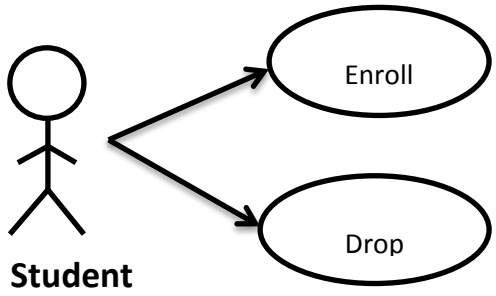
13016224 SOFTWARE VERIFICATION AND VALIDATION
TUTORIAL EXERCISES

International College, King Mongkut's Institute of Technology Ladkrabang

QUESTION 1

The following is part of a use case diagram of a system for course enrollment.



The specification of the use case "**Enroll Course**" is as follows:

Use Case Name: Enroll Course

Summary: Student enrolls a course on the course enrollment system in the current academic semester

Actor: Student

Precondition: The course enrollment system is displaying the welcome screen.

Description:

1. Student enters username and password.
2. If the username and password are correct, the system checks whether the student is currently registered in the University.
3. If the student is currently registered, the system displays a list of courses which is open for enrollment.
4. Student selects a course from the list.
5. The system checks whether the student satisfies the prerequisites of the selected course.
6. If the prerequisites are satisfied, the system asks for confirmation from the student.
7. If the student confirms, the system records the course enrollment and prints out an enrollment report.
8. The system returns to the welcome screen.

Alternatives:

2a. If the username and password are incorrect, the system asks the student to re-enter the username and password.

3a. If the student is not currently registered, the system displays the message informing that the student is not currently registered and then returns to the welcome screen.

4a. If the student cannot find a course to enroll, the student can select "Cancel" and the system returns to the welcome screen.

5a. If the student does not satisfy the prerequisites of the selected course, the system displays a message listing the unfulfilled prerequisites of the selected course. The system then asks the student to select another course from the list.

7a. If the student selects "Change" in the confirmation page, the systems returns to the screen asking the student to select a course from the list.

Your task is to design test cases to test that the system fulfills the “Enroll Course” use case specification above.

Hint:

1. Draw an activity diagram from the use case specification of “Enroll Course”. An activity diagram should have one **initial node** (denoting the start of the workflow) and one or more **terminal nodes** (denoting the termination of the workflow)
2. From the activity diagram, select a set of paths in the diagram that should be tested. Then identify a test case from each selected path by specifying a **sequence of user input** along the path and the **sequence of the expected output** from the system (or the activities that the system should perform).

PROBLEM 2

Suppose you have been hired to test a package management system for a new version of Ubuntu OS under development. You have been given the following specification of the use case "**Install Package**":

Use Case Name: Install Package

Summary: Install the package specified by the user and all the supporting modules that the package requires.

Actor: User

Precondition: The database of installed packages has been created. The user has the right to install packages in the system.

Description:

1. The user specifies the name of the package to install.
2. The system checks that the package has not already been installed.
3. The system connects to the package repository.
4. The system retrieves the detail of the package, including the list of the required modules.
5. The system downloads and installs each required module one at a time until all the required modules have been installed.
6. The system downloads the package.
7. The system installs the package.
8. The system informs the user that the package has been successfully installed.

Alternatives:

2a. If the package with the given name has already been installed, the system reports this to the user and terminates.

3a. If the system fails to connect to the package repository, the system will attempt to re-connect one more time. If the connection still fails, the system reports this to the user and terminates.

4a. If the system is unable to retrieve the detail of the package, the system will report the problem to the user and terminates.

5a. If there is any error while the system is downloading a required module, the system will uninstall every module that it has just installed, and once this is done, report the problem to the user and terminates.

6a. If there is any error when the package is downloaded, the system will uninstall all the required modules that it has just installed, report to the user, and terminate.

2.1 Draw an activity diagram from the use case specification. An activity diagram should have one **initial node** (denoting the start of the workflow) and one or more **terminal nodes** (denoting the termination of the workflow)

2.2 Given an activity diagram, a **scenario** is a path in the activity diagram which starts at the initial node and ends at a terminal node. Each scenario provides a basis for testing the system. Your task is to choose a set of scenarios that satisfies **edge coverage**, i.e. every edge in the activity diagram is covered by some chosen scenario.

QUESTION 3

Imagine a vending machine from which the customer can buy two types of drinks, *BigCola* and *MiniCola*, by inserting 1-Dollar coins. A bottle of *BigCola* costs 2 dollars and a bottle of *MiniCola* costs 1 dollar. The customer may insert up to two one-dollar coins before selecting which type of drinks to buy. And if there are still unused coins in the machine, the user can continue selecting another drink.

The machine responses to the following events.

Events:

- *insertCoin* = The customer inserts a one-dollar coin.
- *buyBigCola* = The customer presses the button to buy a bottle of BigCola.
- *buyMiniCola* = The customer presses the button to buy a bottle of MiniCola.
- *cancel* = The customer presses the button to get the unused coins that he/she inserted back.

The machine can perform the operations.

Actions:

- *ejectBigCola* = The machine dispenses a bottle of *BigCola*.
- *ejectMiniCola* = The machine dispenses a bottle of *MiniCola*.
- *returnCoins* = The machine returns all the unused coins that the customer inserted.
- *rejectCoin* = The machine rejects the coin the customer just inserted because there are already two unused coins inserted into the machine.
- *beep* = The machine sounds a beep when the customer tried to buy the drink that costs more than the money inserted.

To simplify the modeling, we assume the following.

Assumptions:

- The vending machine stores unlimited supply of *BigCola* and *MiniCola*.
- The one-dollar coin is the only type of coins that can be inserted into the machine.

The machine has 3 states, namely

States:

- *S0*= The machine is in this state if no coin has been inserted. This is the **initial state.**
- *S1* = The machine is in this state if there is one unused coin in the machine.
- *S2* = The machine is in this state if there are two unused coins in the machine.

The expected behavior of this machine can be described by the following state-transition table.

Current State	Event	New State	Action
S0	*insertCoin*	*S1*	-
	buyBigCola	*S0*	*beep*
	buyMiniCola	*S0*	*beep*
	cancel	*S0*	-
S1	*insertCoin*	*S2*	-
	buyBigCola	*S1*	*beep*
	buyMiniCola	*S0*	*ejectMiniCola*
	cancel	*S0*	*returnCoins*
S2	*insertCoin*	*S2*	*rejectCoin*
	buyBigCola	*S0*	*ejectBigCola*
	buyMiniCola	*S1*	*ejectMiniCola*
	cancel	*S0*	*returnCoins*

3.1. Draw a state-transition diagram of this vending machine.

3.2. Is there a set of test input that satisfies the **all-state coverage** condition? If so, describe such a set.

Note: In this state-transition diagram, a test input is any sequence of input which can be successively applied on the machine starting from state *S0*. For example, the following is a test input:

insertCoin, buyBigCola, insertCoin, buyMiniCola, insertCoin, insertCoin, cancel, insertCoin

By applying this test input, the machine produces the following sequence of output:

-, beep, -, ejectMiniCola, -, rejectCoin, returnCoins, -

3.3. Is there a set of test input that satisfies the ***all-transition coverage*** condition? If so, describe such a set.

3.4. Is there a set of test input that satisfies the ***all-path-coverage*** condition? If so, describe such a set.

PROBLEM 4

Imagine a microwave oven with a simple operation that allows the user to cook food by setting the timer and press the Start/Stop button.

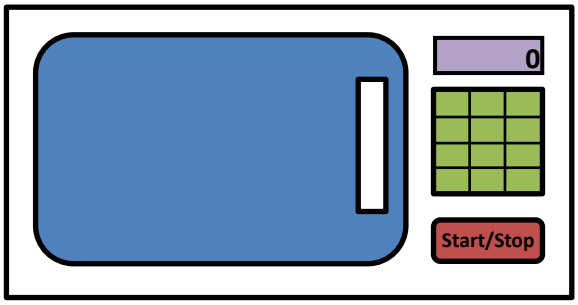


FIGURE 1 : AN ILLUSTRATION OF A SIMPLIFIED MICROWAVE OVEN

The microwave oven is controlled by an embedded microcontroller and a program which is continuously running while the oven is plugged in. You are hired to test that the program controlling the oven works as expected. The manufacturer has provided you with the state machine M described below according to which the control program should operate.

At any time, M can be in one of the following states: S1, S2, S3, S4, S5.

Initially, M will be in state S1. In this initial state, the oven's door is closed, the oven is not cooking, and the oven's timer has value 0 and is not running.

M responses to the following events:

- SetTimer : The user has just set the timer.
- PressStartStop : The user has just pressed the Start/Stop button.
- OpenDoor : The user has just opened the oven's door.
- CloseDoor : The user has just closed the oven's door.
- TimesUp : The timer has just finished running.

M can perform the following actions:

- StartCooking : The oven is commanded to start cooking.
- StopCooking : The oven is commanded to stop cooking.
- StartTimer : The timer is commanded to start running.
- StopTimer : The timer is commanded to stop running and reset its value to zero.
- PauseTimer : The timer is commanded to pause running (while holding its current value).
- PlaySound : The oven is commanded to play sound.

At any state, M responses to certain events by changing its state and perform certain actions as described in the table below.

Current State	Event	New State	Action
S1	OpenDoor	S2	-
	SetTimer	S3	-
S2	CloseDoor	S1	-
S3	OpenDoor	S2	-
	PressStartStop	S4	StartTimer and StartCooking
S4	OpenDoor	S5	PauseTimer and StopCooking
	PressStartStop	S1	StopTimer and StopCooking
	TimesUp	S1	StopCooking and PlaySound
S5	CloseDoor	S4	StartTimer and StartCooking

4.1 Draw a state-transition diagram of this machine M.

4.2 Is there a (finite) set of test inputs that satisfies **all-transition coverage**? If so, describe a set of test inputs that satisfies such coverage criterion.

Note: On a state machine, a test input is any sequence of events which may occur on the machine starting from the initial state, S1. For example, the following is a test input:

[SetTimer, PressStartStop, OpenDoor, CloseDoor]

By applying this test input, the machine produces the following sequence of actions:

[-, StartTimer and StartCooking, PauseTimer and StopCooking, StartTimer and StartCooking]

4.3 Is there a (finite) set of test inputs that satisfies **all-path coverage**? If so, describe a set of test inputs that satisfies such coverage criterion.

4.4 Is there a (finite) set of test inputs that satisfies **prime-path coverage**? If so, describe a set of test inputs that satisfies such coverage criterion.