Mandatory exercise: Hand in number 3 – Integration Test

This mandatory exercise reviews your skills in planning and implementing Integration Tests for a Microwave Oven System, which has already been unit tested. The design and explanation of the Microwave Oven System can be seen further down. However, as with all unit-tested code, some errors may not be detected until the integration step. It is the purpose of this exercise to find as many of the remaining errors as possible.

NOTE: This exercise is *mandatory*. This means that it is a prerequisite for your admission to the exam of this course that you have handed this exercise in and have a grade of PASSED (see details below).

You shall hand in in groups – no solo work. The guidelines for group work are stated in the note "*Teams, lab exercises and hand-ins in I4SWT.pdf*" available on the course Blackboard site under *Lektion 01.1*.

---

1 The exercise

You shall hand in your solution to the exercise introduced in Lecture 11.1 and 11.2.

Some details on the hand-in:

1. Your hand is registered on Blackboard, by delivering exactly one (1) PDF document containing:
    - Your team number
    - A table containing the student number and name of each participant in the exercise.
    - A URL to the Jenkins build job executing your integration tests, e.g. http://ci3.ase.au.dk:8080/job/Team181xxHandin2/
    - A URL to the GitHub repository, you are using as the shared remote repository for your team, e.g. http://github.com/TeamSWTxx/MicrowaveHandin2/
    - A Dependency Tree diagram for the Microwave Oven
    - An Integration Plan, describing the integration steps
    - A short explanation of your choices for your integration strategy
    - A short explanation of any errors you found during the integration tests, and how you found them
    - A short explanation of any corrections you made
    - A short explanation of how you had to change the unit test because of your corrections
2. The GitHub repository shall contain
    - The original code, with any changes you found necessary as a result of your integration testing. This could be corrections you have made to the original code, and any changes to the unit tests to test the corrections
    - Your integration tests according to your plan and selection, correctly implementing at **least one integration step, integrating at least 3** of the classes in the given design
    - **Because of Covid19 consequences, the requirements for this mandatory handin have been reduced compared to previous semesters. So instead of a full implementation of all integration steps and all modules, fulfillment of the previous requirement is enough**
3. Your hand-in shall be made in groups as stipulated in Section 2 in the aforementioned note. No solo work!
4. Your hand-in shall be on time. A late hand-in is not accepted and will be evaluated as "**FAILED**" (see below)
5. You shall hand in your exercise using the Blackboard assignment made available for this exercise.

There are at least two problems in the given code that a properly planned and implemented integration test should be able to detect. **Because of the Covid 19 consequences it is not necessary to find these through the integration test. But if you do, and correct and test them, you will have learned so much more!**

2 Evaluation of your hand-in

2.1 Evaluation criteria

The evaluation of your exercise depends on fulfilling the total set of criteria. The total set of criteria is described in a Rubric, visible at the Assignment Hand-In link on BlackBoard.

2.2 Evaluation grades

The evaluation of this exercise will have 1 of 3 possible grades:

"**2 - PASSED**"	Your hand-in is of sufficient quality to pass the criteria for approval
"**1 - FAILED – RESUBMIT**"	Your hand-in is of insufficient quality to pass the criteria for approval. You are granted one resubmittal. The requirements and deadline for this resubmittal will be conveyed along with the grade. The resubmittal will be graded either "PASSED" or "FAILED" (no 2nd resubmittal)
"**0 - FAILED**"	Your hand-in is of insufficient quality to pass the criteria for approval. You are not granted resubmittal.

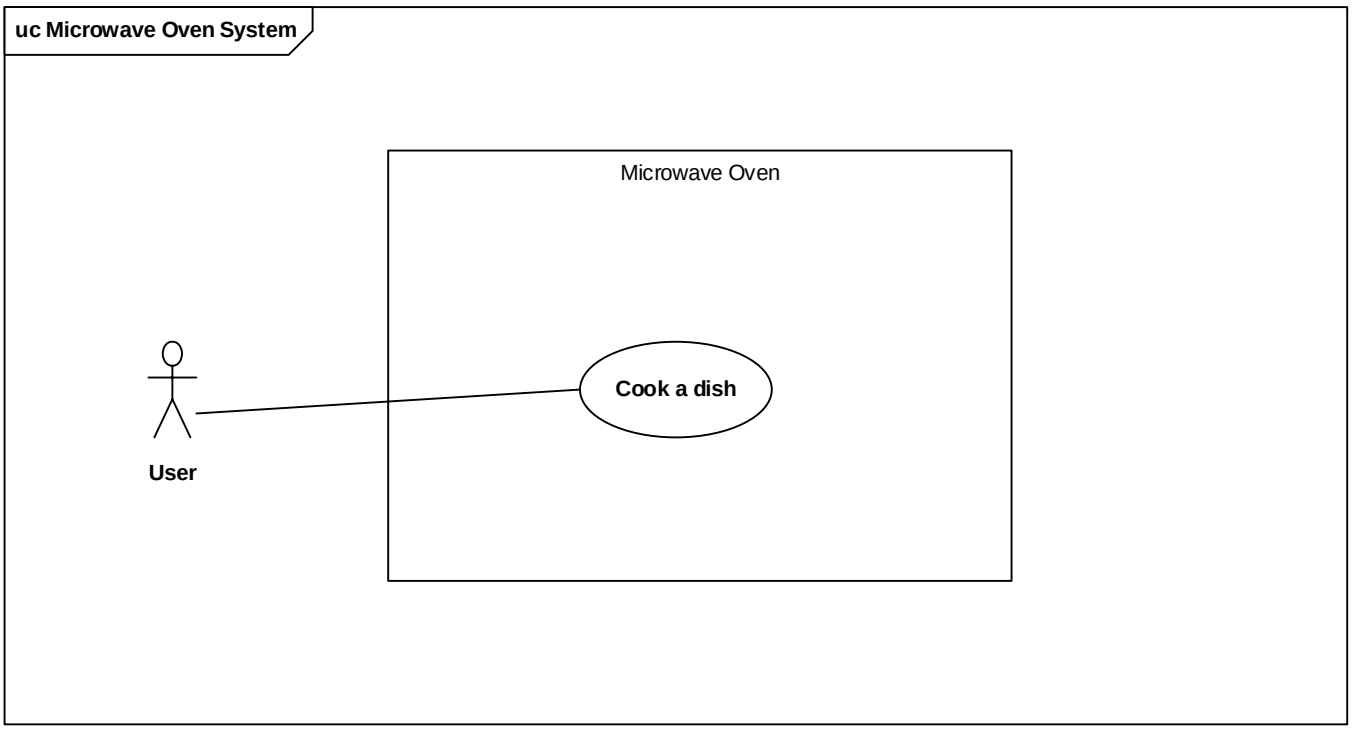
Note that your hand-in must be graded PASSED (in either first or second attempt) for the members of the group to be admitted to the exam in the course.

3 The Microwave Oven system

In the following, the design of the system is described with a Use Case diagram, a Use Case description, a class diagram and some sequence diagrams.

The code for the classes with their unit tests can be found on Blackboard, as a zipped Visual Studio solution. It can also be forked from GitHub from the repository http://github.com/TeamSWTFrabj/MicrowaveOven, and from this forked repository in your own GitHub account, you can do the work for the integration. Be sure to make a "fork", as you cannot use my repository for your integration work. The code also contains a demo application.

Use Case Diagram

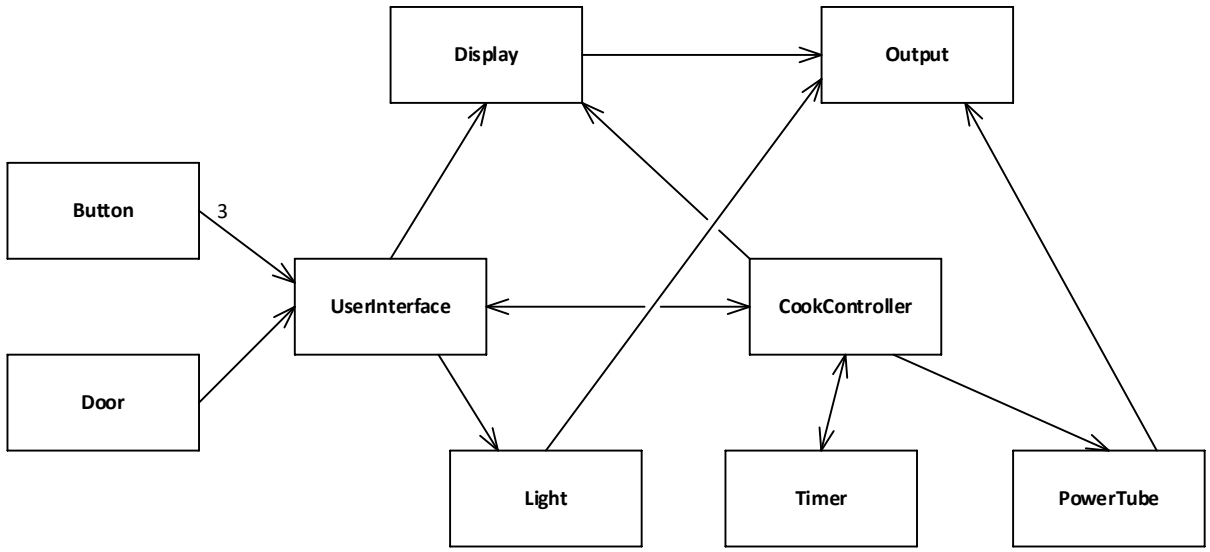


Use Case Description
A beeper has been omitted from this design for the sake of simplicity.

Name	**Cook a dish**
Goal	The user will have a dish of food cooked
Initiation	By the user.
Actors	The User
Precondition	The user has a dish of food on a microwave suitable platter, the MicroWave oven is attached to the mains power supply, no activity is currently taking place in the oven, the display is blank.
Postcondition	The oven is inactive, the food has been heated with the desired microwave power during the desired time. The display is blank.
Main Scenario	1. The user opens the door 2. The light goes on inside the oven 3. The user places the dish in the oven 4. The user closes the door 5. The light goes off inside the oven 6. The user presses the Power button one or more times, to select the desired microwave power. The display shows the currently selected power from 50 to 700 W, starting with 50 W. Each press increases the selected power level with 50 W, until 700, where it will return to 50 W on the next press. [Extension 1: The user presses the Start-Cancel button during power setup] [Extension 2: The user opens the door during setup] 7. The user presses the Time button one or more times to select the desired cooking time. The display shows the currently selected time as minutes:seconds, starting with 01:00. Each press increases the selected time with one minute. [Extension 2: The user opens the door during setup] 8. The user presses the Start-Cancel button. 9. The light goes on inside the oven 10. The powertube starts working at the desired powerlevel 11. The display shows and updates the remaing time every second as minutes:seconds. [Extension 3: The user presses the Start-Cancel button during cooking] [Extension 4: The user opens the Door during cooking] 12. When the time has expired, the power tube is turned off 13. The light inside the oven goes off 14. The display is blanked 15. The user opens the door 16. The light goes on inside the oven 17. The user removes the food 18. The user closes the door 19. The light inside the oven goes off.
Extension	[Extension 1: The user presses the Start-Cancel button during setup] 1. The display is blanked 2. All settings are reset to start values 3. The user can start the Use Case from step 6 or 15, or the Use Case ends. [Extension 2: The user opens the Door during setup] 4. The light goes on inside the oven 5. The display is blanked 6. All settings are reset to start values 7. The user can continue the Use case from step 4 or 17. [Extension 3: The user presses the Start-Cancel button during cooking] 8. The power tube is turned off 9. The display is blanked.

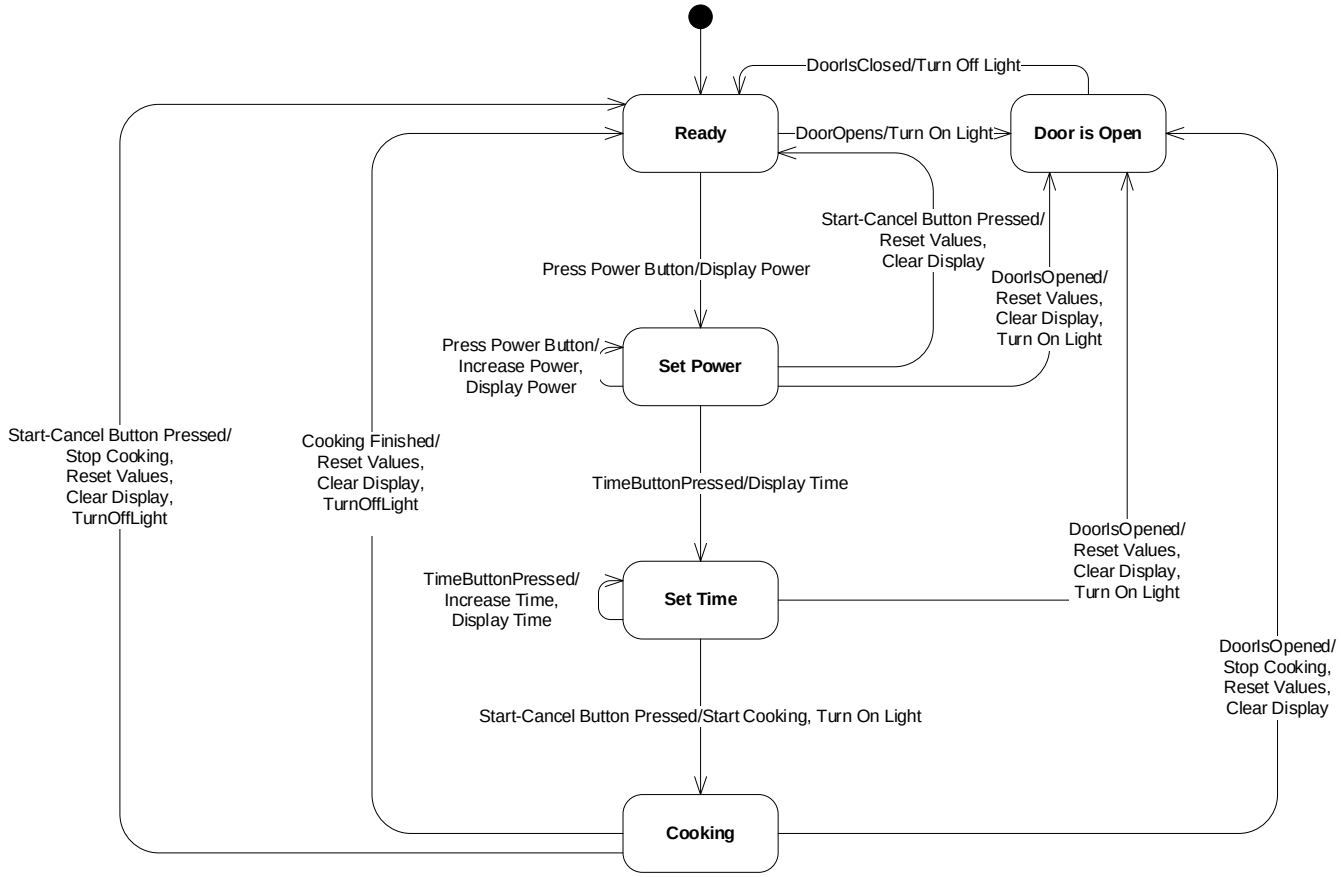
	10. The light inside the oven goes off 11. All settings are reset to start values 12. The user can continue the Use Case from step 6 or 15, or the Use Case ends. [Extension 4: The user opens the Door during cooking] 13. The power tube is turned off. 14. The display is blanked. 15. All settings are reset to start values. 16. The user can continue the Use Case from step 4 or 17.

Class Diagram

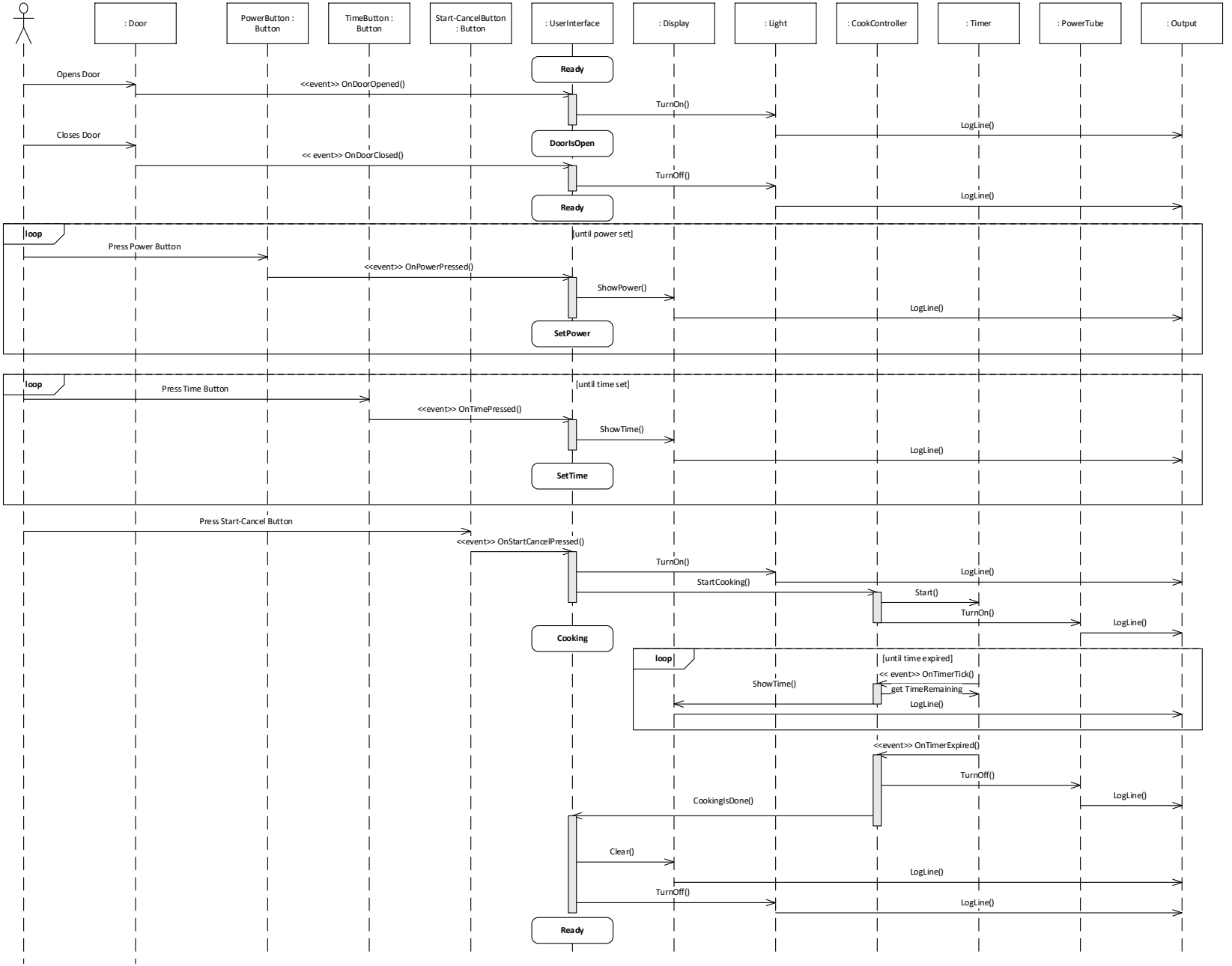


All classes are implementing a relevant interface – not shown on the diagram for simplicity – so dependency isolation is possible for testing.

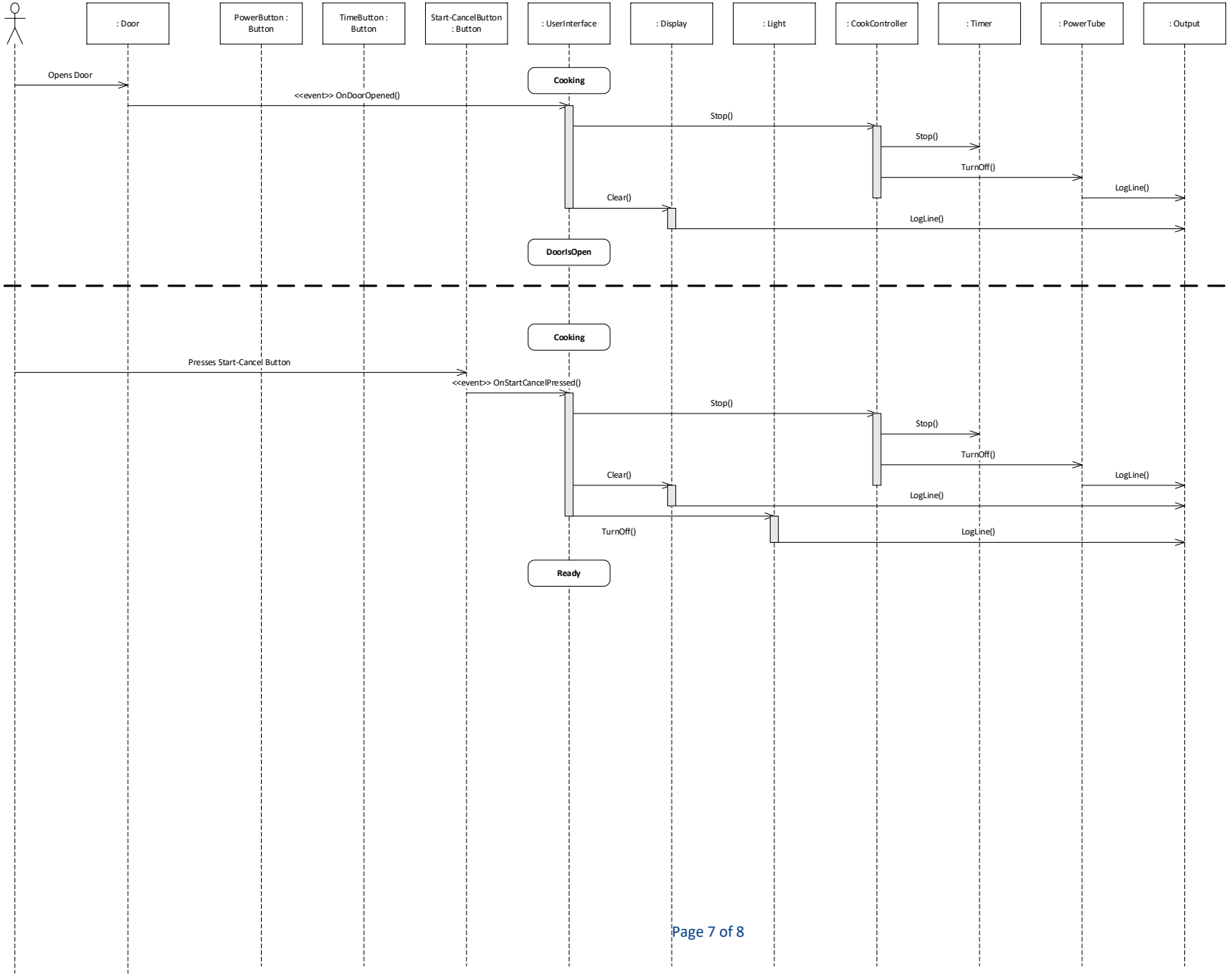
State Machine diagram for UserInterface



On the next page: Sequence diagram for the main scenario


: Door
PowerButton : Button
TimeButton : Button
Start-CancelButton : Button
: UserInterface
: Display
: Light
: CookController
: Timer
: PowerTube
: Output
Ready
Opens Door
<<event>> OnDoorOpened()
TurnOn()
LogLine()
DoorIsOpen
Closes Door
<< event>> OnDoorClosed()
TurnOff()
LogLine()
Ready
loop
[until power set]
Press Power Button
<<event>> OnPowerPressed()
ShowPower()
LogLine()
SetPower
loop
[until time set]
Press Time Button
<<event>> OnTimePressed()
ShowTime()
LogLine()
SetTime
Press Start-Cancel Button
<<event>> OnStartCancelPressed()
TurnOn()
LogLine()
StartCooking()
Start()
TurnOn()
LogLine()
Cooking
loop
[until time expired]
<< event>> OnTimerTick()
ShowTime()
get TimeRemaining
LogLine()
<<event>> OnTimerExpired()
TurnOff()
LogLine()
CookingIsDone()
Clear()
LogLine()
TurnOff()
LogLine()
Ready

And these are are sequence diagrams for the some of the Extensions.



This page is intentionally left blank.