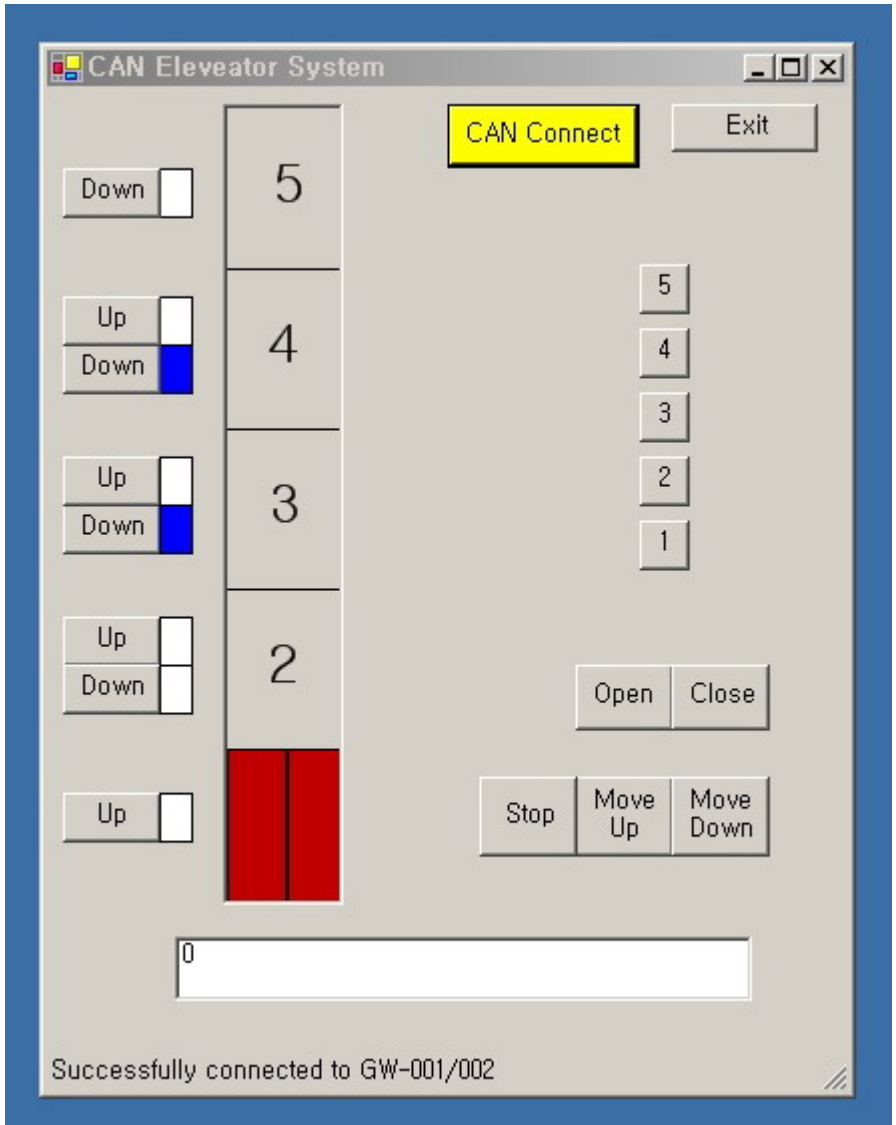
Design Project

Elevator Controller

Controller for CAN Elevator



CAN ID & Data

- Rx message ID=1
- Moving up=01,01
- Moving down=01,02
- Stop moving=01,00

- Door open=02,01
- Door close=02,02

CAN ID & Data

- Tx message ID=2
- When the car is moving, a CAN message with the current position in pixel is transmitted 5 times/sec.
- 1st byte=1,2nd byte=high byte of position,3rd byte=low byte of position
- Each floor height is 80 pixels
- At the first floor, position is 0. At the 5th floor, position is 320.

CAN ID & Data

- Tx message ID=2
- When a call button is pressed, a CAN message is transmitted
- 1st byte=2, 2nd byte=floor, 3rd byte=1 for up & 2 for down. For example when the 4th floor up button is pressed, the data is 2,4,1

CAN ID & Data

- Rx message ID=1
- When a CAN message with 1st byte=3 is received, a call button is off.
- 1st byte=3, 2nd byte=floor, 3rd byte=1 for up button & 2 for down button. For example when a CAN message with the data 3,4,1 is received 4th floor up button is off.

Design Problem

- Design an elevator controller.
- You have to set up your own design objectives to achieve. i.e. the level of complexity for the control algorithm.
- After the design, analyze your own design to determine if your design satisfies your own objectives.

Design Assumption

- To simplify the problem, the following is assumed: the car is waiting at the 1st floor. Everybody wants to go down to the 1st floor for lunch.
- Move up to the floor, turn off the call button lamp, open the door, closed the door, go down to the destination floor, open the door, close the door, etc.

Requirements for Design

- Reliability
- The controller must be able to respond to all the possible key inputs in the reliable and reasonable manner.

- 이 과제는 아래의 두 가지 방법으로 실시한다.
  - Model-Driven Method: Rhapsody, Linux target
  - Legacy Code: C compiler, Cortex-M4 target

보고서 제출 요령

- 보고서에는 각 방법에 대해서 다음 페이지의 내용을 포함한다.
- Rhapsody의 경우 프로젝트 파일이 포함된 폴더를 압축하여 제출한다.
- Cortex-M compiler 의 경우에는 자신이 작성한 C 소스 파일을 제출한다.

보고서 내용

- 본인이 설계하기로 설정한 제어기 동작 시나리오를 기술
- 본인이 설계한 제어기가 본인이 설정한 기준을 만족하는지 분석한 결과
- 본인이 설계한 제어기의 실제 동작 결과에 대한 기술
- 결과에 대한 결론 및 토의

보고서 내용

- 각 방법에 대해서 다음의 비교 테이블을 작성하여 결론에 포함한다. 비교 내용은 본 프로젝트에 한정되지 않고 두 가지 방법에 대한 본인의 의견 또는 생각을 기입한다.

보고서 내용

	Model-Driven (Rhapsody)	
Architectural Design	장점	
	단점	
Detail Design	장점	
	단점	
Coding	장점	
	단점	
Debugging	장점	
	단점	

보고서 내용

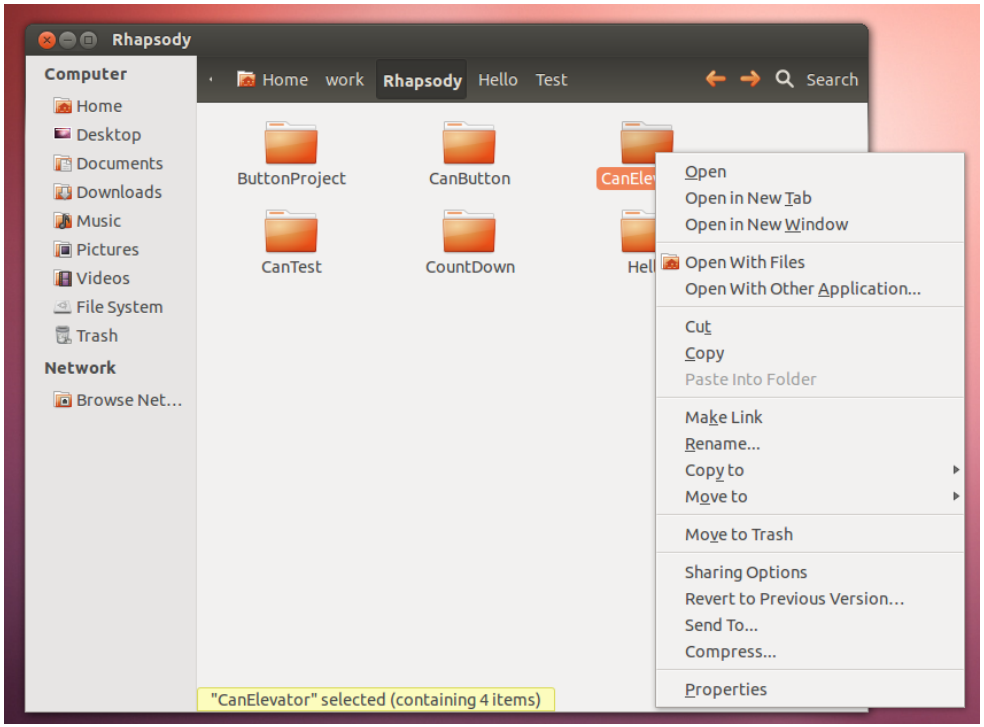
	Legacy Code (C)	
Architectural Design	장점	
	단점	
Detail Design	장점	
	단점	
Coding	장점	
	단점	
Debugging	장점	
	단점	

보고서 내용

- 결론 부분에, 앞 페이지의 비교 내용에 의하여 장래에 임베디드 소프트웨어 개발 프로젝트를 실시할 기회가 있을 경우 Model-Driven Method 를 적용할 의향이 있는지 서술한다.

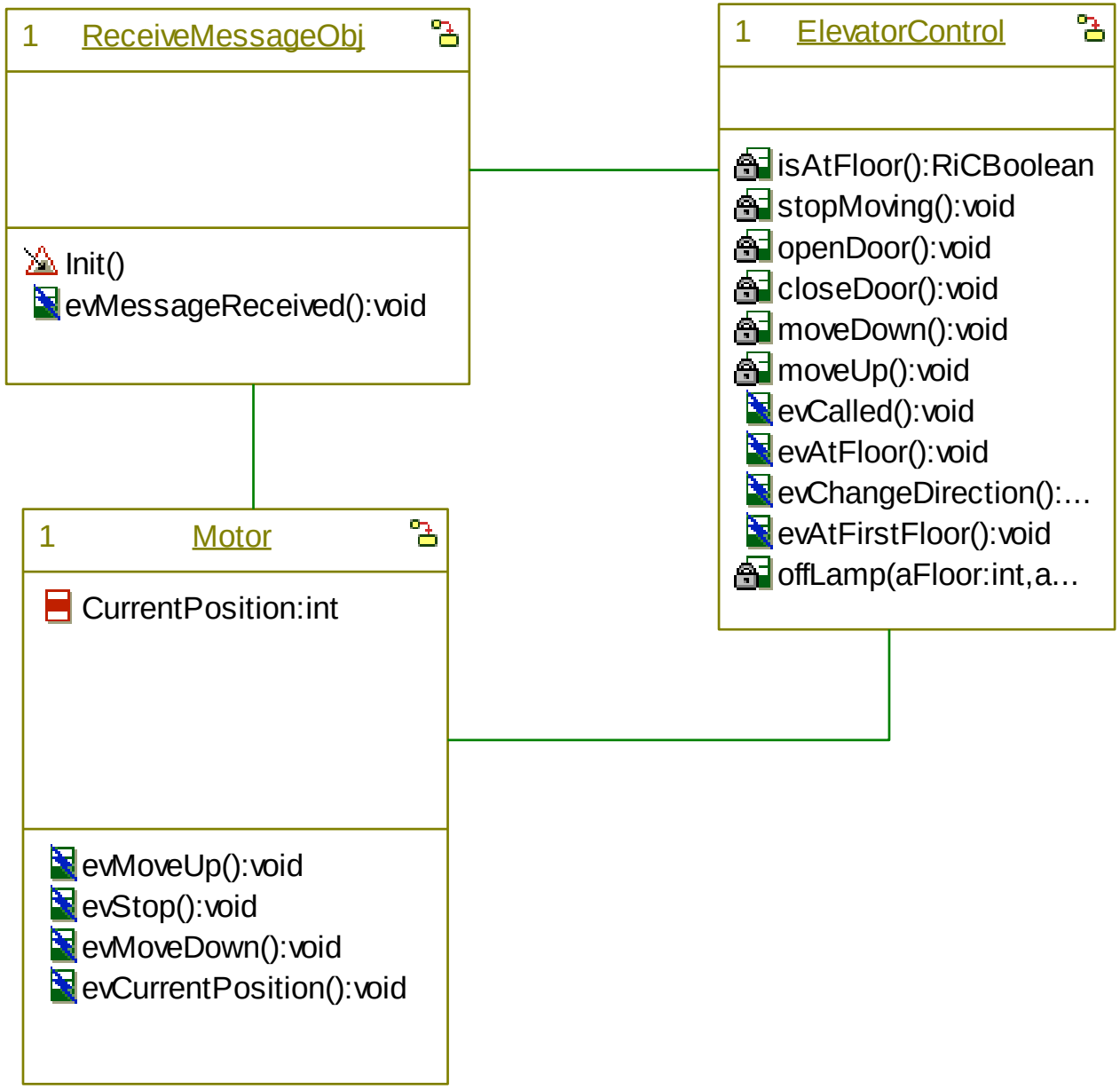
Linux에서 압축 방법

- $ tar czvf CanElevator.tar.gz CanElevator
- 마우스로 선택 후, 오른쪽 버튼 클릭하여 메뉴에서 Compress 선택

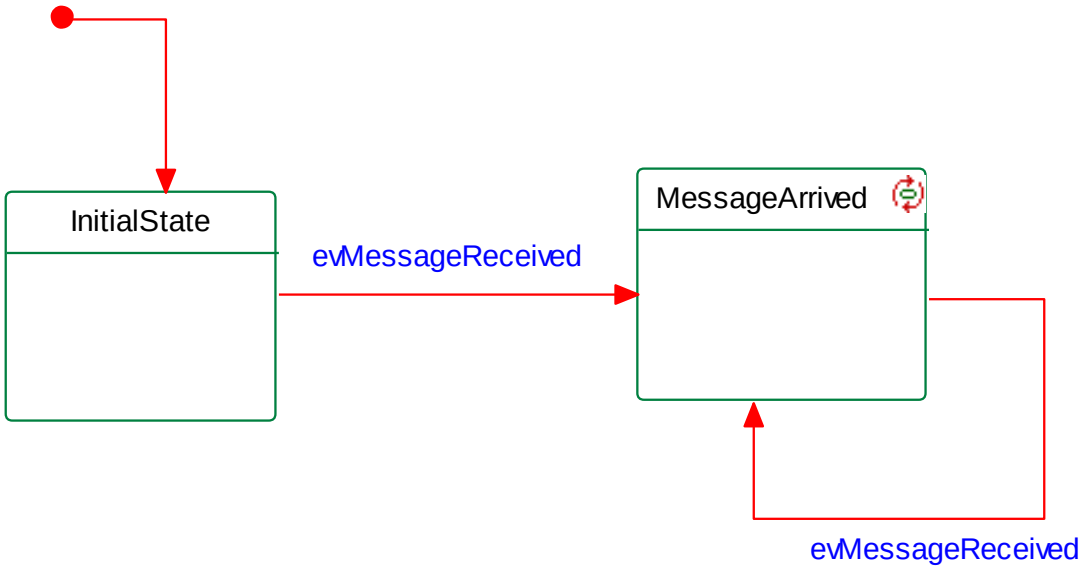


평가

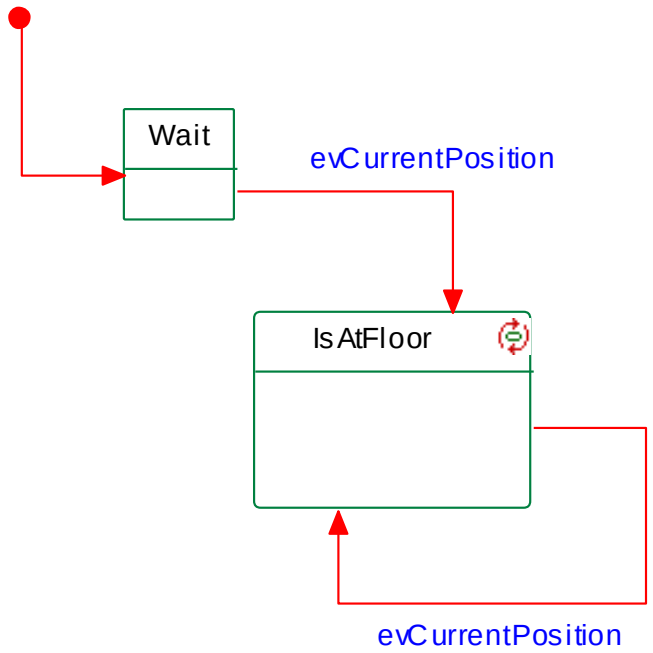
- 기능 구현: 30%
- 설계의 적절성 및 안전성: 30%
- 결과 보고서: 40%
- 본인이 설계하기로 설정한 시나리오의 완성도에 따라서 가점 또는 감점이 있을 수 있음.


1 ReceiveMessageObj
Init()
evMessageReceived():void
1 ElevatorControl
isAtFloor():RiCBoolean
stopMoving():void
openDoor():void
closeDoor():void
moveDown():void
moveUp():void
evCalled():void
evAtFloor():void
evChangeDirection():...
evAtFirstFloor():void
offLamp(aFloor:int,a...
1 Motor
CurrentPosition:int
evMoveUp():void
evStop():void
evMoveDown():void
evCurrentPosition():void

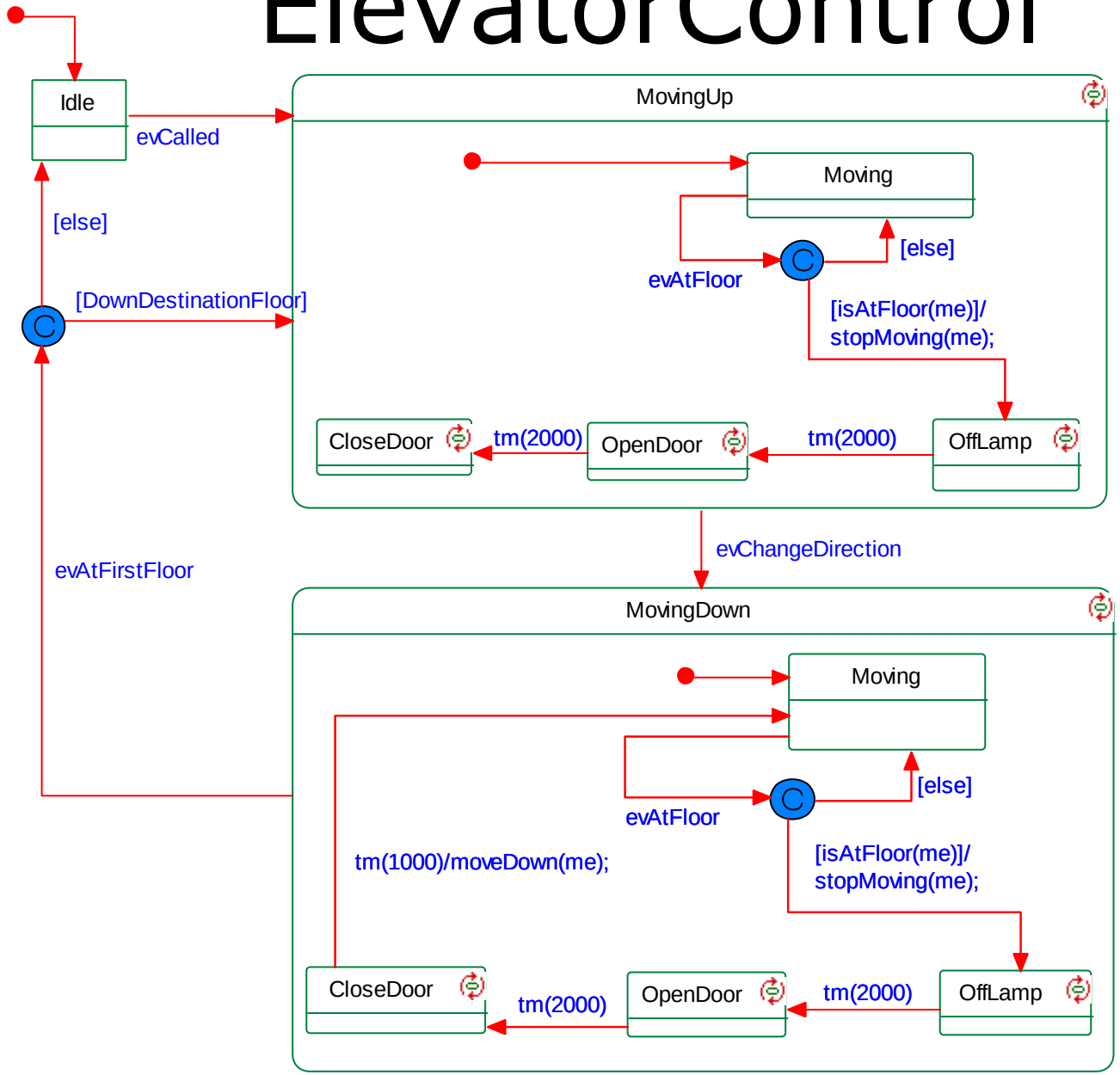
ReceiveMessageObj



Motor



ElevatorControl



Caution

- When CAN messages are received while the program is not ready to accept messages, an error occurs. In that case, start over the program again.
- Be cautious when using ***printf***, since it slows down the program and misses CAN messages.