##### Project Document

System Model (Sequence Diagram) Document

제 9 조

조원 : 201102435 박민수  
 201102440 박주민  
 201102398 김경헌

지도교수: 원유재 (서명)

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Rev# | Date | Affected Section | Author |
| 1 | 2016/05/21 | 1차 작성 | 박민수,박주민,김경헌 |
| 2 | 2016/05/22 | 최종작성 | 박민수,박주민,김경헌 |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Introduction 6](#_Toc451702258)

[1.1. Objective 6](#_Toc451702259)

[2. Use Case Diagram 7](#_Toc451702260)

[3. Sequence Diagram 8](#_Toc451702261)

[3.1. 감시 Sequence 8](#_Toc451702262)

[3.2. 초기패턴등록 Sequence 10](#_Toc451702263)

[3.3. 백업핀번호등록 Sequence 12](#_Toc451702264)

[3.4. 데이터초기화 Sequence 15](#_Toc451702265)

[3.5. 패턴수집 Sequence 18](#_Toc451702266)

[3.6. 잠금 Sequence 20](#_Toc451702267)

[3.7. 패턴비교 Sequence 22](#_Toc451702268)

List of Figure

Figure 1 – Use Case Diagram 6

Figure 2 – 감시 Sequence Diagram 7

Figure 3 – 초기패턴등록 Sequence Diagram 9

Figure 4 – 백업핀번호등록 Sequence Diagram 11

Figure 5 – 데이터초기화 Sequence Diagram 14

Figure 6 – 패턴수집 Sequence Diagram 17

Figure 7 – 잠금 Sequence Diagram 19

Figure 8 – 패턴비교 Sequence Diagram 21

# Introduction

## Objective

이 문서는 사용자 패턴분석을 이용한 본인인증 프로그램의 시스템 모델(시퀀스 다이어그램)에 대한 내용을 기술하고 있다. 요구사항 명세 단계에서 작성한 유스케이스 다이어그램을 기반으로 각 유스케이스의 상세한 내부 동작 흐름을 시퀀스 다이어그램으로 모델링한다.

# Use Case Diagram

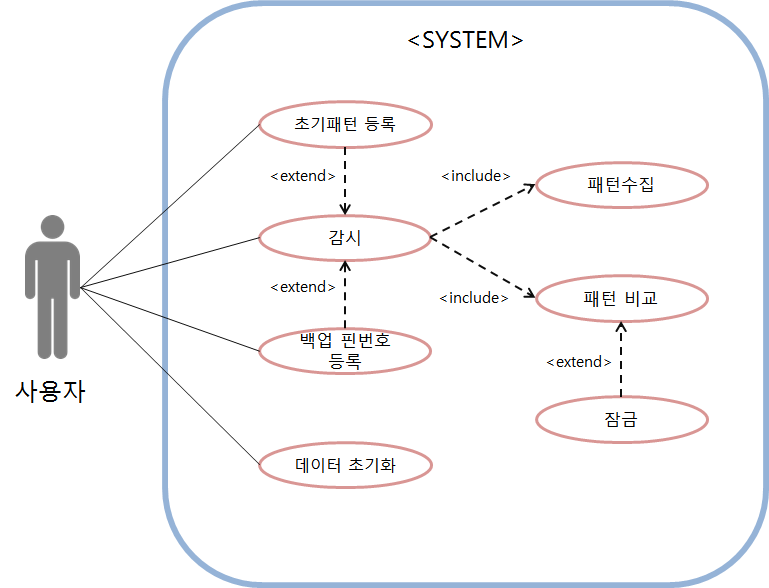


Figure 1 – Use Case Diagram

# Sequence Diagram

## 감시 Sequence

감시 시퀀스는 유저에 의해 실행되며, 실시간으로 유저의 입력데이터를 받아와서 패턴화하고, 이를 비교하는 동작을 실행한다.

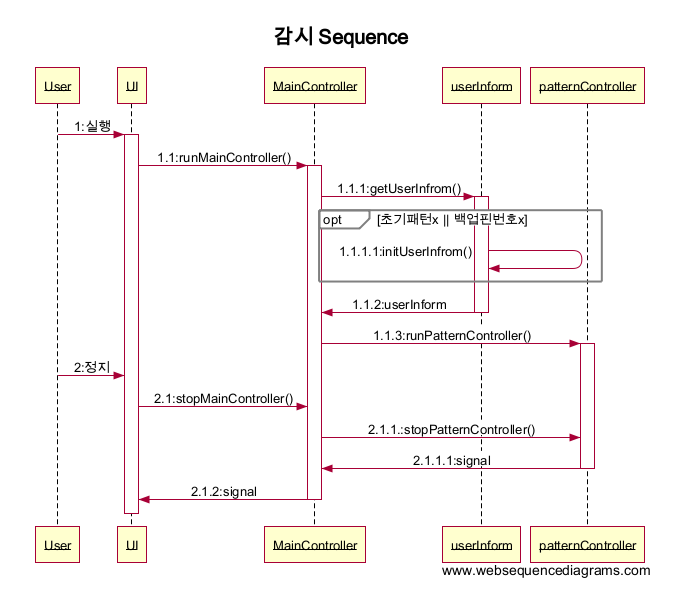


Figure 2 – 감시 Sequence Diagram

1. 사용자는 UI에서 감시항목을 선택하여 실행한다.

1.1. UI에서 runMainController() 함수를 이용하여 MainController를 실행한다.

1.1.1. MainController에서 getUserInform() 함수를 이용하여 userInform에 있는 정보를 요청한다.

1.1.1.1. 초기패턴이나 백업핀번호가 존재하지 않을 경우, initUserInform() 함수를 이용하여 초기값들을 설정해준다.

1.1.2. userInform에 있는 정보들이 리턴된다.

1.1.3 MainController가 runPatternController()를 이용하여 patternController를 실행한다.

2. 사용자는 UI에서 감시항목을 선택하여 정지한다.

2.1. UI에서 stopMainController() 함수를 이용하여 MainController를 정지한다.

2.1.1. MainController에서 stopMainController() 함수를 이용하여 patternController를 정지한다.

2.1.1.1 patternController가 정지되고 정지 signal이 리턴된다.

2.1.2 MainController가 정지되고 정지 signal이 리턴된다.

## 초기패턴등록 Sequence

초기패턴등록 시퀀스는 유저에 의해 실행되며, 유저에게서 초기 입력패턴을 받아와서 DB에 저장하는 역할을 한다.

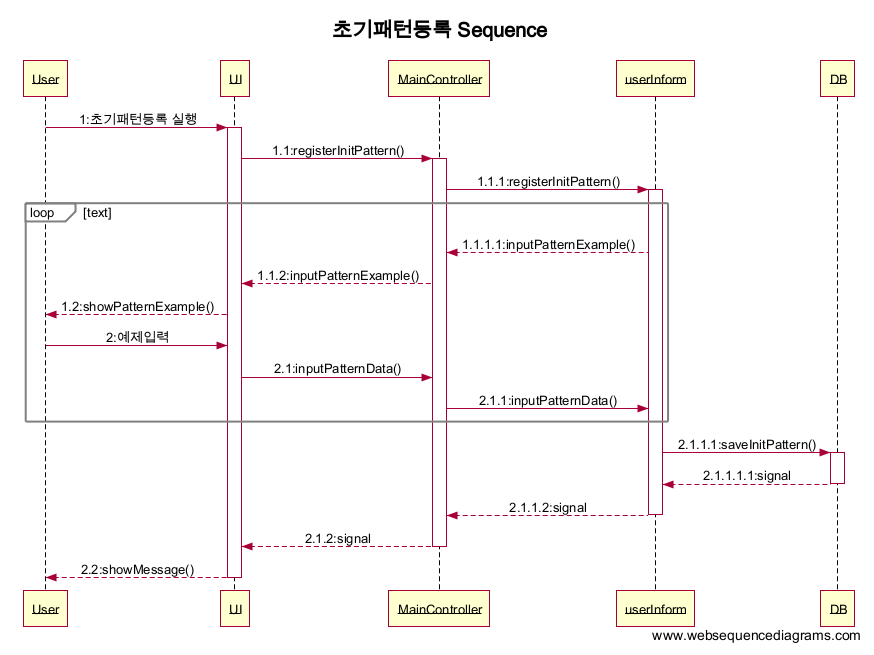


Figure 3 – 초기패턴등록 Sequence Diagram

1. 사용자는 UI에서 초기패턴등록 항목을 선택하여 실행한다.

1.1. UI에서 registerInitPattern() 함수를 실행하여 mainController의 registerInitPattern()함수를 실행한다.

1.1.1. MainController에서 registerInitPattern() 함수를 실행하여 userInform의 입력 예제를 불러한다.

1.1.1.1. userInform 이 입력예제를 리턴해준다.

1.1.2. MainController가 userInform에서 전달된 입력예제를 리턴해준다

1.2. UI가 전달된 입력예제를 화면에 표시해준다

2. 사용자는 입력예제를 보고 입력 디바이스를 이용하여 예제를 입력한다.

2.1. UI에서 inputPatternData () 함수를 이용하여 입력된 데이터를 MainController에 전달한다.

2.1.1. MainController에서 inputPatternData () 함수를 이용하여 전달된 데이터를 userInform에 전달한다.

2.1.1.1 userInform은 전달된 데이터를 saveInitPaterrn() 함수를 이용하여 DB에 저장한다.

2.1.1.1.1 DB에 저장되면 저장되었다는 signal이 리턴된다.

2.1.1.2 DB에 저장되었다는 signal을 MainController로 전달한다.

2.1.2. DB에 저장되었다는 signal을 UI로 전달한다.

2.2 UI에 DB에 저장되었다는 signal이 들어오면 showMessage()함수를 실행하여 화면에 저장되었다는 확인 메시지를 출력해준다.

## 백업핀번호등록 Sequence

백업핀번호등록 시퀀스는 유저에 의해 실행되며, 유저가 원하는 백업핀번호를 입력하면 그 번호를 저장하는 역할을 한다.

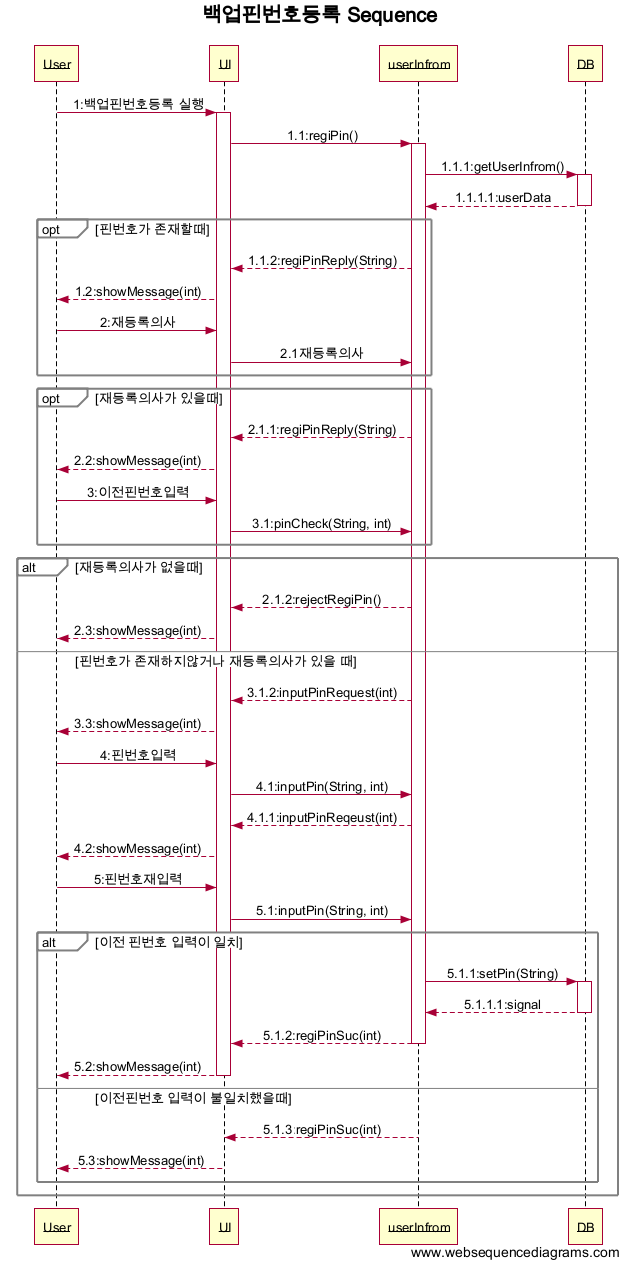


Figure 4 – 백업핀등록 Sequence Diagram

1. 사용자는 UI에서 백업핀번호등록 항목을 선택하여 실행한다.

1.1. UI에서 regiPin() 함수를 실행하여 userInform 의 getUserInform()함수를 실행한다.

1.1.1. userInform 에서 getUserInform () 함수를 실행하여 DB에 저장된 유저 데이터를 불러온다.

1.1.1.1. DB 은 저장된 유저데이터를 리턴해준다.

1.1.2. 저장된 핀번호가 존재할때, userInform이 regiPinReply(String)함수를 실행하여 UI의 showMessage(int)를 실행한다.

1.2. UI는 showMessage(int)를 이용하여 유저가 재등록의사를 물어보는 창을 화면에 표시한다.

2. 사용자는 재등록의사를 확인/취소 버튼을 이용하여 전달한다.

2.1. UI에서 사용자의 재등록의사를 userInform으로 전달한다.

2.1.1 재등록의사가 있다면, userInform regiPinReply(int) 함수를 통해 UI의 showMessage(int) 함수를 실행한다.

2.1.2 재등록 의사가 없다면, userInform rejectRegiPin() 함수를 통해 UI의 showMessage(int) 함수를 실행한다.

2.2 재등록의사가 있다면, UI에서 showMessage(int) 함수를 이용하여 이전 핀번호 입력창을 화면에 표시해준다.

2.3 재등록의사가 없다면, UI에서 showMessage(int) 함수를 이용하여 취소확인창을 화면에 표시해준다.

3. 사용자가 이전 핀번호 입력창을 이용하여 이전 핀번호를 입력한다.

3.1 UI는 pinCheck(String, Int)함수를 이용하여 userInform에 존재하는 이전 핀번호와 입력된 핀번호가 일치하는지 확인한다.

3.1.1 핀번호가 존재하지않거나 재등록의사가 있다면, userInform은 inputPinRequest(int) 함수를 실행하여 UI의 showMessage(int) 함수를 실행한다.

3.2 UI에서 showMessage(int) 함수를 이용하여 핀번호 입력창을 화면에 표시해준다

4. 사용자가 핀번호 입력창을 이용하여 등록할 핀번호를 입력한다.

4.1 UI는 inputPin(String, Int)함수를 실행하여 userInform에 입력이 들어왔음을 알린다.

4.1.1 userInform는 inputPinRequest(int) 함수를 실행하여, UI의 showMessage(int) 함수를 실행한다.

4.2 UI는 showMessage(initSuccess)함수를 이용하여 핀번호 재입력창을 화면에 표시해준다.

5. 사용자가 다시 한번 사용할 핀번호를 입력한다.

5.1 핀번호가 재입력되면, UI는 inputPin(String, Int)함수를 실행하여 입력된 핀번호와 userInform에 이전에 입력한 핀번호가 같은지 확인한다.

5.1.1 입력된 핀번호와 이전에 입력한 핀번호가 같다면, userInform은 setPin(String)함수를 이용하여 DB에 핀번호를 저장한다.

5.1.1.1 DB에 핀번호가 저장되면 저장되었다는 signal이 전달된다.

5.1.2 저장되었다는 signal을 받으면, userInform은 regiPinSuc()함수를 실행하여 등록이 성공했음을 UI에 알린다.

5.1.3 재입력된 핀번호가 이전에 입력된 핀번호와 다르다면, userInform은 regiPinSuc()함수를 실행하여 등록이 실패했음을 UI에 알린다

5.2 UI는 showMessage(int)를 통해 등록에 성공하였다는 창을 화면에 표시해준다.

5.3 UI는 showMessage(int)를 통해 두 입력이 일치하지않아 등록에 실패하였다는 창을 화면에 표시해준다.

## 데이터초기화 Sequence

데이터초기화 시퀀스는 유저에 의해 실행되며, DB에 존재하는 현재 유저 데이터를 초기화 시켜주는 역할을 한다. .

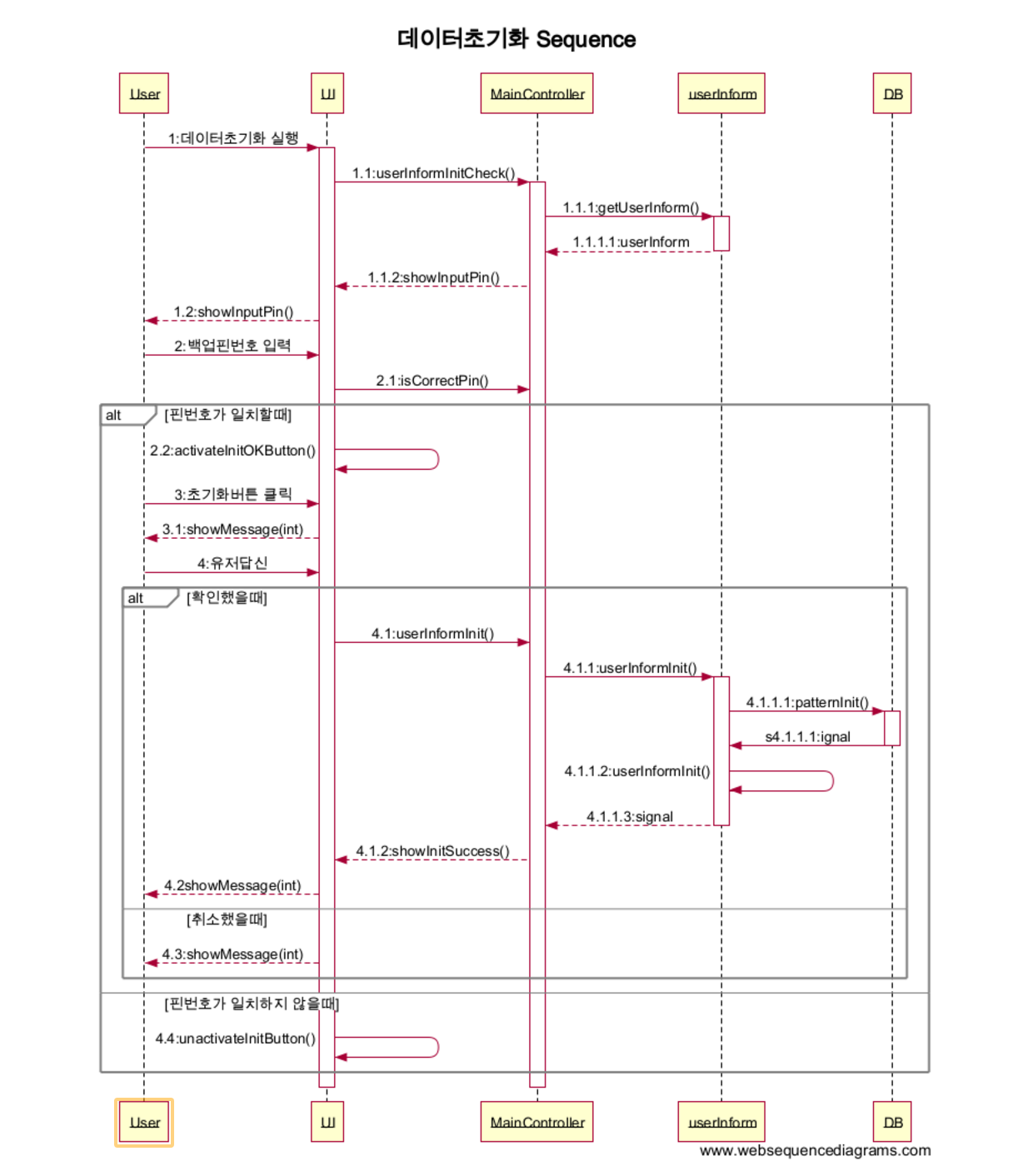


Figure 5 – 데이터초기화 Sequence Diagram

1. 사용자는 UI에서 데이터초기화 항목을 선택하여 실행한다.

1.1. UI에서 userInformInitCheck() 함수를 실행하여 mainController의 getUserInform()함수를 실행한다.

1.1.1. MainController에서 getUserInform () 함수를 실행하여 userInform에 저장된 유저 데이터를 불러온다.

1.1.1.1. userInform 은 유저데이터를 리턴해준다.

1.1.2. MainController가 userInform에서 데이터를 받으면, showInputPin()함수를 실행하여 UI의 showInputPin()함수를 실행한다.

1.2. UI는 showInputPin () 함수를 이용하여 화면에 입력창을 표시해준다.

2. 사용자는 백업핀번호를 입력한다.

2.1. UI에서 isCorrectPin () 함수를 이용하여 입력된 데이터를 MainController에 저장된 백업핀번호와 같은지 비교한다.

2.2 입력된 백업핀번호와 저장된 백업핀번호가 일치한다면, activeInitOKButton()함수를 이용하여 초기화 버튼을 활성화 시켜준다.

3. 사용자가 초기화 버튼을 클릭한다.

3.1 UI는 showMessage(int)함수를 이용하여 의사 재확인 창을 화면에 표시해준다.

4. 표시된 재확인창에서 유저가 확인 또는 취소버튼을 클릭한다.

4.1 유저가 확인버튼을 클릭했다면, UI는 userInformInit()함수를 실행하여 Maincontroller의 userInformInit()함수를 실행한다.

4.1.1 Maincontroller는 userInformInit() 함수를 실행하여, userInform에 patternInit() 함수를 실행한다.

4.1.1.1 userInform은 patternInit()함수를 이용하여 DB 에 저장된 유저 데이터를 초기화한다.

4.1.1.1.1 DB에서 데이터가 삭제되면 삭제되었다는 signal을 전달해준다.

4.1.1.2 삭제되었다는 signal 받으면 userInform 내부의 저장된 값들도 초기화해준다.

4.1.1.3 초기화가 완료되었다는 signal을 MainController로 보내준다.

4.1.2 showInitSuccess()함수를 통해 초기화가 성공했을을 알린다.

4.2 UI는 showMessage(initSuccess)함수를 이용하여 화면에 초기화가 성공했음을 표시해준다.

4.3 유저가 취소버튼을 클릭하면, UI는 showMessage(initFail)함수를 이용하여 화면에 초기화를 취소했음을 표시해준다

4.4 핀번호가 일치하지 않을때, UI는 unactivateButton()함수를 이용하여 초기화버튼을 비활성화 시킨다.

## 패턴수집 Sequence

패턴 수집 시퀀스는 입출력장치에서 입력을 반복적으로 받아오고 입력 받은 데이터를 패턴화하고 데이터베이스에 저장하는 작업을 반복한다.

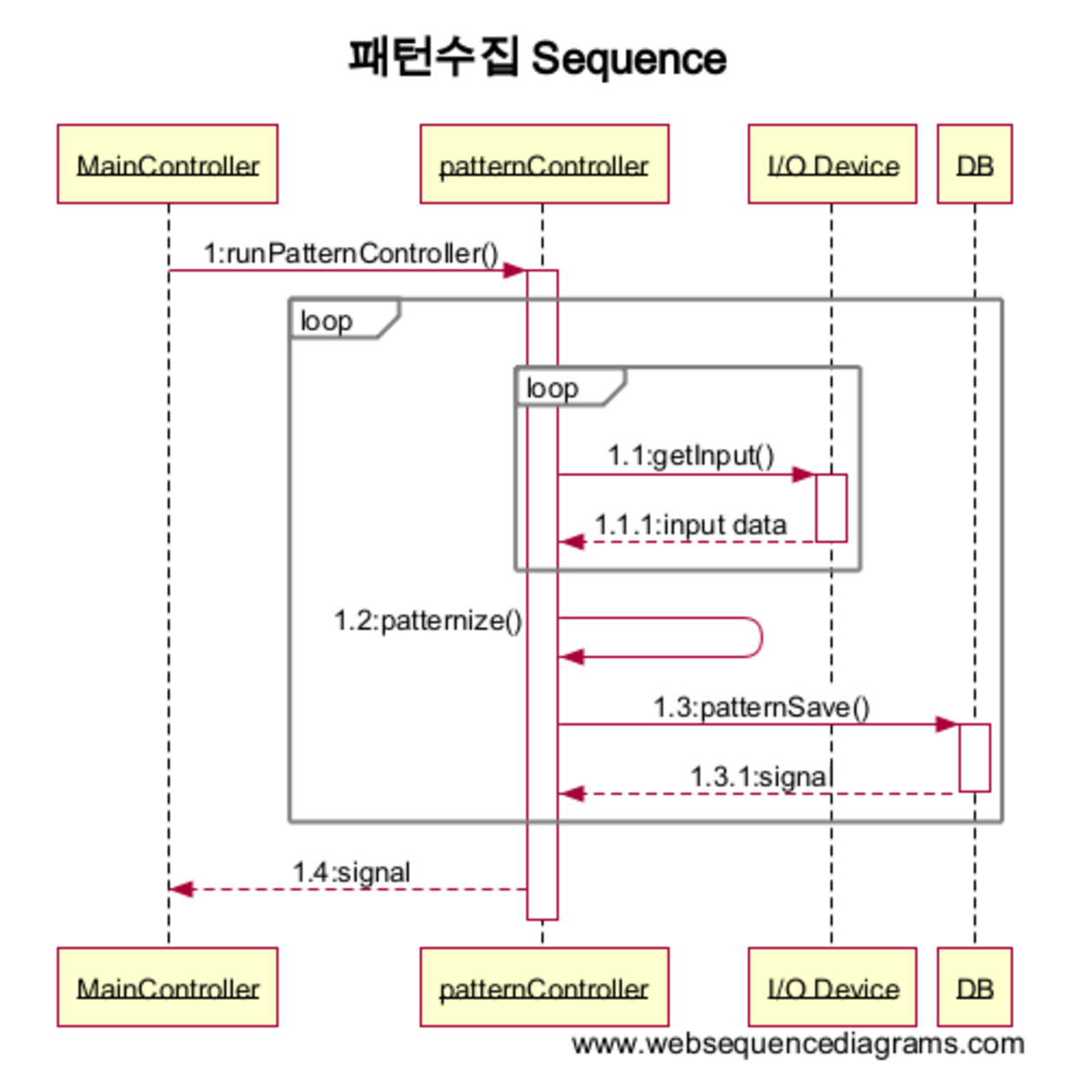


Figure 6 – 패턴수집 Sequence Diagram

1. MainController에서 runPatternController() 함수를 이용하여 patternController를 실행한다.

1.1. getInput()함수를 이용하여 I/O Device의 입력값을 요청한다.

1.1.1. patternController에게 input data를 전달한다.

1.2. input data를 patternize()함수를 이용하여 패턴화한다.

1.3. patternSave()함수를 이용하여 패턴화된 데이터를 DB에 저장한다.

1.3.1. 저장 되었다는 응답 메세지를 patternController에게 보낸다.

1.4. 응답 메세지를 MainController에게 보낸다.

## 잠금 Sequence

잠금 시퀀스는 UI에서 잠금 기능이 실행되고 핀번호 일치여부와 타임아웃 여부에 따라 잠금기능을 계속 실행하거나 잠금 기능을 해제한다.

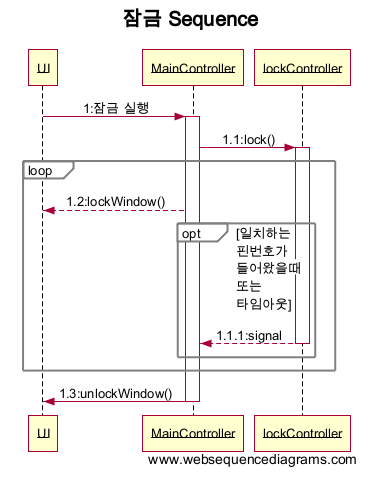


Figure 7 – 잠금 Sequence Diagram

1. UI에서 잠금기능이 실행되며 MainController를 실행한다.

1.1. MainController에서 lock()함수를 이용하여 lockController를 실행한다.

1.1.1. 일치하는 핀번호가 입력되었거나 타임아웃되면 MainController에게 잠금해제 signal를 보낸다.

1.2. lockWindow()함수를 이용하여 모든 기능을 잠근다. 입력된 핀번호가 불일치하면 lockWindow()를 반복한다.

1.3. unlockWindow()함수를 이용하여 잠금을 해제한다.

## 패턴비교 Sequence

패턴비교 시퀀스는 patternController에서 시작한다. 데이터베이스에서 저장된 기존 패턴을 받아와 저장하고 입출력장치의 입력을 반복적으로 받아와 두 데이터를 비교한 후 패턴이 일치하였을 경우에는 인증상태를 '성공'으로 전환하고 패턴이 불일치할 경우에는 인증상태를 '실패'로 전환하고 잠금기능을 실행한다..

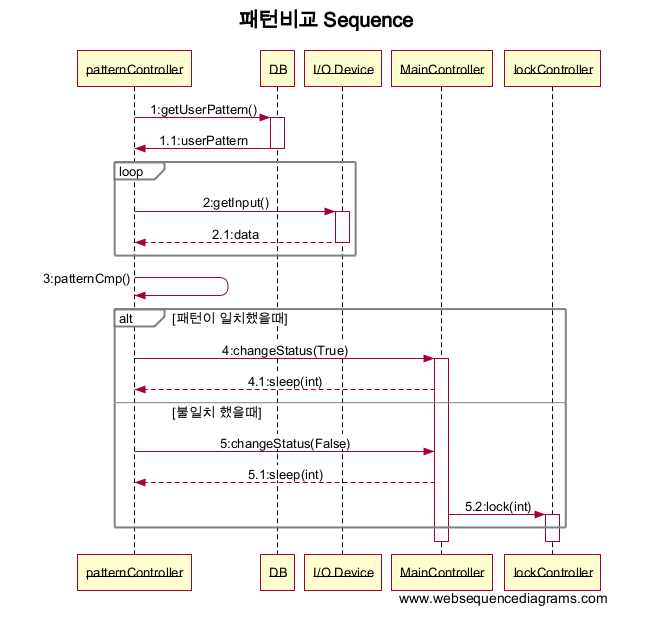


Figure 8 – 패턴비교 Sequence Diagram

1. patternController에서 getUserPattern()함수를 이용하여 DB에 기존 패턴을 요청한다.

1.1. userPattern을 patternController에게 보낸다.

2. patternController에서 getInput()함수를 이용하여 I/O Device에게 입력 데이터 요청을 반복한다.

2.1. input data를 patternController에게 보내는 작업을 반복한다.

3. DB에서 받아온 기존 패턴과 I/O Device에서 받아온 패턴을 비교한다.

4. 패턴이 일치했을 때 changeStatus()함수에 True값을 입력해 MainController에게 보낸다.

4.1. MainController에서 sleep(int)함수를 이용하여 int값의 시간만큼 패턴 비교 기능을 정지한다.

5. 패턴이 불일치했을 때 changeStatus()함수에 False값을 입력해 MainController에게 보낸다.

5.1. MainController에서 sleep(int)함수를 이용하여 int값의 시간만큼 패턴 비교 기능을 정지한다.

5.2. MainController에서 lock()함수를 이용하여 lockController를 int값의 시간만큼 실행한다.