	
3	DOCUMENT –	bigint	8	No	Yes	No	Khóa ngoại tham chiếu đến bảng DMS_DOCUMENT
4	STATE_	Bigint	8		Yes	No	Khóa ngoại tham chiếu đến chính bảng DMS_STATE để cho biết trạng thái trước đó là gì
5	DATE_	DATE TIME	8			Yes	Thời gian upload
	USER_	Varchar	100			No	Người upload
6	NOTE_	Text	255	No		Yes	

Bảng 3-12 Bảng DMS_FILEINFO

○ Bảng DMS_TASKDOCUMENT

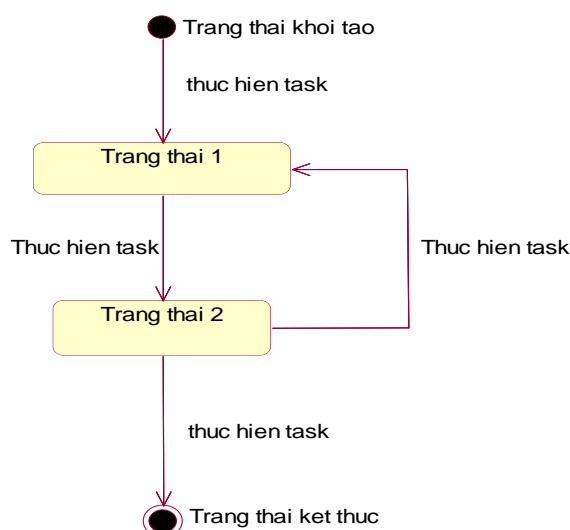
STT	Tên Trường	Kiểu dữ liệu	Dung lượng	Khóa	Cho phép null	Ghi chú
1	ID	Bigint	8	Khóa chính	No	Mã định danh của quan hệ
2	DOCUMENT –	bigint	8	Khóa ngoại	No	Khóa ngoại tham chiếu đến DMS_DOCUMENTS
3	TASK_	bigint	8	Khóa ngoại	No	Khóa ngoại tham chiếu đến bảngJBPM_TASK
4	STATE_	Bigint	8	Khóa ngoại	No	Khóa ngoại tham chiếu đến DMS_STATE
5	ACTION_	Varchar	50			Mô tả hành động được cho phép đối với một tài liệu, upload hoặc download hoặc cả hai
6	NOTE_	Text	255		Yes	

Bảng 3-13 Bảng DMS_TASKDOCUMENT

3.3.2.4 Sự chuyển đổi trạng thái của tài liệu trong quy trình

- Core jBPM không có phân hệ quản trị tài liệu, vì vậy khi phát triển thêm phân hệ này cho jBPM, vấn đề quản lý version của tài liệu phải được đồng bộ hóa với các task trong quy trình nghiệp vụ.

- Mỗi quy trình nghiệp vụ có một tập các tài liệu, mỗi tài liệu có một tập các trạng thái riêng biệt, những task nào có làm việc với tài liệu (đọc, ghi, ký duyệt) thì tại những task đó sẽ mô tả trạng thái làm việc của tài liệu. Khi task được thực hiện xong thì tài liệu sẽ được chuyển đến trạng thái kế tiếp.



Hình 3-20 Sơ đồ trạng thái của tài liệu

3.3.2.5 DMS Administration

- Phân hệ quản trị của DMS được thực hiện thực hiện bởi người dùng có quyền quản trị, bao gồm các chức năng sau đây:
- Tạo một file system entry trên máy chủ, thực chất là tạo một thư mục vật lý trên máy chủ và lưu thông tin về thư mục này trong database phục vụ cho việc lưu các document của process.
- Cập nhật thông tin của work flow document set.
- Map document luận lý với một file tài liệu vật lý.

3.3.2.6 Tổng kết về DMS

- Một hệ thống workflow không có hệ quản trị tài liệu thì hầu như không có ý nghĩa sử dụng thực tế, đây cũng chính là hạn chế của jBPM cho đến phiên bản 3.2.3. Vì vậy, xây dựng một DMS gắn vào jBPM là nhu cầu cần thiết.
- DMS được xây dựng trên cùng nền tảng công nghệ như jBPM sử dụng Hibernate, kết nối CSDL MySQL để tận dụng cấu hình sẵn có của jBPM.
- Việc quản lý các document trong DMS được thực hiện theo mô hình chuyển đổi trạng thái(state chart) gắn kết với việc thực hiện task của process trong jBPM.

3.3.3 Vấn đề security và tích hợp với LDAP

3.3.3.1 Giới thiệu về JAAS

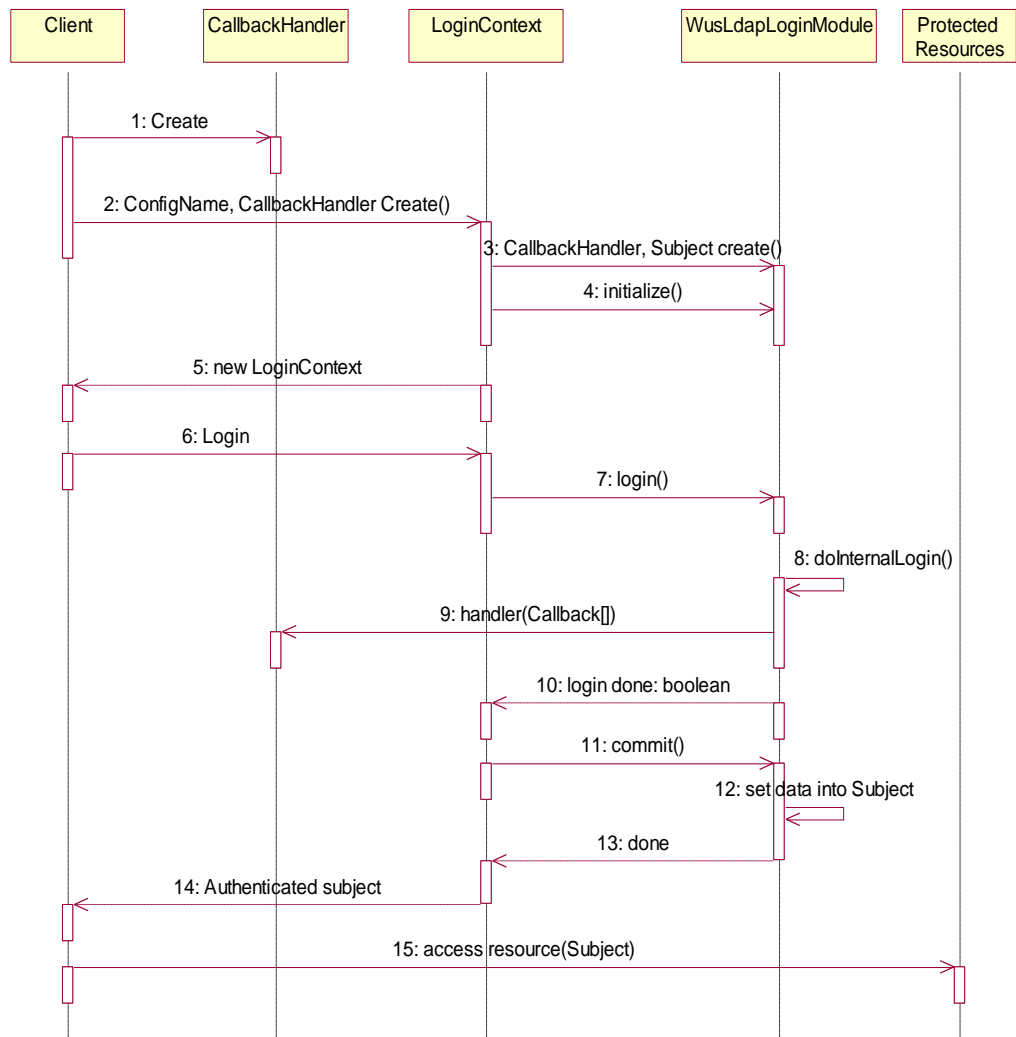
- Vấn đề bảo mật của jBPM được thực hiện nhờ vào công nghệ bảo mật JAAS (Java Authentication & Authorization Service). Đây là dịch vụ chứng thực và cấp quyền truy cập được Sun Microsystem. Inc đưa vào J2SE kể từ phiên bản 1.4.
- Đối với một ứng dụng J2EE, việc thực thi chức thực và cấp quyền dựa trên JAAS được thực hiện qua các bước sau:
 - ❖ Xác định các tài nguyên cần được bảo mật
 - ❖ Xác định một dịch vụ cung cấp cơ chế bảo mật, trong trường hợp của JBOSS, dịch vụ này được cấu hình trong file <server>/default/conf/login-config.xml
 - ❖ Sử dụng dịch vụ cung cấp cơ chế bảo mật để thực thi công việc chứng thực và cấp quyền.

❖ Mỗi khi người dùng có yêu cầu truy cập vào các tài nguyên được bảo mật thì hệ thống phải thông báo cho người dùng cung cấp định danh của mình. Trong đa số trường hợp việc cung cấp định danh này là việc nhập vào username/password hoặc chữ ký điện tử (digital certificates) của người dùng.

- Dịch vụ JAAS bao gồm các lớp chính sau:

- LoginModule: đây là giao diện chứa các phương thức phục vụ cho việc chứng thực và phân quyền. Một ứng dụng muốn sử dụng dịch vụ JAAS phải thực thi giao diện này và định nghĩa lại các phương thức của giao diện. Các phương thức trong giao diện này bao gồm:
 - Initialize(): Phương thức này được đối tượng LoginContext gọi để khởi tạo đối tượng LoginModule.
 - login(): Phương thức này được đối tượng LoginContext gọi để thực hiện việc đăng nhập. Nếu thông tin chứng thực đúng thì hàm trả về true, trường hợp ngược lại trả về false.
 - commit(): Phương thức này được đối tượng LoginContext gọi, thông thường ta cấp quyền trong hàm này.
 - abort(): Phương thức này được gọi khi thông tin đăng nhập không thành công.
 - logout(): Phương thức này được gọi khi người dùng yêu cầu đăng xuất khỏi hệ thống. Thông thường, các công việc giải phóng tài nguyên dành cho một người dùng đã đăng nhập trước đó được thực hiện ở đây.
- LoginContext: Đây là lớp giao tiếp giữa phía khách và phía server. Phía khách sử dụng lớp này để gọi thực hiện việc chứng thực.

- **CallbackHandler and Callback:** Các lớp này cho phép tương tác dữ liệu giữa phía khách và đối tượng LoginModule. LoginModule nhận thông tin đăng từ client thông qua Callback.
 - **Principal and Group:** Đối tượng LoginModule sẽ truy vấn thông tin của người dùng (như tên, địa chỉ, nhóm) từ CSDL hoặc một hệ thống LDAP và lưu các thông tin này vào một đối tượng Principal, danh sách các vai trò của người dùng được lưu trong đối tượng Group. Các đối tượng Principal và Group này sau đó sẽ được lưu giữ trong đối tượng Subject.
 - **Subject:** Đối tượng này là đại diện cho một chứng thực thành công, nó lưu dữ liệu toàn bộ thông tin chức thực và phân quyền của người dùng đang đăng nhập. Và đối tượng này chỉ được hiển thị ở phía khách sau khi việc chứng thực thành công.
- Sơ đồ luồng cho quá trình đăng nhập khi sử dụng JAAS như sau:



Hình 3-21 Sơ đồ luồng xử lý đăng nhập theo JAAS

- Đầu tiên, CallbackHandler sẽ được tạo ra, CallbackHandler và cấu hình đăng nhập sẽ được dùng để tạo LoginContext. Sau đó, LoginContext sẽ tạo và khởi tạo LoginModule.
- Sau đó login() của LoginContext được gọi và hàm này tiếp tục gọi login() của LoginModule. Hàm login() của LoginModule sẽ thực hiện kiểm tra thông tin đăng nhập(thông thường là username/password), thông tin này được đối tượng CallbackHandler cung cấp.

- Nếu thông tin đăng nhập là không hợp lệ thì LoginException sẽ được quăng ra, trong trường hợp ngược lại, quá trình cấp quyền cho người dùng sẽ được bắt đầu.
- Hàm commit() của LoginModule sẽ được gọi bởi LoginContext. Trong hàm này, việc thực hiện cấp quyền cho user sẽ được thực hiện thông qua đối tượng Subject.
- LoginContext trả thông tin chứng thực và cấp quyền này cho phía khách.
- Phía máy khách có thể bắt đầu truy cập vào các tài nguyên được bảo vệ.

3.3.3.2 Vấn đề bảo mật của jBPM

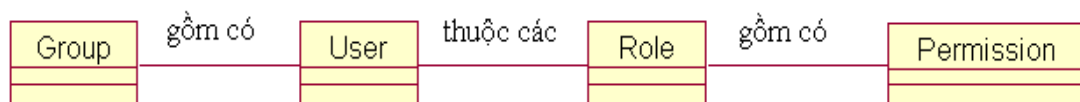
- JAAS cung cấp cơ chế chứng thực và cấp quyền cho các ứng dụng Java. Tuy nhiên, việc chứng thực và cấp quyền cụ thể như thế nào là do nhà phát triển ứng dụng thực hiện.
- Đối với jBPM, việc chứng thực và cấp quyền truy cập hệ thống có một số vấn đề như sau:
 - ❖ Thông tin người dùng được lưu trữ trong cơ sở dữ liệu của ứng dụng, điều này dẫn đến tính bảo mật của hệ thống không cao, trong trường hợp tại đơn vị được triển khai đã có sẵn một hệ thống quản trị thông tin người dùng chuẩn LDAP, ví dụ như Active Directory hoặc OpenLdap thì sẽ gây bất lợi cho người dùng khi phải quản lý thêm một account đăng nhập.
 - ❖ Mật khẩu không được mã hóa dẫn đến mức độ bảo mật không cao.
 - ❖ Quyền truy cập vào các chức năng (hoặc tài nguyên) được thiết đặt cứng trong file thuộc tính (access.properties), không có tính mềm dẻo. Khi có nhu cầu cần thay đổi quyền truy cập cho các role trong hệ thống người dùng quản trị không thể thực hiện trên giao

diện của ứng dụng mà phải thay đổi thủ công trong file cấu hình.

- Những hạn chế này sẽ dẫn đến khó khăn trong triển khai thực tế jBPM tại đơn vị người dùng. Từ đó, việc xây dựng lại phân hệ quản trị người dùng, chứng thực và cấp quyền là một điều cần thiết.

3.3.3.3 Xây dựng lại phân hệ chứng thực và cấp quyền user cho jBPM

- Phân hệ chứng thực và phân quyền được xây dựng lại sẽ lấy thông tin người dùng từ một hệ thống LDAP, cụ thể là OpenLDAP, mật khẩu được mã hóa theo thuật toán SHA-1.
- Sơ đồ lớp mức phân tích



Hình 3-22 Sơ đồ lớp phân hệ chứng thực và cấp quyền người dùng

ST T	Tên lớp/Quan hệ	Loại	Ý nghĩa/ Ghi chú
1	User	Entity	Lớp này kế thừa giao diện Principal, dùng để thể hiện người dùng.
2	Role	Entity	Lớp này kế thừa giao diện Principal, dùng để thể hiện vai trò của người dùng.
3	Group	Entity	Lớp này dùng để thể hiện nhóm người dùng.

4	Permission	Entity	Lớp này dùng để thể hiện quyền truy cập vào các chức năng của ứng dụng.
---	------------	--------	---

Bảng 3-14 Danh sách lớp/ quan hệ phân hệ chứng thực và phân quyền

3.3.3.4 Tổng kết về vấn đề bảo mật và tích hợp LDAP

- Phần chứng thực và phân quyền của jBPM được tách riêng thành một phân hệ riêng, phần này dựa trên dịch vụ bảo mật JAAS của Sun Microsystems Inc. Do yêu cầu về tính mềm dẻo cũng như tăng khả năng bảo mật nên cần phải xây dựng lại phân hệ này.
- Đã xây dựng thành công phân hệ chức thực và phân quyền dựa trên JAAS, thông tin của người dùng được lấy từ OpenLDAP.

4 - Thử nghiệm và cài đặt:

Chương này trình bày phần đặc tả qui trình tổng quát, bao gồm phần đặc tả và phân tích qui trình. Trong đó phát biểu các qui trình ở dạng tổng quát, thông tin cần thiết dùng cho mỗi bước. Sau đó là phần ví dụ với 5 qui trình mẫu được xây dựng để minh họa.

4.1 *Qui trình tổng quát:*

4.1.1 **Đặc tả:**

- Mỗi qui trình sẽ bao gồm nhiều bước thực hiện.
- Bắt đầu ở mỗi qui trình thực thi là phần chỉ định người tổng quát ở mức thiết kế qui trình với người cụ thể khi thực thi qui trình.
 - Có thể tái sử dụng lại thông tin các người dùng đã được chỉ định trước đó.
- Với mỗi bước sẽ do một người đảm nhiệm việc thực thi.
 - Người đảm nhiệm này được xây dựng mức tổng quát khi thiết kế, sau đó sẽ được chỉ định cụ thể là người nào khi bắt đầu một thể hiện của qui trình.
- Mỗi bước có thể không hoặc có đính kèm tài liệu.
 - Mỗi tài liệu sẽ được chỉ định các trạng thái mà tài liệu này sẽ chuyển đổi (ví dụ như trạng thái khởi tạo, trạng thái cập nhật, trạng thái đã ký duyệt, trạng thái đã công bố, ...)
 - Mỗi tài liệu cũng được chỉ định hành vi của người sử dụng đối với tài liệu tại bước cụ thể đó (ví dụ như download hoặc upload).
 - Tài liệu được thiết kế mức tổng quát có thể được chỉ định cụ thể khi thực hiện hành động upload tài liệu.

- Người dùng cũng có thể chỉ định nơi lưu trữ tài liệu ở mức vật lý.

4.1.2 Phân tích qui trình:

- Mỗi qui trình được triển khai có thể được chạy với nhiều thể hiện khác nhau.
- Qui trình có thể được thiết kế với nhiều cấu trúc khác nhau (song song, tuần tự, rẽ nhánh, lựa chọn một trong nhiều nhánh, ...)
- Có thể tại được nhiều thể hiện của một bước nào đó trong qui trình (ví dụ khi phân công nhiệm vụ cho một nhóm). Người quản trị có thể theo dõi thông tin chi tiết của việc phân công tác vụ cho một nhóm nhiều người tham gia vào qui trình.

4.2 Qui trình thông báo tin tức:

4.2.1 Thiết kế qui trình:

- Ban giám hiệu ra thông báo toàn trường.
- Trường sẽ chuyển thông báo xuống khoa.
- Khoa sẽ thông báo đến tất cả cán bộ công nhân viên trong trường.
- Cán bộ công nhân viên gửi phản hồi ngược lại cho khoa.
- Kết thúc qui trình.

4.2.2 Đặc tả qui trình:

4.2.2.1 Swimlanes:

- Giáo vụ.
- Trường/Phó bộ môn.
- Giảng viên.

4.2.2.2 Documents:

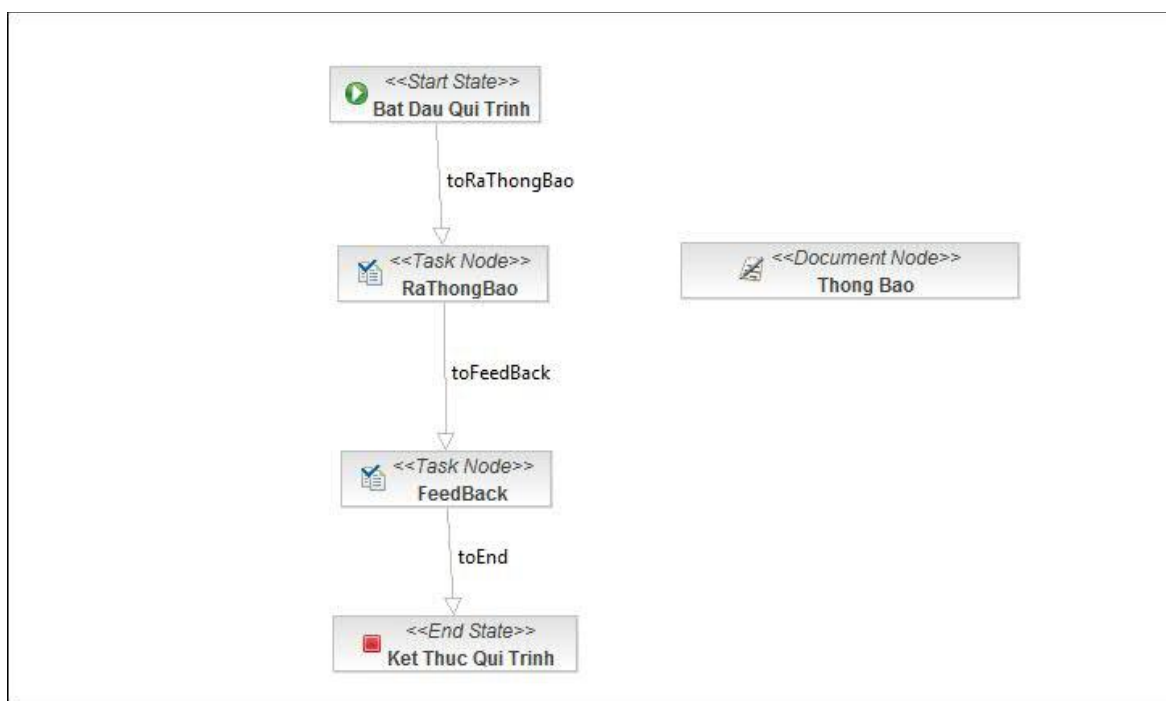
- Thông báo – **ThôngBao**

Tên State	Actor	Người nhận	Trạng thái
-----------	-------	------------	------------

Khởi tạo thông báo	Khoa	Giảng viên	Upload
--------------------	------	------------	--------

Bảng 4-1 Bảng trạng thái document ThôngBao

4.2.2.3 Màn hình triển khai:



Hình 4-1 Quy trình thông báo tin tức

4.2.2.4 Các hàm xử lý:

- Hàm khởi tạo nhiều instance và assign cho group.
- Hàm thực hiện thu nhận phản hồi từ người nhận được thông báo.

4.2.3 Triển khai quy trình: demo

4.3 Quy trình tính lương:

4.3.1 Thiết kế quy trình:

- Giáo vụ thông kê giờ dạy.
- Giáo vụ tính tiền dạy.
- Kế toán kiểm tra dự trù thanh toán.

- Giảng viên kiểm tra dự trù thanh toán và phản hồi.
- Trưởng khoa duyệt.
- Kế toán thực hiện chuyển tiền và soạn thông báo giảng viên.
- Giáo vụ gửi thông báo cho toàn thể giảng viên.

4.3.2 Đặc tả qui trình:

4.3.2.1 Swimlanes:

- Giáo vụ.
- Trưởng/Phó Khoa.
- Giảng viên.
- Kế toán

4.3.2.2 Documents:

- Bảng thống kê giờ dạy – **BangThongKeGioDay**

Tên trạng thái	Người gửi	Người nhận	Trạng thái
Khởi tạo	Giáo vụ	Kế toán	Upload

Bảng 4-2 Bảng thống kê giờ dạy

- Bảng dự thanh toán lương – **BangThanhToanLuong**

STT	Tên trạng thái	Người gửi	Người nhận	Trạng thái
1.	Khởi tạo	Kế toán	Giảng viên	Upload
2.	Phản hồi	Giảng viên	Kế toán	Upload
3.	Chỉnh sửa	Kế toán	Giảng viên	Upload
4.	Tạm duyệt	Kế toán	Trưởng khoa	Upload
5.	Sửa tạm duyệt	Trưởng khoa	Kế toán	Upload
6.	Chính thức	Trưởng khoa	Kế toán	Upload

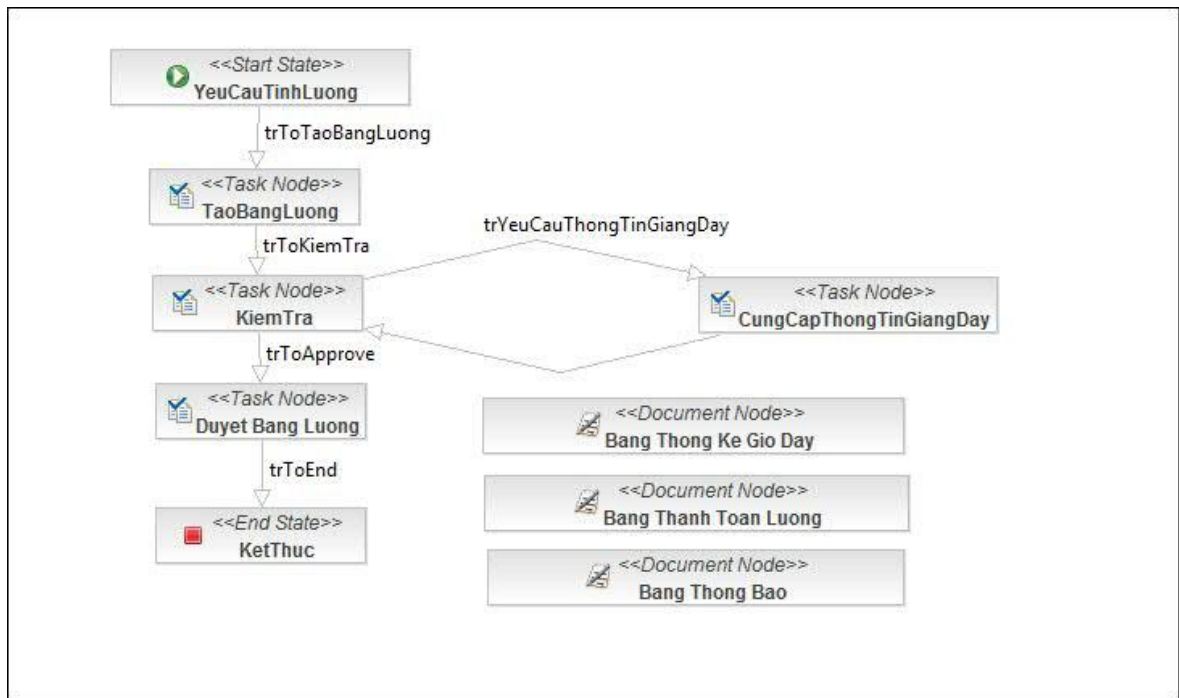
Bảng 4-3 Bảng thanh toán lương

- Bảng thông báo – **BangThongBao**

Tên trạng thái	Người gửi	Người nhận	Trạng thái
Khởi tạo	Giáo vụ	Giảng viên	Upload

Bảng 4-4 Bảng thông báo

4.3.2.3 Màn hình triển khai:



Hình 4-2 Quy trình tính lương

4.3.2.4 Các hàm xử lý:

- Chỉ sử dụng hàm thông báo đã thực hiện xong task để chuyển qua task kế tiếp.

4.4 Quy trình phân công giảng dạy:

4.4.1 Thiết kế quy trình:

- Giáo vụ gửi danh sách môn học.
- Trưởng/Phó bộ môn duyệt danh sách và gửi cho giảng viên đăng ký.

- Giảng viên đăng ký giảng dạy và đăng ký thiết bị.
- Trưởng/Phó bộ môn duyệt danh sách đăng ký giảng dạy và đăng ký thiết bị của giảng viên.
- Kết thúc qui trình.

4.4.2 Đặc tả qui trình:

4.4.2.1 Swimlanes:

- Giáo vụ.
- Trưởng/Phó bộ môn.
- Giảng viên.

4.4.2.2 Documents:

- Danh sách môn học – **DanhSachMonHoc**

STT	Tên State	Actor	Người nhận	Trạng thái
1.	Khởi tạo	Giáo vụ	Trưởng/Phó BM	Upload
2.	Duyệt danh sách	Trưởng/Phó BM		Download
3.	Gui-giang-vien-dang-ky-giang-day	Trưởng/Phó BM	Giảng Viên	Upload

Bảng 4-5 Bảng danh sách môn học

- Danh sách đăng ký giảng dạy – **DanhSachDangKyGiangDay**

STT	Tên State	Actor	Người nhận	Trạng thái
1.	Gui-danh-sach-dang-ky-giang-day	Giảng Viên	Giáo Vụ	Upload

2.	Danh-sach-dang-ky-giang-day-tong-hop	Giáo Vụ	Trưởng/Phó BM	Upload
3.	Duyet-dang-ky-giang-day	Trưởng/Phó BM		Upload

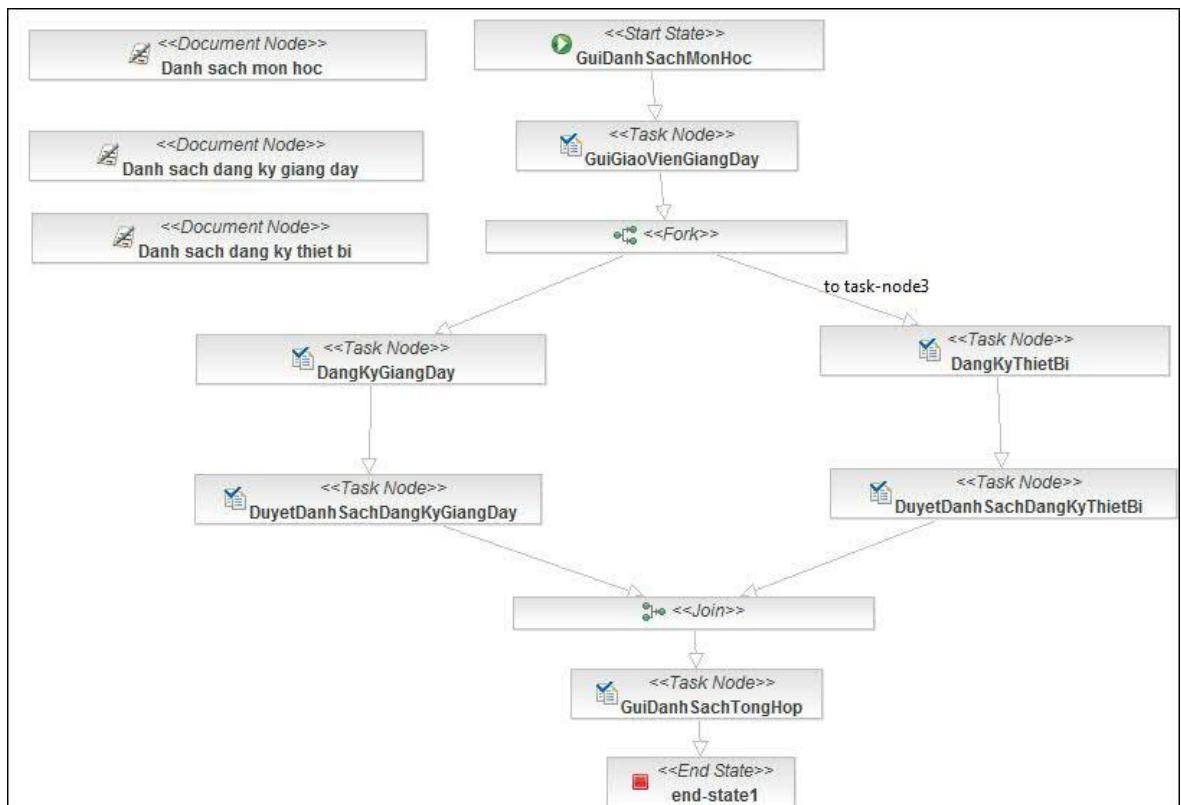
Bảng 4-6 Danh sách đăng ký giảng dạy

- Danh sách đăng ký thiết bị - **DanhSachDangKyThietBi**

STT	Tên State	Actor	Người nhận	Trạng thái
1.	Gui-danh-sach-dang-ky-thiet-bi	Giảng Viên	Giáo Vụ	Upload
2.	Danh-sach-dang-ky-thiet-bi-tong-hop	Giáo Vụ	Trưởng/phó BM	Upload
3.	Duyet-dang-ky-thiet-bi	Trưởng/Phó BM		Upload

Bảng 4-7 Danh sách đăng ký thiết bị

4.4.2.3 Màn hình triển khai:



Hình 4-3 Quy trình đăng ký giảng dạy

4.4.2.4 Các hàm xử lý:

- Chỉ sử dụng hàm thông báo đã thực hiện xong task để chuyển qua task kế tiếp.

4.4.3 Triển khai quy trình:demo

4.5 Quy trình xét học bổng:

4.5.1 Thiết kế quy trình:

- Trợ lý sinh viên gửi đề nghị cấp học bổng đến các công ty.
- Trợ lý sinh viên nhận phản hồi từ công ty và gửi thông báo cho toàn thể sinh viên.
- Sinh viên đăng ký học bổng.
- Trợ lý sinh viên duyệt đăng ký học bổng.

- Trợ lý sinh viên gửi thông báo cho ban chủ nhiệm và sinh viên.
- Kết thúc qui trình.

4.5.2 Đặc tả qui trình:

4.5.2.1 Swimlanes:

- Trợ lý sinh viên.
- Công ty.
- Sinh viên.

4.5.2.2 Documents:

- Thông tin yêu cầu học bổng – **YeuCauHocBong**

STT	Tên State	Actor	Người nhận	Trạng thái
1.	Gửi yêu cầu học bổng	Trợ lý sinh viên	Công ty	Upload
2.	Phản hồi thông tin học bổng	Công ty	Trợ lý sinh viên	Download

Bảng 4-8 Bảng yêu cầu học bổng

- Danh sách và thông tin học bổng – **ThongTinHocBong**

Tên State	Actor	Người nhận	Trạng thái
Thông báo danh sách học bổng	Trợ lý sinh viên	Sinh viên	Upload

Bảng 4-9 Bảng thông tin học bổng

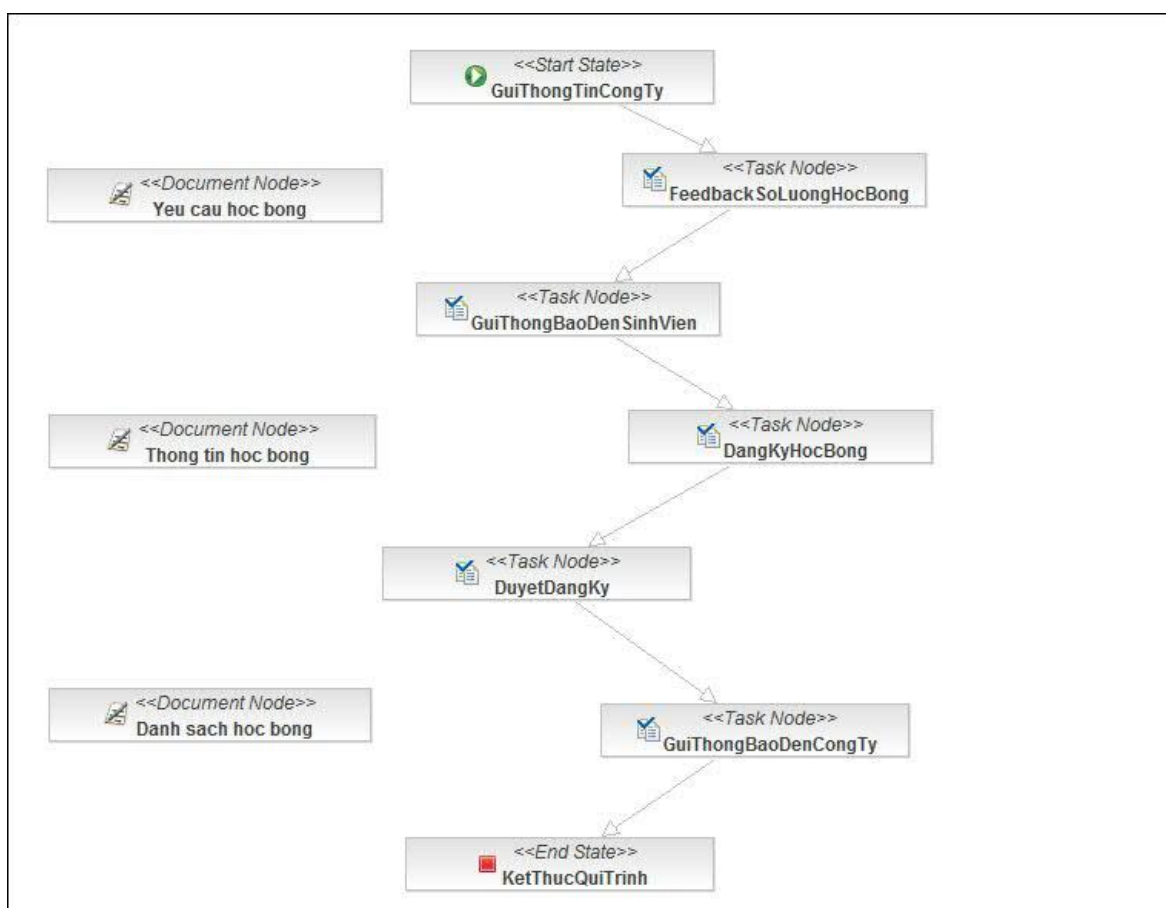
- Danh sách đăng ký học bổng - **DanhSachDangKyHocBong**

STT	Tên State	Actor	Người nhận	Trạng thái

1.	Danh sách đăng ký học bổng	Trợ lý sinh viên	Sinh viên	Upload
2.	Danh sách đã được đăng ký	Sinh viên	Trợ lý sinh viên	Download
3.	Danh sách tổng hợp	Trợ lý sinh viên	Sinh viên	Upload

Bảng 4-10 Bảng danh sách đăng ký học bổng

4.5.2.3 Màn hình triển khai:



Hình 4-4 Quy trình đăng ký học bổng

4.5.2.4 Các hàm xử lý:

- Chỉ sử dụng hàm thông báo đã thực hiện xong task để chuyển qua task kế tiếp.

4.5.3 Triển khai qui trình: demo

4.6 Qui trình đăng ký đi dã ngoại:

4.6.1 Thiết kế qui trình:

- Công đoàn khoa gửi thông báo cho toàn thể cán bộ, công nhân viên.
- Cán bộ, giảng viên đăng ký đi dã ngoại.
- Công đoàn khoa tổng hợp danh sách.
- Kết thúc qui trình.

4.6.2 Đặc tả qui trình:

4.6.2.1 Swimlanes:

- Công đoàn khoa.
- Giảng viên.
- Cán bộ công nhân viên của trường.

4.6.2.2 Documents:

- Thông báo thông tin dã ngoại – **ThôngBaoDaNgoai**:

Tên State	Actor	Người nhận	Trạng thái
Thông báo đăng ký dã ngoại	Công đoàn	Cán bộ, giảng viên	Upload

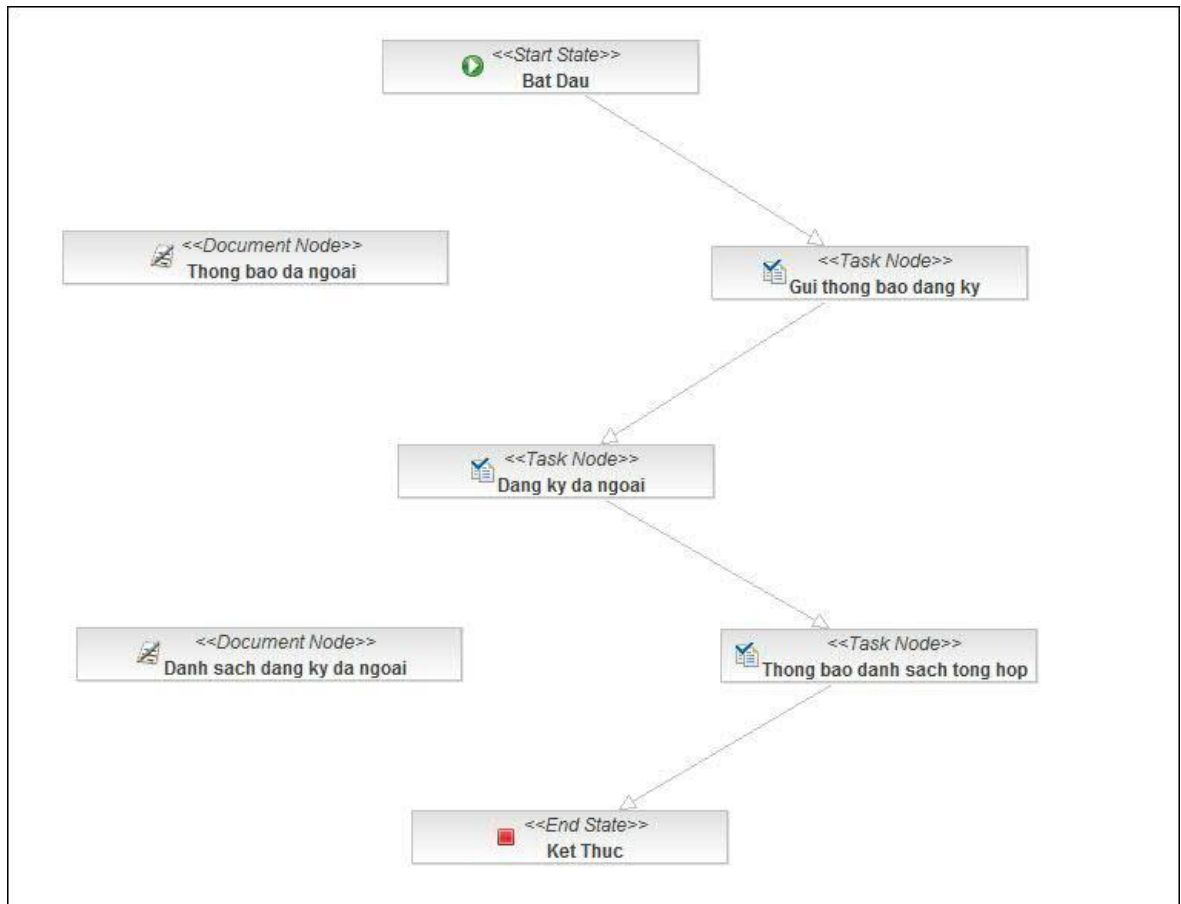
Bảng 4-11 Bảng thông báo dã ngoại

- Danh sách đăng ký dã ngoại – **DanhSachDangKyDaNgoai**:

STT	Tên State	Actor	Người nhận	Trạng thái
1.	Đăng ký thông tin dã ngoại	Cán bộ, giảng viên	Công đoàn	Upload
2.	Danh sách đăng ký tổng hợp	Công đoàn	Cán bộ, giảng viên	Upload

Bảng 4-12 Bảng trạng thái danh sách đăng ký dã ngoại

4.6.2.3 Màn hình triển khai:



Hình 4-5 Quy trình đăng ký dã ngoại

4.6.2.4 Các hàm xử lý:

- Chỉ sử dụng hàm thông báo đã thực hiện xong task để chuyển qua task kế tiếp.

4.6.3 Triển khai quy trình: demo

4.7 Phân tích và đánh giá kết quả thực hiện:

4.7.1 Kết quả đã đạt được:

4.7.1.1 Thực thi với hệ thống ban đầu:

- Document: Với hệ thống ban đầu, chỉ sử dụng được phần đặc tả các qui trình nghiệp vụ mà trong đó document không có sự tham gia vào qui trình thực thi.
- Swimlane: hệ thống jBPM khi chưa cải tiến sẽ ánh xạ trực tiếp swimlane tương ứng với 1 hoặc nhiều actor (trong đó actor chính là user thực sự phía CSDL).
- Multi-instance: hệ thống ban đầu không hỗ trợ multi-instance ở dạng cá nhân, nghĩa là khi thực hiện assign một task cho một group nào đó thực hiện, khi này bất kỳ thành viên nào trong group hoàn thành task thì xem như toàn bộ group đó đã hoàn thành task. Bên cạnh đó, khi assign task cho một nhóm actor thì hệ thống hỗ trợ assign cho cá nhân cụ thể ngay trong giao diện jBPM-console.
- Phản hồi của actor: hệ thống jBPM khi chưa được chỉnh sửa chỉ có thể nhận phản hồi từ người dùng thông qua biến được khởi tạo sẵn hoặc bảng thông tin trạng thái của task mà thôi.

4.7.1.2 Thực thi với hệ thống đã cải tiến:

- Với hệ thống đã được cải tiến, những chức năng đã nêu ở trên sẽ được hỗ trợ một cách đầy đủ.
- Document: ở phiên bản đã được cải tiến, hệ thống document sẽ được gắn cụ thể vào một task. Việc hỗ trợ xây dựng biểu diễn document trong quá trình thiết kế qui trình nghiệp vụ cũng góp phần làm cho hệ thống có khả năng ứng dụng vào thực tế nhiều hơn.
 - ❖ Việc đặc tả hệ thống tập trạng thái của document sẽ xác định được trạng thái cụ thể của từng document ứng với mỗi một tác vụ cụ thể.

- ❖ Bên cạnh đó, đặc tả hành động tương ứng trên document cũng sẽ giúp cho người thực hiện tác vụ biết được thao tác cần phải thực hiện trên document đó (cụ thể là cần upload hay download document).
- Sử dụng định danh trừu tượng cho document: Việc cho phép người thiết kế quy trình thực hiện việc đặt tên document ở dạng tổng quát cũng góp phần làm tăng tính tái sử dụng của quy trình nhiều hơn. Trong trường hợp sử dụng lại cùng một quy trình nhưng với hệ thống tài liệu khác nhau, khi thực thi quy trình, quản trị có thể chỉ định cụ thể tài liệu thực tế tương ứng để hệ thống sử dụng.
- Swimlane: tương tự như trên, tính năng ánh xạ swimlane được định nghĩa tổng quát với swimlane cụ thể có sẵn trong hệ thống LDAP cũng góp phần làm tăng tính linh động và khả năng tái sử dụng của quy trình được thiết kế.
 - ❖ Trong đó người dùng có thể thực hiện ánh xạ 1 : 1, nghĩa là một swimlane trừu tượng sẽ tương ứng với một actor cụ thể trong LDAP. Sử dụng chủ yếu cho chỉ định người quản trị.
 - ❖ Ngoài ra, người dùng có thể thực hiện ánh xạ 1 : n, trong đó n sẽ là một nhóm những người dùng được phân chia sẵn trong hệ thống LDAP. Khi này, swimlane sẽ là một nhóm những người dùng.
- Vấn đề multi-instance: quá trình khởi tạo ra multi-instance sẽ được thiết kế trong giai đoạn đặc tả quy trình. Sau đó, file xử lý dạng .java sẽ được gắn kèm theo lúc thực hiện triển khai hệ thống. Lúc này, việc assign task cho nhiều actor sẽ được thực hiện trong code java kèm theo.

5 - Tổng kết

Chương này sẽ trình bày tóm tắt về kết quả đã đạt được sau khi nghiên cứu và xây dựng ứng dụng. Bên cạnh đó, chương này cũng trình bày những nhận xét trong suốt quá trình thực hiện nghiên cứu, cũng như những khó khăn mà nhóm đã gặp.

5.1 Nhận xét:

5.1.1 Vấn đề khảo sát và lựa chọn hệ thống workflow:

- Trong quá trình bắt đầu nghiên cứu, việc tìm hiểu và lựa chọn hệ thống nguồn mở là một công việc khó khăn. Do trong cộng đồng mã nguồn mở có rất nhiều hệ thống workflow khác nhau.
- Nhóm đã thực hiện khảo sát 19 hệ thống để đánh giá mức độ hỗ trợ cũng như khả năng đáp ứng yêu cầu của bài toán đặt ra. Sau khi khảo sát và đánh giá, ban đầu nhóm lựa chọn hệ thống WfMOpen, tuy hệ thống hỗ trợ tốt về mã nguồn, nhưng khả năng mở rộng và tích hợp các hệ thống khác như LDAP, graphical editor, MySQL, ... rất hạn chế, bên cạnh đó, sự hỗ trợ từ tài liệu đặc tả và cả cộng đồng sử dụng lại không mạnh mẽ và rộng khắp như hệ thống jBPM.
- Chính vì vậy mà sau khi thử nghiệm và đánh giá với hệ thống WfMOpen, nhóm đã chuyển sang nghiên cứu jBPM.

5.1.2 Vấn đề tích hợp nhiều hệ thống khác nhau (Editor, LDAP, MySQL, CA):

- Thuận lợi:
 - Do jBPM được phát triển trên ngôn ngữ Java thuần túy, nên khả năng hỗ trợ của jBPM sẽ tùy thuộc trực tiếp vào khả năng hỗ trợ của ngôn ngữ Java.
 - Chính nhờ khả năng hỗ trợ mạnh mẽ của ngôn ngữ Java nên hệ thống jBPM có khả năng tích hợp với những hệ thống

quản trị cơ sở dữ liệu khác như: MySQL, HyperSonic, LDAP, Active Directory.

- Khó khăn:
 - Hệ thống nội tại của jBPM chỉ hỗ trợ các chức năng đơn giản như quản lý quy trình không có kèm theo document. Chính vì nhu cầu quản lý document, ban đầu nhóm đã thực hiện khảo sát OpenKM, đây là phần mềm nguồn mở hỗ trợ việc quản lý CSDL truyền đi trong hệ thống. Tuy nhiên, phần mềm này chỉ mở version thấp nhất, dù mã nguồn được công bố, nhưng tài liệu hỗ trợ kèm theo không có và không có cộng đồng nguồn mở hỗ trợ, nên nhóm đã quyết định chuyển sang phân tích và xây dựng một hệ thống quản lý tài liệu riêng rẽ.
 - Mặc dù jBPM có cung cấp bộ thiết kế quy trình đồ họa GPD, tuy nhiên mã nguồn của GPD phiên bản mới nhất không được công bố và mã nguồn của phiên bản cũ hơn lại không được kiểm chứng, phiên bản đóng gói công bố trên trang sourceforge không gặp vấn đề lỗi, nhưng mã nguồn công bố lại mắc phải một số lỗi khi thực thi.

5.1.3 Vấn đề thiếu tài liệu đặc tả chi tiết về core engine cũng như các phần khác:

- Hệ thống jBPM là một hệ thống lớn và được sử dụng rộng khắp, tài liệu hỗ trợ của hệ thống này cũng được xây dựng và công bố kèm theo trên trang của jBoss.
- Tuy nhiên, đặc điểm chung của các thành phần trong hệ thống jBPM là tài liệu hỗ trợ không có hoặc nếu có thì cũng chỉ dừng lại ở mức giới thiệu sơ lược.

- Kèm theo đó là hệ thống liệt kê những hàm thư viện lập trình mà hệ thống jBPM cung cấp. Đó cũng là một mặt hạn chế khi tham gia sử dụng mã nguồn mở để xây dựng hệ thống quản lý workflow.
- Phần đặc tả chi tiết của hệ thống không được xây dựng một cách chi tiết, nên phần thử nghiệm chủ yếu dựa vào kinh nghiệm và tìm kiếm thông tin trên cộng đồng đang và đã sử dụng jBPM.

5.1.4 Vấn đề thiếu tài liệu đặc tả editor:

- Riêng phần editor, hoàn toàn không có tài liệu đặc tả kèm theo. Ban đầu khi lựa chọn hệ thống hỗ trợ xây dựng quy trình nghiệp vụ với giao diện đồ họa. Nhóm đã lựa chọn NexusBPM, đây là hệ thống hỗ trợ thiết kế quy trình nghiệp vụ đồ họa với tài liệu đặc tả và hỗ trợ đáng tin cậy. NexusBPM bao gồm cả phần mã nguồn dạng đầy đủ (phát triển dựa trên eclipse và hoàn toàn được tạo ra chỉ để thiết kế quy trình nghiệp vụ đồ họa), ngoài ra còn có dạng plugins vào hệ thống eclipse sẵn có.
- jBPM cung cấp mã nguồn của GPD, tuy nhiên mã nguồn này lại không có bất kỳ tài liệu hướng dẫn nào đi kèm. Chính vì vậy mà nhóm phải dựa trên tìm hiểu NexusBPM để có thể thiết kế thêm và chỉnh sửa mã nguồn của jBPM GPD.

5.1.5 Vấn đề tiêu chuẩn của việc thêm document node:

- Khả năng mở rộng là một ưu thế mà ngôn ngữ đặc tả quy trình jPDL chiếm được so với hệ thống các ngôn ngữ khác. Tuy nhiên, vấn đề đặt ra là khi xây dựng thêm document node, hệ thống file định nghĩa quy trình đã không còn ổn định nữa, nghĩa là khi này nếu không xây dựng phần triển khai phía server ở một hệ thống mới, thì hệ thống mới sẽ không thể hiểu và xử lý được document node được thêm vào.

- Với những ngôn ngữ được qui định nghiêm ngặt và xây dựng theo tiêu chuẩn của các tổ chức workflow thì khả năng chỉnh sửa và cải tiến sẽ vô cùng khó khăn, tuy nhiên, qui trình thiết kế với những ngôn ngữ này sẽ có thể thực thi trên mọi server hỗ trợ ngôn ngữ đó mà không cần phải quan tâm đến việc bổ sung phần xử lý cho server.

5.2 Hướng mở rộng:

5.2.1 Tích hợp thêm với hệ thống Chứng thực tài liệu và chứng thực người dùng:

- Hiện tại, hệ thống jBPM sử dụng công nghệ JAAS để chứng thực người dùng đăng nhập vào hệ thống. Tuy nhiên, jBPM hoàn toàn chưa hỗ trợ hệ thống chứng thực tài liệu và chứng thực lại người dùng trên tài liệu đó.
- Hướng mở rộng của luận văn sẽ là việc xây dựng và tích hợp hệ thống jBPM với hệ thống chứng thực tài liệu (chữ ký điện tử) và chứng thực quyền thao tác của người dùng trên tài liệu đó.

5.2.2 Xây dựng hệ thống hỗ trợ tạo qui trình dạng ad-hoc:

- Trong thực tế, có những qui trình sẽ không được định nghĩa trước, ví dụ như việc chuyển thông báo. Khi người A nhận được thông báo và chuyển cho người B, người B có thể tự do chuyển thông báo mà không cần chỉ định trước, qui trình này kết thúc ở một bước bất kỳ không xác định trước. Đây là qui trình dạng ad-hoc.
- Hiện nay, hệ thống jBPM hoàn toàn không hỗ trợ tính năng này, cho nên một hướng mở rộng khác của jBPM đó là xây dựng thêm tính năng hỗ trợ quản lý ad-hoc, dạng bất kỳ để phục vụ nhu cầu của người sử dụng hệ thống.

- Đây là một vấn đề thực tế có tính cần thiết cao, tuy nhiên cần phải có thời gian để nghiên cứu chi tiết phương án giải quyết bài toán này.

5.2.3 Nghiên cứu vấn đề triển khai qui trình lúc runtime:

- Trong thực tế, một qui trình có thể không được định nghĩa một cách toàn diện và đầy đủ, người dùng có thể tạo một bước X trong qui trình mà bước X này là kết quả thực hiện của một qui trình khác mà qui trình này không được chỉ định trước lúc thiết kế qui trình. Đó là dạng triển khai sub process lúc runtime.
- Hiện tại, hệ thống jBPM đã hỗ trợ triển khai qui trình với một bước bất kỳ có thể là kết quả thực hiện của qui trình khác. Tuy nhiên, qui trình này phải được thiết kế và triển khai đồng bộ lúc xây dựng qui trình nghiệp vụ.
- Về mặt nguyên lý hoạt động, hệ thống jBPM sẽ triển khai đồng thời qui trình chính và mọi qui trình con (nếu có). Khi thông tin kích hoạt gửi đến node có qui trình con, thì qui trình con sẽ được gọi thực hiện, khi này qui trình chính sẽ ở trạng thái chờ cho đến khi hoàn tất qui trình con. Do đó, để thực hiện qui trình bất kỳ, khi này cần can thiệp sâu vào hệ thống jBPM để xây dựng cho người dùng quyền chỉ định thực thi qui trình con sẵn có hoặc dạng ad-hoc.

Tài liệu tham khảo

1. Vondrak: “Business process modeling and workflow automation”.
2. Cesare Pautasso: “Lecture 11-Workflow Management Systems”, Computer Science Management, Swiss Federal Institute of Technology (ETHZ).
3. Tom Baeyens: “ The State of Workflow”, May-2004, tom@jbpm.org
4. JBoss jBPM - ”Workflow in Java”, <http://docs.jboss.org/jbpm/v3.2/-userguide/html/>
5. The Object Management Group (OMGT) <http://www.omg.org/>
6. The Business Process Management Initiative <http://www.bpmi.org/>
7. jBPM business process management, user guide <http://www.jboss.org/jboss-jbpm/jpdl/>
8. State of art of standards in business process modeling and execution: iisl.postech.ac.kr/publication/thesis/2007_jjm.pdf, <http://iisl.postech.ac.kr/>
9. T.Stoilov, K.Stoilova :” E-business workflow modeling and execution tools”, Department Hierarchical Systems, Institute of Computer and Communication Bulgarian Academy of Sciences, Acad. G.Bonchev str. Bl.2, 1113 Sofia, Bulgaria.
10. Domain Specific Languages, <http://www.martinfowler.com/bliki/Domain-SpecificLanguage.html>.
11. Workflow Pattern, <http://www.workflowpatterns.com/>.
12. Process virtual machine , manual page, http://docs.jboss.org/jbpm/pvm/manual/html_single/.
13. Introduction of the Graphical process designer, <http://www.jboss.org/jbossjbpm/gpd/>
14. Eclipse plugin, step by step, power by IBM, <http://www.ibm.com/developer-works/opensource/library/os-ecplug>
15. Plugin developement environment <http://www.eclipse.org/articles/Article-PDE-does-plugins/PDE-intro.html>
16. Eclipse Ganymede help, <http://help.eclipse.org/ganymede/index.jsp>

17. An introduction of Eclipse architecture and technical problems, http://www.eclipse.org/articles/Article-Plug-in-architecture/plugin_architecture.html
18. Understanding how Eclipse plug-ins work with OSGi, <http://www.ibm.com/developerworks/opensource/library/os-ecl-osgi/>
19. Moving Resources Between Eclipse and Rational Software Architect, http://open.ncsu.edu/se/tutorials/import_export
20. A First Look at Eclipse Plug-In Programming , http://www.developer.com/java/other/article.php/10936_3316241_2
21. Plug-in Development Environment (PDE) in Eclipse, http://open.ncsu.edu/se/tutorials/plugin_dev/
22. Eclipse and Jazz, <http://openseminar.org/se/modules/18/index/screen.do>
23. Open Source Product Evaluation <http://www.workflowpatterns.com/evaluations/opensource/index.php>
24. Workflow patterns, introductions and evaluations, <http://www.workflowpatterns.com/documentation/index.php>

Phụ lục

A. Hướng dẫn cài đặt workflow engine:

a. Cấu hình CSDL MySQL 5.0 cho JBPM 3.2.x:

- Tạo CSDL trong MySQL với tên tùy thích ví dụ: jbpmbpm
create database jbpmbpm;
- Chuyển qua sử dụng csdl trên với câu lệnh: use jbpmbpm;
- Trong thư mục \jbpmbpm-jpdl-3.2.x\db có file script [jbpmbpm.jpdl.mysql.sql](#). Thêm dấu chấm phẩy (;) vào tất cả các dòng trong file này.
- Thực thi file script [jbpmbpm.jpdl.mysql.sql](#) trong MySQL với câu lệnh:

source [đường dẫn đầy đủ]\jbpmbpm.jpdl.mysql.sql;

Insert login name

- Chạy các script sau trong MySQL để thêm người dùng vào CSDL

```
INSERT INTO JBPM_ID_GROUP VALUES(1,'G','sales','organisation',NULL);
INSERT INTO JBPM_ID_GROUP VALUES (2,'G','admin','security-role',NULL);
INSERT INTO JBPM_ID_GROUP VALUES (3,'G','user','security-role',NULL);
INSERT INTO JBPM_ID_GROUP VALUES (4,'G','hr','organisation',NULL);
INSERT INTO JBPM_ID_GROUP VALUES (5,'G','manager','security-role',NULL);
INSERT INTO JBPM_ID_USER VALUES (1,'U','user','user@sample.domain','user');
INSERT INTO JBPM_ID_USER VALUES
(2,'U','manager','manager@sample.domain','manager');
INSERT INTO JBPM_ID_USER VALUES (3,'U','admin','admin@sample.domain','admin');
INSERT INTO JBPM_ID_USER VALUES (4,'U','shipper','shipper@sample.domain','shipper');
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (1,'M',NULL,NULL,2,4);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (2,'M',NULL,NULL,3,4);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (3,'M',NULL,NULL,4,4);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (4,'M',NULL,NULL,4,3);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (5,'M',NULL,NULL,1,3);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (6,'M',NULL,NULL,2,3);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (7,'M',NULL,NULL,3,3);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (8,'M',NULL,NULL,3,2);
```

```
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (9,'M',NULL,NULL,2,2);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (10,'M',NULL,NULL,2,5);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (11,'M',NULL,'boss',2,1);
INSERT INTO JBPM_ID_MEMBERSHIP VALUES (12,'M',NULL,NULL,1,1);

UPDATE jbpm_id_membership j, jbpm_id_user u SET j.NAME_ = u.NAME_ WHERE j.USER_
= u.ID_;
UPDATE jbpm_id_membership j, jbpm_id_group g SET j.ROLE_ = g.NAME_ WHERE
j.GROUP_ = g.ID_;
```

- Trong file script db chuẩn của jbpms sau khi tạo DB mới thì bảng jbpms_id_membership sẽ có các trường Name_ và Role_ có giá trị là NULL, chính vì thế khi jbpms check_login với câu lệnh này:

```
SELECT PASSWORD_ FROM JBPM_ID_USER WHERE NAME_=?
```

- Sẽ không thể check được quyền và báo lỗi SQLException.
- Chính vì vậy hai câu lệnh update trên phải được thực hiện để fix lỗi này

- Tạo datasource kết nối vào MySQL trong jBPM
- Tạo một file [jbpm-ds.xml](#) với nội dung như sau và lưu vào thư mục `\jbpm-jpdl-3.2.x\server\server\jbpm\deploy\`

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
  <local-tx-datasource>
    <jndi-name>JbpmDS</jndi-name>
    <connection-url>jdbc:mysql://localhost:3306/jbpm</connection-url>
    <driver-class>com.mysql.jdbc.Driver</driver-class>
    <user-name>root</user-name>
    <password>root</password>
    <metadata>
      <type-mapping>MySQL</type-mapping>
    </metadata>
  </local-tx-datasource>
</datasources>
```

- Sử dụng một trình tháo nén (VD: Winrar) để tháo nén tập tin [jbpm-console.war](#) trong thư mục `\jbpm-jpdl-3.2.x\server\server\jbpm\deploy`
- Bạn sẽ tìm thấy một tập tin xml tên [hibernate.cfg.xml](#) trong thư mục `\jbpm-console\WEB-INF\classes`
- Edit tập tin này như sau:

```
<!-- hibernate dialect -->
  <!--<property name="hibernate.dialect">org.hibernate.dialect.HSQLDialect</property>
-->

  <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<property
name="hibernate.cache.provider_class">org.hibernate.cache.HashtableCacheProvider</pr
operty>
```



```

<!--<property
name="hibernate.connection.datasource">java:comp/env/jdbc/JbpmDataSource</property
> -->

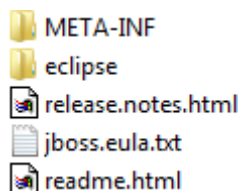
<property name= "hibernate.connection.datasource">
java:/JbpmDS</property>
<!-- JDBC connection properties (begin) ===
<property name="hibernate.connection.driver_class">
org.hsqldb.jdbcDriver</property>
<property name="hibernate.connection.url">
jdbc:hsqldb:mem:jbpm</property>
<property name="hibernate.connection.username">sa </property>
<property name="hibernate.connection.password"> </property>
<!--===== JDBC connection properties (end) -->
<!-- JTA transaction properties (begin) ===

```

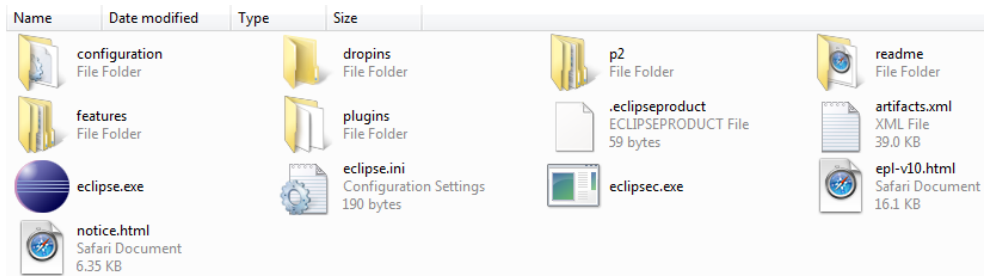
- Lưu lại file này vào file nén ...\\jbpm\\deploy\\[jbpm-console.war](#) và khởi động jBPM.

B. Hướng dẫn cài đặt jPDL-editor.

- Download Eclipse Ganymede.
- Download jbpm-gpd eclipse plugins.
- Giải nén hai file plugins và features vào trong eclipse Ganymede (theo thứ tự liệt kê).

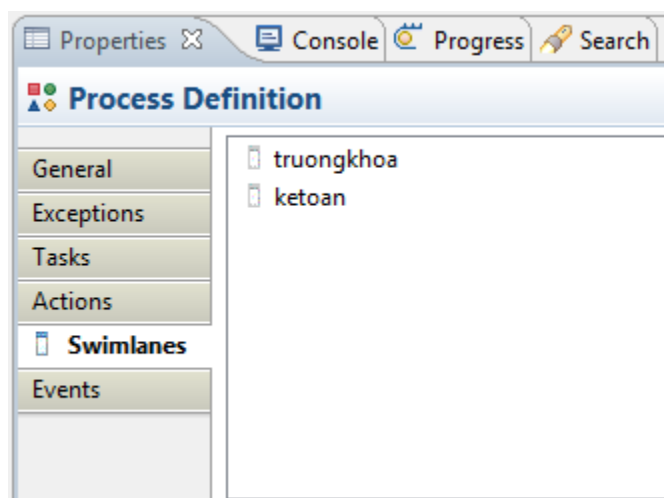


- Giải nén và chép file jbpm-gpd vào thư mục plugins của eclipse Ganymede.



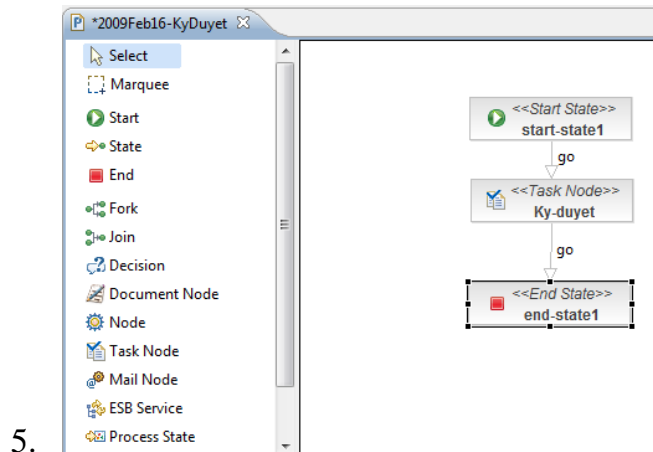
C. Hướng dẫn triển khai một ứng dụng luồng công việc từ jpdI-editor.

1. Khi có một quy trình nghiệp vụ được xác định trước trong một tổ chức hay cơ quan nào đó. Việc mô hình hóa quy trình nghiệp vụ thực tế để có thể quản lý dưới dạng luồng công việc trong ứng dụng jBPM thì có thể làm tuần tự theo các bước sau (bước 1 và 3 có thể hoán đổi thứ tự thực hiện).
2. Thiết kế swimlane (bước 1):
 - Dựa vào bảng phân tích quy trình nghiệp vụ, người thiết kế sẽ xây dựng trước được những người nào sẽ tham gia vào luồng công việc.
 - Ở bước này, người tham gia được trừu tượng hóa lên mức tổng quát, quá trình thực hiện ánh xạ người tham gia thực tế với người tham gia trong lúc thiết kế luồng công việc sẽ được thực hiện khi thực thi một dự án thực tế.



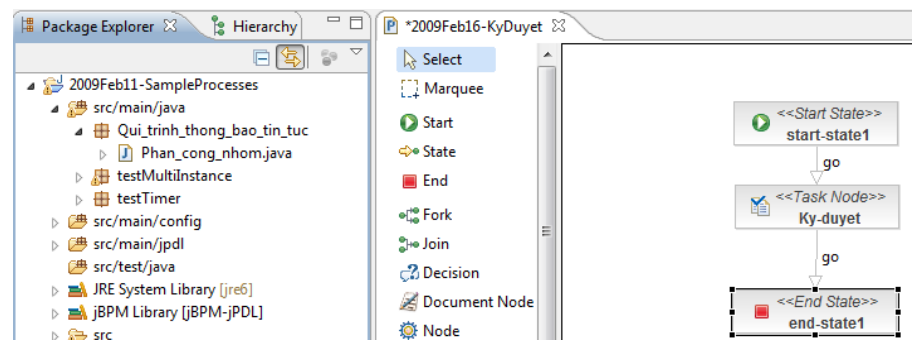
- 3.
4. Thiết kế luồng công việc (bước 2):

- Dựa vào đặc thù của qui trình nghiệp vụ thực tế, người thiết kế sử dụng những ký hiệu được đặc tả trong jpdL-editor để mô tả lại qui trình nghiệp vụ.



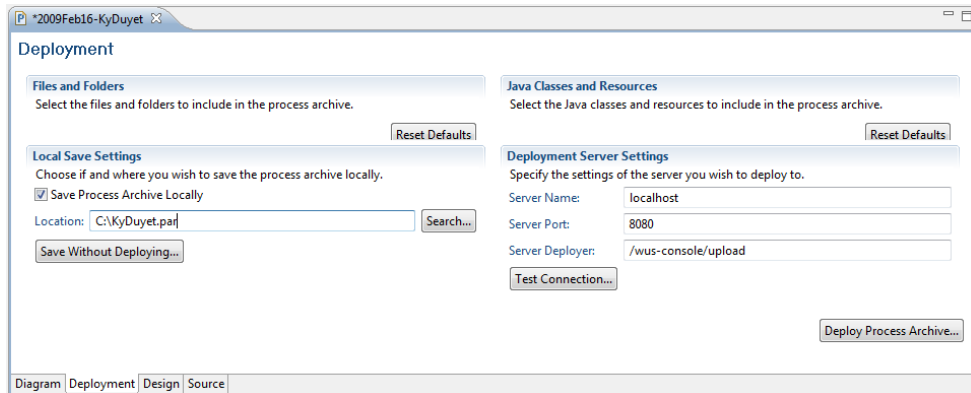
6. Thiết kế các hàm xử lý nghiệp vụ kèm theo (nếu có):

- Ở những bước đòi hỏi phải xử lý tình huống để luồng công việc có thể lựa chọn bước kế tiếp thì người thiết kế phải sử dụng lớp java để hỗ trợ kèm theo luồng công việc.
- Ví dụ: trường hợp gửi thông báo cho nhóm có 100 người, khi task nhận thông báo hoàn tất thì task kế tiếp mới được thực hiện. Do vậy, người thiết kế phải bổ sung thêm hàm xử lý để quyết định bước kế tiếp của qui trình nghiệp vụ.



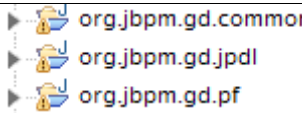
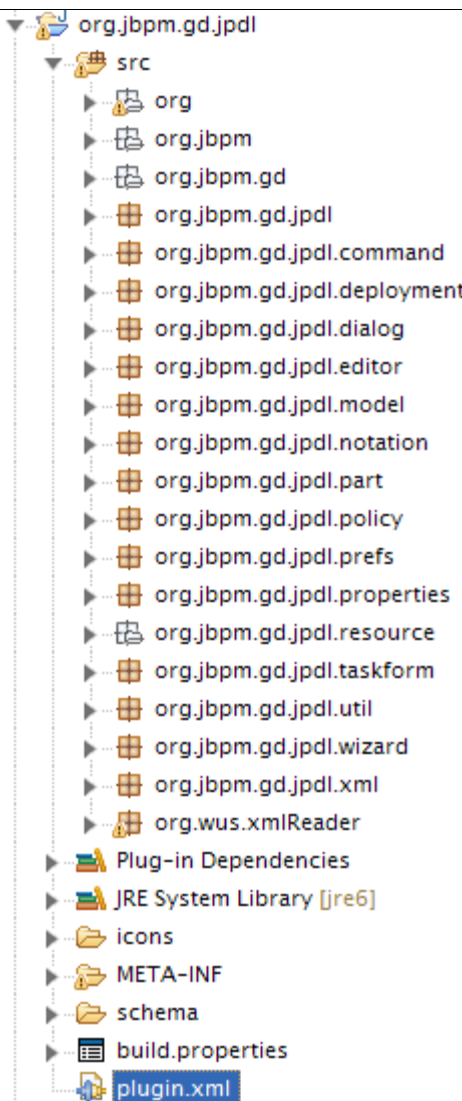
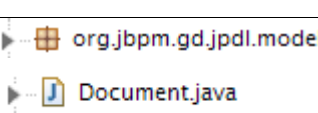
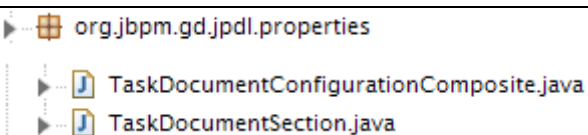
7. Triển khai một luồng công việc trên jBPM-engine:

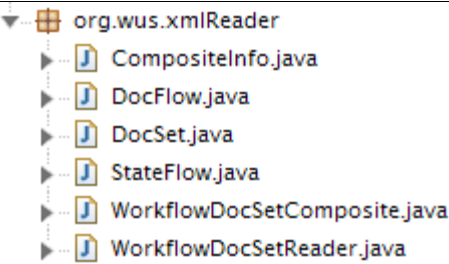
- Sau khi đặc tả và thiết kế hoàn chỉnh luồng công việc và các hàm xử lý bổ sung, người thiết kế có thể khởi động jBPM-engine để triển khai trực tiếp bằng thao tác trên jPDL-editor hoặc bằng cách sử dụng file .par.

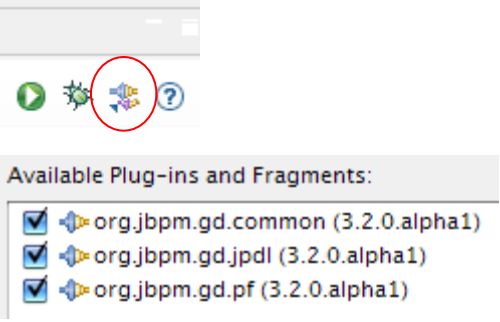


D. Hướng dẫn cách xây dựng document node và property node:

Quá trình tạo Document Element

1.		Customize bộ jbpm GPD – Graphic Process Designer
2.		Project org.jbpm.gd.jpdl thực hiện nhiệm vụ tạo các notation để export ra file jpdl
3.		Dùng để định nghĩa class Document.
4.		Dùng để xây dựng phân giao diện trên Eclipse workbench.

5.		Kế thừa từ lớp XML Adapter để thực hiện insert notation vào file jpdl.
6.		Dùng để đọc thông tin danh sách các document từ file WorkflowDocSet.
7.	<pre> 1<workflowDocumentSets> 2 3 <workflowDocumentSet id="ws1" wdName="ws1-Name"> 4 5 <document id="doc1" docName="doc1-Name"> 6 7 <currentState id="s1"/> 8 9 <stateFlow> 10 <state id="s1" nextStateId="s1" stateName="s1-Name"/> 11 </stateFlow> 12 13 </document> 14 15 </workflowDocumentSet> 16</workflowDocumentSets> </pre> <p>Format của file WorkflowDocSet. Chủ yếu gồm 3 thông tin chính: documentId, currentStateId, uploadAction.</p>	
8.		

	Khai báo các extensions vừa khởi tạo để jbpmm-GPD có thể hiểu được document node vừa thêm vào.	
9.	 <p>Available Plug-ins and Fragments:</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> org.jbpm.gd.common (3.2.0.alpha1) <input checked="" type="checkbox"/> org.jbpm.gd.jpdl (3.2.0.alpha1) <input checked="" type="checkbox"/> org.jbpm.gd.pf (3.2.0.alpha1) 	Thực hiện build thành plugin. Copy vào thư mục Plugins của Eclipse để sử dụng.