https://doi.org/10.32431/kace.2018.21.6.003

엘리베이터 시뮬레이터를 활용한 임베디드 어플리케이션 소프트웨어 교수학습방법 연구

고석훈†

요 약

본 논문에서는 임베디드 시스템의 어플리케이션 계층 소프트웨어 학습 도구로 사용할 수 있는 엘리베이터 시뮬레이터의 설계 및 개발 방법과 이를 이용한 교수학습방법을 제안한다. 본 시뮬레이터는 학생들에게 하드웨어와 임베디드 OS 계층의 이슈를 배제한 어플리케이션 계층에서 엘리베이터 시스템의 동작 원리와 제어 방법을 소프트웨어로 구현할 수 있는 환경을 제공하여, 반응(reactive)적이며 실시간(real-time)적인 특징을 갖는 임베디드 어플리케이션 개발 경험을 가질 수 있도록 한다. 아울러 본 논문에서는 시뮬레이터를 이용하여 단계별로 난이도가 높아지는 실습이 포함된 4주간의 임베디드 어플리케이션 소프트웨어 교육 과정을 제시하고, 실제 학생들을 대상으로 교육을 진행한 결과 학습 성취도 점수 83.3점을 얻어 본 교육 과정이 임베디드 어플리케이션 학습에 유의미한 효과가 있음을 입증하였다.

주제어 : 시뮬레이션, 엘리베이터, 임베디드 시스템, 임베디드 어플리케이션

Study on Teaching and Learning Methods of Embedded Application Software Using Elevator Simulator

Seokhoon Ko†

ABSTRACT

In this paper, we propose a design and development method of an elevator simulator that can be used as an embedded application layer software learning tool and a teaching and learning method using it. The simulator provides students with an environment to implement the operating principle and control method of the elevator system in the application layer excluding the issues of hardware and embedded OS layer. This allows students to have a reactive and real-time embedded application development experience. In addition, we present a four-week embedded application software training course with hands-on exercises that add step-by-step functionality using a simulator. As a result of training for actual students, we obtained 83.3 points of learning achievement score and proved that the curriculum has a significant effect on embedded application learning.

Keywords : Simulation, Elevator, Embedded System, Embedded Application

† 정 회 원: 한국외국어대학교 정보산업공학연구소 교수(교신저자)
논문접수: 2018년 10월 10일, 심사완료: 2018년 11월 22일, 게재확정: 2018년 11월 22일

1. 서론

임베디드 시스템은 특정 목적의 장치에 마이크로 프로세서와 소프트웨어를 탑재하여 서비스를 제공하는 시스템을 말하는데, 자동화가 적용된 공장이나 자동차 등의 제어 분야, 인공지능이 적용된 냉장고와 TV등의 가전 분야, 스마트폰과 스마트워치 등의 단말 분야, 그리고 다양한 방식의 유무선 통신장비 분야 등 광범위한 분야에서 응용되고 있다[1].

임베디드 시스템은 응용 분야가 넓은 만큼 사용되는 기술의 종류도 다양하며, 해당 시스템의 목적에 따라 여러 계층의 기술들이 유기적으로 결합되어 최적화된 시스템을 구성하게 된다. 임베디드 시스템을 구성하는 기술을 기능 관점에서 분류하면 표1과 같이 어플리케이션 계층, 임베디드 OS 계층, 하드웨어 계층으로 나눌 수 있다[2].

임베디드 시스템의 중요성이 부각되면서 각 대학의 컴퓨터 관련 학과에서는 임베디드 시스템 교과목을 개설하여 강의를 하고 있지만, 현재 개설된 교과목의 내용을 살펴보면 대부분 하드웨어 계층과 임베디드 OS 계층을 주요 학습 주제를 다루고 있으며, 임베디드 어플리케이션 계층에 대해서는 LED를 표시하거나 액추에이터(actuator)를 동작 시키는 등의 단순한 주제어서 벗어나지 못하고 있다[3].

일반적으로 임베디드 시스템은 반응 시스템(reactive system)이며 실시간 시스템(real-time system)의 특징을 갖는데, 이러한 특징은 임베디드 어플리케이션에 구현된 알고리즘에 의해 처리된다[4]. 따라서 임베디드 시스템을 깊이 있게 이해하기 위해서는 임베디드 어플리케이션의 특성에 대한 학습이 꼭 필요하다고 할 수 있다. 하드웨어 계층과 임베디드 OS 계층의 기술이 다양한 임베디드 시스템에서 범용적으로 사용되는 특징을 갖는 것과 달리 임베디드 어플리케이션은 그 자체가 임베디드 시스템의 기능 사양을 실현하는 역할을 하기 때문에 각 임베디드 시스템에 따라 매우 차별적인 특징을 갖는다. 또한 하드웨어 부품처럼 취급되기 때문에 범용 소프트웨어와 같이 흔하게 볼 수 없으며 기업의 노하우가 들어있기 때문에 기술 내용이 쉽게 공개되지 않는다. 이러한 특징은 임베디드 어플리케이션을 쉽게 경험하기 어려운 이유가 되는데, 이는 학교에서 임베디드 어플리케이션을 교육하기 어려운 이유가 되기도 한다.

본 논문에서는 임베디드 어플리케이션 학습 도구로 사용하기 위한 엘리베이터 시뮬레이터와 이를 이용한 교육 프로그램을 제안한다.

<표 1> 임베디드 시스템의 기술 계층 분류[2]

임베디드 시스템 기술 계층	해당 기술의 예
어플리케이션 계층	리모컨 기능, 세탁기 프로그램, 엘리베이터 제어기 등
임베디드 OS 계층	플랫폼, 커널, 미들웨어, 디바이스 드라이버
하드웨어 계층	프로세서, 입력장치, 구동장치, 디스플레이

본 논문의 구성은 다음과 같다. 2장에서는 임베디드 시스템 교육과 시뮬레이터 및 엘리베이터 시뮬레이터를 이용한 교육에 대한 관련 연구를 소개한다. 3장에서는 교육용 엘리베이터 시뮬레이터의 설계와 구현 방법을 설명하고, 엘리베이터 시뮬레이터를 이용한 교육 프로그램의 구성과 적용 사례를 소개한다. 마지막으로 4장에서는 연구 결과를 요약하고 향후 연구 방향을 제시한다.

2. 관련연구

2.1 임베디드 시스템 교육

한국교원대학교의 김명중과 이태욱은 ARM9 프로세서가 탑재되어 프로그래밍이 가능한 레고 마인드스톰 로봇교구를 이용하여 14주차의 로봇 제어 소프트웨어 학습 프로그램을 제시하였고[3], 동국대학교 신연순은 스포츠 종목으로부터 마인드스톰을 이용한 공학설계과제 선정 방법에 대해 연구하여 다양한 스포츠 로봇과제를 수행하였다[5]. 마인드스톰은 레고 블록을 사용하여 다양한 형태의 로봇을 창작하여 만들 수 있다는 장점이 있지만, 기자재 구입을 위한 비용적인 부담이 크며, 마인드스톰의 비주얼 코딩은 소프트웨어 작성 기능이 제한되므로 소프트웨어 공학적인 측면의 교육에는 적합하지 않은 특징이 있다.

오픈소스 하드웨어로 개발된 아두이노(Aduino)

는 저렴한 가격과 뛰어난 확장성 때문에 프로토타이핑 보드로 각광을 받고 있다. 조선이공대학교의 김송주는 아두이노 보드를 사용한 프로젝트 기반의 임베디드 시스템 실습 교육 과정을 소개하였는데, 전반부에는 다양한 센서와 액추에이터, 통신 기능의 사용법을 익히고 후반부에는 공학 설계 절차에 따라 팀 프로젝트 진행하여 매우 창의적인 결과물을 만들어 내었다[6]. 아두이노를 비롯하여 프로토타입 보드를 활용한 임베디드 시스템 교육은 하드웨어와 소프트웨어를 포함하여 임베디드 시스템의 모든 요소를 직접 만들어볼 수 있다는 점에서 큰 의미가 있지만, 프로젝트 특성상 하드웨어 제작 및 안정화에 많은 시간과 노력이 필요하기 때문에 임베디드 어플리케이션 계층의 소프트웨어에 대해서는 심도 있는 관심을 두지 못한다는 한계가 있다. 본 논문에서는 기존의 임베디드 시스템 교육과정에서 등한시한 임베디드 어플리케이션 계층에 대해 깊이 있게 학습할 수 있는 교육 방안을 제안하고자 한다.

2.2 시뮬레이터를 이용한 교육

제한된 시간과 비용 내에서 진행해야 하는 실습 교육에 있어서 소프트웨어로 구현한 시뮬레이터는 물리적인 하드웨어를 구현하는 과정을 거치지 않고 시스템의 동작 과정 및 결과를 확인할 수 있는 효과적인 방법이다.

고려대학교 김우찬이 구현한 논리회로 시뮬레이터는 논리연산과 논리회로 교육의 보조도구로써 하드웨어를 구현하는 것에 비해 매우 간단한 조작으로 논리회로를 만들 수 있으며, 각 게이트를 지나는 와이어를 점등하는 시각적 효과를 제공하여 실제 하드웨어에서는 확인하기 어려운 게이트의 중간 동작 결과를 쉽게 확인시켜주어 교육 효과를 높일 수 있음을 보여주었다[7]. 성균관대학교의 박선주와 안성진은 자동차 동작 제어에 대해 자동차를 하드웨어로 구현하는 대신 시뮬레이션 도구인 ProDG를 활용하여 소프트웨어로 구현하고 가상 환경에서 3D 그래픽으로 동작 결과를 확인하는 교육 프로그램을 설계하였다[8]. 이러한 가상환경을 이용한 시뮬레이션 실험은 실제 하드웨어를 사용하는 실험에 비해 훨씬 적은 노력과 비용으로 결과를 확인할 수 있는 장점이 있으며, 또한 빠른 시간 내에 결과를 확인할 수 있기 때문에 학습에 참여하는 학생들의 몰입감을 높이는 효과도 얻을 수 있다. 본 논문에서는 임베디드 시스템의 어플리케이션 계층 소프트웨어 학습에 있어서 시뮬레이터를 활용하는 교육방법을 도입하여 실제 실험용 하드웨어를 준비하기 어려운 엘리베이터 시스템의 어플리케이션 소프트웨어 개발 경험을 제공하고자 한다.

2.3 엘리베이터 시뮬레이터

엘리베이터는 전기, 전자, 기계와 관련된 약 5만여 개의 부품으로 구성되는 복잡한 임베디드 시스템으로서[9] 여러 가지 관점에서 연구 대상이 되는 만큼 다양한 목적의 엘리베이터 시뮬레이터가 개발되었다. Peters Research의 Elevate는 건축 공학적인 측면에서 엘리베이터의 성능을 분석하는 시뮬레이터로써 다양한 공학 분야의 이론을 근거로 엘리베이터 동작을 정교하게 시뮬레이션하여 신규 건물에 알맞은 엘리베이터의 개수, 크기 및 속도를 선택하는데 사용되는데[10], 이러한 전문가용 시뮬레이터를 사용하기 위해서는 많은 비용을 지불해야 하고, 사용 방법도 복잡하기 때문에 교육용으로 활용하기에 적합하지 않다.

버지니아 공대에서는 C/C++ 프로그래밍 과목의 프로젝트로 학생들에게 텍스트 파일로 주어진 엘리베이터의 동작 명령에 대해 결과를 출력하는 엘리베이터 시뮬레이터를 작성하도록 하였고[11], 멜버른대학교의 Programming Usable Interface 과목에서는 제시된 알고리즘에 따라 세대의 엘리베이터가 동작하는 시각적 시뮬레이터를 구현하도록 하였다[12]. 미시건 주립대에서는 C프로그래밍 과제로 엘리베이터 시뮬레이션 라이브러리와 기본 동작에 대한 알고리즘을 제공하고 모터, 브레이크 레벨의 제어함수를 구현하도록 하였다[13]. 이와 같이 엘리베이터 시뮬레이터는 여러 학교에서 컴퓨터 프로그래밍 과목의 과제로 사용되었지만, 대부분 엘리베이터 기능의 일부분에 대해 프로그램을 작성하는 일회성 과제의 주제로 사용되었으며 엘리베이터 시스템 전체를 이해하기 위한 시뮬레이션의 목적으로 활용되지는 않았다.

본 논문에서는 엘리베이터 시스템을 구성하는 주요 구성요소를 모두 포함하는 시뮬레이터를 제작하고 시각화 기능을 구현하여 학생들이 임베디드 어플리케이션 프로그래밍 학습용 교구로 활용할 수 있는 방안을 제안하고자 한다. 이 시뮬레이터를 통해 학생들은 엘리베이터 시스템의 동작 원리를 이해하게 되며 주요 기능을 직접 구현함으로써 임베디드 어플리케이션 작성 능력을 향상시킬 수 있을 것으로 기대한다.

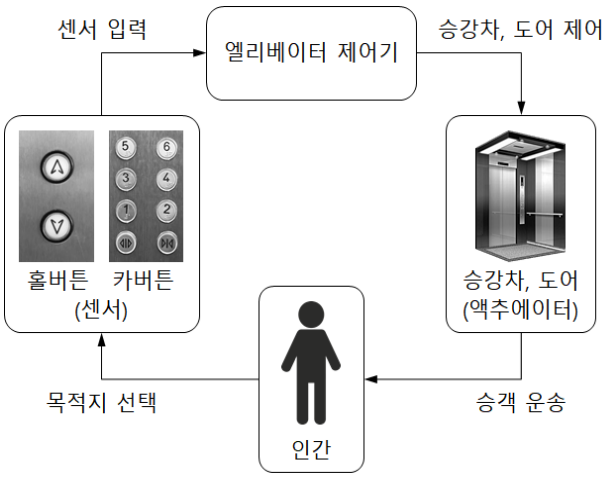
3. 본론

3.1 엘리베이터 시뮬레이터

3.1.1 엘리베이터 시스템 모델

엘리베이터 시스템은 자동차와 비교될 만큼 고도로 복잡한 임베디드 시스템이지만[14], 누구나 매일 한번 이상 사용할 만큼 친숙한 장치이기 때문에 다른 임베디드 시스템에 비해 매우 쉽게 동작 원리를 이해할 수 있다. 본 논문에서 제안하는 엘리베이터 시뮬레이터는 어플리케이션 계층의 임베디드 소프트웨어의 구조와 동작에 대한 교육을 목적으로 하므로 가장 대중적인 엘리베이터 동작 방식을 소프트웨어 관점에서 추상화하여 모델링한다.

엘리베이터 시스템은 그림 1과 같이 사용자가 누른 버튼 센서의 입력에 대해 엘리베이터 제어기가 승강차의 움직임을 제어하는 반응 시스템 모델(reactive system model)로 정의할 수 있다. 이 모델에서 센서의 역할을 하는 버튼은 엘리베이터가 멈추는 각 층에 있는 위/아래 방향 홀버튼, 그리고 엘리베이터 승강차 내부에서 도착층을 지정하는 층버튼과 도어를 여닫는 open/close 버튼이 포함된 카버튼으로 구성되는데, 사용자가 가고자 하는 층을 누르면 그 정보를 엘리베이터 제어기에 전달하는 역할을 한다. 액추에이터에 해당하는 승강차는 엘리베이터 제어기가 지정하는 도착층으로 이동하고, 승강차가 멈추면 도어가 열리고 닫히는 동작을 한다. 엘리베이터 제어기는 버튼 입력 정보를 바탕으로 승강차 제어 알고리즘을 수행하여 승강차의 이동을 스케줄링 하는 지능적이며 핵심적인 역할을 담당한다.



[그림 1] 엘리베이터 반응 시스템 모델

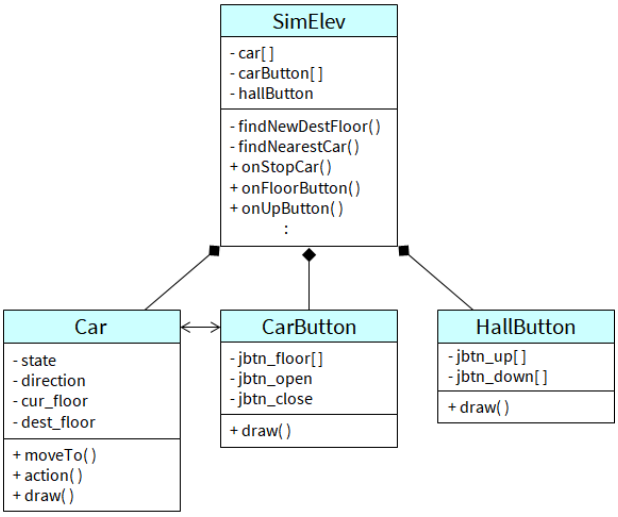
실제 엘리베이터 시스템은 설치 장소와 목적에 따라 스케줄링 알고리즘도 다르고 세부적인 동작 방식도 다르게 설정할 수 있으므로[9], 논문에서 다루는 엘리베이터 시스템의 동작에 대한 기준을 정해놓을 필요가 있다. 본 논문에서 개발하는 엘리베이터 시뮬레이터가 따르는 엘리베이터 동작 기준은 다음과 같이 정의한다.

- 승강차는 1층부터 시뮬레이터에 지정된 최고층까지 상하로 이동한다.
- 승강차의 도어는 승강차가 이동 중에는 열리지 않고 특정 층에 멈췄을 때만 열린다.
- 승강차 안에 있는 카버튼은 1층부터 최고층까지의 층버튼과 open/close 버튼으로 구성된다. 층버튼은 선택 및 선택 해제 할 수 있고, open/close 버튼은 선택만 할 수 있다.
- 승강차가 멈추는 모든 층에는 위/아래 방향의 홀버튼이 있는데, 1층에는 위쪽 방향의 홀버튼만 있고, 최고층에는 아래쪽 방향의 홀버튼만 존재한다. 홀버튼은 선택 및 선택 해제 할 수 있다.
- 엘리베이터 시스템에는 승강차가 1대이거나 또는 2대 이상일 수 있다.
- 승강차가 2대 이상인 경우, 승강차는 서로 연동하여 동작한다. 따라서 승강차마다 대응되는 카버튼이 하나씩 존재하지만 홀버튼은 오로지 한 세트만 존재한다.

● 승강차가 2대 이상인 경우, 홀버튼으로 승강차를 호출하면 현재 승강차의 위치와 이동방향을 기준으로 가장 먼저 도착할 수 있는 승강차가 선택된다.

3.1.2 시뮬레이터 설계

엘리베이터 시뮬레이터는 엘리베이터를 구성하는 승강차, 카버튼, 홀버튼, 그리고 엘리베이터 제어기를 Car, CarButton, HallButton, 그리고 SimElev 객체로 구분하여 설계하였다. 그림 2는 각 객체의 관계를 UML로 표현한 것이다.



[그림 2] 엘리베이터 시뮬레이터 UML

엘리베이터 시뮬레이터의 동작 과정은 엘리베이터 반응 시스템 모델의 동작 순서를 그대로 따른다. 사용자가 화면에 나타난 홀버튼을 눌러 승강차를 호출하거나 카버튼을 눌러 목적층을 지정하면 CarButton 객체와 HallButton 객체에서 SimElev 객체로 목적층 정보를 이벤트로 전달하고, SimElev 객체는 findNewDestFloor 메소드에서 도착층 검색 알고리즘을 실행하여 도착층을 결정하여 Car 객체에게 넘겨주고, Car 객체는 action 메소드를 반복적으로 실행하면서 도착층을 향해 승강차를 이동하는 순서로 동작한다.

이러한 반응 동작을 완성하기 위해 구현해야 하는 모듈은 크게 이벤트 처리기 부분과 핵심 알고리즘 부분으로 구성된다. 표 2는 버튼을 선택하거나, 승강차가 도착층에 도착하는 등 시뮬레이터 동작 과정에서 사용되는 이벤트의 처리기를 설명한 것이고, 표 3은 시뮬레이터에서 사용되는 3가지 제어 알고리즘의 역할에 대해 정리한 것이다.

<표 2> 엘리베이터 시뮬레이터에 정의된 이벤트의 종류와 이벤트 처리기의 역할

이벤트 주체	이벤트 처리기	이벤트 발생 시점과 이벤트 처리기의 역할
Car	onStopCar	승강차가 도착층에 도착하면, 도어를 여는 작업을 한다.
	onDoorOpened	도어가 완전히 열린 상태가 되면, 도어 타이머를 설정하여 타임아웃이 될 때 도어가 닫히도록 한다.
	onDoorClosed	도어가 완전히 닫힌 상태가 되면, 도착층을 검색하고 이동을 시작한다.
CarButton	onFloorButton	카버튼의 층버튼이 선택/해제되면, 도착층을 다시 검색한다.
	onOpenButton	승강차가 정지된 상태에서 카버튼의 open 버튼이 눌리면, 승강차의 도어를 연다.
	onCloseButton	승강차가 정지된 상태에서 카버튼의 close 버튼이 눌리면, 승강차의 도어를 닫는다.
HallButton	onUpButton	홀버튼의 위쪽 버튼이 선택/해제되면, 승강차를 호출하거나 호출을 취소한다.
	onDownButton	홀버튼의 아래쪽 버튼이 선택/해제되면, 승강차를 호출하거나 호출을 취소한다.

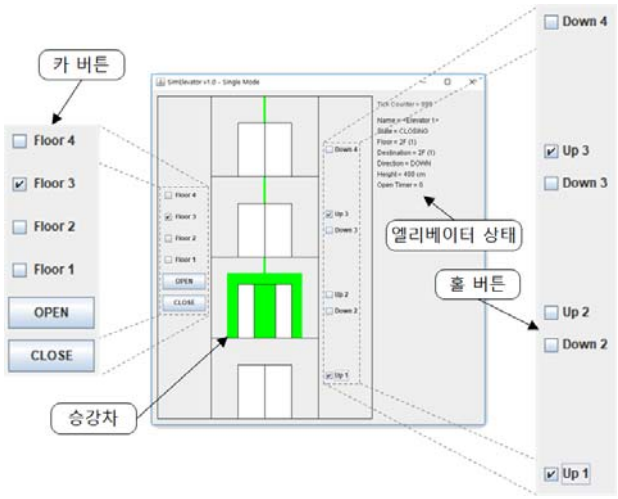
<표 3> 엘리베이터 시뮬레이터의 세 가지 제어 알고리즘

알고리즘	알고리즘의 역할
Car.action	승강차의 상태 변화를 관리하며, 현재 상태에 따라 승강차를 도착층으로 이동시키거나 도어를 열고 닫는 동작을 하고 각 동작이 완수되는 순간 대응되는 이벤트를 발생시킨다.
findNewDestFloor	홀버튼과 카버튼의 선택 정보와 현재 승강차의 위치, 이동방향을 바탕으로 다음번에 정차해야 하는 도착층을 결정한다.
findNearestCar	2대 이상의 승강차를 운영하는 환경에서 홀버튼으로 승강차를 호출했을 때 각 승강차의 위치, 이동방향, 승강차를 호출한 층의 높이를 바탕으로 가장 빠른 시간에 도착할 수 있는 승강차를 결정한다.

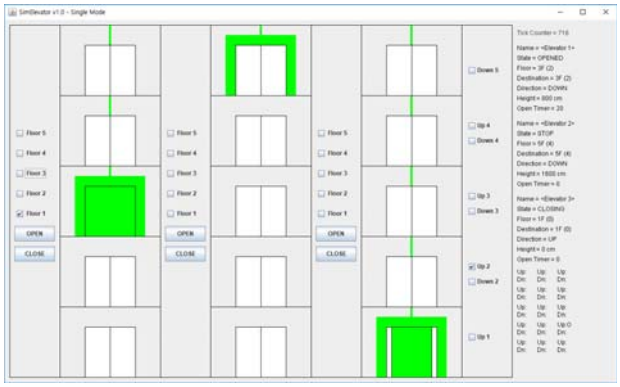
3.1.3 시뮬레이터 구현

설계를 마친 엘리베이터 시뮬레이터는

SimElevator란 이름의 Java 프로그램으로 구현하였는데, 설계 단계에서 정의한 각 객체에 대응되는 이벤트 처리기(표 2)와 세 가지 제어 알고리즘(표 3)을 작성하여 주요 모듈의 구현을 완료하였다. 또한 승강차가 1대인 경우에 대한 Single-Mode 시뮬레이터를 먼저 구현하고, 이를 확장하여 승강차가 2대 이상인 경우에 대한 Multi-Mode 시뮬레이터를 별도로 구현하였는데, 시뮬레이션 하는 건물의 층수와 승강차 대수를 소스코드의 상수 값으로 설정함으로써 손쉽게 변경이 가능하도록 하였다.



(a) SimElevator - Single Mode



(b) SimElevator - Multi Mode

[그림 3] 엘리베이터 시뮬레이터 실행 장면

엘리베이터의 동작을 시각적으로 보여주고 버튼 입력 기능을 사용하기 위해 Java 표준 그래픽 패키지인 AWT와 Swing을 사용하여 GUI를 구현하였다. GUI가 완성된 시뮬레이터의 모습은 그림 3과 같다. 그림 3-(a)는 Single-Mode 실행 중에 승강차가 2층에 멈추고 도어가 열리는 장면을 보여주고 있는데, 화면의 가장 큰 사각형은 4층 건물을 나타내며, 각 층의 중앙에는 엘리베이터 도어가 위치하며, 도어 뒤쪽으로 녹색 승강차가 매달려 있고, 승강차 왼쪽에는 카버튼이 위치하며 오른쪽에는 각 층의 홀버튼이 위치하고 화면 우측 상단에는 현재 시뮬레이터의 각종 상태 변수의 값을 자세히 보여주고 있다. 그림 3-(b)는 Multi-Mode 실행 화면으로 5층 건물에 3대의 승강차가 운영되는 모습을 보여주고 있는데, 복수개의 승강차가 연동되어 운영되므로 카버튼은 각 승강차마다 하나씩 대응되지만 홀버튼은 엘리베이터 시스템 전체에 한 세트뿐인 것을 알 수 있다.

3.2 임베디드 어플리케이션 교육 과정

3.2.1 교육 과정 제작

엘리베이터 시뮬레이터를 사용한 교육 프로그램의 목표는 하드웨어와 임베디드 OS 계층의 이슈를 배제하고, 임베디드 어플리케이션 계층에서 엘리베이터 시스템의 동작 원리와 제어 방법을 이해하고 그것을 소프트웨어로 구현함으로써 반응(reactive)적이며 실시간(real-time)적인 특징을 갖는 문제에 대한 개발 경험을 갖도록 하는 것이다. 이러한 목표는 시뮬레이터의 이벤트 처리기(표 2)와 세 가지 제어 알고리즘(표 3)의 구조와 동작 원리를 공부하고, 학생들이 직접 구현하는 과정을 통해 성취될 수 있다. 엘리베이터 시뮬레이터를 개발하는 과정에서 모든 이벤트 처리기와 핵심 알고리즘을 구현하여 시뮬레이터의 동작을 검증하였지만, 그것은 교육 과정 중에 참고용으로만 사용될 것이며, 교육용으로 학생들에게 제공되는 시뮬레이터는 이벤트 처리기와 세 가지 제어 알고리즘을 미구현 상태로 두고 수업 과정을 통해 학생들이 직접 작성하도록 한다.

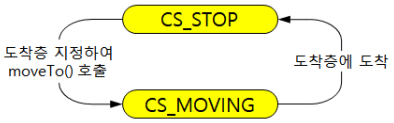
엘리베이터 시뮬레이터를 완성하기 위해 공부하고 이해해야 하는 내용의 양이 적지 않으므로 학생들에게 모든 기능을 한 번에 구현하라고 요

구하기는 어렵다. 따라서 엘리베이터의 기능을 여러 단계로 나누어 쉬운 기능부터 먼저 공부한 다음 소프트웨어를 작성하고, 단계 별로 기능을 추가하여 최종 단계에서 모든 기능을 완성하는 방식의 단계별 교육 프로그램을 작성하였다. 최종적으로 완성된 SimElevator를 이용한 임베디드 어플리케이션 교육 과정은 표 4와 같이 총 7개 단원으로 나누어 약 4주간의 수업으로 진행할 수 있는 PPT 강의 자료와 실습 문제로 구성하였다. 4주간의 교육 과정 동안 총 5회의 실습을 진행하는데 단순한 기능부터 점진적으로 기능을 추가하여 3주차의 실습 4에서 Single-Model 엘리베이터의 기능을 완성하고, 4주차의 마지막 실습 5에서 엘리베이터 군관리 제어[15] 개념을 포함하는 Multi-Mode 엘리베이터의 동작을 구현하도록 하였다. 그림 4는 각 단계별로 승강차를 제어하는 Car.action 알고리즘의 상태 전이도를 나타낸다. 엘리베이터 시스템의 기능이 추가되면서 처리해야 하는 상태의 개수와 이벤트의 종류가 늘어나고, 그에 따라 복잡도가 높아지는 것을 확인할 수 있다.

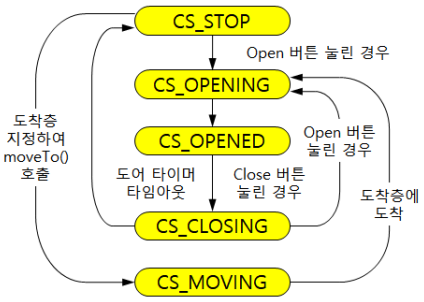
본 논문에서 제안하는 임베디드 어플리케이션 교육 프로그램은 일반적으로 한 학기 동안 진행되는 임베디드 시스템 교과목의 일부로 사용되는 것을 목표로 하였는데, 총 16주 동안 하드웨어 계층, 임베디드 OS 계층, 그리고 임베디드 어플리케이션 계층에 대한 교육을 진행해야 하므로 어플리케이션 계층의 교육 과정으로 4주를 배정한 것은 실제 수업에 적용하기 매우 적합한 기간이라 생각된다.

3.2.2 교육 수행 평가

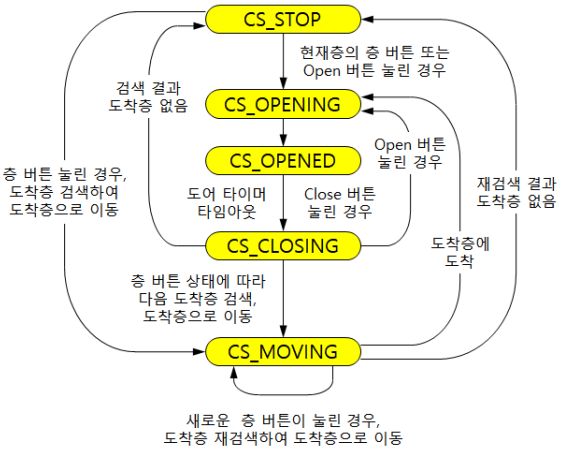
SimElevator를 이용한 임베디드 어플리케이션 교육 프로그램은 컴퓨터 관련 학과에 개설된 전공 선택과목인 시스템프로그래밍 강좌에서 실제 수업의 일부로 진행하였다. 해당 강좌에는 총 9명의 4학년 학생이 참가하여 교육 프로그램 강의를 듣고 5개의 소프트웨어 개발 실습을 진행하였다. 교육 프로그램을 마친 후 표 5에 정리한 평가 기준을 사용하여 학생들이 제출한 실습 결과물에 대해 학습 성취도 평가를 진행하였다.



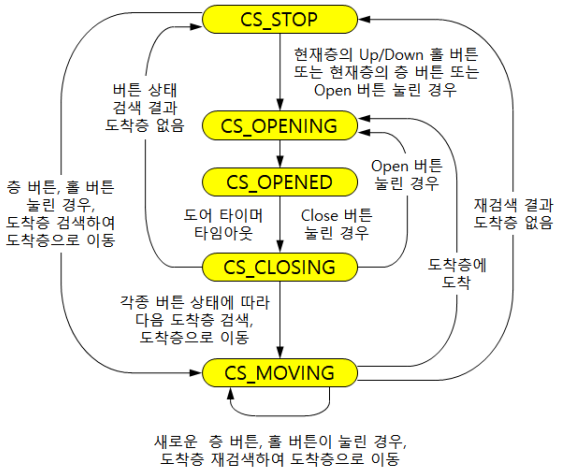
(a) 단순 이동 기능의 상태 전이도



(b) 도어 여닫기 기능이 추가된 상태 전이도



(c) 카버튼 호출에 대응하는 상태 전이도



(d) 카버튼과 홀버튼에 대응하는 상태 전이도

[그림 4] 교육 과정별 승강차의 상태 전이도

<표 4> SimElevator를 이용한 임베디드 어플리케이션 교육 과정

주차	단원	교육 내용
1주	1장	엘리베이터 시스템 학습 • 엘리베이터의 역사, 동작 원리 • 엘리베이터의 기본 동작: (1) 엘리베이터 호출, (2) 엘리베이터 이동 • 엘리베이터 컨트롤 문제: 동적 스케줄링, 군관리 제어 Single-Mode SimElevator 구조 학습 • 주요 클래스: Car, CarButton, HallButton, SimElev 클래스
	2장	실습 1: 층버튼을 누르면 해당 층으로 승강차가 이동하는 기능 구현 (그림 4-(a)) • onFloorButton 이벤트 핸들러: 도착층을 지정하여 Car.moveTo 메소드 호출 • Car.moveTo 메소드: 도착층이 현재층이 아니면 CS_MOVING 상태로 변경하고 승강차 이동 시작 • Car.action 메소드: CS_MOVING 상태면 엘리베이터 이동. 도착층에 도착하면 CS_STOP 상태로 변경하고 onStopCar 이벤트 핸들러 호출 • onStopCar 이벤트 핸들러: 도착층의 층버튼 선택 해제
	3장	실습 2: 엘리베이터 도어가 열리고 닫히는 기능 구현 (그림 4-(b)) • 도어 상태 추가: Car.action 메소드에 CS_OPENING, CS_OPENED, CS_CLOSING 상태 추가 구현 • 도착층 도착시 도어 자동 열림: CS_MOVING 상태에서 도착층에 도착하면 CS_OPENING로 변경 • 승강차 정차시 열림/닫힘 버튼에 의해 도어 열거나 닫음: onOpenButton, onCloseButton 이벤트 핸들러와 Car.open, Car.close 메소드 구현 • 도어가 열려있는 상태에서는 승강차가 이동하지 않도록 고정: CS_OPENING, CS_OPENED, CS_CLOSING 상태에서 다른 상태로 변경되지 않도록 수정
2주	4장	실습 3: 복수개의 층버튼이 눌린 경우, 가까운 순서로 차례대로 정차하는 기능 구현 (그림 4-(c)) • 층버튼 상태에 따른 도착층 검색 알고리즘 findNewDestFloor 작성: (1) 현재 이동 방향에서 눌려 있는 가장 가까운 층버튼 검색 (2) 검색 결과가 없으면 반대 방향으로 가장 가까운 층버튼 검색 • 도착층에 도착하여 도어가 열렸다 닫히면 도착층 검색 알고리즘 재실행: onDoorClosed 이벤트 핸들러에 기능 구현 • 승강차 이동 중에 층버튼을 선택 또는 해제하면 도착층 검색 알고리즘 재실행: onFloorButton 이벤트 핸들러에 기능 구현
3주	5장	실습 4: 홀버튼의 승강차 호출 기능 구현 (그림 4-(d)) • 층버튼과 홀버튼에 모두 대응하도록 도착층 검색 알고리즘 findNewDestFloor 수정: 다음 순서에 의해 도착층 검색 (1) 현재 이동 방향 쪽에서 같은 방향으로 눌려 있는 가장 가까운 층버튼 또는 홀버튼 검색 (2) 현재 이동 방향 쪽에서 반대 방향으로 눌려 있는 가장 먼 층버튼 또는 홀버튼 검색 (3) 이동 방향을 반대로 바꾸고 (1), (2) 검색 과정 반복 수행 • 승강차 이동 중에 홀버튼을 선택 또는 해제하면 도착층 검색 알고리즘 재실행: onUpButton, onDownButton 이벤트 핸들러에 기능 구현 • 도착층에 도착했는데 홀버튼이 눌려져 있는 경우, 홀버튼 해제: onStopCar 이벤트 핸들러 수정
4주	6장	엘리베이터 군관리 제어 방법 학습 • 군관리 제어의 개념 • 군관리 제어의 종류: 사용 정보, 제어 목적, 지능 제어 Multi-Mode SimElevator 구조 학습 • Car, CarButton, HallButton 클래스는 변경 없음 • SimElev 클래스의 변경 사항 학습
	7장	실습 5: Multi-Mode SimElevator 홀버튼의 승강차 호출 기능 구현 • 각 승강차에 독립적인 홀버튼 호출 플래그 추가: 각 승강차에 Up/Down 호출 플래그 추가, 각 홀버튼에 할당된 승강차를 저장하는 속성 추가 • 승강차를 호출한 층에서 가장 가까운 승강차 검색 알고리즘 findNearestCar 작성: 다음 순서에 의해 가장 가까운 승강차를 검색 (1) 호출한 층에 정지해 있는 승강차 (2) 호출 방향의 반대쪽에서 다가오는 승강차 중 가장 가까이 있는 승강차 (3) 호출 방향과 반대방향으로 이동하는 승강차 중 가장 많이 진행한 승강차 (4) 호출 방향에서 멀어지는 승강차 중 가장 많이 진행한 승강차 • 홀버튼을 누르면 가장 가까운 승강차를 선택하여 호출 플래그를 설정하도록 onUp/DownButton 이벤트 핸들러 수정: 선택된 승강차의 호출 플래그를 설정하고, 선택된 승강차를 홀버튼에 저장. 홀버튼을 취소하는 경우 호출 플래그 해제 • 호출 플래그를 참고하여 도착층을 결정하도록 도착층 검색 알고리즘 findNewDestFloor 수정: 각 승강차에 대해 홀버튼 상태 대신 승강차 별 호출 플래그를 참조하도록 수정

<표 5> 학습 성취도 평가 기준

평가 (점수)	실습 결과의 평가 기준
A (100)	실습 과제에 제시된 모든 요구사항을 만족함 • 모든 버튼 선택에 대해 상태 전이도의 시나리오에 따라 승강차가 동작하도록 구현 • 승강차 호출 시 가장 가까운 승강차를 선택하는 승강차 알고리즘 구현
B (75)	실습 과제의 대부분의 요구사항을 만족함 • 버튼 선택에 대해 승강차가 해당 층으로 이동하기는 하지만, 불필요한 층을 방문 하는 등 최적화 되지 않은 동작을 수행함 • 승강차 호출 시 가장 가까운 승강차를 선택하지 못하고 멀리 있는 승강차가 선택됨
C (50)	실습 과제의 주요 요구사항을 만족하지 못함 • 특정 버튼 선택에 대해 상태 전이도의 시나리오에 따라 동작하지 못함
D (25)	실습의 대부분의 요구사항을 만족하지 못함 • 시뮬레이터 프로그램 실행 도중 심각한 오류로 프로그램이 중지됨

4주간의 교육 프로그램에 대한 학습 성취도 평가 결과는 표 6과 같다. 9명의 학생 중에 6명 학생이 과제에 제시된 모든 요구사항을 만족하여 A로 평가되었으며, 3명의 학생이 각각 1명씩 B, C, D로 평가되었다. 이러한 평가 결과를 표 5의 평가 기준에 따라 점수로 환산하면 평균 83.3점이 된다. 모든 학생이 실습 과제를 완수하지는 못했지만 성취도 평가 결과 83.3점을 얻은 것은 본 논문에서 제안한 엘리베이터 시뮬레이터를 이용한 교육 프로그램이 반응적이며 실시간적인 특징을 갖는 임베디드 어플리케이션 학습에 유의미한 효과가 있음을 입증하는 점수라 할 수 있다.

<표 6> 학습 성취도 평가 결과

평가 (점수)	A (100)	B (75)	C (50)	D (25)	합계
인원	6	1	1	1	9
비율	66.7%	11.1%	11.1%	11.1%	100.0%
환산점수	66.7	8.3	5.6	2.8	83.3

다음 목록은 교육 프로그램에 참가한 학생들이 자유롭게 작성한 과제 수행 후기를 요약한 것이다. 학생들이 작성한 후기와 교육 프로그램의 내용 비교하여 분석하면 본 교육 프로그램을 통해 다음과 같은 부수 효과를 기대할 수 있을을 알 수 있다. 첫째, 이벤트 핸들러를 구현하는 과정을 통해 객체지향 언어 기반의 이벤트-드리븐(event-driven) 프로그램 설계 및 개발 능력을 키울 수 있다. 둘째, 승강차의 상태를 제어하는 Car.action 알고리즘 구현을 통해 상태-기계(state-machine)를 이용한 제어 프로그램 동작 원리를 경험할 수 있다. 셋째, 단계별 개발 과정을 통해 복잡한 문제를 분할하여 정복하는 원리를 터득하여 문제 해결 능력을 향상시킬 수 있다.

- 단계별로 요구사항에 맞게 코드를 작성하니 원하는 결과를 얻을 수 있었다. 하지만, 전체 요구사항을 한 번에 받았다면 프로그램을 완성할 수 없었을 것이다.
- 실습 과제를 처음 보았을 땐 막막했지만 PPT 자료를 보면서 차근차근 코드를 작성하니 원하는 코드를 완성할 수 있었다.
- 기존 코드에 기능을 추가하는 것이 어렵게 느껴졌지만, 기존 코드를 학습하면서 엘리베이터 동작의 전체적인 흐름을 파악하니 문제를 해결 할 수 있었다.
- 객체와 객체를 연결하는 방법에 대해 알게 되었고, 현재 진행 중인 프로젝트의 코드를 좀 더 나은 방식으로 수정하는 방법을 생각하게 되었다.
- 객체지향 프로그래밍의 장점을 알게 되었다. 메소드 호출로 객체와 객체를 연동하는 법을 알게 되니 어려울 것 같은 프로그램도 비교적 쉽게 만들 수 있었다.

4. 결론

본 논문에서는 임베디드 시스템의 어플리케이션 계층의 소프트웨어 학습 도구로 사용할 수 있는 엘리베이터 시뮬레이터의 설계 및 개발 방법과 이를 이용한 교육 프로그램을 제안하였다. 본 시뮬레이터는 학생들에게 하드웨어와 임베디드 OS 계층의 이슈를 배제한 임베디드 어플리케이션 계층에서 엘리베이터 시스템의 동작 원리와 제어 방법을 소프트웨어로 구현할 수 있는 환경을 제공하여, 반응(reactive)적이며 실시간(real-time)적인 특징을 갖는 임베디드 시스템 문제를 해결하는 경험을 가질 수 있도록 한다. 아울러 본 논문에서는 시뮬레이터를 이용하여 단계별로 난이도

를 높여가며 실습을 진행하는 4주간의 임베디드 어플리케이션 교육 과정을 제시하고, 학생들을 대상으로 실제 교육을 수행하여 성취도 평가 점수 83.3점을 얻어 제안하는 교육 과정이 유의미한 학습 효과가 있음을 입증하였다. 또한 부수적으로 이벤트-드리븐(event-driven) 프로그래밍 및 상태-기계(state-machine)를 사용하는 제어 프로그래밍 교육에도 효과가 있음을 확인하였다.

본 논문의 성과는 그동안 임베디드 시스템 교육에서 비중 있게 다루지 않았던 어플리케이션 계층에 대해 시뮬레이터를 이용한 학습방법을 제시했다는데 의의가 있다. 이를 계기로 앞으로 다양한 임베디드 시스템 분야에서 시뮬레이터를 이용한 교육 방법이 활성화되기를 기대해 본다.

엘리베이터 시뮬레이터와 관련된 향후 연구 계획으로 시뮬레이터 요소에 시나리오에 따라 엘리베이터를 이용하는 승객 객체를 추가하여 엘리베이터 제어 알고리즘의 성능을 측정하는 기능을 추가할 계획이다. 그리고 다양한 군관리 제어 알고리즘을 적용할 수 있도록 Multi-Mode 시뮬레이터의 기능을 확대하여 교육용 시뮬레이션뿐만 아니라 군관리 제어 연구용 시뮬레이션으로 사용할 수 있도록 개선 할 생각이다.

참 고 문 헌

[1] 은성배, 한상숙, 진성기 (2002). 임베디드 시스템 프로그래밍 교육론 및 교육용 장비 개발 사례. **정보과학회지, 20**(7), 45-51.

[2] 김수홍 (2013). **임베디드 시스템 개발을 위한 임베디드 소프트웨어.** 경기: 21세기사.

[3] 김명중, 이태욱 (2014). 레고 마인드스톰을 활용한 임베디드 SW 학습프로그램 개발. **한국컴퓨터정보학회 학술발표논문집, 22**(2), 157-160.

[4] 로버트 오샤나, 마크 크랠링, 윤희병 (2015). **임베디드 시스템을 위한 소프트웨어 공학 총론.** 경기: 에이콘

[5] 신연순, 손대근, 이경호, 홍성호, 이강우, 정진우 (2016). 레고 마인드스톰 NXT를 활용한 기초설계 교과목에서의 효과적인 공학설계과제 선정방안 연구. **공학교육연구, 19**(2), 60-69.

[6] 김송주 (2017). 아두이노를 활용한 프로젝트 기반의 임베디드 시스템 교육. **한국정보기술학회논문지, 15**(12), 173-180.

[7] 김우찬 (2010). 고등학교 정보교과 논리회로 교육을 위한 논리회로 시뮬레이터 설계 및 구현, **정보창의교육논문지, 4**(1), 1-7.

[8] 박선주, 안성진 (2017). 3D 시뮬레이션을 활용한 소프트웨어 교육프로그램 설계, **한국컴퓨터교육학회 동계 학술발표논문지, 21**(1), 65-68.

[9] 서병화 (1996). **엘리베이터 하이테크 기술.** 서울: 성안당

[10] About Elevate, Peters Research (1997), https://www.peters-research.com/index.php/elevate/about-elevate

[11] Project 4: Elevator Simulator (2006), http://courses.cs.vt.edu/~cs1044/summerII06/projects/p4/project4.pdf

[12] Assignment 2a, Programming Usable Interfaces - Spring 2009 (2009), https://people.eng.unimelb.edu.au/vkostakos/courses/pui09S/hw2a.pdf

[13] Project 2: Elevator Simulator (2014), https://cse.msu.edu/~cse251/project2.html

[14] EBS 원더풀 사이언스: 과학, 엘리베이터를 타다 (2016), https://youtu.be/3dSh0ynThCw

[15] 김영수, 박성미, 박성준 (2016). 초고층빌딩의 엘리베이터 군관리시스템에 관한 개발동향. **조명·전기설비, 30**(1), 53-61.

고 석 훈

1993 동국대학교
전자계산학과(공학학사)
1995 KAIST
전산학과(공학석사)
1995～1997 LG종합기술원 주임연구원
1997～1999 SK텔레텍 SW개발팀 선임연구원
2000～2007 ㈜신지소프트 대표이사
2010 동국대학교 컴퓨터공학과(공학박사)
2012～2013 서강대학교 컴퓨터공학과
강의전담교수
2017～현재 한국외국어대학교
정보산업공학연구소 교수
관심분야: HCI, 임베디드 소프트웨어, 컴퓨터교육
E-Mail: shko99@gmail.com