ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

TÔN LONG PHƯỚC

SINH MÃ NGUỒN CHO CÁC THIẾT BỊ ĐEO TAY TRONG CÁC GIẢI PHÁP THÔNG MINH TRÊN NỀN TẢNG INTERNET VẠN VẬT BẰNG MÔ HÌNH HÓA VÀ TẬP LUẬT

Ngành: Khoa học máy tính
Mã số ngành: 62480101

TÓM TẮT LUẬN ÁN TIẾN SĨ

TP. HỒ CHÍ MINH - NĂM 2022

Công trình được hoàn thành tại **Trường Đại học Bách Khoa – ĐHQG-HCM**

Người hướng dẫn 1: TS. Lê Lam Sơn
Người hướng dẫn 2: TS. Phạm Hoàng Anh

Phản biện độc lập 1:
Phản biện độc lập 2:

Phản biện 1:
Phản biện 2:
Phản biện 3:

Luận án sẽ được bảo vệ trước Hội đồng đánh giá luận án họp tại
.....
.....
vào lúc giờ ngày tháng năm

Có thể tìm hiểu luận án tại thư viện:

- Thư viện Trường Đại học Bách Khoa – ĐHQG-HCM
- Thư viện Đại học Quốc gia Tp.HCM
- Thư viện Khoa học Tổng hợp Tp.HCM

DANH MỤC CÔNG TRÌNH ĐÃ CÔNG BỐ

Tạp chí quốc tế

1. **L-P. Tôn,** L-S. Lê and M-S. Nguyen "Micraspis: A Computer-aided Proposal towards Programming and Architecting Smart IoT Wearables", *IEEE Access*, vol. 9, pp 105393-105408, 2021 (**ISI (SCIE), Q1, Impact Factor: 3.745**).

Tạp chí trong nước

1. **L-P. Tôn** and M-H. Nguyen "Model checking early requirements specifications in Alloy", *In Journal of Science and Technology, Vietnam Academy of Science and Technology*, vol. 54 (3A), *Special issue of Intelligent System and its Applications*, pp. 140-151, (selected papers from Proceedings of *International Symposium Intelligent Systems and Applications 2016* (ISA2016), Ho Chi Minh city, Vietnam).

Kỷ yếu hội nghị quốc tế

1. **L-P. Tôn** and T-M. Truong "Linking Rules and Conceptual Model in a Domain Specicfic Language", *in Proceedings of 9th International Conference on Advanced Computing and Applications*, pages 35-42, IEEE Computer Society, 2015 (**Scopus**).
2. T-M. Truong, L-S. Lê and **L-P. Tôn** "Re-engineering Enterprises Using Data Warehouse as a Driver and Requirements as an Enabler", i*n Proceedings of 21st IEEE International Conference on Enterprise Distributed Object Computing (EDOC)*, pp 67-72, IEEE Computer Society, Quebec, Canada, October 2017 (**Qualis conference ranking A2**)
3. **L-P. Tôn**, L-S. Lê and H-A. Pham "Towards a Domain Specific Framework for Wearable Applications in Internet of Things", *in Proceedings of 4th International Conference on Future Data and Security Engineering*, pp 309-324, Springer, November 2017 (**Scopus**).
4. **L-P. Tôn**, H-A. Pham and B. Dao "Software Abstraction for Casual Games Using Temporal Model: an Alloy-based Approach", *in Proceedings of 11th International Conference on Advanced Computing*

and Applications, pp 3-9, IEEE Computer Society, November 2017 (**Scopus**).

5. **L-P. Tôn**, L-S. Lê, H-A. Pham and B. Dao "Monitoring IoT Objects in Wearable Applications: An Alloy-Based Approach", *in Proceedings of 12th International Conference on Advanced Computing and Applications*, pp 35-41, IEEE Computer Society, November 2018 (**Scopus**).
6. T-M. Truong, **L-P. Tôn** "Augmenting a Process Model with State Machine and Business Artifacts", *in Proceedings of 13th International Conference on Advanced Computing and Applications*, pp 7-15, IEEE Computer Society, 2019.
7. **L-P. Tôn**, L-S. Lê "Enacting a Rule-Based Alert Business Process in Smart Healthcare Using IoT Wearables", *in Proceedings of 21st IEEE International Conference on Enterprise Distributed Object Computing*, pp 104-107, IEEE Computer Society, Paris, France, October 2019 (**Qualis conference ranking A2**)

Đề tài nghiên cứu khoa học

1. Đề tài cấp cơ sở, Tên đề tài "*Xây dựng mô hình chuyển đổi ngôn ngữ đặc tả sang ngôn ngữ thực thi*", Mã đề tài: IUH KTT10/16, Thời gian: 2015-2016, vai trò: Chủ nhiệm, Kết quả: Tốt
2. Đề tài cấp cơ sở, Tên đề tài "*Sinh mã nguồn theo cách tiếp cận ngôn ngữ mô hình chuyên biệt hóa cho ứng dụng games dựa trên ngôn ngữ Alloy*", Mã đề tài: TNCS-KHMT-2016-11, Thời gian: 2016-2017, vai trò: Đồng chủ nhiệm, Kết quả: Tốt
3. Đề tài cấp cơ sở, Tên đề tài "*Phát sinh mã nguồn giao diện cho các ứng dụng vận hành thiết bị đeo tay trong lĩnh vực IoT*", Mã đề tài: To-KHMT-2017-05, Thời gian: 2017-2018, vai trò: Đồng chủ nhiệm, Kết quả: Tốt

1 GIỚI THIỆU

1.1 Bài toán sinh mã nguồn

Bài toán sinh mã nguồn cho các ứng dụng chạy trên thiết bị đeo tay trong các giải pháp thông minh của lĩnh vực Internet vạn vật là thực tiễn và nhiều vấn đề cần giải quyết. Trong đó việc đặc tả các ứng dụng thông minh kết hợp đặc tả phần cứng của thiết bị sao cho phù hợp. Đồng thời, mã nguồn tạo ra có thể tái sử dụng trên các thiết bị khác nhau hoặc ngược lại. Các thiết bị cùng kiến trúc có thể được sử dụng trong các ứng dụng khác nhau. Tỷ lệ mã nguồn tạo ra cũng như chất lượng mã nguồn tạo ra phải đảm bảo tính đúng đắn và hiệu quả. Từ đó, chúng tôi xây dựng hướng tiếp cận, vận dụng sự kết hợp giữa mô hình hóa và xử lý ngữ nghĩa dựa vào các luật để đặc tả hệ thống. Cũng từ hướng tiếp cận này, chúng tôi xây dựng công cụ sinh mã nguồn cho các ứng dụng trong các giải pháp thông minh chạy trên thiết bị đeo tay. Để thực hiện nghiên cứu này, luận án cần phải xem xét giải quyết các mục tiêu nghiên cứu sau:

[**OB1**] Tìm hiểu các công trình nghiên cứu về sinh mã nguồn theo hướng mô hình, đặc biệt trong lĩnh vực Internet vạn vật. Từ đó, đề xuất một khung thức tổng quát cho việc sinh mã nguồn ứng dụng chạy trên các thiết bị đeo tay. Khung thức này cho phép đặc tả ứng dụng và kiến trúc phần cứng của thiết bị đeo tay một cách hợp lý.

[**OB2**] Nghiên cứu đưa ra cách tiếp cận đặc tả ứng dụng và kiến trúc phần cứng của thiết bị đeo tay bằng mô hình hóa. Trong đó ứng dụng chạy trên thiết bị đeo tay sẽ đặc tả bằng lược đồ mang tính khái quát cao (như lược đồ máy trạng thái - state machine), phần cứng của thiết bị đeo tay cũng sẽ dùng mô hình hóa để đặc tả. Mô hình đặc tả này phải được kiểm chứng đúng đắn với sự kết hợp các tập luật nhằm đảm bảo mã nguồn sinh ra sẽ hiệu quả.

[**OB3**] Xây dựng công cụ với giao diện đồ họa thân thiện, giúp triển khai cho khung thức mô tả ở trên. Đồng thời, công cụ cũng cho phép người dùng

đặc tả ứng dụng và phần cứng của thiết bị một cách hợp lý. Công cụ cũng cho phép tái sử dụng mã nguồn hay kiến trúc phần cứng của các thiết bị đeo tay. Mã nguồn sinh ra từ công cụ cũng đảm bảo chất lượng cũng như đúng đắn và tỷ lệ sinh mã tự động cao nhằm tiết kiệm thời gian phát triển ứng dụng. Cuối cùng, luận án cần phải xây dựng phương án kiểm thử phần mềm nhằm đảm bảo đúng đắn và hiệu quả của công cụ một cách khách quan. Trong đó, sẽ vận dụng nhiều tiêu chí nhằm kiểm chứng tính hiệu quả của công cụ thông qua các tiêu chí được đánh giá (tỷ lệ mã nguồn tạo ra, chất lượng hay độ hài lòng của người dùng). Từ đó, giúp công cụ có thể tiếp cận với ngành công nghiêp phần mềm trong lĩnh vực Internet vạn vật.

1.2 Câu hỏi nghiên cứu của luận án

Câu hỏi nghiên cứu của luận án là các câu hỏi nhằm giải quyết các mục tiêu nghiên cứu mà luận án đã đề ra. Trong đó, mục tiêu tổng quát sẽ là xây dựng khung thức cho bài toán sinh mã nguồn tự động cho các ứng dụng trong các giải pháp thông minh trong lĩnh vực Internet vạn vật. Từ mục tiêu tổng quát này, luận án xây dựng các câu hỏi nghiên cứu cụ thể như sau:

[**RQ1**] Đã có một khung thức hay một công cụ tổng quát nào cho việc sinh mã nguồn của các ứng dụng chạy trên thiết đeo tay trong lĩnh vực Internet vạn vật hay chưa? Khung thức phải bao hàm các bước từ đặc tả ứng dụng, thiết kế phần cứng, hỗ trợ kiểm tra tính đúng đắn của đặc tả phần cứng cũng như ứng dụng nhằm sinh mã nguồn đáp ứng yêu cầu của người dùng. Câu hỏi nghiên cứu này sẽ được giải quyết bởi mục tiêu nghiên cứu **OB1** của luận án.

[**RQ2**] Làm sao để đặc tả phần cứng của thiết bị đeo tay và các ứng dụng chạy trên thiết bị đó bằng sự kết hợp giữa mô hình hóa và các tập luật nhằm đảm bảo tính đúng đắn và hiệu quả cho bài toán sinh mã nguồn? Câu hỏi nghiên cứu này sẽ được giải quyết bởi mục tiêu nghiên cứu **OB2** của luận án.

[**RQ3**] Từ khung thức và cách đặc tả trên, xây dựng một công cụ hỗ trợ người dùng như thế nào để sinh mã nguồn cho các giải pháp thông minh chạy trên thiết bị đeo tay? Đồng thời kiểm chứng như thế nào để đảm bảo khách quan tính đúng đắn, hiệu quả của công cụ được xây dựng? Từ đó có hướng đưa công cụ tiếp cận công nghiệp phần mềm trong lĩnh vực Internet vạn vật. Câu hỏi nghiên cứu này sẽ được giải quyết bởi mục tiêu nghiên cứu **OB3** của luận án.

2 KHUNG THỨC TỔNG QUÁT XÂY DỰNG NGÔN NGỮ CHUYÊN BIỆT HÓA TRONG NGỮ CẢNH HẸP CHO LĨNH VỰC IoT

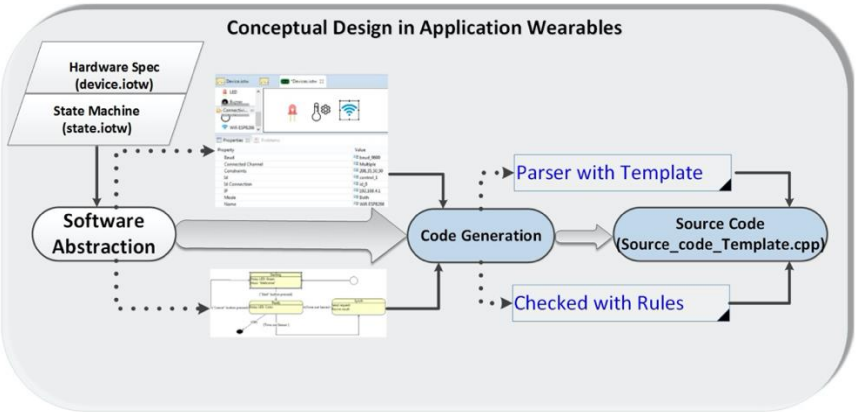
2.1 Khung thức đề xuất

Như những khảo sát các kỹ thuật cho bài toán sinh mã nguồn ở các công cụ, chúng tôi nhận thấy rằng hướng phát triển phần mềm theo hướng mô hình là một giải pháp phù hợp cho bài toán trong luận án này. Tư tưởng chính cho hướng tiếp cận này vẫn là nâng cao mức độ trừu tượng cho việc đặc tả phần mềm và thiết bị đeo tay. Mục tiêu giúp cho việc phát triển ứng dụng phần mềm trở nên nhanh chóng và cho phép các thiết kế phần cứng dễ dàng tái sử dụng. Tuy nhiên, theo chúng tôi được biết chưa có công cụ nào được đề xuất cho việc sinh mã các ứng dụng cụ thể trên thiết bị đeo tay. Trong khi đó, các nghiên cứu về mảng này tương đối rộng, trong nhiều lĩnh vực, nhưng chỉ giới hạn cho việc sinh mã cho các xử lý trong quá trình kết nối các thiết bị với nhau, hay xử lý trên các sự kiện riêng biệt giữa các thành phần trong hệ thống. Từ đó, chúng tôi mạnh dạn đề xuất một khung thức cho bài toán sinh mã nguồn các ứng dụng chạy trên các thiết bị đeo trong các giải pháp thông minh. Từ khung thức này, chúng tôi có thể xây dựng một công cụ hỗ trợ máy tính (computer-aided solutions) nhằm tăng cường quá trình tự động sinh mã nguồn cho ứng dụng.

Để có thể thiết kế được khung thức này, chúng tôi đề xuất công cụ hỗ trợ máy tính được tạo ra từ khung thức này có các khả năng như sau: cho phép xây dựng một thiết bị đeo tay đã sẵn sàng về mặt chức năng cho xử lý; cho phép thiết kế phần cứng đơn giản cho thiết bị với các thành phần phổ biến (ví dụ: đèn LED,

bộ rung, bàn phím, v.v.); khả năng tự kiểm tra các ràng buộc về hệ thống (ví dụ số lượng bàn phím, đèn LED, cảm biến, số pins, v.v.) đồng thời sinh mã nguồn cho việc cấu hình các pins khi kết nối trên bo mạch chủ; khả năng kết nối giữa mô hình đặc tả ứng dụng và mô hình kiến trúc phần cứng cần tách biệt nhằm tăng cường khả năng tái sử dụng mã nguồn; cuối cùng, khả năng sinh mã hiệu quả, chất lượng, cho các phần mềm trong các giải pháp thông minh.

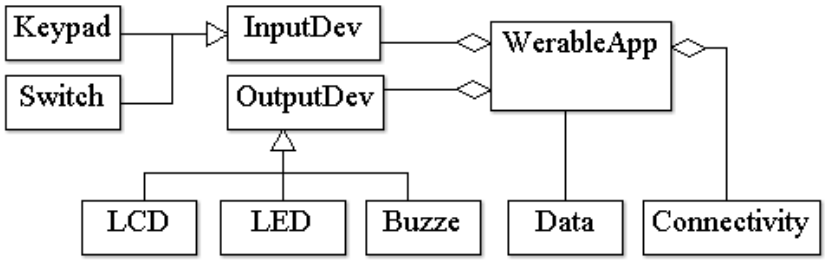
Với những yêu cầu và cách tiếp cận giải pháp trên, chúng tôi đề xuất khung thức tổng quát cho bài toàn sinh mã nguồn các ứng dụng chạy trên thiết bị đeo tay như sau: (Hình 3.6)



Hình 3.6 Khung thức tổng quát qui trình các bước cho việc phát triển một ứng dụng chạy trên thiết bị đeo tay.

Một trong những thành phần chính của khung thức mà chúng tôi đề xuất là siêu mô hình (Meta-model). Khung thức của chúng tôi chỉ thu hẹp lại miền ứng dụng, cụ thể là các ứng dụng chạy trên thiết bị đeo tay. Cho nên, chúng tôi chỉ đề xuất các thành phần chính của phần cứng như InputDev, OutputDev, Connectivity và WearableApp. Trong đó InputDev mô tả các thành phần nhận dữ liệu của thiết bị phần cứng như Keypad, Button, còn OutputDev mô tả các thành phần xuất của thiết bị như LCD, LED, Buzz. Riêng WearableApp, dùng mô tả các thành phần của ứng dụng chạy trên thiết bị đeo

tay. Phần này bao gồm những phần hỗ trợ như Data để mô tả dữ liệu của ứng dụng, Connectivity mô tả các kết nối với thiết bị. Tất cả các thành phần trong khung thức chúng tôi đề xuất nhằm hỗ trợ đặc tả các phần cứng của thiết bị đeo tay theo hướng mô hình hóa. Những mô hình này có thể được biểu diễn bằng các mô hình đồ họa có tính trừu tượng cao. Hình 3.7 là siêu mô hình trong khung thức mà chúng tôi đề xuất.



Hình 3.7 Meta-model của các thành phần trong khung thức đề xuất cho các ứng dụng chạy trên thiết bị đeo tay.

2.2 Cơ chế sinh mã của khung thức tổng quát

Trong khung thức chúng tôi đề xuất (Hình 3.6) gồm có ba pha chính: đặc tả phần mềm, sinh mã nguồn và hoàn thiện mã với mẫu và các tập luật. Trong đó, mỗi pha thực hiện một bước trong quá trình sinh mã của bài toàn mà luận án cần giải quyết. Chi tiết các pha được mô tả như sau:

Pha 1 (Software Abstraction): Pha đầu tiên của khung thức chúng tôi đề xuất là đặc tả phần mềm ở mức trừu tượng (Software Abstraction) gồm hai phần: Phần thứ nhất sẽ cho phép khai báo đặc tả phần cứng (Hardware Specification); Phần thứ hai cho phép đặc tả ứng dụng bằng lược đồ máy trạng thái (State Machine). Cả hai phần đặc tả này được tổ chức thành hai cấu trúc tập tin riêng biệt. Hai tập tin được lưu trữ nhằm mục đích tăng cường khả năng tái sử dụng mã nguồn của ứng dụng cũng như kiến trúc phần cứng của thiết bị. Ngoài ra, ở phần thứ nhất khung thức sẽ kiểm tra các ràng buộc cho việc thiết kê phần cứng

của thiết bị như số lượng, cấu hình chân pins. Phần thứ hai dùng lược đồ máy trạng thái để mô tả các xử lý của ứng dụng. Trong lược đồ này chúng tôi đề xuất một bộ văn phạm để mô tả cho việc đặc tả các xử lý bên trong của các trạng thái. Bản thân các trạng thái và việc chuyển đổi giữa chúng bên trong các lược đồ cũng được ràng buộc theo các cú pháp và các luật mà khung thức đã định nghĩa trước.

Pha 2 (Code Generation): Pha thứ hai, khung thức sẽ có nhiệm vụ phân tích cú pháp của pha thứ nhất, nhằm kiểm tra văn phạm mô tả các ứng dụng trong các lược đồ máy trạng thái đảm bảo tính đúng đắn hay không? Trong pha này, khung thức sẽ dựa vào một bảng ánh xạ (mapping) nhằm đưa ra các thành phần cho các khung chương trình tạo ta như hàm, biến, biểu thức, điều kiện. Ở pha này, khung thức sẽ thực hiện một bước được xem như là xây dựng các thành phần chính cho các đoạn mã chương trình được tạo ra. Trong đó các thông tin mô tả các thành phần khai báo phần cứng liên quan đến cấu hình hệ thống. Ngoài ra, các phần còn lại sẽ mô tả cho các hàm xử lý, sự kiện, biểu thức và điều kiện. Pha này chính là tiền đề cho pha tiếp theo dùng để hoàn thiện mã nguồn được tạo ra.

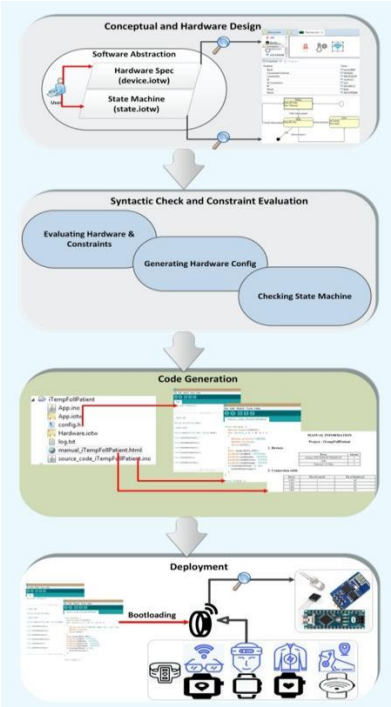
Pha 3 (Completion): Pha này là pha hoàn thiện mã nguồn cho các tập tin chương trình sinh mã khi hoàn tất khung thức. Ở pha này, khung thức sẽ sử dụng hai thành phần hỗ trợ gắn vào là mã nguồn mẫu (source code template) và các tập luật (rules). Trong đó, các mã nguồn mẫu dùng để tạo khung và dựa vào phân tích các thành phần của các khai báo phần cứng cũng như các xử lý của các hàm trong ứng dụng để hoàn thiện mã nguồn. Các tập luật sử dụng để tăng cường các xử lý cho các hàm của ứng dụng cũng như cách chuyển đổi trạng thái của các sự kiện bên trong lược đồ máy trạng thái.

3 MICRASPIS: CÔNG CỤ SINH MÃ NGUỒN CHO CÁC ỨNG DỤNG CHẠY TRÊN THIẾT BỊ ĐEO TAY

Micraspis là công cụ sinh mã nguồn cho các ứng dụng trong lĩnh vực IoT chạy trên thiết bị đeo tay. Công cụ này chúng tôi xây dựng dựa vào các thư viện EMF (Eclipse Modeling Framework) và GEF (Graphical Editing Framework). Với Micraspis, chúng tôi chia quá trình khai báo gồm hai pha, trong đó pha đầu tiên sẽ cho phép người dùng khai báo các phần cứng của thiết bị đeo tay, pha thứ hai cho phép người dùng đặc tả ứng dụng chạy trên thiết bị đeo tay. Cả hai pha này đều thực hiện bằng giao diện đồ họa với các thao tác kéo thả kèm theo. Đồng thời công cụ cho phép khai báo các thuộc tính, các trạng thái và các xử lý của trạng thái một cách dễ dàng. Công cụ cho phép cấu hình các thành phần trên thiết bị một cách tự động và đồng thời kiểm tra các ràng buộc cho tính hợp lý của thiết bị. Chi tiết cho công cụ này, chúng tôi sẽ trình bày ở các phần dưới đây.

3.1 Kiến trúc tổng quát

Trong khung thức Micraspis, chúng tôi đề xuất một hướng tiếp cận sinh mã nguồn tự động cho ứng dụng dựa vào đặc tả phần mềm bằng lược đồ máy trạng thái và kiến trúc thiết bị đeo tay bằng mô hình hóa kết hợp xử lý ràng buộc qua các tập luật. Mô hình này được thiết kế theo 4 tầng như sau (Hình 4.3):



Hình 4.3 Kiến trúc 4 tầng của công cụ Micraspis được đề xuất cho quá trình thiết kế phần cứng & ứng dụng cho thiết bị đeo tay.

Tầng thứ nhất (**Conceptual and Wearable Design**) sẽ cho phép người dùng đặc tả các ứng dụng của chương trình chạy trên thiết bị đeo tay bằng lược đồ máy trạng thái. Mỗi ứng dụng sẽ được biểu diễn là một lược đồ gồm đầy đủ các trạng thái bắt đầu, kết thúc và các điều kiện chuyển đổi trạng thái lẫn nhau. Riêng phần thiết bị phần cứng (thiết bị đeo tay), Micraspis cho phép đặc tả các thành phần cấu hình nên thiết bị từ bo mạch chủ (Arduino R3, UNO, v.v.), các thành phần nhập xuất (Keypad, Button, Sensor, Led, LCD, Buzzz, v.v.), kết nối với các máy chủ thông qua (Bluetooth, Wifi).

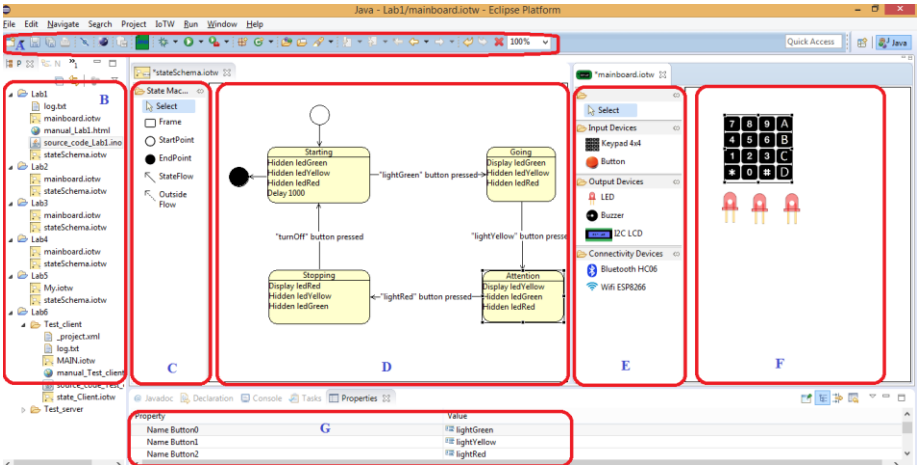
Tầng thứ hai (**Rule Checker**) nhằm kiểm tra các ràng buộc cho các đặc tả theo các tập luật (rules) đã định nghĩa trong hệ thống. Tầng này sẽ giúp cho việc sinh mã nguồn theo các biểu mẫu (template) dễ dàng và thuận tiện hơn ở bước sau.

Tầng thứ ba (**Code Generation**), sinh mã nguồn C cho chương trình ứng dụng được đặc tả bằng lược đồ máy trạng thái và đã được kiểm chứng ở các bước trước. Ở tầng này, hệ thống sẽ dựa vào việc phân tích các thành phần trong nội dung đặc tả phần cứng và ứng dụng nhằm xác định cấu trúc của mã nguồn. Từ cấu trúc này kết hợp với các biểu mẫu sẽ sinh mã nguồn hoàn chỉnh cho chương trình.

Tầng cuối cùng (**Deployement**), ở tầng này các mã nguồn được sinh ra từ các tầng trên sẽ được biên dịch sang ngôn ngữ máy chạy trên bo mạch Arduino tương ứng. Kiến trúc và cấu hình của thiết bị có bo mạch trên đều được khai báo trong phần đặc tả ở tầng đầu tiên với tài liệu hướng dẫn mà công cụ tạo ra.

3.2 Tầng thiết kế phần cứng và đặc tả ứng dụng

Công cụ Micraspis là một dạng plug-in được xây dựng bằng ngôn ngữ lập trình Java. Công cụ này sau khi cài đặt vào Eclipse Modeling Tool sẽ có giao diện như Hình 4.4. Màn hình giao diện chính gồm các phần như sau: Phần **A** là thanh công cụ (toolbar), cho phép người dùng thao tác như tạo project, biên dịch và thực hiện chức năng sinh mã. Phần **B** mô tả toàn bộ nội dung của project được biểu diễn cấu trúc cây project của ứng dụng. Phần này cho phép người dùng xem được kiến trúc của một project thông thường trên Micraspis. Trong phần kiến trúc này sẽ chứa 2 tập tin dùng để mô tả ứng dụng và kiến trúc phần cứng của thiết bị đeo tay. Từ hai tập tin nguồn này, công cụ cho phép sinh mã nguồn cho ứng dụng. Mã nguồn sinh ra của ứng dụng được tổ chức gồm có 03 tập tin. Tập tin thứ nhất có định dạng **.html* là tập tin hướng dẫn cấu hình thiết bị. Tập tin thứ hai có định dạng *.txt lưu lại thông tin hệ thống của thiết bị đeo tay. Tập tin thứ ba có định dạng *.ino là tập tin mã nguồn chính của ứng dụng, được tổ chức bằng ngôn ngữ C++.



Hình 4.4 Giao diện chính cho công cụ Micraspis trong khung thức đề xuất của bài toán sinh mã nguồn cho các ứng dụng chạy trên thiết bị đeo tay.

Phần **C**, thanh palette mô tả các thành phần trong lược đồ máy trạng thái, bao gồm trạng thái đầu, trạng thái kết thúc, trạng thái, dòng chuyển đổi trạng thái, dòng chuyển đổi trạng thái từ yếu tố ngoài. Với thanh palette này, người dùng dễ dàng thiết kế các lược đồ máy trạng thái theo cách kéo thả vào phần **D**. Mỗi thành thành đều có thuộc tính và người dùng có thể hiệu chỉnh thông số của chúng qua cửa sổ (Phần **G**). Phần **D** là cửa sổ cho phép đặc tả ứng dụng chạy trên thiết bị đeo tay bằng lược đồ máy trạng thái. Phần **E** cũng là palette cho phép người dùng chọn các thành phần phần cứng dùng thiết kế cho thiết bị đeo tay. Tất cả các thành phần phần cứng này, người dùng có thể hiệu chỉnh các thuộc tính qua cửa sổ phần **G**. Phần **F** là cửa sổ dùng để hiển thị các thành phần trên thiết bị đeo tay.

3.3 Tầng kiểm tra cú pháp và ràng buộc dựa trên tập luật

Micraspis là công cụ dùng thiết kế và sinh mã, do đó nó cũng yêu cầu các ràng buộc về thiết kế cũng như các cú pháp khai báo. Công cụ sẽ không chuyển sang trạng thái sinh mã nếu như các bước kiểm tra các ràng buộc hay các cú pháp đặc tả vi phạm. Nó chỉ chuyển sang quá trình sinh mã khi tất cả các thiết kế và

khai báo đảm bảo định dạng đúng, các ràng buộc được thỏa mãn. Để thực hiện việc này, công cụ Micraspis sẽ thực hiện hai giai đoạn một cách tự động khi chúng ta thao tác trên giao diện. Giai đoạn thứ nhất, Micraspis sẽ kiểm tra các ràng buộc về phần cứng thông qua bảng mô tả các tập luật dành cho thiết bị đeo tay như số lượng linh kiện nhập, xuất, kết nối. Giai đoạn hai, Micraspis sẽ quét toàn bộ nội dung trong các đặc tả máy trạng thái của ứng dụng.

Sub-phase 1: Kiểm tra ràng buộc

Theo kiến trúc về phần cứng, tất cả các thiết bị đeo tay được cấu trúc gồm một bo mạch chính (hiện công cụ Micraspis hỗ trợ được các dòng Arduino). Micraspis sẽ sắp xếp các linh kiện để gắn kết với bo mạch qua các chỉ số pins cho phù hợp. Hiện linh kiện được phân loại theo linh kiện dùng cho chức năng nhập, xuất, kết nối. Mỗi linh kiện cũng có thông số chân pins để kết nối với bo mạch, trong khi đó chân pins của bo mạch cũng có những phân loại như chân pins dùng cho linh kiện nhập, xuất hay cả hai.

Như những ràng buộc mô tả trong *Bảng 4.2*, Micraspis sẽ đưa ra các qui định về phần cứng như sau: (i) Một thiết bị đeo tay không có quá một linh kiện Keypad/LCD/Wifi; (ii) Mỗi bo mạch sẽ có số lượng chân pins đủ để gắn với các linh kiện; (iii) Chân pins trên bo mạch chủ sẽ có được sắp xếp thiết lập kết nối với từng linh kiện cho tối ưu. Trong quá trình thiết kế phần cứng của thiết bị hay điều chỉnh cấu hình phần cứng phải đảm bảo các ràng buộc này. Nếu như một trong những ràng buộc trên bị vi phạm, công cụ Micraspis sẽ đưa ra các cảnh báo ngay lập tức.

Bảng 4.2: Tập mô tả các ràng buộc trong công cụ Micraspis.

Ràng buộc	**Ý nghĩa**
Số pins tối đa trên bo mạch	Số pins I/O có trên bo mạch chủ Arduino R3 là 14 và số pins cho thành phần nhập Analog là 6
Số pins tối đa cho thành phần Input (Keypad, Button)	Số chân đầu vào kỹ thuật số (digital input) tối đa trên bo mạch Arduino R3 là 12
Số pins tối đa cho thành phần Output (LED, LCD)	Không có nhiều hơn 8 chân đầu ra kỹ thuật số (output digital) trên bo mạch Arduino R3
Tính duy nhất của Keypad	Số bàn phím (Keypad) tối đa được gắn vào bo mạch Arduino R3 là một
Tính duy nhất của LCD	Không được gắn nhiều hơn một màn hình LCD vào bo mạch Arduino R3
Tính duy nhất của bộ thu phát Wifi	Không được gắn nhiều hơn một thành phần Wifi vào bo mạch Arduino R3
Tính duy nhất của kết nối Bluetooth	Không được gắn nhiều hơn một bộ kết nối Bluetooth vào một bo mạch Arduino R3
Số lượng thành phần nhập thông tin (input components)	Có ít nhất một thành phần nhập thông tin (input components) gắn vào bo mạch Arduino R3
Số lượng thành phần xuất thông tin (output components)	Có ít nhất một thành phần xuất thông tin (output components) gắn vào bo mạch Arduino R3

Sub-phase 2: Kiểm tra cú pháp

Như đã giới thiệu, công cụ Micraspis cho phép các nhà phát triển ứng dụng có thể đặc tả các hành vi của thiết bị đeo tay thông qua lược đồ máy trạng thái.

Bảng 4.3: Các cú pháp sử dụng cho các đặc tả trong công cụ Micraspis.

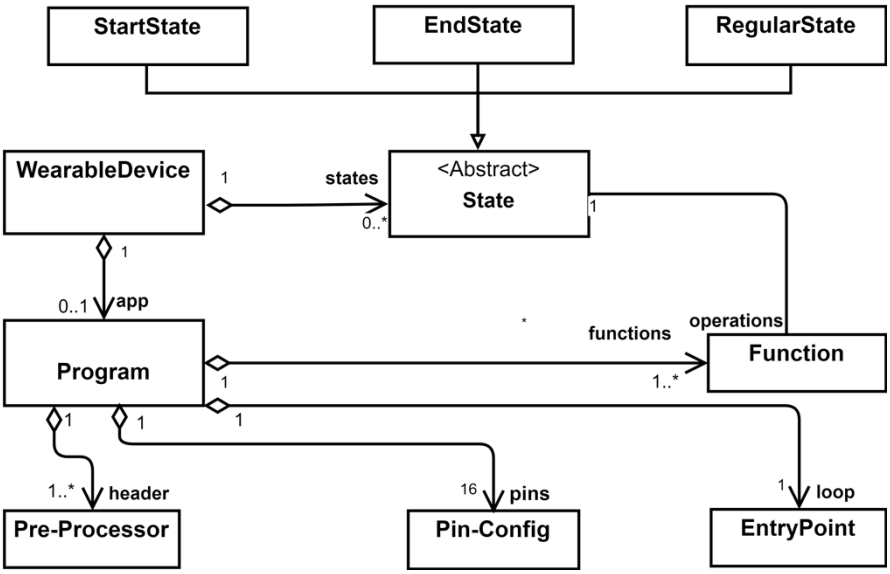
Cú pháp	**Mô tả**	**Thành phần**	**Ví dụ**
`Display` <component>	Bật một component	LED	Display redLED
`Halt` <component>	Tắt một component	LED, I2LCD	Halt redLED
`Blink` <component>	Xóa nội dung	LED, I2CLCD	Blink redLED, Blink myLCD
`Show` <String>	Hiển thị chuỗi	I2CLCD	Show "Welcome"
`Beep` <id>	Phát âm thanh	Buzzer	Beep myBuzzer
<String> button `pressed`	Button <String> được nhấn	Keypad	"Cancel" button pressed
<id> `pushed`	Nhận sự kiện khi button nhả	Button	Arlambutton pushed
<String> `sent`	Gửi dữ liệu qua Wifi	Esp8266 Wifi	bodyTemp sent
<String> `received`	Nhận dữ liệu từ Wifi	Esp8266 Wifi	bodyTemp received

Trong giai đoạn này, các lược đồ được biểu diễn bằng các dòng lệnh để mô tả các thao tác trong từng trạng thái. Đồng thời, những dòng lệnh này phải đảm bảo các văn phạm mà công cụ Micraspis đã định nghĩa trước đó. *Bảng 4.3* mô tả chi tiết các cú pháp và ý nghĩa của các thao tác bên trong các trạng thái. Do công cụ Micraspis cho phép thiết kế phần cứng trước cho nên những ràng buộc về phần cứng đã được kiểm tra ở giai đoạn trước đó. Trong giai đoạn này công cụ Micraspis cũng sẽ ánh xạ các linh kiện mà đã được khai báo trong phần mô tả phần cứng.

3.4 Sinh mã nguồn cho chương trình

Micraspis được xây dựng như là một công cụ plugin vào Eclipse - một môi trường dùng cho việc phát triển các ứng dụng trong máy tính. Chúng tôi đã

chọn lựa khung thức meta-modeling để xây dựng công cụ này. Trong công cụ này, chúng tôi sinh mã cho các ứng dụng chạy trên thiết bị đeo tay với các quan tâm chính sau: (i) sinh mã nguồn cho các trạng thái, trạng thái lặp; (ii) sinh mã nguồn cho các chuyển đổi trạng thái; (iii) sinh mã nguồn cho các nội dung đặc tả bên trong các trạng thái; (iv) sinh mã nguồn cho các khai báo linh kiện, thư viện cũng như việc cấu hình các chân pins cho các linh kiện; (v) cuối cùng sinh tài liệu hướng dẫn lắp ráp linh kiện của thiết bị.



Hình 4.9 Meta-model cho các trạng thái mô tả ứng dụng trong công cụ Micraspis.

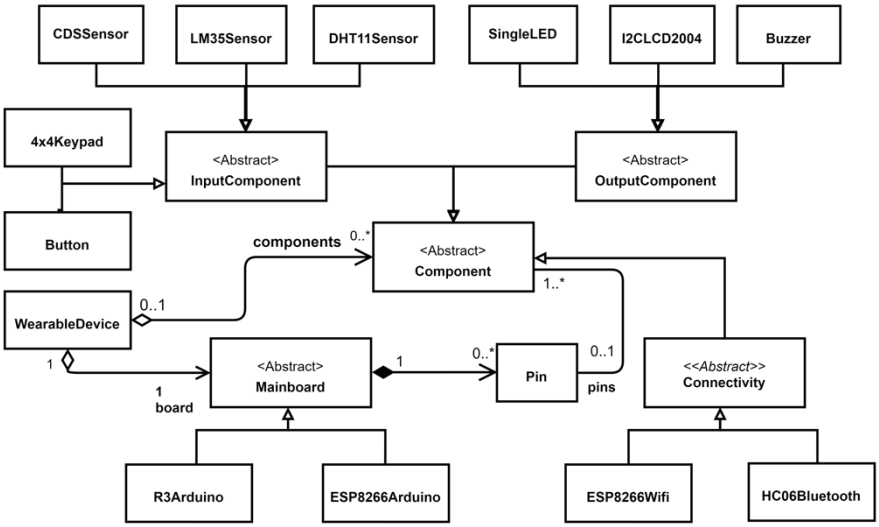
Sinh mã nguồn theo đặc tả cụ thể

Các thiết bị đeo tay thường hoạt động theo cơ chế không đồng bộ. Vòng lặp xử lý chính của thiết bị là lắng nghe để thay đổi các trạng thái. Mỗi trạng thái sẽ có

những hành động, những hành động này có thể được phân tích thành các hàm trong mã nguồn tạo ra. Micraspis sẽ dựa vào các trạng thái để phân tích và xác định các hàm để chuyển đổi. Ví dụ như, cú pháp `OFF greenLED` khai báo trong các trạng thái sẽ chuyển thành lệnh `digitalWrite(greenLED, LOW)`; trong ngôn ngữ lập trình C++.

Sinh mã nguồn theo đặc tả trừu tượng

Công cụ Micraspis phát triển dựa vào kỹ thuật phát triển phần mềm theo hướng kiến trúc MDA (Model-Driven Architecture). Cho nên, việc sinh mã nguồn theo các mô hình sẽ dựa vào nhiều kiến trúc meta-model được định nghĩa sẵn. Micraspis định nghĩa ra hai mô hình Meta-Model cho bài toán gồm một mô hình cho đặc tả phần cứng (Hình 4.10) và một cho ứng dụng chạy trên thiết bị đeo tay (Hình 4.9).



Hình 4.10 Meta-model cho các mô tả phần cứng trong công cụ Micraspis.

Sinh mã theo khai báo trực quan

Một trong những kênh thông tin để hỗ trợ sinh mã nguồn như theo mã cơ bản, theo siêu mô hình (meta-model) và theo thông tin dựa vào các khai báo trực quan. Những khai báo trực quan này được công cụ Micraspis hỗ trợ trong phần thiết kế phần cứng và ứng dụng trên giao diện đồ họa. Trong các bảng thiết kế này, các mô hình sẽ cung cấp thêm các thông tin trực quan cho phần sinh mã nguồn.

3.5 So sánh kết quả

Nhằm đánh giá về mặt thiết kế và hiệu quả của công cụ Micraspis với các công cụ khác mà chúng tôi đã khảo sát. Chúng tôi đã liệt kê ra một số đặc điểm về cách tiếp cận sinh mã nguồn theo hướng kiến trúc MDE như sau:

- *Luôn hỗ trợ một trình soạn thảo cho thiết kế trực quan* trong giai đoạn đầu của phát triển phần mềm / hệ thống.
- *Tách bạch giữa đặc tả phần cứng và các xử lý trong phần mềm ứng dụng*. Đối với đặc điểm này, các công cụ mà chúng tôi khảo sát gần như không hỗ trợ hoặc có hỗ trợ nhưng rất giới hạn. Riêng công cụ Micraspis hỗ trợ đầy đủ đặc điểm này.
- *Hỗ trợ tài liệu thiết kế phần cứng*. Đây được xem là chức năng mà các công cụ chúng tôi khảo sát hầu như không hỗ trợ. Với đặc điểm này, người dùng có thể dễ dàng thiết kế, lắp ráp các thành phần phần cứng cho thiết bị đeo tay.
- *Khả năng tái sử dụng mã nguồn hay kiến trúc phần cứng*. Đây là đặc điểm nổi bật của công cụ Micrapis. Do đặc điểm của công cụ Micrapis là cho phép người dùng đặc tả phần cứng và ứng dụng riêng biệt nên một thiết bị cùng kiến trúc có thể chạy được một ứng dụng trên một thiết bị khác với kiến trúc phần cứng tương đương.
- *Khả năng kiểm tra mô hình đặc tả*. Đa số các công cụ chúng tôi khảo sát đều không có chức năng này. Nghĩa là, mô hình đặc tả cho ứng dụng ngay thời điểm ban đầu được kiểm tra từ cú pháp cho đến tính hợp lý các xử lý

bên trong các trạng thái. Riêng công cụ Micrapis, việc kiểm tra mô hình này được thiết kế bằng cách sử dụng ngôn ngữ đặc tả Alloy. Tuy nhiên, việc triển khai cho chức năng này vào công cụ Micraspis vẫn chưa hoàn thiện.

4 KIỂM CHỨNG

Đánh giá một công cụ phần mềm ngoài các yếu tố về kiểm tra chương trình trong qui trình phát triển phần mềm như kiểm thử phần mềm (testing), chúng ta cần phải đánh giá yếu tố khác như độ hài lòng, hiệu suất công việc, chất lượng phần mềm, v.v. Hiện công cụ Micraspis cũng đã hoàn thiện các chức năng mà luận án đề ra, chúng tôi cũng đã tiến hành khảo sát đánh giá công cụ này với nhiều cách thức khác nhau. Từ đó, giúp cho các nhà khoa học cũng như các kỹ sư phần mềm có cách nhìn khách quan về công cụ này.

4.1 Tỷ lệ độ hài lòng với công cụ Micraspis

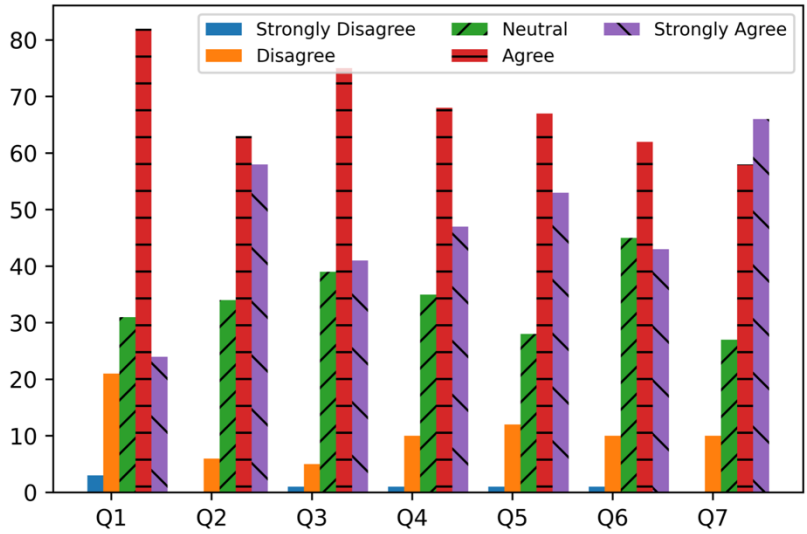
Bảng 5.2: Bảng thống kê các câu trả lời được đưa ra bởi những người được phỏng vấn về khả năng sử dụng của công cụ Micraspis, 19.3 % trong số họ là các kỹ sư có kinh nghiệm trong các lĩnh vực IoT.

Câu hỏi	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**
Strongly Disagree	3	0	1	1	1	1	0
Disagree	21	6	5	10	12	10	10
Neutral	31	34	39	35	28	45	27
Agree	*82*	*63*	*75*	*68*	*67*	*62*	*58*
Strongly Agree	*24*	*58*	*41*	*47*	*53*	*43*	*66*
Tỷ lệ hài lòng %	**65.8**	**75.2**	**72.0**	**71.4**	**74.5**	**65.2**	**77.0**

Bên cạnh độ hài lòng của của người dùng (Bảng 5.2), chúng tôi cũng đánh giá độ hiệu quả của công cụ thông qua tỷ lệ sinh tự động mã nguồn so với mã nguồn hoàn thiện của chương trình. Chúng tôi cũng đánh giá chất lượng của mã

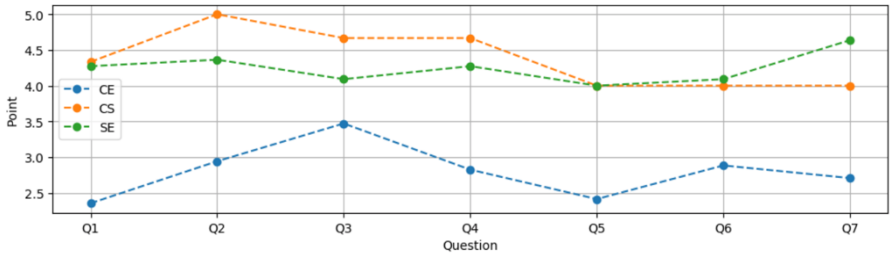
nguồn sinh ra thông qua việc thống kê các ghi chú cũng như tài liệu hướng dẫn lắp ráp thiết bị đeo tay tạo ra một cách tự động. Từ các đánh giá này chúng tôi cũng minh chứng được tính hiệu quả của công cụ Micraspis.

Ngoài ra từ dữ liệu trong Hình 5.3 cũng cho chúng ta có cách nhìn về cách đánh giá công cụ Micraspis sẽ phụ thuộc vào chuyên ngành của người dùng. Theo dữ liệu, chúng ta nhận ra rằng tỷ lệ người dùng thuộc chuyên ngành Khoa học máy tính (CS) đánh giá rất cao về công cụ này. Ngược lại, tỷ lệ những người chuyên ngành về Kỹ thuật máy tính (CE) thì đánh giá không cao cho công cụ này.



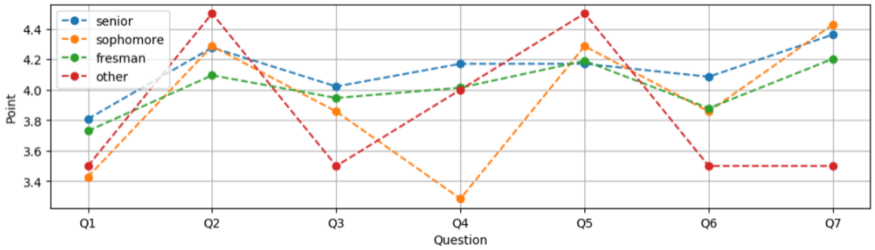
Hình 5.2 Biểu đồ hiển thị các tỷ lệ hài lòng của người dùng với câu trả lời mà chúng tôi thu thập được cho bảng câu hỏi về Micraspis.

Điều này cũng cho chúng ta nhận ra rằng, yêu cầu đặt ra của người chuyên ngành Kỹ thuật máy tính cao hơn nhiều khả năng đáp ứng của công cụ. Như vậy, để đáp ứng cho sinh viên chuyên ngành này, công cụ Micraspis cần phải cải tiến nhiều hơn.



Hình 5.3 Tỷ lệ độ hài lòng của người dùng theo chuyên ngành với công cụ Micraspis.

Hay thông qua dữ liệu từ Hình 5.4 cũng cho chúng ta có cách nhìn về cách đánh giá công cụ giữa nhóm người dùng có kinh nghiệm hoặc chưa có kinh nghiệm. Với dữ liệu này, chúng ta kết luận cách đánh giá của những người có kinh nghiệm (senior) ổn định qua các tiêu chí và độ hài lòng trên mức trung bình (3.8/5) cho tất cả các tiêu chí.



Hình 5.4 Tỷ lệ độ hài lòng của người dùng theo vị trí với công cụ Micraspis.

Ngoài ra, các chỉ số đánh giá của các đối tượng là những người mới tiếp cận trong lĩnh vực IoT với việc phát triển các ứng dụng và các thiết bị đeo tay trong lĩnh vực này (freshman) cũng đặt tỷ lệ cao (trên mức 3.7/5). Dữ liệu này cũng phản ánh được độ hài lòng của đội ngũ (freshman) mà công cụ Micraspis mong muốn hỗ trợ.

4.2 *Tỷ lệ mã nguồn tạo ra tự động*

Để kiểm định tính hiệu quả của công cụ Micraspis, chúng tôi đã thống kê kết quả của mã nguồn tạo ra tự động từ công cụ. Cách thực hiện như sau: chúng tôi dựa vào mã nguồn tạo ra từ công cụ Micraspis của các ví dụ mà sinh viên thực hành (Phụ lục A). Sau đó, chúng tôi cho phép sinh viên bổ sung mã nguồn nếu như chương trình chưa hoàn thiện đúng với yêu cầu. Sau khi hoàn thiện, chúng tôi đếm các dòng mã lệnh của chương trình sau khi phát sinh và sau khi hoàn thiện bằng công cụ đếm dòng lệnh - Code Line. Kết quả của tỷ lệ sinh mã nguồn cho các ứng dụng đạt tỷ lệ trên 65%. Trong đó có những ứng dụng đơn giản, mã nguồn phát sinh có thể đạt tỷ lệ 100% (Bảng 5.3).

Bảng 5.3: Kết quả thử nghiệm công cụ Micraspis cho các project nhỏ.

Ứng dụng	**LOCs sinh mã**	**LOCs hoàn thiện**	**Linh kiên**	**N° of Trạng thái & N° of Tác vụ**	**% Code hoàn thiện**
Traffic Light	58	58	LED, keypad	4 states & 12 tasks	100%
LED Passing	70	90	LED, Button	3 states & 6 tasks	78%
Welcoming Screen	82	99	LCD, Keypad	2 states & 2 tasks	83%
Alarming Devices	102	142	LED, Buzzer, Keypad	3 states & 7 tasks	72%
Wifi Signaling Device	148	197	LED, Wifi	3 states & 8 tasks	75%
iTempFoll	172	265	LED, Wifi, LCD, Keypad, Sensor	7 states & 12 tasks	65%

4.3 *Chất lượng mã nguồn tạo ra từ công cụ Micraspis*

Mã nguồn sinh ra cho ứng dụng từ công cụ Micraspis được tổ chức trong tập tin có định dạng **.ino*. Đây là tập tin chính của ứng dụng chạy trên thiết bị đeo tay. Chúng tôi đã thống kê tỷ lệ % mã nguồn tạo ra tự động so với mã nguồn hoàn thiện. Ngoài ra, chúng tôi cũng đã phân tích chất lượng của mã nguồn thông qua các thông số như các ghi chú (comments) cho mã nguồn hay tập tin hướng dẫn cấu hình thiết bị (Bảng 5.4).

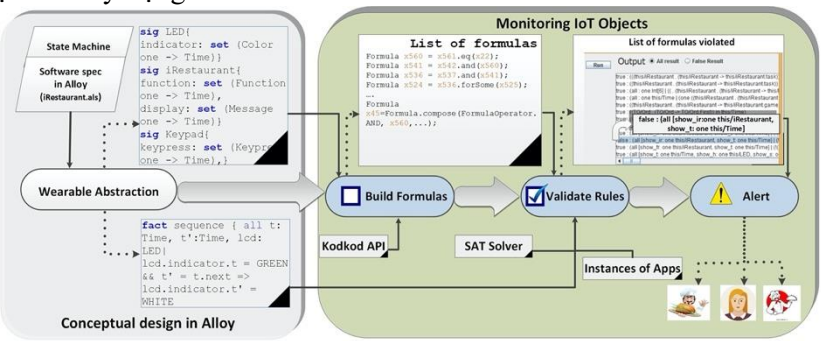
Bảng 5.4: Kết quả sinh mã nguồn với các comments trong các project từ công cụ Micraspis.

Ứng dụng	Head comments	Section comments	Code comments	Inline comments	Task comments
Traffic Light	6	5	3	0	4
LED Passing	8	3	3	0	3
Welcoming Screen	10	6	3	0	4
Alarming Devices	8	5	4	0	3
Wifi Signaling Device	11	8	10	3	7
iTempFoll	24	18	24	6	16

5 MÔ HÌNH HÓA ĐẶC TẢ NGHIỆP VỤ CÁC GIẢI PHÁP THÔNG MINH BẰNG NGÔN NGỮ ALLOY

Alloy là ngôn ngữ cho phép đặc tả yêu cầu và mô hình hóa các hệ thống phần mềm ở mức độ trừu tượng cao. Ngôn ngữ này được phát triển bởi một nhóm nghiên cứu về trừu tượng hóa phần mềm của MIT. Ngôn ngữ Alloy đi kèm với một số thư viện API và đặc biệt là công cụ Alloy Analyzer cho phép thử nghiệm mô hình, tạo mô hình mẫu hay tìm kiếm mâu thuẫn tiềm tàng về mặt ngữ nghĩa trong thiết kế hoặc đặc tả. Trong bài toán sinh mã nguồn cho các ứng dụng chạy trên thiết bị đeo tay, chúng tôi dùng ngôn ngữ Alloy cho hai vấn đề chính:

i) Ngôn ngữ Alloy có thể đặc tả đầy đủ một ứng dụng; ii) Dùng ngôn ngữ Alloy kiểm chứng tính đúng đắn của các đối tượng trong ứng dụng được đặc tả bằng lược đồ máy trạng thái.



Hình 6.6 Khung thức đề xuất để kiểm soát các đối tượng IoT trong ứng dụng đeo tay bằng ngôn ngữ đặc tả Alloy.

Nhằm kiểm chứng khung thức đã đề xuất, chúng tôi đã xây dựng một công cụ nhằm kiểm soát sự thay đổi trạng thái của đối tượng theo thời gian với các sự kiện trong ứng dụng (Hình 6.9). Đầu tiên, chúng tôi đã biểu diễn nội dung ứng dụng *iRestaurant* bằng ngôn ngữ đặc tả Alloy. Trong đó chúng tôi có sử dụng đối tượng Time nhằm sử dụng kỹ thuật hướng thời gian để kiểm soát trạng thái

Hình 6.9: Công cụ kiểm tra các tập luật bị vi phạm trong quá trình mô phỏng ứng dụng *iRestaurant* bằng ngôn ngữ Java.

của các đối tượng. Tiếp theo, chúng tôi kiểm chứng đặc tả này bằng công cụ *Alloy Analyzer* theo khung thức đề xuất Hình 6.6.

6 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết luận

Theo kết quả nghiên cứu trong luận án này, chúng tôi đã xây dựng thành công một phương pháp kết hợp giữa mô hình hóa cho việc đặc tả ứng dụng cũng như thiết bị phần cứng cùng với tập luật cho bài toán sinh mã nguồn. Mã nguồn tạo ra là các ứng dụng chạy trên các thiết bị đeo tay trong các giải pháp thông minh trong lĩnh vực Internet vạn vật. Từ phương pháp này, chúng tôi cũng đã hoàn thiện một công cụ có tên là Micraspis dùng để giải quyết bài toán sinh mã với tỷ lệ sinh mã nguồn hoàn thiện trên 60%. Công cụ cũng nhận được sự đánh giá tốt và khách quan từ người dùng trong các lĩnh vực giáo dục, công nghiệp. Công cụ cũng đã giải quyết được đầy đủ các thách thức mà bài toán đã được nêu ra trong luận án. Các đóng góp chính của luận án được trình bày tóm tắt như sau:

- Xây dựng một khung thức tổng quát cho bài toán sinh mã nguồn trong ngữ cảnh là các ứng dụng chạy trên thiết bị đeo tay thuộc lĩnh vực Internet vạn vật.
- Đề ra giải pháp để tăng tỷ lệ tự động tạo mã nguồn hoàn thiện cho quá trình sinh mã. Đó là giới hạn phạm vi ứng dụng kết hợp việc đặc tả ứng dụng bằng lược đồ máy trạng thái giúp cho việc sinh mã của ứng dụng trở nên hiệu quả hơn.
- Đề xuất cách tiếp cận bổ sung các tập luật vào phần khai báo trong quá trình sinh mã nguồn. Đồng thời, tổ chức cấu trúc của chương trình đặc tả gồm hai tập tin riêng biệt giúp cho việc tái sử dụng mã cũng như tái sử dụng thiết bị phần cứng dễ dàng.

- Đề xuất sử dụng ngôn ngữ Alloy nhằm kiểm tra tính đúng đắn của các mô hình đặc tả cũng như kiểm soát các đối tượng trong ứng dụng theo các tập luật ngay giai đoạn đầu của bài toán sinh mã nguồn.
- Từ các đề xuất trên, luận án xây dựng một công cụ hoàn chỉnh có tên là Micraspis cho việc sinh mã nguồn các ứng dụng chạy trên thiết bị đeo tay trong các giải pháp thông minh thuộc lĩnh vực Internet vạn vật. Đây là đóng góp mang tính thực tế cho ngành công nghiệp phần mềm trong lĩnh vực IoT hiện nay.
- Kiểm chứng công cụ Micraspis trong nhiều môi trường khác nhau như giáo dục và công nghiệp. Quá trình kiểm chứng này được đánh giá qua nhiều tiêu chí như tỷ lệ mã nguồn tạo ra, chất lượng mã nguồn cũng như độ hài lòng của người dùng. Tất cả quá trình này được đánh giá trực tiếp và khách quan tạo tiền đề cho công cụ Micraspis tiếp cận ngành công nghệ phần mềm trong thực tế.

2. Hướng phát triển

Đối với công việc tương lai, nghiên cứu này vẫn còn nhiều vấn đề cần được xem xét, phát triển. Chi tiết các hướng mở rộng, phát triển cho nghiên cứu này là:

- *Mở rộng nền tảng Web:* Hiện công cụ Micraspis được triển khai ở dạng plug-in trên nền tảng Eclipse. Tuy nhiên, với nhu cầu thực tế hiện nay, môi trường Web được nhiều người quan tâm và sử dụng. Cho nên, bài toán sinh mã nguồn tiếp cận theo hướng mô hình hóa và tập luật trên nền tảng Web cũng được xem là hướng mở cho bài toán trong tương lai.
- *Mở rộng các thành phần phần cứng*: Hiện các ứng dụng trên thiết bị đeo tay được mở rộng trên nhiều lĩnh vực, bên cạnh đó nhu cầu xử lý các ứng dụng cũng càng cao dẫn đến cấu trúc phần cứng cũng được mở rộng. Hiện nay, công cụ Micraspis chỉ thực hiện sinh mã nguồn trên các

bo mạch dòng Arduino (ESP8266 và R3) nên việc mở rộng các dòng bo mạch khác là cần thiết. Ngoài ra, các sensor hỗ trợ trong công cụ cũng chỉ dừng ở các dòng sensor cơ bản, chưa được bổ sung nhiều các loại sensor khác nhằm mở rộng phạm vi ứng dụng.

- *Mở rộng phạm vi sinh mã nguồn*: Hiện mã nguồn sinh ra từ công cụ Micraspis chỉ tập trung cho các thiết bị đeo tay. Tuy nhiên, các mã nguồn cho ứng dụng này từ phía máy chủ (server) vẫn còn bỏ ngỏ. Trong khi đó, các giải pháp thông minh trong lĩnh vực IoT luôn có sự đóng góp nhiều từ phía máy chủ. Cho nên, công cụ Micraspis cũng cần có hướng mở rộng nhằm hỗ trợ việc sinh mã nguồn cho ứng dụng phía máy chủ trong tương lai.
- *Một số hướng nghiên cứu mở rộng khác*: Ngoài hướng tiếp cận mô hình hóa đặc tả ứng dụng kết hợp tập luật cho bài toán sinh mã nguồn trên các thiết bị đeo tay, vẫn còn nhiều hướng nghiên cứu đang bỏ ngỏ trên các thiết bị khác. Cụ thể cho các hướng nghiên cứu đó là sinh mã nguồn cho các hệ thống robot, sinh mã nguồn cho ứng dụng chạy trên thiết bị di động trong lĩnh vực IoT.