Software Engineering

Books or notes are **not** allowed.
Write only on these sheets. **Concise** and **readable** answers please.

Surname, name, matricola ____

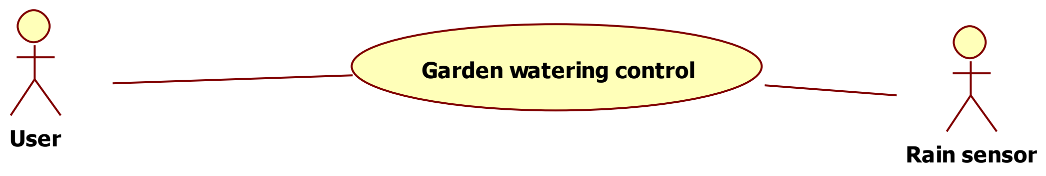
Garden watering controller
A watering controller is a device that opens / closes a water valve when defined by the user. The water valve sections a water pipe that waters a garden (or anything else).

In the following you should analyze and model a controller with these characteristics:

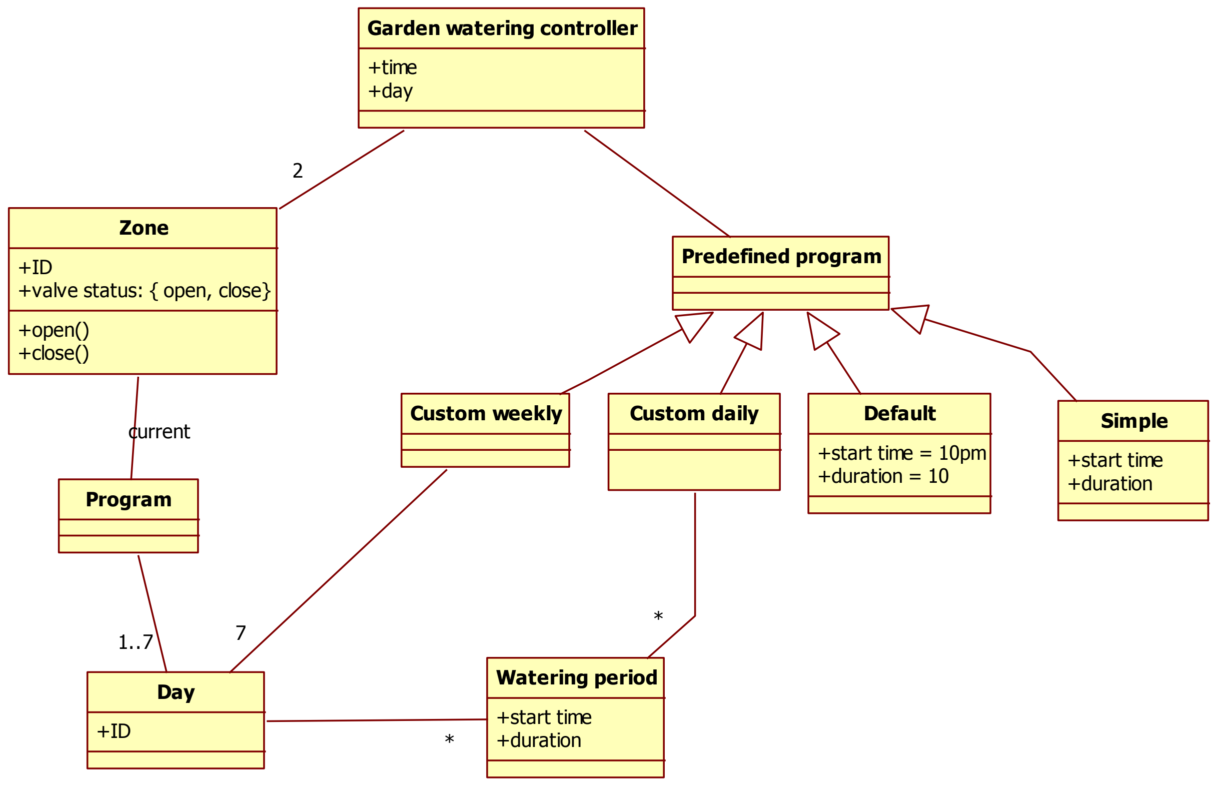
- The user interacts via a simple user interface (a few buttons for input, 3-4 inches LCD black and white screen)
- The controller controls 2 zones, in other words controls 2 independent water valves
- The user can, for each zone,
    - open / close directly the valve
    - Use a default watering program (open 10 minutes per day at 10 pm, every day)
    - Use a simple watering program (one watering period per day, the user defines the start time and the duration)
    - Use a custom water program (x watering periods per day, every y hours, each of duration z
    - Use a weekly custom water program (same as above but each day in the week can have different parameters)
- The controller can be connected to an external rain sensor (bipolar connector, 0-3V, if closed no rain). In case of rain no watering program is applied.
- The controller is battery operated

1 (15 points) – a. Define the context diagram (including relevant interfaces)

Actor	Physical interface	Logical interface
User	Buttons, screen	Simple screen shots
Rain sensor	Bipolar connector 0-3V	Boolean, open = rain, closed = no rain



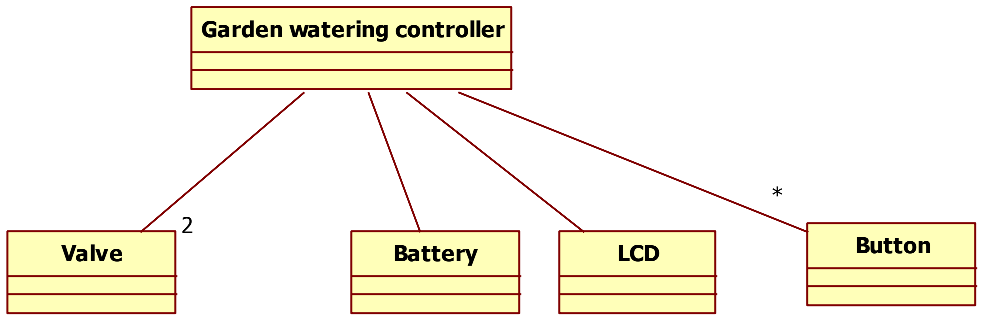
Define the glossary (key concepts and their relationships) (UML class diagram) for the application



List the requirements in tabular form (do not forget to list important NF requirements)

ID	Type (Functional Non Functional)	Description
1	F	Switch on / off
2	F	Show status of battery (or ask to change battery)
3	F	Set up time day
4	F	Open close zone x
5	F	Select default program for zone x
6	F	Define and select program for zone x
7	F	Close valve if rain sensor signals rain. Active program remains active
8	NF	System should be water proof (ex IP25)
9	NF	Energy efficiency: given battery of Y W x hour the controller should be capable of handling Z open close cycles (or should last at least 120 days with default program)

Define the system design model (UML class diagram)



System design must be consistent with Context diagram. Since rain sensor is an external actor, then it must not appear in system design. Valves are assumed to be part of system, so they appear here and not as actors in context diagram.

Sketch the User interface for the device

Many options are possible

2 (7 points) -Define black box tests for the following function, using equivalence classes and boundary conditions.

```
int triangleType(int side1, int side2,  int side3 )
```

This function receives three integers that represent the length of three sides of a triangle.
It returns 1 if the three sides define an equilateral triangle, 2 if isosceles,
3 if scalene, -1 if the three sides cannot be composed in a triangle.

Ex. triangleType(1, 1, 1) → 1
triangleType(1, 1, 3) → 2
triangleType(3, 4, 5) → 3
triangleType(1, 1, 2) → -1

Side1	Side2	Side3	Not a triangle	N equal sides	Position of equals		Test cases
[minint, 0]	[minint, 0]	[minint, 0]	-	-	-	I	(-1, -1, -2, err) Boundary (0, -1, -2, err)
		[1, maxint]	-	-	-	I	(-1, -1, 2, err)
	[1, maxint]	[minint, 0]	-	-	-	I	(-1, 1, -2, err)
		[1, maxint]	-	-	-	I	(-1, 2, 2, err)
[1,maxint]	[minint, 0]	[minint, 0]	-	-	-	I	(1, -1, -2, err)
		[1, maxint]	-	-	-	I	(1, -1, 2, err)
	[1, maxint]	[minint, 0]	-	-	-	I	(1, 1, -2, err)
		[1, maxint]	T	-	-	I	(1,1,2, err)
			F	0	-	V	(3, 4, 5) 3
				3	-	V	(1,1,1) 1
				2	A C A	V	(1,3,1) 2
					A A C	V	(1,1,3) 2
					C A A	V	(3,1,1) 2

3 (7 points) – For the following function define the control flow graph, and define test cases to obtain the highest possible node coverage, edge coverage, multiple condition coverage, loop coverage, path coverage.
For the test cases, **write only the input value**.

WRITE control flow graph here

```
1       int compute_tax(int wage) {
2
3               int ranges[] = {5000,15000,30000};
4               int amount_due = 0;
5               int level=0;
6
7               for(int i=0;i<3;i++){
8                       if(wage>ranges[i])
9                               level++;
10              }
11              if(level==1)
12                      amount_due = 500;
13              else if(level == 2)
14                      amount_due = 1500;
15              else if(level == 3)
16                      amount_due = 3000;
17              return amount_due;
18      }
```

Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined
Node	3	100%	T1,T2, T3
Edge	4	100%	T1,T2, T3, T4
Multiple condition line 8	Not a multiple, 2 are enough (even only 1 because of for cycle line 7)	100%	T1
Loop line 7	3, but not controllable	33%	Any input
Path	For in line 7, in theory 2^3, in practice 4 only Line 11 to 16: 4 Should be 4*4 However paths in the two parts of the function are correlated, so overall 4 paths Feasible		

T1: 600   T2: 1600   T3: 3100   T4: 400

4 (2 points) – Someone asks you to develop a simple web site, a work that requires around 3 person months.

- Which milestones would you define?

Month1, only GUI without logic, month 2 Gui + logic iteration1, month3 delivery

- Which deliverables?

GUI mock up and requirements (M1, design (M1), code (M3)

5 (1 point) – Describe what is an oracle in testing, and the key problems related to it

See slides

6 (1 point) – List the types of defects that can be found in a requirement document

See slides

7 (1 point) – What are the main differences between an iterative and a waterfall process?

Waterfall only performs one iteration

8 (1 point) – Describe shortly the 'Singleton' Design pattern

See slides