BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC HOA SEN

KHOA KHOA HỌC VÀ CÔNG NGHỆ

KHÓA LUẬN TỐT NGHIỆP

Đề tài: Xây dựng ngôn ngữ theo hướng tác tử

Giảng viên hướng dẫn: TS. Trần Vũ Bình

Nhóm sinh viên thực hiện:

Ngô Thị Đan Thanh	061496
Trần Thị Thu Thảo	061475
Nguyễn Ngọc Nhã Vy	060445

Tháng 12 / năm 2010

TRÍCH YẾU

Trong thời gian gần đây, đã trở nên rõ ràng rằng các hệ thống Công nghệ thông tin truyền thống không còn phù hợp để tương tác với những môi trường năng động. Thực hiện tương tác đã chứng minh được việc tốn nhiều thời gian và dễ gây lỗi. Nó đặc biệt có vấn đề khi có nhiều thực thể bên ngoài phải được kiểm soát. Trong quá khứ, thực hiện hệ thống như thế đã kéo theo việc lập trình một cách rõ ràng với sự tương tác giữa các thực thể bên ngoài. Nếu có nhiều hơn một tương tác giữa client/server, thì ứng dụng sẽ rất khó thực hiện, duy trì hay thay đổi.

Nghiên cứu về tác tử phát triển rất nhanh trong những năm gần đây, và ngày càng có nhiều người quan tâm đến vấn đề nghiên cứu các hệ thống theo hướng tác tử.

Luận án đưa ra vài đóng góp trong lĩnh vực nghiên cứu chiều hướng phát triển của việc xây dựng một ngôn ngữ lập trình mới theo hướng tác tử. Kết quả này cung cấp một nền tảng cho việc thu hẹp khoảng cách giữa lý thuyết và triển khai thực tế của hệ thống tác tử. Việc khép lại khoảng cách này hứa hẹn sự cải tiến, triển khai các lý thuyết tác tử và các phương pháp kĩ nghệ phần mềm tốt hơn để xây dựng hệ thống trực quan và sử dụng được.

MỤC LỤC

LỜI CÁM ƠN

Sau những năm tháng học tập trên ghế nhà trường, nhằm tạo điều kiện cho sinh viên có thể tốt nghiệp và áp dụng những kiến thức sinh viên đã thu thập được trong quá trình tích lũy kiến thức vào thực tế, trường Đại học Hoa Sen đã đưa ra những đề tài luận án trong những năm cuối của cuộc đời sinh viên. Có thể nói luận án tốt nghiệp như một bài học tổng hợp về khả năng lập trình, phân tích, và hoạt động nhóm. Đây còn được xem như một tác phẩm vô cùng quan trọng, là đứa con tinh thần, là mốc son của cuộc đời của sinh viên.

Trước hết, nhóm xin cám ơn Thầy Trần Vũ Bình đã tạo cơ hội cho nhóm được tiếp xúc với một đề tài mang tính mới lạ hơn so với các đề tài đã làm trước đó, giúp nhóm có thêm kinh nghiệm để ứng dụng cho công việc tương lai sau này. Đây là một đề tài mà để hoàn thành, đòi hỏi phải tìm tòi, nghiên cứu nhiều. Vì vậy, đề tài đã giúp nhóm năng động hơn khi tìm cách thích hợp nhất để giải quyết, tạo sự hứng thú, không bị gò bó theo khuôn khổ nhất định.

Trong suốt quá trình thực hiện luận văn, nhóm xin cám ơn Thầy đã tận tình giúp đỡ cũng như hướng dẫn nhóm hoàn thành khóa luận. Chính nhờ sự giúp đỡ của Thầy mà nhóm đã giải quyết được những khúc mắc cũng như vấn đề khó khăn gặp phải. Thêm vào đó, nhóm đã được trang bị thêm kiến thức mới và hệ thống lại kiến thức cũ trong quá trình thực hiện luận. Thầy đã nhiệt tình hướng dẫn và chỉ ra những phần mà nhóm làm chưa tốt để nhóm có thể khắc phục, sửa chữa nhằm đạt kết quả tốt và chính xác hơn. Ngoài ra, Thầy còn động viên nhóm tích cực hoàn thiện tốt đề án.

Một lần nữa, nhóm xin cảm ơn sự nhiệt tình dìu dắt và hướng dẫn của Thầy trong suốt quá trình làm đề án.

DANH MỤC CÁC BẢNG BIỂU

NHẬP ĐỀ

Trong thời buổi hệ thống công nghệ thông tin phát triển như hiện nay, Việt Nam chúng ta là một nước có nền công nghệ đang phát triển và có tiềm năng rất lớn. Không những quan tâm tới các vấn đề tiện ích công nghệ phục vụ cho đời sống, hệ thống công nghệ thông tin hiện nay cũng góp phần vào việc giúp đỡ cho các công trình nghiên cứu mang tính khoa học. Một trong những đề tài đó là về 'Xây dựng ngôn ngữ theo hướng tác tử'.

Mục tiêu của luận án khi thực hiện đề tài này là xây dựng được ngôn ngữ hỗ trợ theo hướng tác tử. Các yêu cầu về ngôn ngữ ấy gồm có:

- Một ngôn ngữ hoàn chỉnh sử dụng cho việc đặc tả các thành phần hệ thống và hành vi.
- Một ngữ nghĩa rành mạch được sử dụng để xác định những thuộc tính của tác tử và dựa vào đó, các thực thi của hệ thống có thể được đo lường và đánh giá.
- Một ngôn ngữ cho phép người dùng mô tả thực tế sau đó chuyển đổi sang để tương tác với hệ thống

Nhóm thực hiện đề tài và báo cáo dưới sự phân công như sau:

Bảng 1: Bảng phác thảo phân công công việc

Tuần	Tên công việc	Người thực hiện	Tiến độ hoàn thành
1	Tìm hiểu và thực hiện một complier đơn giản.	Nguyễn Ngọc Nhã Vy	100%
	Tìm hiểu modal logic, BDI, Jack agent.	Ngô Thị Đan Thanh	100%
	Tìm hiểu cách thức lưu trữ của cơ sở dữ liệu.	Trần Thị Thu Thảo	100%
2	Tiếp tục tìm hiểu về complier đơn giản.	Nguyễn Ngọc Nhã Vy	100%
	Tìm hiểu về các các khái niệm Reflection, Custom Attribute.	Ngô Thị Đan Thanh	100%
	Nghiên cứu thêm về lưu trữ dữ liệu trong database.	Trần Thị Thu Thảo	100%
3	Phân tích theo hướng agent về một vài ví dụ cơ bản. Bước đầu phân tích bài toán thực tế về bệnh viện, chuyển bệnh nhân giữa các bệnh viện với nhau.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
4	Chuyển bài toán phân tích ứng dụng, phân tích về việc quản lý bệnh án điện tử, chuyển bệnh nhân vào các khoa của bệnh viện. Bước đầu xây dựng cú pháp cho ngôn ngữ hướng tác tử.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
5	Tiếp tục phân tích, tìm hiểu quy trình hoạt động và nghiệp	Ngô Thị Đan Thanh	100%

	vụ, kiến thức về các hoạt động trong bệnh viện.	Trần Thị Thu Thảo	
	Tiếp tục chỉnh sửa, hoàn thiện cú pháp ngôn ngữ.	Nguyễn Ngọc Nhã Vy	100%
6	Chỉnh sửa, thực hiện xong cú pháp. Chỉnh sửa về cú pháp, ngoài các cú pháp đã được thực hiện như bind, add, create, remove, delete, update,… thì tiếp tục chỉnh sửa và thêm các cú pháp mới (cho phép người dụng tạo tập điều kiện cho method, xây dựng tập đại diện).	Nguyễn Ngọc Nhã Vy	100%
	Tìm kiếm và thu thập dữ liệu thực tế. Thiết kế mô hình tương tác ở bên dưới của ngôn ngữ, thể hiện mối liên hệ giữa các Context, Agent, Condition,… và các Library.	Ngô Thị Đan Thanh Trần Thị Thu Thảo	100%
7	Mô tả cơ chế map Capability với Capability, Role trong Context, tập đại diện, sự di chuyển từ Context này sang Conext kia của Agent. Thực hiện cơ bản việc kết nối các method từ các Object cụ thể thông qua Interface.	Nguyễn Ngọc Nhã Vy	
	Xây dựng cơ sở dữ liệu, một vài chức năng cơ bản của bài toán bệnh án điện tử. Tiếp tục chỉnh sửa và hoàn thiện mô hình tương tác bên dưới.	Ngô Thị Đan Thanh	

	Xây dựng interface để kết nối giữa ngôn ngữ và Object cụ thể do người dùng định nghĩa.	Trần Thị Thu Thảo	100%
8	Xây dựng các Library hỗ trợ: Việc tập hợp các Condition của Method lên thành tập hợp Condition của Capability, tương tự từ Capability lên Role, Role lên Context. Xây dựng tập đại diện và tiến hành map các tập điều kiện với tập đại diện đó để di chuyển Agent giữa các Context. Xây dựng việc so sánh Cap của Agent và Cap của Role để đưa Agent và Role đó trong Context. Chuyển đổi việc xuất Agent dưới dạng một Object.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
9	Xây dựng các Object, method của hệ thống bệnh án điện tử. Kiểm tra việc thực hiện của các Library và việc kết nối với ngôn ngữ đang xây dựng.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
10	Tiếp tục chỉnh sửa, hoàn thiện các object. Test toàn bộ hệ thống hướng tác tử.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
11	Chỉnh sửa và hoàn chỉnh các object. Chuyển bài toán ứng dụng thành Tư vấn đăng ký môn	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%

	học.		
12	Rà soát và kiểm tra, đánh giá lại những gì đã thực hiện. Tiếp tục phân tích bài toán Đăng ký môn học theo hướng agent. Viết dàn bài cho báo cáo.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
13	Tiếp tục chỉnh sửa và viết báo cáo. Phân tích và hoàn thiện bài toán Đăng ký môn học theo hướng agent. Chuẩn bị Slides cho buổi báo cáo bảo vệ luận án.	Ngô Thị Đan Thanh Trần Thị Thu Thảo Nguyễn Ngọc Nhã Vy	100%
14	Góp ý, chỉnh sửa báo cáo	Nguyễn Ngọc Nhã Vy	100%
	Hoàn tất báo cáo.	Ngô Thị Đan Thanh Trần Thị Thu Thảo	100%

Xem chi tiết về phân công công việc trong tập tin PhanCongLA.mpp đính kèm.

Kế hoạch sơ bộ cho việc thực hiện Luận án	70 days?	Mon 9/13/10	Fri 12/17/10
Tuần 1	5 days?	Mon 9/13/10	Fri 9/17/10
Tuần 2	5 days?	Mon 9/20/10	Fri 9/24/10
Tuần 3	5 days?	Mon 9/27/10	Fri 10/1/10
Tuần 4	5 days?	Mon 10/4/10	Fri 10/8/10
Tuần 5	5 days?	Mon 10/11/10	Fri 10/15/10
Tuần 6	5 days?	Mon 10/18/10	Fri 10/22/10
Tuần 7	5 days?	Mon 10/25/10	Fri 10/29/10
Tuần 8	5 days?	Mon 11/1/10	Fri 11/5/10
Tuần 9	5 days?	Mon 11/8/10	Fri 11/12/10
Tuần 10	5 days?	Mon 11/15/10	Fri 11/19/10
Tuần 11	5 days?	Mon 11/22/10	Fri 11/26/10
Tuần 12	5 days?	Mon 11/29/10	Fri 12/3/10
Tuần 13	5 days?	Mon 12/6/10	Fri 12/10/10
Tuần 14	5 days?	Mon 12/13/10	Fri 12/17/10

Hình 1: Sơ đồ các phần công việc chính

GIỚI THIỆU BÀI TOÁN

I. Mô tả bài toán

1. Đặt vấn đề bài toán

Như chúng ta đã biết, một trong những ưu điểm của lập trình hướng đối tượng khi so với lập trình cấu trúc chính là ở đặc tính đa hình. Tuy nhiên, phương pháp ấy không phải lúc nào cũng đạt được kết quả chính xác như mong đợi, do không dựa vào hoàn cảnh. Vấn đề đặt ra là, giả sử có nhiều hoàn cảnh khác nhau, làm sao để đảm bảo độ chính xác khi cùng một method nhưng sẽ tùy theo hoàn cảnh thay đổi mà method đó sẽ được thực hiện khác nhau?

Chúng ta sẽ xét một ví dụ cụ thể.

Khi áp dụng câu chào ‘Hello’ trong tiếng Anh vào tiếng Việt, chữ ‘Hello’ sẽ không chỉ đơn thuần là ‘Xin chào’, mà sẽ được gắn với một câu chào nhất định từ một hoàn cảnh cụ thể, gồm ‘Xin chào’ + chức danh, như chào anh, chào chị…

Việc xác định để chào như thế nào là hợp lý được dựa trên rất nhiều điều kiện, như quan hệ gia đình, tuổi tác, giới tính, và trong từng hoàn cảnh cụ thể thì có thể phát sinh thêm nhiều điều kiện khác. Bản chất của ví dụ trên chính là cùng một method, nhưng tùy hoàn cảnh mà method đó sẽ thực hiện khác nhau. Để xác định câu chào hợp lý, ta phải xác định dựa trên hoàn cảnh và những điều kiện phát sinh. Tuy nhiên, nếu áp dụng theo mô hình hướng đối tượng thì kết quả sẽ không rõ ràng và chi tiết, vì trong lập trình hướng đối tượng thường tạo class và gom chung các điều kiện thành method, sau đó khi tương tác ta chỉ có thể gọi class đó nên kết quả sinh ra sẽ thiếu rõ ràng.

Xét một ví dụ khác. Đặt vấn đề xảy ra như sau

Hôm nay là ngày nghỉ lễ đối với các sinh viên trường Đại học Hoa Sen. Trong ngày nghỉ lễ ấy, tuy đều là sinh viên nhưng mỗi người cụ thể lại có những việc làm khác nhau. Chẳng hạn như có người mua sắm, có người ở nhà, có người xem phim.

Từ ví dụ trên, ta thấy, trong cùng một hoàn cảnh (ngày nghỉ lễ), cùng một vai trò (sinh viên), nhưng với từng đối tượng cụ thể lại sinh ra những hành động khác nhau. Vấn đề đặt ra là, giả sử chúng ta cần mô tả use case của một bài toán, nhưng use case cần mô tả trở nên quá phức tạp tùy theo điều kiện sinh ra tình huống. Vì vậy, khi tình huống thay đổi thì toàn bộ mô tả cũng phải thay đổi theo. Nếu có quá nhiều tình huống xảy ra, và áp dụng phân tích theo hướng đối tượng, sẽ có rất nhiều tình huống cần phải nhắc đến một cách rõ ràng và chi tiết khi mô tả usecase. Với số lượng tổ hợp các tình

huống tăng lên một con số rất lớn, hệ thống hướng đối tượng sẽ không thể đáp ứng cho một phương pháp tối ưu.

Từ ví dụ đã nêu, ta có thể nhận thấy môi trường thay đổi rất đa dạng, với nhiều tình huống có thể phát sinh, nên từ đó nhìn ra những khuyết điểm còn thiếu sót của lập trình hướng đối tượng.

2. Hướng giải quyết

Từ những khiếm khuyết đã nêu ra của lập trình hướng đối tượng, lập trình theo hướng tác tử đã ra đời, nhằm bổ sung và giải quyết những bài toán mà lập trình hướng đối tượng vẫn chưa giải quyết được.

Để có thể áp dụng bài toán theo hướng tác tử, chúng ta phải giải quyết được ba bài toán chính.

1. Bài toán thứ nhất là bài toán về cơ sở lý luận, phương pháp nào để thay thế đa hình, cho phép lựa chọn phương thức linh hoạt hơn so với tính kế thừa trong lập trình hướng đối tượng.

2. Bài toán thứ hai là bài toán về phân tích thiết kế, làm sao để nắm bắt được những điều kiện hình thành nên hoàn cảnh, các tình huống sử dụng khác nhau như thế nào, và quá trình nào hỗ trợ điều đó.

3. Bài toán thứ ba là bài toán về công cụ hỗ trợ.

Như chúng ta đã biết, trong vấn đề hướng tiếp cận và cơ sở lý luận, để có thể xem xét chính xác một vấn đề, ta cần đặt nó vào một hoàn cảnh, vì khi hoàn cảnh thay đổi, thì những tương tác của tác nhân với môi trường ngoài xung quanh cũng sẽ thay đổi theo. Trong hướng đối tượng, thông thường chỉ có một hoàn cảnh duy nhất. Tuy nhiên, theo hướng tác tử, chúng ta có nhiều hoàn cảnh khác nhau, vì vậy để có thể mô tả trên nhiều hoàn cảnh khác nhau, khái niệm về hoàn cảnh cụ thể (possible word) được ra đời.

Xét tiếp bài toán chọn công cụ xây dựng ngôn ngữ hỗ trợ. Trong bài toán này, chúng ta sẽ có khái niệm về capability.

Khái niệm *capability* là một cách cơ cấu lại những yếu tố lý luận của các toán tử vào những tập hợp được lựa chọn để thực hiện khả năng lý luận. Kỹ thuật này đơn giản hóa việc thiết kế hệ thống tác tử, cho phép sử dụng lại mã số, và đóng gói các chức năng tác tử.

Khả năng của đại diện về khía cạnh chức năng của một tác tử có thể được đưa vào theo yêu cầu. Khả năng này được xem như cách tiếp cận thành phần, cho phép một

kiến trúc sư hệ thống tác tử xây dựng một thư viện các khả năng theo thời gian. Các thành phần này sau đó có thể được sử dụng để thêm chức năng lựa chọn với tác tử.

Thêm vào đó, khả năng có thể được cấu trúc để một số khả năng-con có thể được kết hợp để cung cấp chức năng phức tạp trong một khả năng-lớn. Khả năng này có thể được thêm vào một tác tử để tác tử đó có chức năng mong muốn.

Theo hướng tác tử, bản chất của capability là một tập hợp các method, mặt khác các method là một tập hợp các condition, và condition là một tập hợp các vị từ.

II. Những khái niệm liên quan

1. Lập trình hướng tác tử là gì [7]

Lập trình theo hướng tác tử là một mô hình phần mềm tiên tiến xuất phát từ nghiên cứu trong trí tuệ nhân tạo. Lập trình theo hướng tác tử chỉ ra sự cần thiết cho các hệ thống phần mềm để đưa ra sự hợp lý, hành vi của con người giống như trong các vấn đề tương ứng, vì hệ thống phần mềm truyền thống sẽ gặp khó khăn trong việc mô hình hành vi một cách hợp lý, và thường các chương trình bằng văn bản trong hệ thống sẽ hạn chế những kinh nghiệm này, đặc biệt là khi cố gắng hoạt động trong các môi trường thời gian thực.

Lập trình theo hướng tác tử rất phù hợp với nhiều lĩnh vực ứng dụng, bao gồm các hệ thống kinh doanh phân phối, chỉ huy và kiểm soát, các thiết bị thông minh và mô phỏng. Mặc dù còn trẻ và đang được phát triển, lập trình hướng tác tử cũng đã thể hiện được mình trong một loạt các vấn đề phân phối, giải quyết các nhiệm vụ, như tổ chức đội tàu, quản lý giao thông trên không và mô phỏng không chiến. Bởi do cung cấp một giải pháp với nhiều vấn đề cần đối mặt trong xử lý phản ứng, lập trình theo hướng tác tử rất lý tưởng cho các môi trường này.

Mô hình theo hướng tác tử đi theo cùng một nguyên tắc cơ bản của lập trình hướng đối tượng – đó là phát triển sự tin cậy và khả năng mở rộng được tăng cường bởi các hành vi mong muốn, bằng cách đóng gói hành vi mong muốn đó trong các đơn vị module có chứa tất cả các định nghĩa và cơ cấu cần thiết để chúng hoạt động một cách độc lập. Tác tử mở rộng khái niệm về đóng gói bao gồm một đại diện của hành vi ở mức độ cao hơn – phương pháp tiếp cận hướng đối tượng.

2. Modal logic

Như đã đề cập đến trong mục phần đặt vấn đề bài toán của mục I.1, bài toán thực tế là bài toán với nhiều hoàn cảnh cụ thể thay đổi khác nhau. Vì vậy, khái niệm về modal logic đã xuất hiện để hỗ trợ cho khái niệm về hoàn cảnh cụ thể. Modal logic được sử dụng để mô tả cho những hoàn cảnh cụ thể.

Không giống các ngôn ngữ cổ điển mô tả mối quan hệ cấu trúc của những class tương tự nhau, modal logic mô tả cấu trúc trong nội bộ. Các thông tin được lưu giữ nơi những hoàn cảnh khác có thể được thăm dò, tuy nhiên, chỉ có các thông tin được truy cập trực tiếp từ hoàn cảnh cụ thể hiện tại mới được các phương thức nắm giữ.

Thông thường, để có thể xem xét chính xác một vấn đề, ta cần đặt nó vào một hoàn cảnh, vì khi hoàn cảnh thay đổi, thì những tương tác của tác nhân với môi trường ngoài xung quanh cũng sẽ thay đổi theo. Trong hướng đối tượng, thường chỉ có một hoàn cảnh duy nhất. Tuy nhiên, theo hướng tác tử, chúng ta có nhiều hoàn cảnh khác nhau. Vì vậy, trong modal logic, mỗi vị từ sẽ ứng với từng hoàn cảnh cụ thể, vị từ trong hoàn cảnh này có thể đúng nhưng sai trong hoàn cảnh khác, và vị từ chính là điều kiện để xác định phải gọi hàm gì.

[12] Modal logic là một loại của logic chính thức và được mở rộng từ logic chính thức, bao gồm các yếu tố của phương thức (ví dụ như khả năng, và sự cần thiết). Modal logic bổ nghĩa cho sự thật trong việc đưa ra phán xét. Ví dụ như, nếu sự thật là "Anh ấy hạnh phúc." chúng ta có thể bổ nghĩa cho phát biểu trên bằng cách nói rằng "Anh ấy *rất* hạnh phúc.", với từ *'rất'* là một phương thức.

Một modal logic chính thức đại diện cho phương thức bằng cách sử dụng phương thức toán tử. Ví dụ như, "Trời có thể mưa hôm nay" và "Có thể mưa ngày hôm nay" đều chứa các khái niệm về khả năng. Trong một modal logic, điều này được thể hiện bằng toán tử *có thể,* thuộc về một loại khả năng.

Theo truyền thống, có ba dạng của phương thức hay trạng thái được biểu diễn trong modal logic, gồm khả năng, xác suất, và sự cần thiết.

3. Đặt nhãn trong modal logic

Một khái niệm khác cần được đề cập tiếp theo trong cơ sở lý luận chính là khái niệm đặt nhãn trong modal logic. Việc đặt nhãn trong modal logic sẽ giúp ta mô tả những thay đổi của hoàn cảnh.

Thông thường, để chứng minh vấn đề, chúng ta sẽ thay các giá trị đã có vào mệnh đề biết trước. Tuy nhiên, nếu có quá nhiều giá trị cần thay vào, hay nói cách khác, nếu tập các giá trị cần thay vào để kiểm chứng tương đối lớn, vậy sẽ dẫn đến việc khó xác định kết quả.

Từ những điều trên, ta sẽ sinh ra khái niệm đặt nhãn trong modal logic.

Đặt vào các trường hợp cụ thể, ta sẽ dễ dàng xét được các giá trị, như

W_0 gồm có $w_1 \longrightarrow a = true$

$w_2 \longrightarrow a = true$

$w_3 \longrightarrow a = false$

Vấn đề: Điều kiện dẫn đến a true có 2 trường hợp đúng

W_0 $w_1 \longrightarrow a = true$

$w_2 \longrightarrow a = true$

Vậy, để xác định đường đi từ W_0 đến a = true thì sẽ đi như thế nào?

Giải quyết: Tiến hành đặt nhãn cho các trường hợp cụ thể.

Khi đó ta có: $T_0w_0 \rightarrow w_1$: T_1 a = true

$T_1w_1 \rightarrow w_2$: T_2 a = true

Lúc này có thể dễ dàng xác định chính xác các đường đi qua các nhãn vì mối tương xứng hiện tại là 1:1

4. Hybrid logic

Khái niệm cuối cùng cần tìm hiểu trong cơ sở lý luận chính là khái niệm về hybrid logic.

Như đã được đề cập đến ở phần II.3, khái niệm modal logic ra đời để hỗ trợ cho việc mô tả những hoàn cảnh cụ thể. Tuy nhiên, chỉ riêng mỗi modal logic sẽ không thể mô tả được cấu trúc hệ thống cũng như cách di chuyển giữa những hoàn cảnh đó. Hơn nữa, những ngôn ngữ mô tả trước đây chưa đáp ứng được yêu cầu đặt ra trong việc mô tả cấu trúc này. Vì vậy, khái niệm hybrid logic ra đời. Thông qua hybrid logic để tăng cường chức năng mô tả của các ngôn ngữ đặc tả tác tử.

Hybrid logic là một trong những phong trào cách mạng trong phương thức logic để tăng độ biểu đạt của phương thức ngôn ngữ. Mỗi hoàn cảnh có thể được đại diện bởi một *nominal* (danh nghĩa) trong ngôn ngữ đối tượng. Nó cung cấp một phương tiện để nghiên cứu mối quan hệ giữa các model frame, các biểu tượng và những sự diễn giải. Một giải thích của bất kỳ ý kiến nào có thể thay đổi trong những hoàn cảnh cụ thể khác nhau, nhưng việc giải thích của một *nominal* là duy nhất. Nominals trở thành điểm mốc trong hệ thống miêu tả mà bất kỳ thay đổi nào cũng đều được so sánh.

[10] Hybrid logic chính là sự mở rộng của modal với sức mạnh biểu đạt nhiều hơn. Không giống như phương thức logic thông thường, hybrid logic khiến bản thân nó có thể tìm đến một trạng thái trong hoàn cảnh cụ thể (possible world) trong công thức. Điều này đạt được bởi danh nghĩa. Danh nghĩa sẽ đúng trong chính xác một trạng thái.

Để làm rõ khái niệm trên, chúng ta sẽ xét kỹ hơn.

[6] Trong modal logic, sự thật mang ý nghĩa tương đối và phụ thuộc vào điểm mốc. Vì vậy, một ký hiệu mệnh đề có thể có giá những giá trị đúng khác nhau tùy thuộc vào những điểm mốc khác nhau. Thông thường, những điểm này được đưa ra để đại diện cho những hoàn cảnh cụ thể, thời gian, không gian, trạng thái trong một máy tính, hay cái gì khác. Lấy ví dụ, như lời phát biểu

Trời đang mưa

Câu phát biểu trên rõ ràng có giá trị đúng hoặc sai tùy vào những thời gian khác nhau. Như vậy, rõ là những phát biểu của ngôn ngữ tự nhiên chỉ đúng vào một thời gian xác định, một hoàn cảnh xác định. Ví dụ như câu

Bây giờ là năm giờ ngày 15 tháng Ba năm 2007

Thì câu trên chỉ đúng vào thời điểm năm giờ ngày 15 tháng Ba năm 2007, nhưng sai ở tất cả các thời điểm khác.

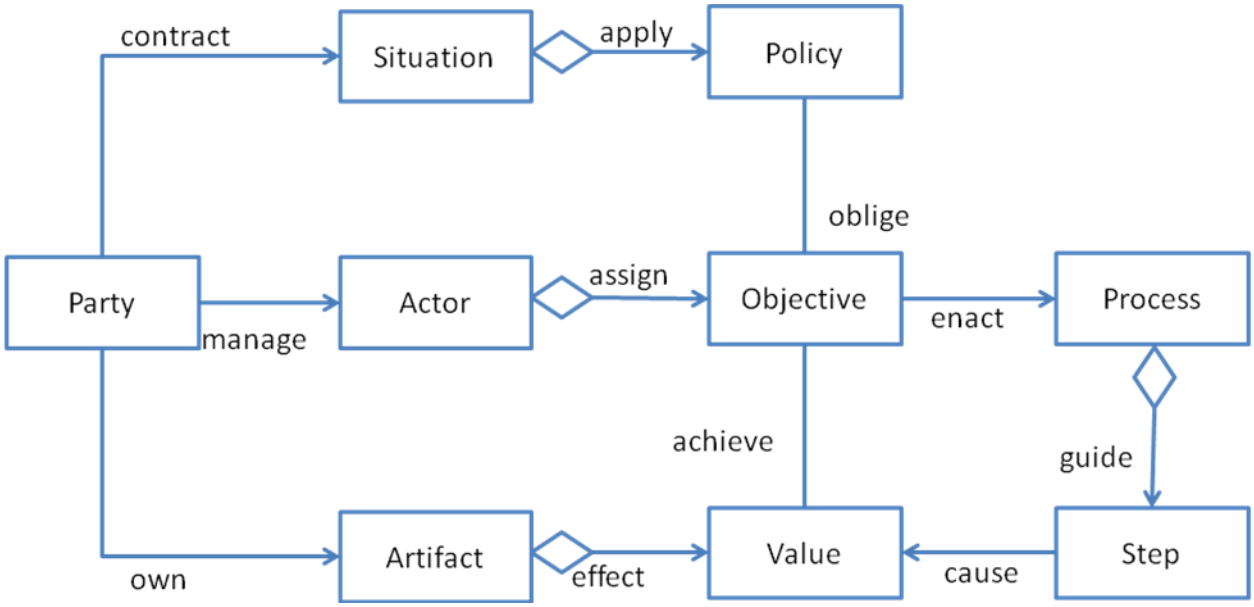
5. Mô hình hệ thống nghiệp vụ

Như đã nhắc đến trong đặt vấn đề bài toán ở phần I.1, trong phân tích thiết kế hướng đối tượng, ta sẽ sử dụng mô hình usecase trong quá trình thiết kế. Tuy nhiên, mô hình usecase chỉ dừng lại ở quá trình phân tích các quy trình nghiệp vụ và phân tích các thực thể. Nếu có quá nhiều ràng buộc hoặc nhiều hoàn cảnh khác nhau, thì làm cách nào để miêu tả những usecase đó? Làm cách nào để biết usecase được nhắc đến trong hoàn cảnh, tình huống nào? Nếu usecase có quá nhiều những luật về nghiệp vụ liên quan, và những luật đó phụ thuốc vào nhiều hoàn cảnh, thì làm cách nào để mô tả hết và đầy đủ?

Và thông thường, để phân tích usecase, ta hay vẽ ra các tình huống riêng bằng sequence diagram, hay activity diagram cho các trường hợp này. Dù vậy, khi tình huống tăng lên quá nhiều, đồng thời còn có thể phát sinh thêm tình huống mới trong quá trình vận hành thì việc phân tích có thể trở nên không đủ.

Do đó, từ những câu hỏi và vấn đề trên, thay vào việc mô tả từng tình huống nhất định trong usecase, ta sẽ mô tả đặc điểm chung của các tình huống bằng hoàn cảnh, chủ yếu là sẽ mô tả điều kiện để hoàn thành các tình huống. Vì thế, với những tình huống và usecase thay đổi liên tục, để có thể mô tả một cách chính xác hơn, ta phát sinh thêm khái niệm Policy và Situation, trong đó Policy sẽ tương ứng với khái niệm điều kiện và Situation sẽ tương ứng với khái niệm hoàn cảnh cụ thể. Với những khái niệm phát sinh ấy, nhóm chọn mô hình Hệ thống nghiệp vụ được giới thiệu trong sách *Hệ thống nghiệp vụ Modeling with UML,* để mô tả cho một bài toán nghiệp vụ bất kỳ nhằm phát triển việc phân tích theo hướng tác tử.

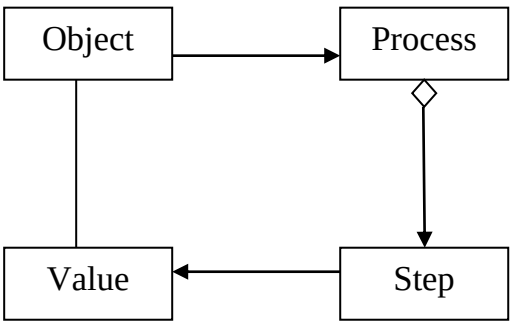
Mục đích chính của mô hình Hệ thống nghiệp vụ là mô tả những giá trị được chia sẻ giữa khách hàng, nhà cung cấp, nhân viên… và định nghĩa vì sao lại tồn tại các mối quan hệ đó. Mô hình còn đưa ra định nghĩa về việc công việc được hoàn thành như thế nào.



Hình 2: Mô hình Hệ thống nghiệp vụ

Với mô hình Hệ thống nghiệp vụ, trong phân tích thiết kế hướng đối tượng chỉ áp dụng được một phần của mô hình, chủ yếu là các khái niệm Object, Process, Step, Value, mà không quan tâm đến các giá trị về Situation và Policy.

Phần dưới của mô hình Hệ thống nghiệp vụ tương ứng với quy trình của mô hình usecase trong phân tích thiết kế hướng đối tượng.



Hình 3: Một phần của mô hình Hệ thống nghiệp vụ

Như ta đã thấy, lập trình hướng đối tượng chỉ đáp ứng một phần của mô hình Hệ thống nghiệp vụ, nhưng với lập trình hướng tác tử, thì chúng ta có thể áp dụng triệt để. Nếu áp dụng vào lập trình hướng tác tử, chúng ta sẽ áp dụng được khái niệm Situation và Policy để hoàn thiện hơn cho quá trình phân tích.

6. Giới thiệu về công cụ Coco/R [1]

Trong quá trình thực hiện luận án, nhóm lựa chọn công cụ Coco/R trong việc hỗ trợ việc xây dựng ngôn ngữ. Coco/R là một công cụ sinh ra trình biên dịch, và sẽ sinh ra hai tập tin tên là Parser và Scanner.

Công việc chính của scanner là chia nhỏ các đoạn mã thành các token để parser có thể sử dụng. Scanner định nghĩa token nào sẽ được gửi cho parser, và đưa những gì không được định nghĩa trong cú pháp ngôn ngữ ra ngoài. Parser được xem như trái tim của trình biên dịch. Parser có các công việc như đảm bảo rằng mã nguồn chương trình phù hợp với cú pháp của ngôn ngữ, và sẽ thông báo lỗi khi không phù hợp.

Coco/R có các phiên bản hỗ trợ cho nhiều ngôn ngữ, như C#, Java, C++, Delphi, Modula-2, Oberon và những ngôn ngữ khác. Đề án của nhóm sẽ sử dụng phiên bản Coco/R dùng cho C#.

Người dùng phải đưa ra một lớp chính, lớp chính này sẽ gọi tập tin Parser, cũng như gọi các lớp ngữ nghĩa (ví dụ như phần sinh ra code).

7. Giới thiệu .NET

Trên cơ sở của những khái niệm về modal logic và đặt nhãn đã được nhắc đến trong phần II.2 và II.3, khi thực hiện luận án, nhóm chọn sử dụng .NET mà không phải một ngôn ngữ khác như Java. Nhóm chọn công cụ là .NET là do .NET có hỗ trợ Custom Attribute. Bản chất của Custom Attribute là một dạng như đánh dấu, hỗ trợ metadata, trong khi Java không hỗ trợ những điều đó. Vì vậy sẽ tốt hơn khi chọn .NET để chuyển nền tảng đặt nhãn vào thực tế.

Custom attribute [5]

Là những thuộc tính được tạo ra vì mục đích riêng của người sử dụng. Thuộc tính là những public class. Mỗi thuộc tính phải có ít nhất một constructor. Các constructor có thể được overload để cho phép thuộc tính được ứng dụng vào các class bằng nhiều cách.

Để tạo một custom attribute, ta phải bắt nguồn từ new custom attribute class từ class System.Attribute.

Phạm vi và mục tiêu của thuộc tính có thể được định nghĩa bằng cách áp ụng AttributeUsage.

Sử dụng Custom Attribute [8]

Việc tạo ra Custom Attribute là một tiến trình đơn giản. Các bước được tiến hành như sau:

- Tạo ra Custom Attribute bởi một lớp bắt nguồn từ System.Attribute

- Nếu cần, xác định các properties gắn liền với một thuộc tính. Một số thuộc tính như các thuộc tính Serializable không có property.
- Định nghĩa một constructor cho thuộc tính có chứa các đối số cần thiết (nếu có) để xây dựng các thuộc tính.
- Tùy chọn, xác định phạm vi của Custom Attribute bằng cách xác định các thuộc tính AttributeUsage

Metadata (Siêu dữ liệu)

Metadata là thông tin về dữ liệu nghĩa là thông tin về kiểu, code, assembly,… mà được lưu trữ cùng với chương trình.

Attibutes là một cơ chế để thêm metadata, chẳng hạn như hướng dẫn trình biên dịch và các dữ liệu khác về dữ liệu, phương thức, và các class tới chương trình riêng của mình. Thuộc tính được đưa vào metadata và có thể nhìn thấy qua *ILDasm* và các công cụ metadata khác.

Reflection là quá trình mà theo đó một chương trình có thể đọc dữ liệu riêng của mình. Một chương trình được cho là để phản ánh về chính nó phải giải nén metadata từ assembly và sử dụng metadata hoặc thông báo cho người dùng hoặc để sửa đổi hành vi của chính mình.

III. Hướng giải quyết bài toán

Sau khi tìm hiểu được những khái niệm được đặt ra trong bài toán thứ nhất, ta sẽ áp dụng những khái niệm đó lần lượt vào hai bài toán tiếp theo, bài toán về hướng phân tích và bài toán xây dựng ngôn ngữ.

1. Hướng tiếp cận bài toán phân tích

Chúng ta có thể thấy rằng, ngày nay, với các phần mềm ngày càng phát triển, hệ thống của chúng sẽ trở nên phức tạp hơn. Khả năng nắm bắt và kiểm soát sự phức tạp đó đi kèm với khả năng trình bày hệ thống một cách toàn diện. Xuất phát từ những bài toán thực tế, các phương pháp phân tích đã ra đời.

Như đã trình bày, phân tích theo hướng tác tử có thể giải quyết những vấn đề mà phân tích theo hướng đối tượng chưa làm được. Và quá trình phân tích của hai hướng tác tử và đối tượng rõ ràng là có sự khác biệt. Như đã chỉ ra trong mô hình Hệ thống nghiệp vụ, phân tích hướng đối tượng trước hết sẽ đi từ mô tả giả định đến usecase. Tuy nhiên, một usecase thường có rất nhiều những luật nghiệp vụ liên quan, mà những luật đó lại phụ thuộc vào hoàn cảnh. Để phân tích bằng usecase, ta hay vẽ ra các tình huống riêng bằng sequence diagrams, hay activity diagrams cho các trường hợp này. Dù vậy, khi các tình huống tăng lên quá nhiều, đồng thời tình huống còn có thể phát sinh thêm trong quá trình vận hành thì việc phân tích có thể trở nên không đủ. Do đó, thay vào việc mô tả từng tình huống nhất định trong usecase, ta sẽ mô tả đặc điểm

chung của các tình huống bằng hoàn cảnh, chủ yếu ở đây là điều kiện để hoàn thành các tình huống.

Tương tự như trong phân tích hướng đối tượng, khi đi theo phân tích hướng tác tử, chúng ta cũng bắt đầu từ mô tả giả định. Dựa vào mô tả giả định đó, chúng ta sẽ xác định các thành phần của bài toán, như Agent, Condition, Context, Role…

1.1. Context

Khái niệm Context

Như chúng ta đã biết, để có thể xem xét chính xác một vấn đề, ta cần đặt nó vào một hoàn cảnh cụ thể (posible word), vì khi hoàn cảnh thay đổi, thì những tương tác của tác nhân với môi trường ngoài xung quanh cũng sẽ thay đổi theo.

Các hoàn cảnh cụ thể thường được xác định bởi một tập hợp các điều kiện.

Với khái niệm về possible word hay hoàn cảnh cụ thể ấy, khi áp dụng vào việc mô tả lý thuyết vào bài toán, chúng ta có khái niệm tương ứng là Context.

Xác định Context

Ta xác định context từ bản mô tả trong lập trình hướng tác tử, bằng cách dựa vào các dòng thông tin lưu hành trong quá trình tương tác giữa tác nhân với hệ thống, ta xác định được những context tồn tại.

1.2. Condition

Khái niệm Condition

Từ khái niệm gán nhãn trong modal logic, ta nhận thấy việc context được gán nhãn sẽ tương ứng với Condition do người dùng định nghĩa. Khái niệm Condtion cũng tương ứng với khái niệm về Policy trong mô hình Hệ thống nghiệp vụ.

Xác định Condition

Từ những quy tắc, ràng buộc trong hệ thống, ta chú ý đến với những quy tắc ấy, ta sẽ có những điều kiện cụ thể tương ứng nào. Condition là tập hợp các điều kiện quy định với điều kiện này thì method sẽ thực thi như thế nào.

1.3. Role

Khái niệm Role

Ta thấy Context sẽ sinh ra usecase, trong usecase có khái niệm về Actor và các mối tương tác giữa Actor. Mặt khác, context lại mô tả tình huống chung của tất cả các

usecase mà actor đóng vai trò một role, từ đó chúng ta sẽ nảy sinh ra khái niệm về Role.

Xác định Role

Bản chất của role cũng là một tập hợp các capability, và role cũng được gán nhãn trong một hoàn cảnh cụ thể. Vì vậy, vấn đề đặt ra là làm sao phân biệt được role và agent trong bài toán.

Để xác định role, ta thường trả lời các câu hỏi sau:

Ai là người có tương tác với hệ thống?

Những người có tương tác đó đóng vai trò như thế nào trong hệ thống?

Còn có vài trò nào tương tác với hệ thống nữa không?

Từ những câu hỏi đó sẽ giúp ta xác định được trong hệ thống có những role nào.

1.4. Agent

Khái niệm Agent

Từ khái niệm về Context và Role, khi đi vào phân tích, chúng ta sẽ sinh ra khái niệm về Agent. Tương tự như phân tích hướng đối tượng có Software Object, trong phân tích hướng tác tử chúng ta có Software Agent, từ đó dẫn đến việc phải lựa chọn mô tả agent như thế nào, xác định có bao nhiêu agent, mỗi agent có thể đóng bao nhiêu role.

Định nghĩa Agent [4]

Một tác tử là một hệ thống máy tính đóng gói mà nó được đặt trong vài môi trường nào đó, và tác tử có khả năng tự trị, linh động trong môi trường mà nó được đặt vào để đáp ứng mục tiêu thiết kế.

Đặc tính của Agent [4]

Khả năng phản ứng: Các agent đều được đặt trong một môi trường cụ thể và có khả năng đáp ứng, nhận thức với những thay đổi từ môi trường bên ngoài một cách kịp thời. Tuy nhiên, theo cách nhìn từ hướng quan hệ và chức năng, nơi mà các chương trình được coi là chức năng từ một trạng thái ban đầu đến trạng thái cuối, thì khả năng phản ứng phải được miêu tả trong các hành động đang diễn ra trong khi tương tác với môi trường.

Tính năng động: Agent có thể áp dụng theo mục tiêu mới và chủ động làm hài lòng các đối tượng của thiết kế.

Khả năng xã hội: Khi được đặt trong một môi trường cụ thể, agent có thể tương tác với các tác tử khác bằng cách hợp tác hay cưỡng ép các hành vi khác để đạt được mục tiêu. Chúng cũng có thể yêu cầu để hiểu và lập luận về các mục đích của những tác tử khác để đưa ra hành động. Sẽ phức tạp hơn nhưng ít được hiểu rõ hơn so với khả năng trao đổi thông tin nhị phân.

Môi trường hoạt động của tác tử thường năng động và không thể đoán trước, vậy nên bản chất và phạm vi của sự tương tác không thể nhìn thấy trước được tại thời gian thiết kế. Agent là khả năng tự giải quyết vấn đề của hệ thống về tính tự trị, linh động và hành vi xã hội, do đó tác tử rất phù hợp cho các hệ thống như vậy.

Từ ví dụ cụ thể, như ví dụ về bài toán ngày nghỉ lễ đã nêu ra trong mục I.1 của phần Giới thiệu bài toán, việc đầu tiên chúng ta sẽ xác định các thành phần bài toán.

Xác định Agent

Bản chất của agent cũng là một tập hợp các capability. Agent sẽ được gán nhãn trong một hoàn cảnh cụ thể. Vì là một tập hợp của các capability, agent cũng được quy định bởi một tập hợp các condition. Tập hợp của các condition này bản chất là các condition của capability.

Để xác định agent, ta chú ý đến những tác nhân cụ thể được nêu ra trong hệ thống, mà các agent này ứng với mỗi role đã xác định trước đó.

Những agent như một ví dụ cụ thể về role.

Xác định agent đó đóng vai trò gì trong hệ thống.

Phân biệt giữa Agent và Role

Vào giai đoạn phân tích, thông thường, trong phân tích hướng đối tượng, ta có tác nhân và hệ thống tương tác lẫn nhau. Khi chuyển qua context, mỗi tác nhân mà hệ thống sẽ phải mô tả lại một phần sẽ trở thành một role. Dựa trên nhóm tương tác giữa các role, ta có thể khái quát hóa chung các capability cho từng role.

Sang giai đoạn thiết kế chúng ta mới xác định các agent, dựa trên cơ sở phân tích role trong nhiều context. Agent là cách chúng ta nhóm các capability trong các role để cài đặt theo cách sao cho dễ thực hiện nhất.

1.5. Xác định Method

Method là phương thức thể hiện sự tác động qua lại giữa các đối tượng, thực thi một hành động nào đó và trả ra kết quả.

Method trong hướng tác tử cũng tương tự như usecase trong phân tích hướng đối tượng, tuy nhiên method này có gắn thêm các điều kiện cụ thể, để khi hoàn cảnh thay đổi thì method và các điều kiện tương ứng sẽ thay đổi theo.

1.6. Xác định Capability

Bản chất của Capability là một tập hợp của các method có những điểm chung trong cùng một hoàn cảnh cụ thể. Trong Capability, Condition quy định từ method hình thành một mối quan hệ method và condition luôn luôn đi chung với nhau.

1.7. Mối quan hệ

Mối quan hệ biểu diễn quan hệ giữa các thành phần trong bài toán.

Để xác định mối quan hệ, ta thường trả lời các câu hỏi sau:

Các thành phần liên hệ với nhau như thế nào?

Mối quan hệ đó được thể hiện ra sao?

Các mối quan hệ tiêu biểu trong lập trình theo hướng tác tử được dựa theo các mối quan hệ trong phân tích hệ thống hướng đối tượng, như mối quan hệ giữa agent và role, giữa agent và context, giữa capability với nhau…

Sau khi xác định được những thành phần bài toán và các mối quan hệ của chúng, tiếp theo chúng ta sẽ chuyển những dữ liệu ấy thành mô hình để có cái nhìn trực quan hơn. Các mô hình trong phân tích hướng tác tử cũng dựa trên những mô hình cơ bản trong phân tích hướng đối tượng, như State Diagram, Activity Diagram, State Transition Diagram, Class Diagram… nhưng được phát triển theo hướng tác tử.

Mô hình di chuyển Agent vào Context

Mô hình này sẽ thể hiện việc di chuyển một agent vào những context thích hợp, và việc di chuyển agent vào context này sẽ phụ thuộc vào từng điều kiện cụ thể. Tùy theo điều kiện phát sinh ra mà ta sẽ biết được phải di chuyên agent đến context nào cho phù hợp.

State Transition Diagram

Là biểu đồ mô tả tất cả trạng thái có thể có, những sự kiện gây ra sự thay đổi trạng thái, điều kiện để sự thay đổi xảy ra, và những hoạt động trong vòng đời của một đối tượng. Mô hình nãy hữu ích cho việc mô tả hành vi của đối tượng cụ thể trong toàn bộ các trường hợp sử dụng có ảnh hưởng đến những đối tượng đó.

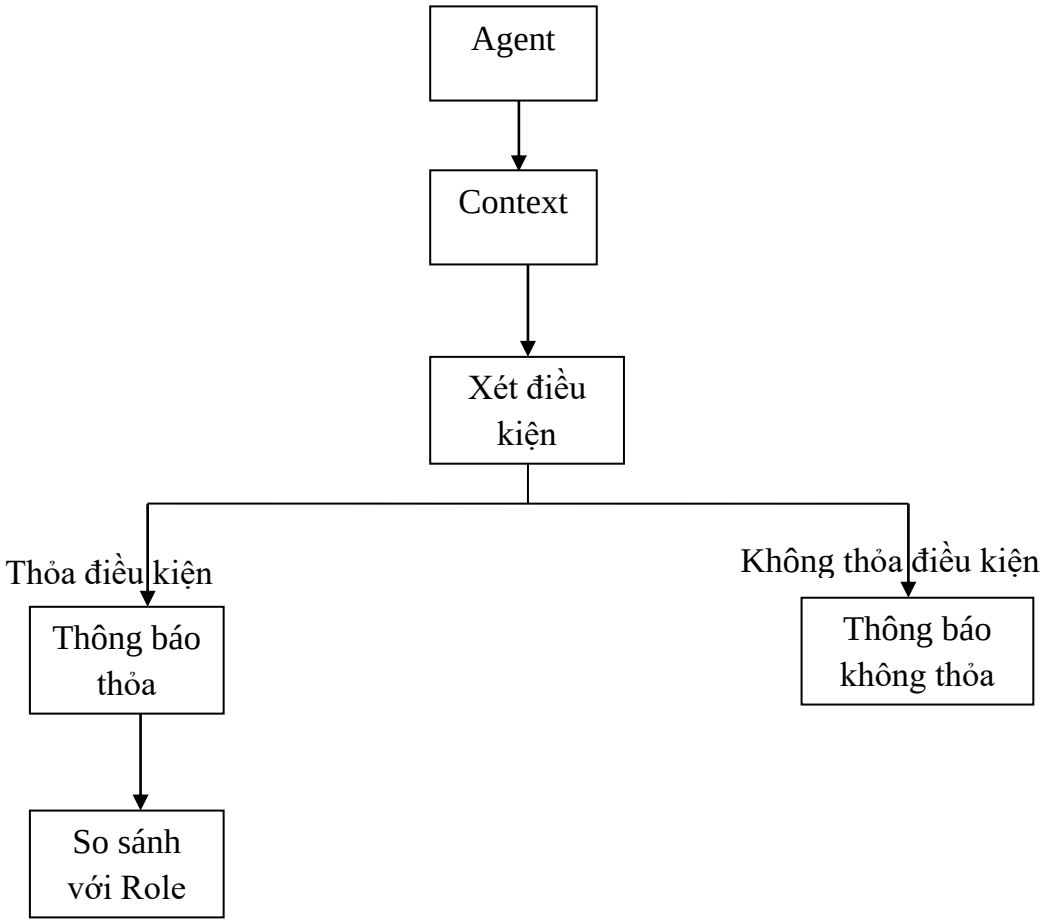
Quá trình phân tích kết thúc sau khi ta thể hiện hoàn tất những mô hình trên.

Sau quá trình phân tích, chúng ta sẽ tiến hành xây dựng ngôn ngữ theo hướng tác tử.

2. Mô hình trong bài toán xây dựng ngôn ngữ

2.1. WorkFlow tương tác giữa Agent và Context

Mô hình bên dưới thể hiện WorkFlow chung trong quá trình tương tác giữa agent và context.

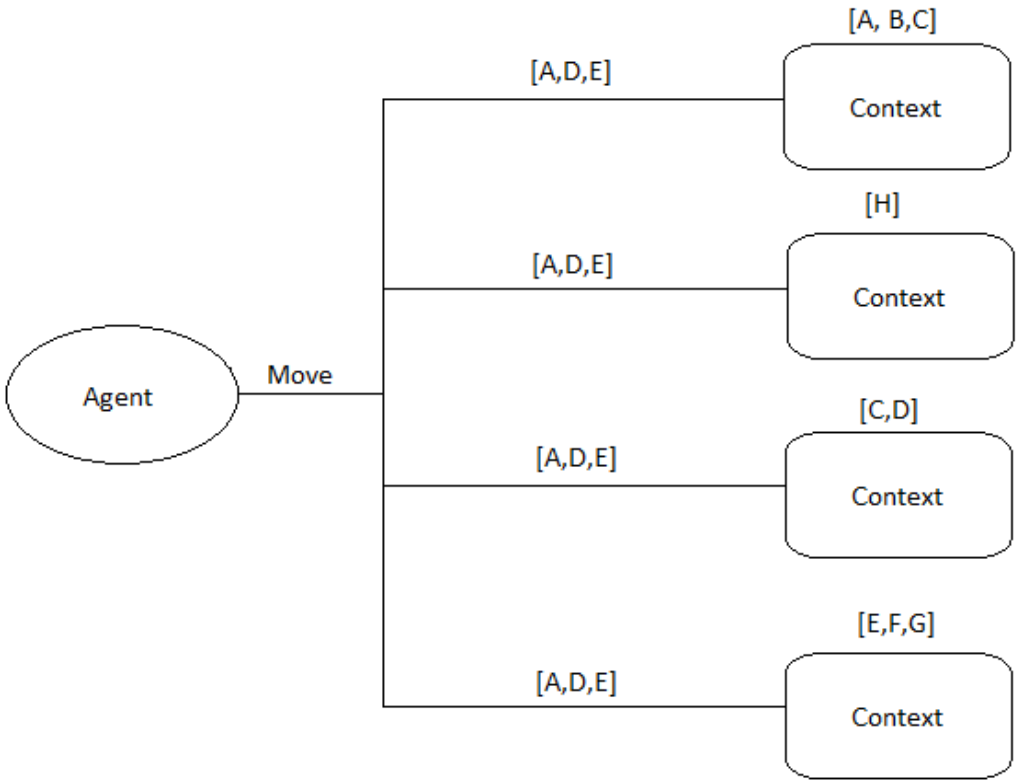


Hình 4: WorkFlow thể hiện dòng quy trình tương tác giữa agent và context

2.2. Chọn lựa Context thích hợp

Khi thực hiện lệnh move, trình biên dịch sẽ so sánh condition kèm theo lệnh move với các condition của các context đang có. Nếu thỏa điều kiện thì sẽ tiến hành move agent vào context, sau đó sẽ tiến hành so sánh điều kiện của agent với từng role trong context để thực hiện việc gán agent vào role.

Mô hình bên dưới diễn tả việc chọn lựa context thích hợp của agent.



[ ] Biểu tượng cho điều kiện

Hình 5: Việc chọn lựa Context thích hợp của Agent

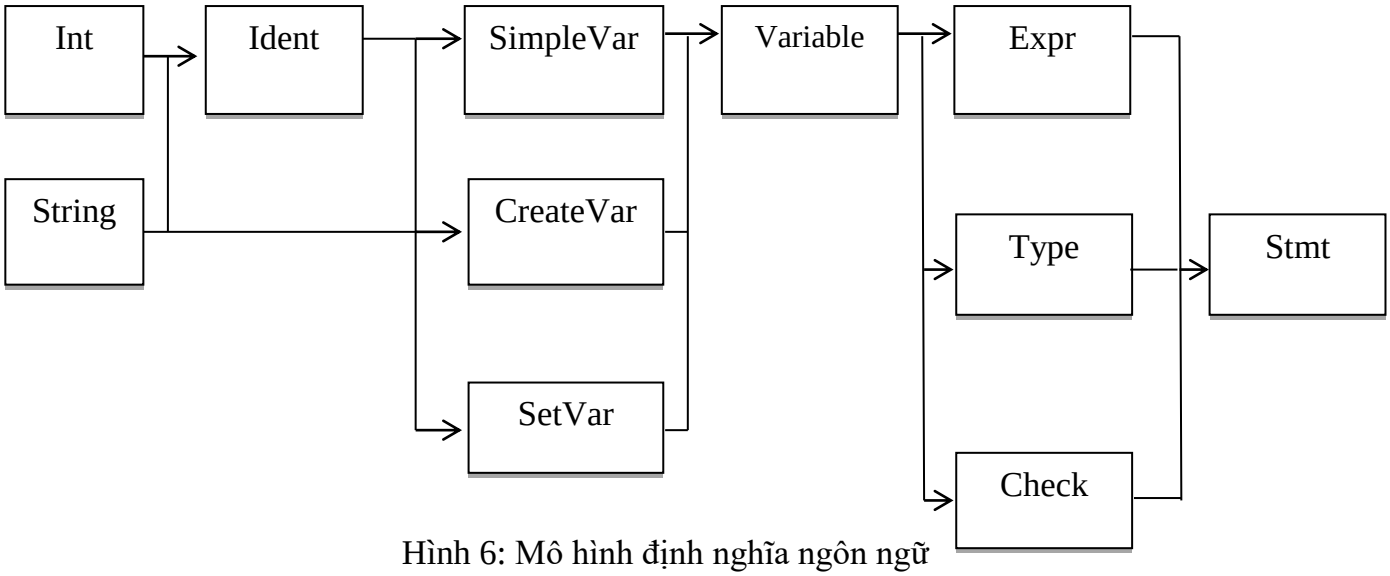
3. Xây dựng ngôn ngữ hướng tác tử

3.1. Mô hình định nghĩa ngôn ngữ

Các ngôn ngữ phần mềm đều bắt đầu từ một mục đích cụ thể, vì vậy các ngôn ngữ máy tính phải rất chính xác để từ đó các lập trình viên có thể mô tả chính xác những gì được yêu cầu, từ đó trình biên dịch có thể hiểu chính xác và tạo ra các mã thực thi chính xác cho những gì được mô tả. Bản thiết kế của của một ngôn ngữ phải cụ thể để loại bỏ tính không rõ ràng trong khi trình biên dịch thực thi.

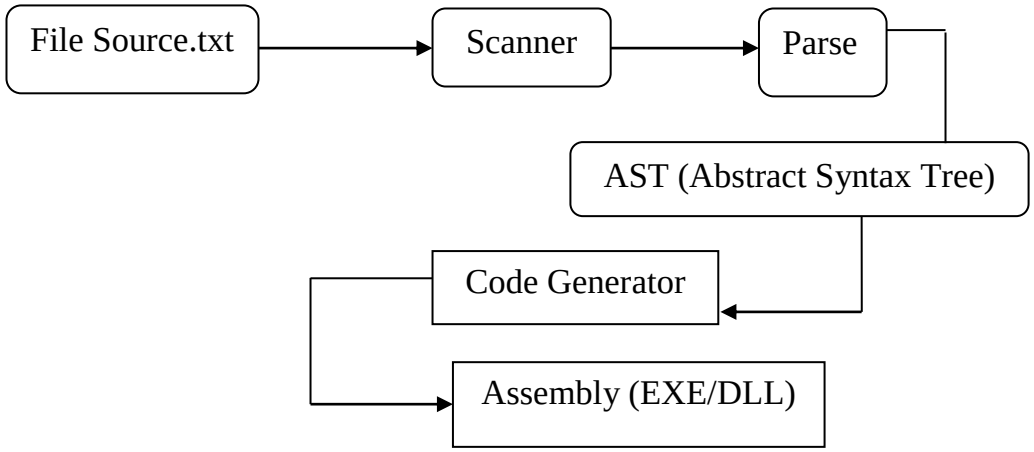
Ngôn ngữ hướng tác tử được viết dưới dạng descripted tương tự như ngôn ngữ SQL. Trong đó, Stmt có thể khai báo, gán điều kiện hoặc chọn các thành phần để in ra màn hình, được cách nhau bởi dấu ‘;’. Expr có thể là chuỗi, số nguyên, các định danh hoặc một tập hợp các giá trị được thiết lập. Ident có thể là tên của của thành phần bao gồm các chữ cái alphabetic làm kí tự đầu tiên, theo sau là số hoặc chữ…

SimpleVar là định nghĩa một tên biến với loại và tên của biến, trong khi CreateVar và SetVar được dùng để khai báo đồng thời tên biến và giá trị của nó. CreateVar và SetVar được dùng phổ biến cho Condition. Check là cơ chế cho phép người dùng định nghĩa chế độ mapping là union hay intersect.



Hình 6: Mô hình định nghĩa ngôn ngữ

3.2. Giai đoạn thực thi



Hình 7: Giai đoạn thực thi của trình biên dịch

Hình trên mô tả các thành phần thực thi các giai đoạn của một trình biên dịch: scanner, parser, code generator. Trong mỗi giai đoạn, ngôn ngữ sẽ được chia nhỏ xuống và các thông tin về dự định của chương trình được phục vụ cho giai đoạn kế tiếp.

Trình biên dịch thường nhóm các giai đoạn lại thành hai phần là front end và back end. Front end bao gồm scanning và parsing, trong khi back end thường chỉ bao gồm code generator. Công việc của front end là khám phá cấu trúc ngữ nghĩa của một chương trình từ text vào trong một đại diện cho bộ nhớ cao cấp, gọi là Abstract Syntax Tree (AST). Back end có nhiệm vụ lấy AST và chuyển đổi nó thành thứ có thể thực thi bằng máy.

Ba giai đoạn thường được chia thành front end và back end bởi do scanner và parser thường đi chung với nhau, trong khi code generator thì thường gắn chặt với một target platform. Thiết kế này cho phép một nhà phát triển có thể thay thế code generator cho những platform khác nếu ngôn ngữ cần là cross-platform.

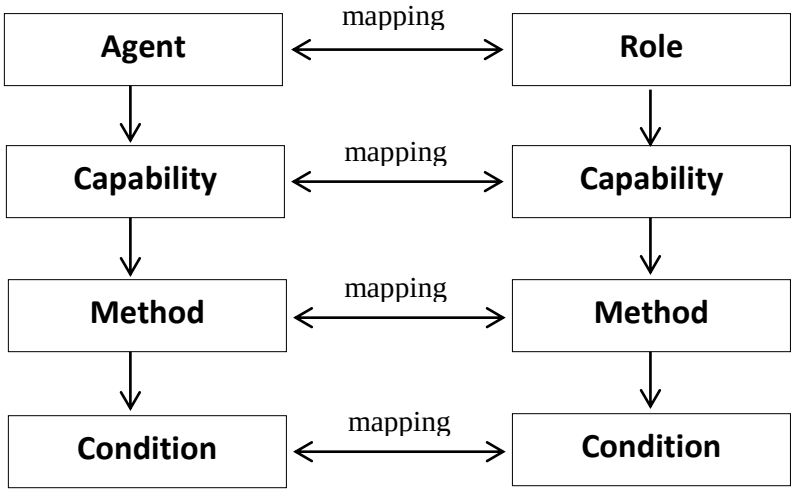
3.3. Cơ chế mapping

Như chúng ta đã biết, các thành phần trong hệ thống hướng tác tử vốn là các tổ hợp đi từ thấp đến cao. Vì vậy, khi thực hiện việc so sánh giữa agent và role để gán agent và role vào context nhất định, ta cũng thực hiện tương tự.

Có thể nói bản chất của việc mapping giữa agent và role là so sánh các cặp condition với nhau. Có hai cơ chế để so sánh là cơ chế so sánh union (hợp) và intersect (giao). Việc chọn hình thức so sánh nào sẽ được lập trình viên quyết định khi họ tạo role, capability hay method.

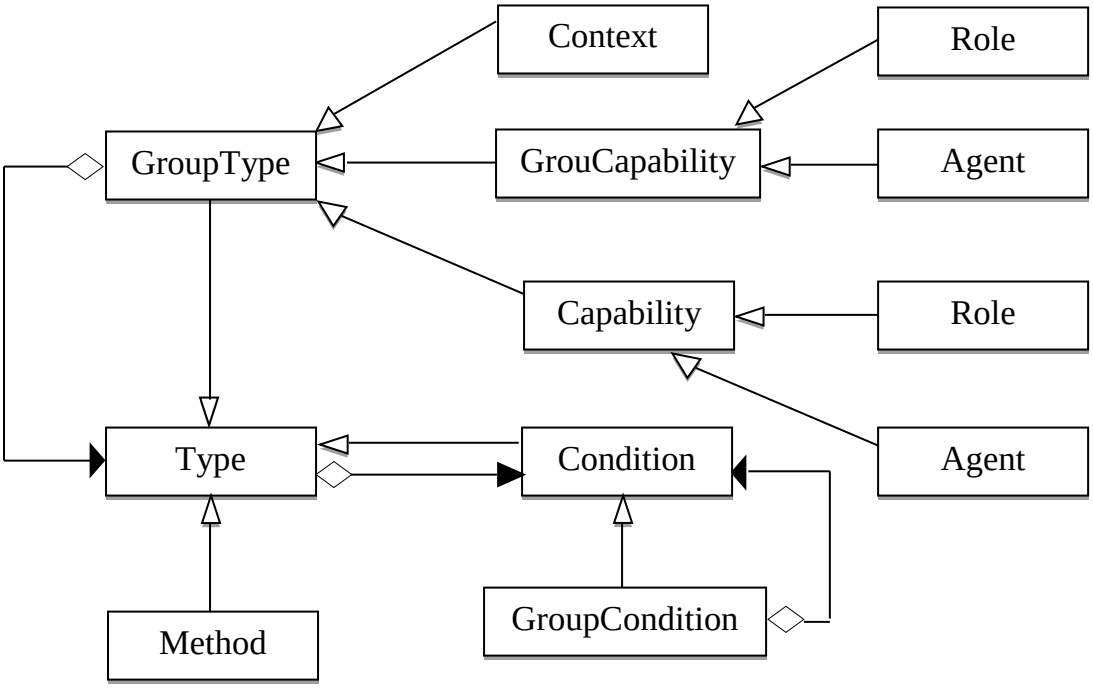
Cơ chế của so sánh union tương tự như cách thực hiện phép hợp trong toán học. Để so sánh, ta chỉ cần một giá trị hay điều kiện, method hoặc capability thỏa với điều kiện thì giá trị sẽ được trả về true. Cơ chế so sánh intersect là việc thực hiện phép giao trong tập hợp và đòi hỏi toàn bộ các điều kiện, method hoặc capaility thỏa, khi đó giá trị mới được trả về true.

Để làm rõ hơn về hai cơ chế so sánh, ta sẽ lấy một ví dụ cụ thể. Giả sử như ta có method A gồm các condition là C1, C2 và C3. Method B gồm các condition là C1, C4, C5. Vậy, khi so sánh Method A và Method B, nếu áp dụng cơ chế union thì giá trị sẽ trả về là true. Ngược lại, khi áp dụng so sánh với cơ chế là intersect thì giá trị trả về là false. Nếu muốn giá trị là true khi so sánh bằng cơ chế intersect thì Method B phải gồm có C1, C2, C3.



Hình 8: Cơ chế Mapping

3.4. Mô hình thư viện



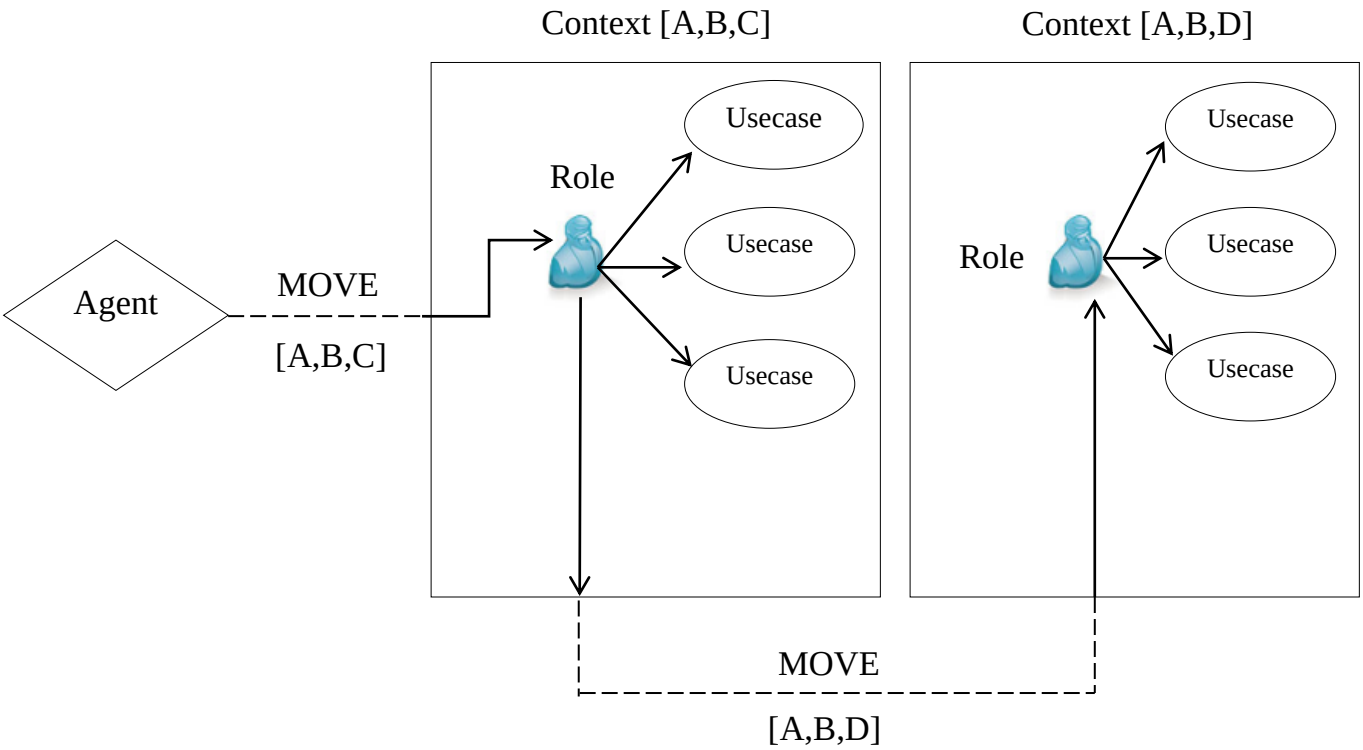
Hình 9: Mô hình thư viện bên dưới

Các thành phần chính trong hệ thống hướng tác tử gồm có Context, Role, Agent, Capability, Method và Condition. Trong đó Method có nhiều Condition, Capability có nhiều Method, Role hay Agent có nhiều các Capability và Context thì chứa một tập hợp Role trong nó.

Với mối quan hệ như vậy, ta có thể thấy các thành phần trong hệ thống hướng tác tử thực chất là một tổ hợp đi từ thấp lên cao. Trong đó, thành phần của hướng tác tử có thể tương tác với nhau và tổ hợp lại để thành một loại cao hơn.

Khi tiến hành di chuyển một Agent, chương trình sẽ tiến hành so sánh với điều kiện của Context để xem Agent có phù hợp với Context đó hay không. Điều này có thể dẫn đến trường hợp một Agent thỏa điều kiện của hai hay nhiều Context cùng một lúc. Tương tự, khi gán Agent vào Role cũng có thể xảy ra trường hợp một Agent có thể gán vào nhiều Role, hay nhiều Agent gán vào một Role. Khi mối quan hệ trở thành n-n thì mọi việc sẽ trở nên rất khó kiểm soát, cho nên hiện tại phạm vi bài toán của nhóm sẽ giới hạn lại việc xử lý chỉ là 1-1, nghĩa là một Agent thỏa một Context và thỏa một Role trong đó (nếu có). Việc giải quyết quan hệ n-n sẽ được nói rõ hơn trong phần phương hướng phát triển.

3.5. Mô hình di chuyển Agent vào Context



Hình 10: Mô hình di chuyển Agent vào Context

3.6. Cú pháp ngôn ngữ

Cũng tương tự như các ngôn ngữ khác, ngôn ngữ theo hướng tác tử cũng được xây dựng dựa theo những cú pháp nhất định.

Sau đây là cú pháp của ngôn ngữ do nhóm xây dựng

Type: Agent, Capability, Conditions, Context, Role, Object, Class.

Keywords: Create, Delete, Bind, Update, Remove, Alter, Insert, Select, Move, Interface.

Combine Keywords: Into, Drop, Set, From, Values, Add.

Bảng 2: Bảng cú pháp ngôn ngữ

Lệnh	Mô tả	Ràng buộc	Combine keyword
Create	Dùng để tạo mới một đối tượng, như context, agent, condition, method… Optional: check intersect,		Values

	check union		
Delete	Dùng để xóa một đối tượng đã tồn tại, như context, agent, condition, method…	Phần tử muốn delete phải không có quan hệ với phần tử khác	
Update	Dùng để cập nhật một đối tượng đã có sẵn, như condition, method…		Values
Remove	Dùng để đưa một đối tượng ra khỏi đối tượng khác.		From
Alter	Dùng để thay đổi một Condition, thay đỗi các field trong Condition.		Drop, Add
Select	Dùng để chọn một ra một đối tượng nào đó.		
Move	Dùng để di chuyển từ context này sang context khác		With
Bind	Dùng để ràng condition với method, method với capability, capability với role hay agent, role hay agent với context.		Into
Interact	Dùng để thực hiện việc kết nối giữa hai role cùng context	Hai role đều phải được gán bằng agent	With

ỨNG DỤNG VÀO BÀI TOÁN CỤ THỂ

Với những lý thuyết đã nêu, ta sẽ áp dụng vào những bài toán cụ thể, gồm hai bài toán được nêu ra trong I.1 mục Giới thiệu bài toán là bài toán về Lời chào, bài toán Ngày nghỉ lễ, và bài toán áp dụng chính của nhóm là bài toán Đăng ký môn học.

I. Bài toán Lời chào

1. Hoàn cảnh phát sinh bài toán

Như đã phân tích trong phần đặt vấn đề bài toán, khi áp dụng câu chào 'Hello' trong tiếng Anh vào tiếng Việt, chữ 'Hello' sẽ không chỉ đơn thuần là 'Xin chào', mà sẽ được gắn với một câu chào nhất định từ một hoàn cảnh cụ thể, gồm 'Xin chào' + chức danh, như chào anh, chào chị… (xem thêm I.1 trong mục Giới thiệu bài toán)

Trong bài toán Lời chào, xét một usecase cụ thể, usecase 'Kiểu chào'. Ta hãy thử xét các quy tắc, điều kiện và tình huống tạo nên usecase này.

Trước hết, chúng ta sẽ xét những policy có trong quá trình áp dụng câu chào. Quy tắc đầu tiên là về *giới tính*.

Do tính chất của việc áp dụng câu chào theo giới tinh, ta sẽ có điều kiện về *câu chào áp dụng cho nam* và *câu chào áp dụng cho nữ*. Vậy nên usecase kiểu chào phải phân rã thành hai usecase kiểu chào cho nam, và kiểu chào cho nữ.

Xét kiểu chào áp dụng cho giới tính *nam*

Trong việc áp dụng kiểu chào cho giới tính là nam, ta xét policy đầu tiên là *Tuổi tác*, trong policy tuổi tác ta có các điền kiện cho tuổi như tuổi lớn hơn, nhỏ hơn hay bằng tuổi người chào.

Tiếp đến ra xét Policy về *Quan hệ* giữa những người chào nhau. Trong policy quan hệ ta có các điều kiện như họ hàng trong gia đình, người mới quen trong công việc, người chưa quen biết...

Sau khi xét tất cả các policy tiêu biểu cho kiểu chào áo dụng cho giới tính nam, ta sẽ xét tiếp đến kiểu chào dành cho nữ. Việc áp dụng câu chào cho nữ có policy tương tự cho nam cũng gồm tuổi tác, quan hệ gia đình.

Tóm lại ta có:

Giới tính: nam và nữ

Tuổi tác: lớn hơn hay nhỏ hơn hoặc bằng người đưa ra câu chào.

Quan hệ: họ hàng trong gia đình, người mới quen trong công việc, người chưa quen biết...

Như vậy việc tính toán áp dụng câu chào sao cho hợp lý rất phức tạp, một câu chào có thể thay đổi thành nhiều dạng khác nhau tùy theo hoàn cảnh nhất định. Ví dụ: A gặp một người giới tính nam, mới quen chưa biết tuổi của người đó. Khi đó, việc áp dụng câu chào sẽ dựa trên sự linh hoạt, A có thể dựa trên ngoại hình, vóc dáng để phân vùng câu chào là chào anh hay chào em hay chào bác… Giới tính nam của người được chào còn giúp A xác định rõ là phải chào anh hay chào em, chào bác… chứ không được chào chị hay chào cô. Từ những sự linh hoạt đó, nếu áp dụng phân tích theo hệ thống theo hướng đối tượng sẽ gặp rất nhiều khó khăn, vì usecase cho trường hợp này là vô tận và luôn luôn biến đổi theo hoàn cảnh. Vì thế, ta sẽ gom nhóm các điều kiện và hình thành nên một tổ hợp các điều kiện xác định câu chào sao cho hợp lý nhất. Đó là cách giải quyết theo hướng tác tử.

2. Giải quyết bài toán theo hướng tác tử

2.1. Các thành phần bài toán

Do sự linh hoạt tùy hoàn cảnh của bài toán Lời chào, để giảm bớt các trường hợp phân rã nhỏ của usecase kiểu chào, ta sẽ hình thành các điều kiện dựa trên policy đã phân tích, điều kiện được chia theo kiểu chào cho giới tính nam và kiểu chào cho giới tính nữ.

Tập hợp Condition

Dựa trên những Policy đã được mô tả, ta sẽ có được tập hợp những condition, gồm

Giới tính = [nam, nữ]

Quan hệ = [bà cháu, người lạ, người quen, đồng nghiệp]

Tuổi tác = [già, trẻ]

Ngoài ra, việc xác định các Condition trong bài toán Lời chào còn có thể dựa trên một vài condition khác bổ sung thêm, như

Địa điểm = [gia đình, trường học, đường phố]

Dựa vào những condition trên, ta sẽ lấy khoảng 10 tổ hợp condition tiêu biểu

Condition 1 = [nam, người lạ, trẻ]

Condition 2 = [nữ, người lạ, trẻ, đường phố]

Condition 3 = [nam, người quen, lớn tuổi]

Condition 4 = [nữ, bà cháu, gia đình]

Condition 5 = [nam, đồng nghiệp, trẻ]

Xác định Method

Dựa vào những tập hợp những Condition đã có, ta sẽ xác định được Method cho mỗi Condition. Ta sẽ lấy ví dụ một Condition sẽ có hai Method tương ứng.

Condition 1

Method 1 = Chào anh; Method 2 = Chào em.

Condition 2

Method 3 = Chào chị; Method 4 = Chào bạn.

Condition 3

Method 5 = Chào bác; Method 6 = Chào chú.

Condition 4

Method 7 = Chào bà; Method 8 = Chào cụ.

Condition 5

Method 9 = Chào anh; Method 10 = Chào cậu.

Xác định Context

Với các method đã được xác định, ta nhận thấy rằng trong bản thân mỗi method sẽ có điều kiện riêng của nó. Chẳng hạn như với *Method 1* Chào anh, ta còn phải xét xem lời chào này diễn ra trong hoàn cảnh nào.

Như vậy, từ những method đã xác định được, ta nhận thấy những tập hợp Condition đã xét chỉ là những điều kiện chung nhất, vì khi xét vào một lời chào cụ thể, lời chào được xét còn phải phụ thuộc vào hoàn cảnh phát sinh tình huống đó. Từ đó, dễ thấy rằng những Condition tiêu biểu chẳng qua sẽ là các Context trong bài toán Lời chào. Khi đó, những method đã xác định sẽ là những method tương ứng trong mỗi context cụ thể, và mỗi method sẽ có Condition riêng biệt của chính nó.

Tiếp tục phân tích, nhóm sẽ chọn ra hai context tiêu biểu.

Context 1 = Trong trường học

Context 2 = Trên đường phố

Xác định Role

Khi phân tích theo hướng đối tượng, trong mô hình usecase ta sẽ có Actor và System tương tác với nhau. Khi chuyển qua Context, mỗi Actor mà System phải mô tả lại một

phần sẽ trở thành một Role. Trong bài toán Lời chào ta xác định Actor là người chào và được thực hiện trong một hoàn cảnh nhất định, như vậy Role người chào trong một hoàn cảnh nhất định như trường học, hay Role người chào trên đường phố.

Xác định Agent

Trên cơ sở phân tích Role trong nhiều Context, ta sẽ xác định được Agent. Agent là cách chúng ta nhóm các capability trong các role để cài đặt theo cách sao cho dễ thực hiện nhất. Vì vậy, với Role là Người chào trong một địa điểm, ta sẽ có Agent là một người chào cụ thể.

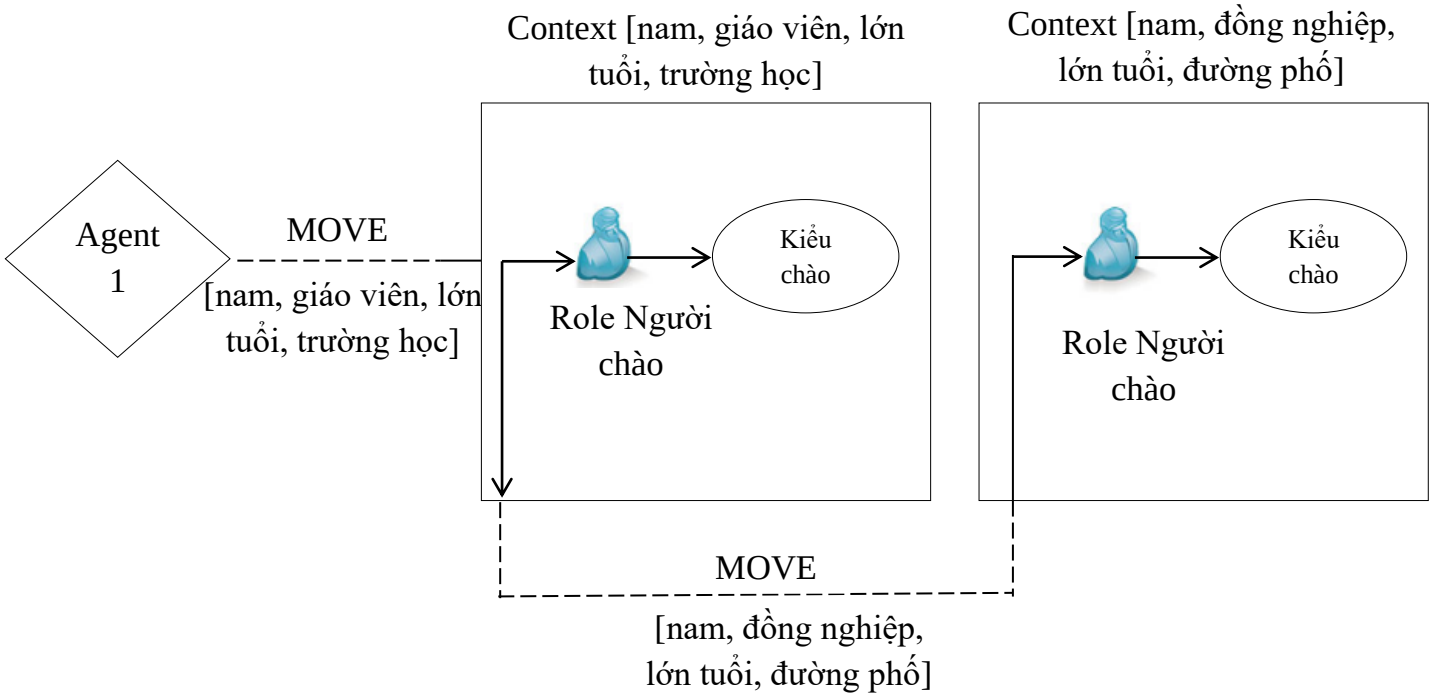
Ta sẽ xây dựng ba Agent cụ thể như sau:

Agent 1: Trần Văn A [tuổi: 21, sinh viên]

Agent 2: Trần Văn B [tuổi: 51, giảng viên]

Agent 3: Trần Văn C [tuổi: 38, công nhân viên]

2.2. Mô hình di chuyển Agent vào Context



Hình 11: Mô hình di chuyển Agent vào Context trong bài toán Lời chào

II. Bài toán Ngày nghỉ lễ

1. Hoàn cảnh phát sinh bài toán

Như đã nếu ra trong mục I.1 phần Giới thiệu bài toán, với bài toán ngày nghỉ lễ, ta giả sử hôm nay là ngày nghỉ đối với các sinh viên trường Đại học Hoa Sen. Trong ngày nghỉ lễ ấy, tuy đều là sinh viên nhưng mỗi người cụ thể lại có những việc làm khác nhau. Chẳng hạn như có người mua sắm, có người ở nhà, có người xem phim.

2. Giải quyết bài toán theo hướng tác tử

2.1. Các thành phần của bài toán

Trong hướng phân tích này, ký hiệu [] sẽ biểu hiện cho Condition.

Tập hợp những Condition

Sở thích = [cắm trại, du lịch, mua sắm, xem phim, dạo phố, đi bơi, yên tĩnh]

Phương tiện = [xe máy, xe đạp, xe buýt, xe đò, xe taxi]

Địa điểm = [khu trung tâm, khu ngoại ô]

Dựa vào những condition đó, ta sẽ lấy tổ hợp khoảng 5 condition tiêu biểu

Condition 1 = [cắm trại, khu ngoại ô, xe máy]

Condition 2 = [mua sắm, khu trung tâm, xe máy]

Condition 3 = [xem phim, xe máy]

Condition 4 = [dạo phố, xe đạp]

Condition 5 = [yên tĩnh, xe đò]

Xác định Method

Dựa vào những tập hợp những Condition đã có, ta sẽ xác định được Method cho mỗi Condition. Ta sẽ lấy ví dụ một Condition sẽ có hai Method tương ứng.

Condition 1

Method 1 = Picnic; Method 2 = Leo núi.

Condition 2

Method 3 = Shopping ; Method 4 = Đi nhà sách.

Condition 3

Method 5 = Xem phim; Method 6 = Xem kịch.

Condition 4

Method 7 = Đi công viên; Method 8 = Dạo phố.

Condition 5

Method 9 = Về quê; Method 10 = Đi café sách.

Xác định Context

Với các method đã được xác định, ta nhận thấy rằng trong bản thân mỗi method sẽ có điều kiện riêng của nó. Chẳng hạn như với *Method 1* Đi picnic, ta sẽ có các điều kiện riêng của nó như thích đi cắm trại, với phương tiện là xe máy và địa điểm là những khu ngoại ô thành phố.

Như vậy, từ những method đã xác định được, ta nhận thấy những tập hợp Condition đã xét chỉ là những điều kiện chung nhất, vì khi xét vào hoàn cảnh cụ thể, kế hoạch ngày nghĩ còn được xét phải phụ thuộc vào sở thích. Từ đó, dễ thấy rằng những Condition tiêu biểu chẳng qua sẽ là các Context trong bài toán Ngày nghỉ lễ. Khi đó, những method đã xác định sẽ là những method tương ứng trong mỗi context cụ thể, và mỗi method sẽ có Condition riêng biệt của chính nó.

Tiếp tục phân tích, nhóm có một context tiêu biểu.

Context 1 = Ngày nghỉ lễ

Xây dựng Role

Với những Context, Condition và Method đã phân tích như trên, tiếp theo chúng ta sẽ xác định Role cho bài toán.

Trong bài toán nghỉ lễ, Role được xác định là Sinh viên

Xây dựng Agent

Trên cơ sở phân tích Role trong nhiều Context, ta sẽ xác định được Agent. Agent là cách chúng ta nhóm các capability trong các role để cài đặt theo cách sao cho dễ thực hiện nhất. Vì vậy, với Role là Sinh viên, ta sẽ có Agent là một sinh viên cụ thể để gán vào Role.

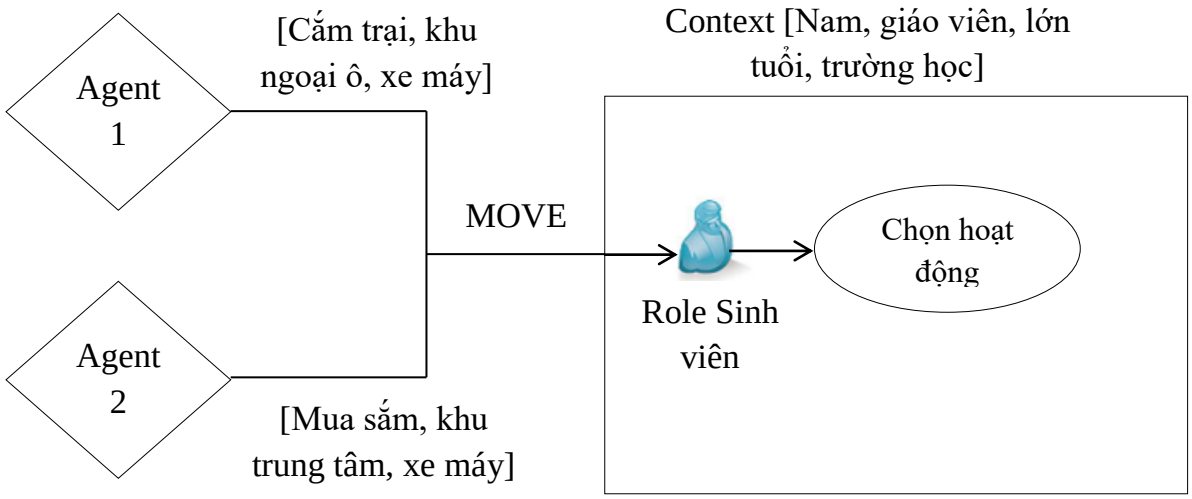
Ta sẽ xây dựng ba Agent cụ thể

Agent 1: Sinh viên A [cắm trại, khu ngoại ô, xe máy]

Agent 2: Sinh viên B [mua sắm, khu trung tâm, xe máy]

Agent 3: Sinh viên B [xem phim, xe máy]

2.2. Mô hình di chuyển Agent



Hình 12: Mô hình di chuyển Agent trong bài toán Ngày nghỉ lễ

III. Bài toán Đăng ký môn học

1. Giới thiệu bài toán cụ thể

Trước khi bắt đầu mỗi học kỳ mới, nhà trường sẽ tổ chức cho sinh viên đăng ký môn học trực tuyến. Việc đăng ký môn học cho phép sinh viên có thể tự chọn môn học và lớp môn học phù hợp với thời gian biểu cá nhân và nguyện vọng của mình. Quá trình chọn môn học là một quá trình phải xét rất nhiều điều kiện cũng như nảy sinh ra rất nhiều tình huống, hoàn cảnh khác nhau. Chủ nhiệm chương trình thông qua Chương trình đào tạo sẽ giải quyết những tình huống và hoàn cảnh xảy ra trong quá trình chọn môn học của sinh viên.

Với phân tích hướng đối tượng, giai đoạn đầu tiên của phân tích thiết kế chính là mô tả usecase. Với bài toán đăng ký môn học, vì các tình huống rất đa dạng và không cố định, để mô tả usecase một cách đầy đủ và chính xác, ta phải nắm bắt được các mối tương tác liên kết tương đối chặt chẽ, từ đó khoanh vùng lại thành một usecase mô tả cố định, còn sự linh hoạt là để actor lựa chọn. Sau đó các tình huống thay đổi còn lại sẽ được mô tả trong các tình huống khác nhau.

Tuy nhiên, nếu các tình huống xảy ra quá nhiều, các điều kiện cần mô tả để sinh ra usecase quá lớn, thì khi tổ hợp các điều kiện lại sẽ ra số usecase và tình huống cần viết là một con số rất lớn, khi ấy bằng cách nào để mô tả hết? Hơn nữa, trong mỗi tình huống, usecase có thể tạo ra trình tự không biết trước.

Trong bài toán Đăng ký môn học, xét một usecase cụ thể, usecase ‘Chọn môn học’. Ta hãy thử xét các quy tắc, điều kiện và tình huống tạo nên usecase này.

Trước hết, chúng ta sẽ xét những policy có trong quá trình đăng ký môn học. Quy tắc đầu tiên là về *Tính chất môn học*.

Do tính chất môn học, ta sẽ có điều kiện về *môn học bắt buộc* và *môn học tự do*. Vậy nên usecase Chọn môn học phải phân rã thành hai usecase Đăng ký môn học bắt buộc theo chương trình đào tạo và Đăng ký môn học tự do.

Tiếp đến, xét đăng ký môn học theo chương trình đào tạo.

Trong đăng ký môn học theo chương trình đào tạo, policy đầu tiên ta phải xét đến là *Quan hệ giữa các môn học*. Trong policy về Quan hệ giữa các môn học, ta có điều kiện về môn tiên quyết, môn song hành, môn tương đương, môn thay thế. Ngoài ra, ta còn phải xét policy về *Chương trình đào tạo*, để xét xem môn học cần đăng ký nằm trong chương trình đào tạo nào, do với từng khóa đào tạo khác nhau ta lại có vài môn học khác nhau với yêu cầu về quan hệ môn học cũng khác nhau. Trong policy về Chương trình đào tạo, ta sẽ có giá trị điều kiện là Chương trình đào tạo khóa *06, 07, 08, 09*.

Sau đó, ta phải xét tiếp policy về *Hệ*. Hệ: 5 hệ gồm đại học, cao đẳng, liên thông, trung cấp chuyên nghiệp, kỹ thuật viên.

Sau khi đã xét xong các policy về Đăng ký môn học theo chương trình đào tạo, ta sẽ xét tiếp đến các policy của Đăng ký môn học tự do.

Trong Đăng ký môn học tự do, ta cũng xét tương tự như với Đăng ký môn theo chương trình đào tạo, với các Policy gồm *Quan hệ giữa các môn học, Hệ, Khóa, Học phí*. Tuy nhiên, trong Đăng ký môn tự do, ta còn phải xét thêm Policy liên quan nữa là *Ngành*.

Tóm lại, ta có

Tính chất môn học: môn bắt buộc và môn tự do.

Quan hệ giữa môn học: 4 quan hệ gồm môn tiên quyết, môn song hành, môn tương đương, môn thay thế.

Ngành: 43 ngành gồm Đại học: 13 ngành, Cao đẳng: 7 ngành, Trung cấp chuyên nghiệp: 3 ngành, Kỹ thuật viên: 7 ngành, liên thông 13 ngành.

Hệ: 5 hệ gồm đại học, cao đẳng, liên thông, trung cấp chuyên nghiệp, kỹ thuật viên.

Chương trình đào tạo: 4 khóa 06, 07, 08, 09.

Đăng ký Môn học = Đăng ký môn học theo CTDT + Đăng ký môn hoc tự do (1)

Đăng ký theo CTDT = Quan hệ môn học * Hệ * Khóa (2)

Đăng ký Tự do = Ngành + Quan hệ môn học * Hệ * Khóa (3)

Từ (2) (3) ta thấy:

Đăng ký môn học = (4*5*4) + (43 +4*5*4) = 80+ 123 = 203 (Usecase)

Như đã phân tích, nếu ta giải quyết bài toán theo hướng phân tích đối tượng, thì số tổ hợp usecase là một con số vô cùng lớn.

Ứng với mỗi Usecase ta có các mô hình đi kèm theo để phân tích rõ các bước di chuyển của người dùng với hệ thống. Việc có quá nhiều các thay đổi giữa các bước di chuyển qua lại của các usecase hình thành nên một mạng lưới di chuyển phức tạp gây khó khăn và vô cùng phức tạp để xác định sửa chữa khi có lỗi cũng như dễ gây xung đột khi chương trình có nhiều sinh viên vào cùng một lúc như đăng ký môn học.

Vì thế để giảm bớt sự phức tạp chồng chéo lên nhau của các bước di chuyển Usecase, ta sẽ áp dụng theo hướng mới, gom những bước di chuyển quan trọng nhất, có quan hệ chặt chẽ với nhau như Chọn ngành phải luôn luôn chọn trước chọn lớp, phải chọn môn học mới được chọn thời khóa biểu... Từ đó ta hình thành những điều kiện, và hoàn cảnh chung cho quá trình di chuyển. Cách gom nhóm và hình thành các điều kiện chính là ta đã đi phân tích bài toán theo một hướng mới, hướng tác tử.

2. Giải quyết bài toán theo hướng tác tử

2.1. Các thành phần của bài toán

Từ những Policy đã nêu ra, ta nhận thấy rằng ứng với mỗi Usecase ta có các mô hình đi kèm theo để phân tích rõ các bước di chuyển của người dùng với hệ thống. Việc có quá nhiều các Policy dẫn đến việc để có thể mô tả đầy đủ những quy tắc và ràng buộc đòi hỏi phải phân rã thành rất nhiều usecase. Với con số tổ hợp quá lớn, việc phân tích và mô tả sẽ trở nên rất phức tạp, rườm rà.

Vì thế để giảm bớt sự phức tạp của các bước di chuyển Usecase, từ các mối quan hệ đã phân tích trước đó, ta sẽ hình thành các điều kiện chọn môn học thay cho việc phân rã usecase. Việc hình thành điều kiện ban đầu được chia theo chọn môn học tự do hay theo chương trình đào tạo.

Trong hướng phân tích này, ký hiệu [] sẽ biểu hiện cho Condition.

Tập hợp Condition dựa trên những Policy đã được mô tả

Hệ = [Đại học, Cao đẳng, Trung cấp chuyên nghiệp, Kỹ thuật viên, Liên thông]

Khóa = [06, 07, 08, 09]

Ngành = [Mạng máy tính, Công nghệ thông tin, Quản trị kinh doanh, Quản trị nhân lực, Marketing, Quản trị Du lịch và Khách sạn Nhà hàng, Tài chính ngân hàng, Tiếng

Anh, Toán ứng dụng trong tài chính, Thống kê và tính toán khoa học, Quản lý công nghệ môi trường, Thiết kế thời trang, Kế toán]

Dựa vào những Condition trên, ta sẽ lấy khoảng 10 tập tổ hợp Condition tiêu biểu

Condition 1 = [Đại học, 06, Mạng máy tính]

Condition 2 = [Đại học, 08, Tài chính ngân hàng]

Condition 3 = [Cao đẳng, 07, Công nghệ thông tin]

Condition 4 = [Trung cấp chuyên nghiệp, 07, Mạng máy tính]

Condition 5 = [Kỹ thuật viên, 09, Tiếng Anh]

Condition 6 = [Đại học, 06, Công nghệ thông tin]

Condition 7 = [Cao đẳng, 08, Quản trị nhân lực]

Condition 8 = [Trung cấp chuyên nghiệp, 07, Kế toán]

Condition 9 = [Cao đẳng, 09, Thiết kế thời trang]

Condition 10 = [Liên thông, 06, Tiếng Anh]

Xác định Method

Dựa vào những tập hợp những Condition đã có, ta sẽ xác định được Method cho mỗi Condition. Ta sẽ lấy ví dụ một Condition sẽ có ba Method tương ứng.

Condition 1

Method 1 = Kỹ thuật truyền số liệu; Method 2 = Cơ sở an ninh mạng; Method 3 = PTTK hệ thống mạng.

Condition 2

Method 4 = Toán vi mô; Method 5 = Nhập môn tài chính; Method 6 = Tài chính nâng cao.

Condition 3

Method 7 = Cấu trúc dữ liệu và giải thuật; Method 8 = Lập trình ứng dụng; Method 9 = Nhập môn công nghệ phần mềm.

Condition 4

Method 10 = Kiến trúc máy tính; Method 11 = Cơ sở quản trị mạng; Method 12 = Cơ sở an ninh mạng.

Condition 5

Method 13 = Văn chương anh; Method 14 = Tiếng anh trong thương mại; Method 15 = Tiếng anh giao tiếp.

Condition 6

Method 16 = Lập trình hướng đối tượng; Method 17 = Lý thuyết đồ thị; Method 18 = Cơ sở dữ liệu.

Condition 7

Method 19 = Nghệ thuật lãnh đạo; Method 20 = Quản trị nguồn nhân lực; Method 21 = Quản trị dự án và tài chính.

Condition 8

Method 22 = Nguyên lý kế toán; Method 23 = Kế toán tài chính; Method 24 = Kinh tế vi mô.

Condition 9

Method 25 = Thiết kế dựng hình tổng hợp; Method 26 = Thẩm mỹ học; Method 27 = Vẽ mỹ thuật.

Condition 10

Method 28 = Tiếng Pháp 1; Method 29 = Dẫn luận ngôn ngữ; Method 30 = Âm vị học.

Xác định Context

Với các method đã được xác định, ta nhận thấy rằng trong bản thân mỗi method sẽ có điều kiện riêng của nó. Chẳng hạn như với *Method 1* Kỹ thuật truyền số liệu, ta còn phải xét xem môn học này có cần môn tiên quyết hoặc môn học này có phải là một môn song hành của môn nào khác hay không, và nếu có môn tiên quyết, ta phải làm gì tiếp đó.

Như vậy, từ những method đã xác định được, ta nhận thấy những tập hợp Condition đã xét chỉ là những điều kiện chung nhất, vì khi xét vào một môn học cụ thể, môn học được xét còn phải phụ thuộc vào quan hệ hay tính chất môn học. Từ đó, dễ thấy rằng những Condition tiêu biểu chẳng qua sẽ là các Context trong bài toán Đăng ký môn học. Khi đó, những method đã xác định sẽ là những method tương ứng trong mỗi context cụ thể, và mỗi method sẽ có Condition riêng biệt của chính nó.

Tiếp tục phân tích, nhóm sẽ chọn ra hai context tiêu biểu.

Context 1 = Đại học, 06, Mạng máy tính.

Context 2 = Đại học, 06, Công nghệ thông tin.

Xây dựng Role

Với những Context, Condition và Method đã phân tích như trên, tiếp theo chúng ta sẽ xác định Role cho bài toán.

Khi phân tích theo hướng đối tượng, trong mô hình usecase ta sẽ có Actor và System tương tác với nhau. Khi chuyển qua Context, mỗi Actor mà System phải mô tả lại một phần sẽ trở thành một Role. Trong bài toán Đăng ký môn học, sinh viên là một Actor. Actor sinh viên cần được mô tả trong hệ thống, do đó sẽ trở thành Role sinh viên trong Context.

Ngoài ra chủ nhiệm chương trình cũng đóng vai trò quan trọng trong việc kiểm tra và tương tác giữa Sinh viên và hệ thống. Do đó chủ nhiệm chương trình đóng vai trò là Role và Role chủ nhiệm chương trình sẽ nắm toàn bộ chương trình đào tạo.

Xây dựng Agent

Trên cơ sở phân tích Role trong nhiều Context, ta sẽ xác định được Agent. Agent là cách chúng ta nhóm các capability trong các role để cài đặt theo cách sao cho dễ thực hiện nhất. Vì vậy, với Role là Sinh viên, ta sẽ có Agent là một sinh viên cụ thể để gán vào Role.

Với Role sinh viên, ta sẽ xây dựng ba Agent cụ thể

Agent 1: Nguyễn Văn A, sinh viên năm nhất, hệ đại học, ngành công nghệ thông tin, đăng ký môn Cấu trúc dữ liệu và giải thuật, Cơ sở dữ liệu, Kế toán tài chính, Tin học văn phòng.

Agent 2: Nguyễn Văn B, sinh viên năm 2, hệ cao đẳng, ngành mạng máy tính, đăng ký môn Cấu trúc dữ liệu và giải thuật, Mạng máy tính, PTTK hệ thống mạng.

Agent 3: Nguyễn Văn C, sinh viên năm 4, hệ đại học, ngành Tiếng Anh, đăng ký môn Anh văn thương mại, Văn chương anh, Tin học văn phòng, Kinh tế lượng.

Với Role chủ nhiệm chương trình, ta sẽ có Agent cụ thể như

Agent 4: CNCT A, Chương trình đào tạo đại học khóa 06, danh sách môn tiên quyết, tương đương, thay thế.

Agent 5: CNCT B, Chương trình đào tạo cao đẳng khóa 07, danh sách môn tiên quyết, tương đương, thay thế.

2.2. Hướng giải quyết

Từ những mô tả theo hướng ban đầu của hệ thống, chúng ta nhận thấy sẽ có hai trường hợp cụ thể phát sinh.

Trường hợp 1: Đưa một Agent vào nhiều Context cụ thể.

Trong trường hợp này, với một Agent là sinh viên, khi đăng ký môn học, nếu môn học sinh viên đăng ký không phù hợp với điều kiện của context nào, sinh viên sẽ được chuyển qua một context khác với môn đăng ký phù hợp hơn.

Danh sách các môn học sẽ được đưa vào Chương trình đào tạo của khóa 06. Từ danh sách ban đầu, khi đưa vào chương trình đào tạo thì danh sách sẽ phân làm 2 loại: những môn có trong chương trình đào tạo khóa 06 và những môn đã được thay thế trong chương trình đào tạo mới, hiện tại không còn mở lớp.

Những môn có tên trong chương trình đào tạo khóa 06 sẽ được đưa vào Hệ thống đăng ký môn học, và thông báo đăng ký môn học thành công.

Những môn không có trong chương trình đào tạo khóa 06 cũng được đưa vào Hệ thống đăng ký môn học, và thông báo môn này không còn lớp, chủ nhiệm chương trình sẽ kiểm tra môn thay thế, hay còn gọi là môn tương đương trong danh sách các môn thay thế và tương đương, sau đó chủ nhiệm chương trình sẽ chọn môn học tương đương hay thay thế cho môn học, sau đó chương trình sẽ xuất ra môn học thay thế cho môn không đăng ký được.

Nếu đăng ký môn học thay thế thì chương trình đào tạo khóa 06 sẽ không còn phù hợp nữa, vì vậy chủ nhiệm chương trình sẽ tìm chương trình đào tạo phù hợp và di chuyển qua các chương trình đào tạo khác như chương trình đào tạo của khóa 07, 08. Nếu có môn tương ứng, chương trình sẽ cho phép đăng ký môn học theo chương trình đào tạo của khóa khác.

Trường hợp 2: Đưa nhiều Agent vào một Context.

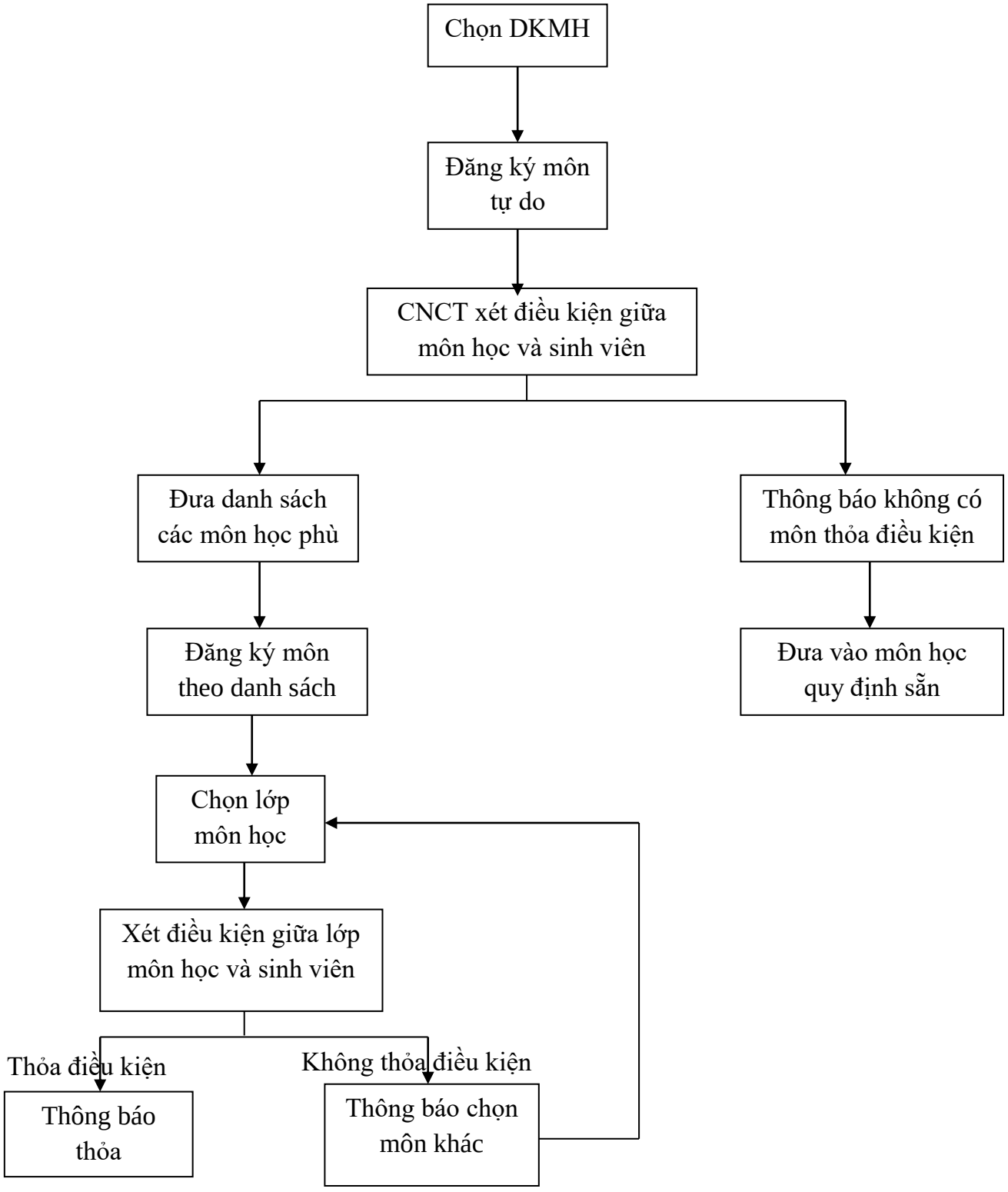
Trong trường hợp này, giả sử có hai agent khác nhau, agent A là chủ nhiệm chương trình đại học hệ chính quy, agent B là chủ nhiệm chương trình của đại học hệ liên thông. Như vậy, khi gán hai agent chủ nhiệm chương trình A và B vào role Chủ nhiệm chương trình để xét việc đăng ký môn học của sinh viên, sẽ xảy ra hai trường hợp tùy theo từng hoàn cảnh cụ thể.

Giả sử như cùng đăng ký một môn Công nghệ phần mềm, nhưng hoàn cảnh Hệ chính quy sẽ yêu cầu điều kiện là môn tiên quyết, ngược lại hoàn cảnh Hệ liên thông sẽ không yêu cầu điều kiện môn tiên quyết. Vì vậy, tùy hoàn cảnh khác nhau mà sẽ dẫn đến kết quả trả về khác nhau.

2.3. Sơ đồ WorkFlow

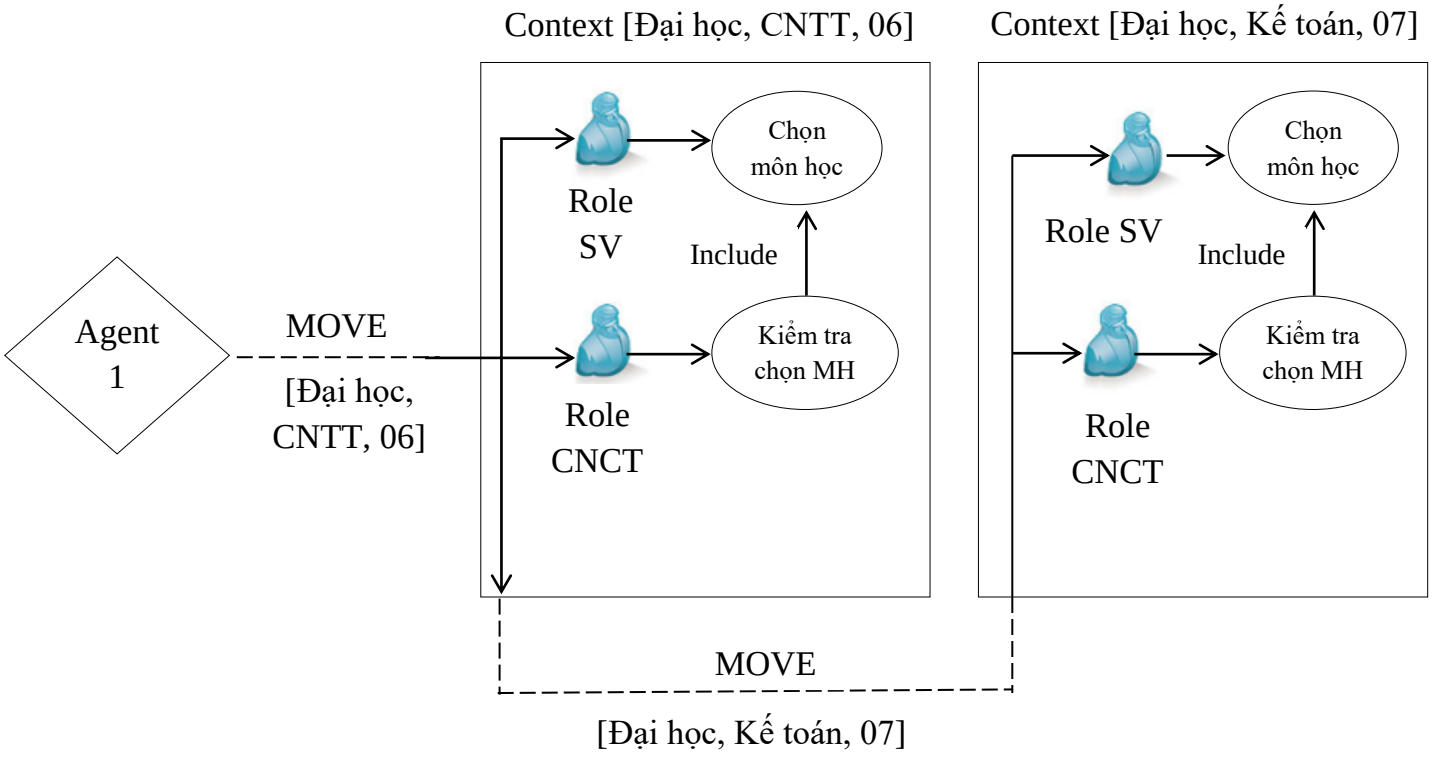
Để nhận rõ quy trình hệ thống, chúng ta có sơ đồ WorkFlow, State Diagram và Activity Diagram.

Hình sau là workflow của quá trình chọn môn học tự do của sinh viên



Hình 13: Sơ đồ WorkFlow của quá trình chọn môn học cho sinh viên

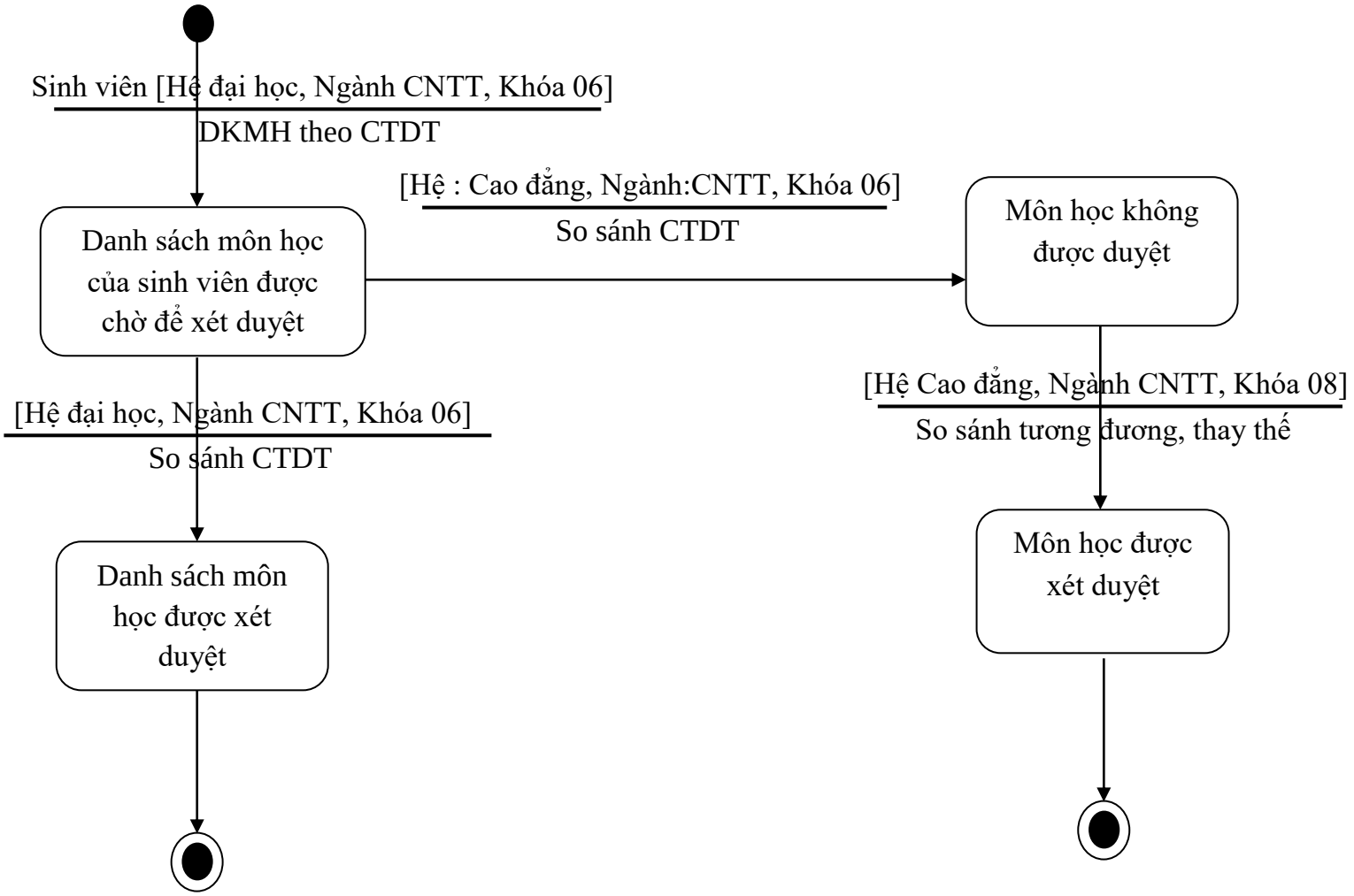
4. Mô hình di chuyển Context



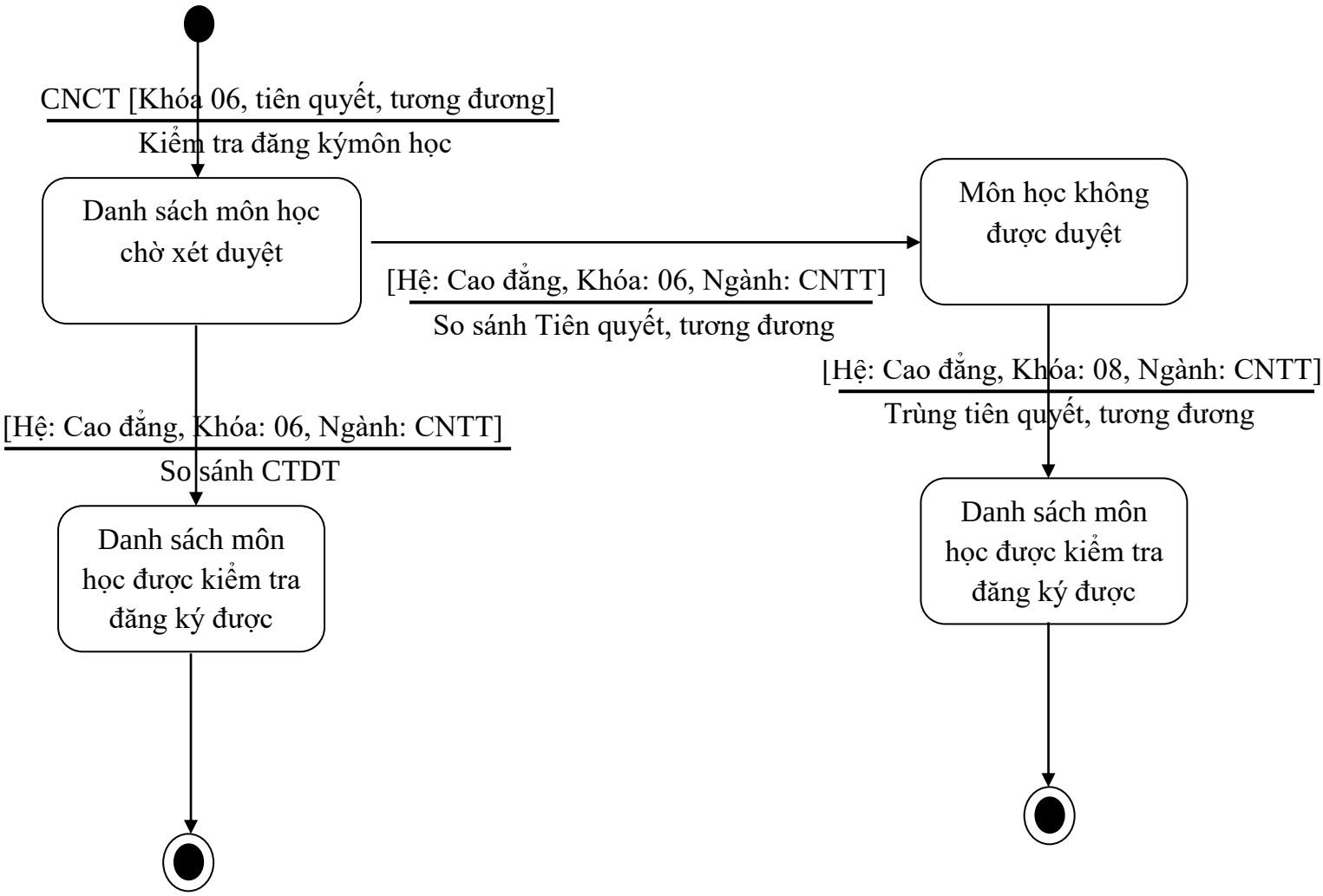
Hình 14: Mô hình di chuyển Agent vào Context trong Đăng ký môn học

5. State Transiton Diagram

Trong mô hình State Transition Diagram, ta sẽ có hai mô hình chính gồm mô hình của sinh viên khi đăng ký môn học, và của chủ nhiệm chương trình khi kiểm duyệt môn học.



Hình 15: Mô hình State Transition Diagram trong đăng ký môn học của sinh viên



Hình 16: Mô hình State Transition Diagram trong Kiểm duyệt môn học của Chủ nhiệm chương trình

Kết luận và đề nghị

I. Kết luận

1. Đánh giá kết quả

Sau mười lăm tuần thực hiện luận án, so với những mục tiêu ban đầu đã đề ra, nhóm đã đạt được những kết quả nhất định.

Với bài toán đầu tiên, bài toán về cơ sở lý luận, nhóm đã tiến hành tìm tòi, nghiên cứu những khái niệm hãy còn rất mới mẻ. Tuy thời gian tìm hiểu không dài, nhưng nhóm cũng đã nắm bắt được các mức khái niệm về mặt cơ bản, nắm được những cách thức cũng như hiểu được nguyên tắc khi làm việc với ngôn ngữ theo hướng tác tử.

Với bài toán thứ hai, bài toán về phân tích thiết kế, tuy gặp nhiều khó khăn và trở ngại trong hướng phân tích do đây là lần đầu tiên tiếp xúc với một hướng phân tích còn mới lạ, nhóm cũng đã phần nào hiểu được những bước đi, cũng như cách thức của quy trình phân tích theo hướng tác tử.

Với bài toán cuối cùng, bài toán về xây dựng công cụ hỗ trợ, ngôn ngữ hướng tác tử phần nào đã hoàn thành một số mục tiêu ban đầu khi đã có thể hỗ trợ mô tả các khái niệm chung như context, role, agent, capability, method và condition, cũng như thể hiện được việc gán agent vào context hay role, và qua đó làm rõ được vấn đề đã từng được đặt ra trong nêu vấn đề của bài toán, là cho thấy được cùng một context và role nhưng agent khác nhau sẽ dẫn đến các method khác nhau.

Ngoài ra, trong bài toán cuối cùng, về mức độ hoàn thành thì ngôn ngữ hướng tác tử cũng phần nào liên kết được với hệ thống hướng đối tượng, bằng việc lấy các method và condition mà người dùng khai báo để sử dụng, và qua đó tạo ra tập tin exe để hệ thống hướng đối tượng sử dụng sau này.

Bên cạnh việc hoàn thành những mục tiêu đã đề ra, trong thời gian thực hiện khóa luận vẫn còn chưa giải quyết hoàn toàn một vài vấn đề cũng như những vấn đề mới nảy sinh ra trong quá trình thực hiện ngôn ngữ. Vấn đề đầu tiên chính là mối quan hệ 1 Agent – n Context và n Agent – n Role. Sau đó là cơ chế mapping hiện tại vẫn còn nhiều thiếu sót, custom attribute sử dụng để khai báo condition chưa hỗ trợ việc khai báo động, và việc thực thi hệ thống như thế nào khi kết hợp hướng tác tử và hướng đối tượng?

Tóm lại, trong quá trình thực hiện luận văn, với những mục tiêu đã đề ra ban đầu, bên cạnh những kết quả đạt được nhóm vẫn còn vài vấn đề chưa giải quyết triệt để. Những vấn đề phát sinh sẽ được mô tả kỹ hơn trong phần hướng mở rộng.

2. Ưu điểm

Sau quá trình thực hiện luận án, với những kết quả đạt được, ngôn ngữ do nhóm tạo ra so với mục tiêu đề ra có những ưu điểm nhất định

- Ngôn ngữ hỗ trợ người dùng mô tả các khái niệm mới về hướng tác tử.
- Hỗ trợ giải quyết các bài toán liên quan nhiều đến các điều kiện tác động dẫn đến quá nhiều usecase, như bài toán ứng dụng đăng ký môn học.
- Cung cấp cơ chế mapping tự động giúp đơn giản hóa bài toán.
- Hỗ trợ phần nào kết nối với hệ thống hướng đối tượng.

3. Khuyết điểm

Tuy nhiên, bên cạnh những ưu điểm thì ngôn ngữ cũng tồn tại một vài khuyết điểm, như

- Việc kết nối với hệ thống hướng đối tượng chưa thật sự tốt.
- Cơ chế mapping hiện tại vẫn còn nhiều thiếu sót, không thể mapping chính xác đối với những điều kiện thuộc dạng “<=” hoặc “>=”.
- Chưa thể hỗ trợ việc gán một agent vào nhiều role hay nhiều agent vào nhiều role.

4. Những khó khăn gặp phải

Khi nhóm tiến hành hiện thực vào bài toán cụ thể theo hướng tác tử, thì gặp rất nhiều khó khăn, do các ứng dụng xây dựng dựa trên mô hình tác tử đã được triển khai trong các lĩnh vực lớn bao gồm thương mại điện tử, kiểm soát không lưu và một loạt các vấn đề phức tạp khác. Vì thế, nhóm đã gặp khó khăn để tìm được một bài toán ứng dụng phù hợp trong thời gian mười lăm tuần của luận án.

Mặt khác, tính khả thi của bài toán ứng dụng ảnh hưởng rất lớn đến quá trình hiện thực. Như bài toán ứng dụng đầu tiên mà nhóm áp dụng là bài toán về chuyển bệnh nhân giữa các bệnh viện. Đây là một ứng dụng hay và hữu ích, đã được áp dụng trong hệ thống y tế của nhiều nước tiên tiến trên thế giới. Nhưng, khi áp dụng vào hệ thống y tế ở Việt Nam, thì điều này lại không khả thi, do đa số các bệnh viện ở Việt Nam đều quản lý một cách đơn lẻ, hầu hết đều không có bệnh án điện tử, nên việc chuyển bệnh nhân sang bệnh viện khác thường mang tính thủ công.

Khó khăn tiếp theo mà nhóm gặp phải trong quá trình thực hiện luận án là do thời gian tiếp cận hướng tác tử quá ngắn, nên trong việc phân tích nhóm còn bị ảnh hưởng nhiều bởi hướng đối tượng. Vì vậy, việc phân tích gặp phải nhiều khó khăn cũng như phải chỉnh sửa nhiều lần.

Một khó khăn khác nhóm gặp phải là thường hay phân tích vấn đề theo hướng mở rộng, nên dễ dẫn đến việc phức tạp hóa bài toán, dẫn đến tốn thời gian thực hiện những bước không cần thiết.

5. Bài học kinh nghiệm

Thông qua luận văn này, nhóm đã học được thêm nhiều kiến thức bổ ích. Từ những ưu khuyết điểm rút ra được trong quá trình thực hiện luận án, nhóm đã rút ra bài học kinh nghiệm

- Không quá tập trung vào giải quyết một vấn đề bế tắc. Nên đi tiếp để không chậm tiến độ, sau đó sẽ đi lặp lại quá trình để giải quyết, xem xét kỹ lại vấn đề hơn.
- Củng cố lại những kiến thức về phân tích hướng đối tượng.
- Tiếp cận và học hỏi được một phương pháp mới, phương pháp phân tích theo hướng đối tượng, mặc dù vẫn còn nhiều bỡ ngỡ.
- Ghi báo cáo từng tuần đã làm được những gì để khi tổng kết lại nhanh gọn và không bị bỏ sót ý.

II. Hướng mở rộng

Trong tương lai, nếu có điều kiện tiếp tục nghiên cứu về đề tài Phân tích câu theo ngữ cảnh, nhóm sẽ có những mở rộng so với chương trình hiện tại. Một số những điểm mở rộng mà nhóm cần đạt tới như

Về Custom attribute

Khi dùng Custom attribute, custom attribute không cho phép thay đổi các điều kiện mà sẽ giữ cố định chúng. Để khắc phục nhược điểm này, chúng ta sẽ tạo tập tin xml từ bên Object Oriented, tập tin xml có tên và cú pháp theo quy định, với dữ liệu là phần dữ liệu được lấy từ database đã có sẵn. Khi ấy, nếu muốn lấy điều kiện, ta sẽ sử dụng custom attribute, chỉ cần đưa ra ID sẽ lập tức lấy được điều kiện cần thiết.

Cơ chế so sánh

Trong chương trình, khi tiến hành so sánh giữa các điều kiện với nhau, nhóm sẽ tiến hành thực hiện bằng phương pháp so sánh bằng. Ví dụ như điều kiện yêu cầu ứng viên đi phỏng vấn xin việc phải biết hai ngoại ngữ, thì chương trình sẽ xét giữa điều kiện của bên phỏng vấn đưa ra và điều kiện giữa người ứng viên đến xin việc, xem người đó có thỏa mãn yêu cầu biết hai ngoại ngữ hay không.

Tuy nhiên, với phương pháp so sánh bằng này, chúng ta không thể so sánh để đưa ra một kết quả chính xác dựa trên các điều kiện.

Lấy ví dụ với một ứng viên đi xin việc làm. Trong những yêu cầu cần có của ứng viên là người đó phải có ít nhất hai năm kinh nghiệm trong lĩnh vực dự tuyển. Vậy nếu đi

theo phương pháp giải quyết so sánh bằng, thì chỉ có những ai với số năm kinh nghiệm là hai mới thỏa điều kiện, trong khi nếu số năm kinh nghiệm từ hai trở lên, nghĩa là có ba, hoặc bốn năm kinh nghiệm đều thỏa. Trong trường hợp này, để so sánh chính xác, ta phải áp dụng cơ chế so sánh lớn hơn hoặc bằng.

Xét tiếp ví dụ về ứng viên đi xin việc làm. Nếu trong những điều kiện dự tuyển yêu cầu để đảm nhận vị trí công việc được nêu, yêu cầu ứng viên tham gia phải 30 tuổi hoặc trẻ hơn, thì lúc này để có thể so sánh chính xác, chúng ta phải áp dụng cơ chế so sánh nhỏ hơn hoặc bằng.

Từ những ví dụ đã đưa ra, chúng ta nhận thấy hiện tại phương pháp so sánh bằng chưa thể đáp ứng được sự chính xác trong việc so sánh điều kiện. Để có thể so sánh một cách chính xác, chúng ta phải nhận biết được khi nào cần so sánh lớn hơn hoặc bằng, khi nào cần so sánh nhỏ hơn hoặc bằng. Đó chính là hạn chế của nhóm trong phần giải quyết bài toán. Để giải quyết được hạn chế đó, cần phải thay đổi cú pháp đã đề ra cũng như đưa ra một cơ chế so sánh cụ thể hơn. Tuy nhiên với quãng thời gian hiện tại, nhóm vẫn chưa có cơ hội để hoàn thành. Trong tương lai, nếu có thể tiếp tục nghiên cứu và phát triển đề tài của luận án, nhóm sẽ mở rộng và hoàn thành việc so sánh một cách chính xác hơn.

Một agent gán vào nhiều role

Trong thời gian tiến hành luận án, nhóm chỉ dừng lại ở việc thực hiện gán một agent vào một role. Tuy nhiên, trong tương lai, nếu được tiếp tục nghiên cứu và phát triển, mở rộng đề tài luận án, nhóm sẽ mở rộng hơn, bắt đầu bằng việc gán nhiều agent vào nhiều role.

Để thực hiện việc gán nhiều agent vào nhiều role, hướng giải quyết là ta sẽ tiến hành phân chia theo thời gian. Ở vào những mốc thời gian nhất định thì sẽ có một agent cụ thể nhất định được gán vào một role. Cũng cùng một role đó, nhưng vào thời điểm khác nhau sẽ được gán bởi agent khác nhau. Cơ chế phân chia thời gian sẽ là cách thức để giải quyết vấn đề gán nhiều agent vào nhiều role.

Nhiều agent gán vào một role

Để thực hiện việc gán nhiều agent vào một role, ta sẽ tiến hành việc định nghĩa role đó như một type, và agent cần gán vào chính là một mảng các giá trị của type đó.

PHỤ LỤC

Tài liệu tham khảo

[1] Hanspeter Mössenböck, Johannes Kepler University Linz, *User Manual – The Compiler Generator Coco/R.*

[2] http://en.wikipedia.org/wiki/Common_Language_Runtime

[3] http://en.wikipedia.org/wiki/Belief-Desire-Intention_software_model#BDI_Agents

[4] Trần Vũ Bình, *Observation, Expectation and Attention Logics – A Foundation for Agent Theories.*

[5] http://www.devarticles.com

[6] http://plato.stanford.edu/entries/logic-hybrid/

[7] Jack Intelligent Agents – Agent Manual

[8] http://www.devarticles.com

[9] http://www.agentlab.de/aose.html

[10] http://en.wikipedia.org/wiki/Hybrid_logic

Code ứng dụng bài toán Lời chào

```
create context GiaDinh;

create context DuongPho;

create context TruongHoc;

create conditions GDCond (location) values (‘family’);

create conditions DPCond (location) values (‘street’);

create conditions THCond (location) values (‘school’);

bind conditions GDCond into context GiaDinh;

bind conditions DPCond into context DuongPho;

bind conditions THCond into context TruongHoc;

create role NguoiChaoGD;

create role NguoiChaoDP;

create role NguoiChaoTH;

bind role NguoiChaoGD into context GiaDinh;

bind role NguoiChaoDP into context DuongPho;

bind role NguoiChaoTH into context TruongHoc;

create capabilities ChaoHoiGD;

bind capabilities ChaoHoiGD into role NguoiChaoGD;

create capabilities ChaoHoiDP;

bind capabilities ChaoHoiDP into role NguoiChaoDP;

create capabilities ChaoHoiTH;

bind capabilities ChaoHoiTH into role NguoiChaoTH;

create method ChaoGD;

bind method ChaoGD into capabilities ChaoHoiGD;

create method ChaoDP;

bind method ChaoDP into capabilities ChaoHoiDP;
```

create method ChaoTH;

bind method ChaoTH into capabilities ChaoHoiTH;

create conditions Chao_1 (gender, age, location, relation) values (‘female’, ‘old’, ‘family’, ‘grandmother’) check intersect;

bind conditions Chao_1 into method ChaoGD;

create conditions Chao_2 (gender, age, location) values (‘female’ ,’young’, ‘street’) check intersect;

bind conditions Chao_2 into method ChaoDP;

create conditions Chao_3 (gender, age, location) values (‘male’ ,’old’, ‘school’) check intersect;

bind conditions Chao_3 into method ChaoTH;

create agent NguyenVanTi;

create capabilities ChaoHoi;

bind capabilities ChaoHoi into agent NguyenVanTi;

bind method ChaoChi into capabilities ChaoHoi;

bind method ChaoDi into capabilities ChaoHoi;

bind method ChaoCo into capabilities ChaoHoi;

bind method ChaoAnh into capabilities ChaoHoi;

bind method ChaoBan into capabilities ChaoHoi;

bind method ChaoAnh into capabilities ChaoHoi;

bind method ChaoChu into capabilities ChaoHoi;

bind method ChaoCau into capabilities ChaoHoi;

bind method ChaoThay into capabilities ChaoHoi;

move agent NguyenVanTi with conditions GDCond;

move agent NguyenVanTi with conditions DPCond;

move agent NguyenVanTi with conditions THCond;

Code Ứng dụng bài toán Ngày nghỉ lễ

#condition for agent method

#create conditions Condition1 (relation, hobby, center) values (‘family’, ‘shopping’, ‘vincom’) check intersect;

#create conditions MethodCond2 (major, age, gender) values (‘informatic’, ‘young’, ‘male’) check intersect;

#create conditions MethodCond3 (vehicle, location) values (‘moto’, ‘hsu’) check intersect;

#create conditions MethodCond4 (relation, hobby, center) values (‘friend’, ‘picnic’, ‘dainam’) check intersect;

#create conditions MethodCond4A (relation, hobby, center) values (‘friend’, ‘tech’, ‘expo’) check intersect;

#method for agent

#create method DiShopping check union;

#bind conditions MethodCond1 into method DiShopping;

#create method HopNhom check union;

#bind conditions MethodCond2 into method HopNhom;

#create method DiDaNgoai check intersect;

#bind conditions MethodCond3 into method DiDaNgoai;

#bind conditions MethodCond4 into method DiDaNgoai;

#create method DiHoiCho check intersect;

#bind conditions MethodCond4A into method DiHoiCho;

#bind conditions MethodCond2 into method DiHoiCho;

#capabilities for agent

create capabilities HocTap check union;

bind method HopNhom into capabilities HocTap;

create capabilities GiaiTri check union;

bind method DiShopping into capabilities GiaiTri;

bind method DiDaNgoai into capabilities GiaiTri;

create capabilities DiExpo check union;

bind method DiHoiCho into capabilities DiExpo;

#agent

create agent NguyenVanA;

bind capabilities GiaiTri into agent NguyenVanA;

create agent NguyenVanB;

bind capabilities HocTap into agent NguyenVanB;

create agent NguyenVanC;

bind capabilities DiExpo into agent NguyenVanC;

#condition for role method

create conditions MethodCond5 (relation, hobby) values(‘family’, ‘shopping’) check intersect;

create conditions MethodCond6 (age, gender) values (‘young’, ‘male’) check intersect;

create conditions MethodCond7 (vehicle, location) values (‘bus’, ‘hsu’) check intersect;

#method for role

create method RDiChoi check union;

bind conditions MethodCond5 into method RDiChoi;

bind conditions MethodCond6 into method RDiChoi;

bind conditions MethodCond7 into method RDiChoi;

#create method RHopNhom;

#capabilities for role

create capabilities RGiaiTri check union;

bind method RDiChoi into capabilities RGiaiTri;

#create role

create role SinhVien check union;

bind capabilities RGiaiTri into role SinhVien;

#condition for context

create conditions ContextCond(relation, age, gender) values (‘family’, ‘young’, ‘male’) check intersect;

#create context

create context NghiLe;

bind conditions ContextCond into context NghiLe;

bind role SinhVien into context NghiLe;

#move agent

move agent NguyenVanA with conditions ContextCond;

#move agent NguyenVanB with conditions ContextCond;

#move agent NguyenVanC with conditions ContextCond;

select capabilities GiaiTri;

select conditions ContextCond;

select agent NguyenVanA;

select role SinhVien;

select context NghiLe;

select method RDiChoi;

Code Ứng dụng bài toán Đăng ký môn học

create context DKMH06;

create conditions DK06 (year, major, type) values (‘06’, ‘IT’, ‘university’) check intersect;

bind conditions DK06 into context DKMH06;

#============Role SinhVien============#

create role SinhVien check union;

bind role SinhVien into context DKMH06;

create capabilities DangKyMH check intersect;

bind capabilities DangKyMH into role SinhVien;

create method DangKy check intersect;

bind method DangKy into capabilities DangKyMH;

create conditions DangKy_01 (year, major, type, semester) values (‘06’, ‘IT’, ‘university’, ‘01’) check intersect;

#create conditions DangKy_02 (course_type) values (‘basic’) check intersect;

create conditions DangKy_03 (course_type) values (‘high’) check intersect;

bind conditions DangKy_01 into method DangKy;

#bind conditions DangKy_02 into method DangKy;

bind conditions DangKy_03 into method DangKy;

#============Role CNCT============#

create role CNCT check union;

bind role CNCT into context DKMH06;

create capabilities KiemTraDK check union;

bind capabilities KiemTraDK into role CNCT;

create conditions DKTQ (course_type) values (‘basic’) check intersect;

create conditions DKCN (course_type) values (‘high’) check intersect;

create conditions DKTD (course_type) values (‘plus’) check intersect;

```
create method DKTQ check intersect;
bind conditions DKTQ into method DKTQ;
bind method DKTQ into capabilities KiemTraDK;
create method DKTD check intersect;
bind conditions DKTD into method DKTD;
bind method DKTD into capabilities KiemTraDK;
create method DK_LTHDT check intersect;
bind method DK_LTHDT into capabilities KiemTraDK;
bind conditions DKCN into method DK_LTHDT;
create conditions DK_LTHDT (course_studied) values (‘NMLT’) check intersect;
bind conditions DK_LTHDT into method DK_LTHDT;
create method DK_UML check intersect;
bind method DK_UML into capabilities KiemTraDK;
bind conditions DKCN into method DK_UML;
create conditions DK_UML (course_studied) values (‘LTHDT’) check intersect;
bind conditions DK_UML into method DK_UML;
create method DK_PTNV check intersect;
bind method DK_PTNV into capabilities KiemTraDK;
bind conditions DKCN into method DK_PTNV;
create conditions DK_PTNV_01 (course_studied) values (‘LTHDT’) check intersect;
bind conditions DK_PTNV_01 into method DK_PTNV;
create conditions DK_PTNV_02 (course_studied) values (‘UML’) check intersect;
bind conditions DK_PTNV_02 into method DK_PTNV;
#============Agent NguyenVanA============#
create agent NguyenVanA;
```

create conditions Obl (year, major, type, semester) values (‘06’, ‘IT’, ‘university’, ‘01’);

create conditions CType_01 (course_type) values (‘basic’);

create conditions CType_02 (course_type) values (‘plus’);

create conditions CType_03 (course_type) values (‘high’);

create capabilities DangKyMonHoc;

bind capabilities DangKyMonHoc into agent NguyenVanA;

create method KTTSL check intersect;

bind method KTTSL into capabilities DangKyMonHoc;

bind conditions Obl into method KTTSL;

bind conditions CType_01 into method KTTSL;

create method MHHDL check intersect;

bind method MHHDL into capabilities DangKyMonHoc;

bind conditions Obl into method MHHDL;

bind conditions CType_01 into method MHHDL;

create method Marketting check intersect;

bind method Marketting into capabilities DangKyMonHoc;

bind conditions Obl into method Marketting;

bind conditions CType_02 into method Marketting;

create method LTHDT check intersect;

bind method LTHDT into capabilities DangKyMonHoc;

bind conditions Obl into method LTHDT;

bind conditions CType_03 into method LTHDT;

create method UML check intersect;

bind method UML into capabilities DangKyMonHoc;

bind conditions Obl into method UML;

bind conditions CType_03 into method UML;

create conditions UML (course_studied) values (‘LTHDT’);

bind conditions UML into method UML;

select capabilities DangKyMonHoc;

move agent NguyenVanA with conditions DK06;

interact role CNCT with role SinhVien;