Solution to the TDT4140 Exam Summer 2013

Developed by Letizia Jaccheri, checked by Per H. Meland and J. Austvik

Part 1 – Total 30 p.

Fill the following table. Each right answer counts 1.5 points. Wrong answers count -0.5. No answer counts zero.

		True or False?
1	Software specification, software testing, software validation and software evolution are the four fundamental activities of software engineering	F
2	For generic software product development, the specification is owned by the product developer.	T
3	The user requirements are developed after system requirements have been established.	F
4	One of the four principles underlying agile development is that plans are not important	F
5	Test-first development is one of the three important characteristics of extreme programming?	T
6	Functional requirements have direct impact on the developers while not-functional requirements have direct impact on the user.	F
7	There are three main requirements validation techniques. Test-case generation is one of them.	T
8	All requirements should ideally be specified without any knowledge of the proposed solution.	F
9	One of the benefits of reuse is compliance to standards	T
10	A component has both a Requires and a Provides interface	T
11	A service has only a Requires interface	F
12	Dependability is defined as Reusability, Availability, Safety, Security	F
13	If a system has reliability "four nines", it means that it will be available 99% of the time, giving a predicted down time of 3.65 days per year	F
14	Inspections do not require execution of a system so may be used before implementation.	T
15	Testing is more important in Water Fall software development than in Agile software development.	F
16	Development Testing may consist of three subphases: 1) Unit Testing 2) Component testing 3) System testing	T
17	It is not possible to automate unit testing so that tests are run and checked without manual intervention.	F
18	Activity diagrams model both the activities involved in a process and the relationships to the environment	F
19	State diagrams model how the system react to internal and external events	T
20	Sequence diagrams describe only the interaction between actors and the system	F

Merknad [LJ1]: It should be development not testing Ch 1

Merknad [LJ2]: Ch 1

Merknad [LJ3]: It is the opposite CH

Merknad [LJ4]: the plan must exist b must be flexible Ch3 Agile

Merknad [LJ5]: together with Requirements expressed as scenarios, Pair programming, Ch3

Merknad [LJ6]: F – functional requirements have impact on the users. And among the not functional some, like maintainability have impact on the developers but others like reliability on t users.

Merknad [LJ7]: T the other 2 are 1. Requirements reviews
2. Prototyping, CH4

Merknad [LJ8]: You should know something about the solution – see user a system.

Merknad [LJ9]: Ch16 Reuse

Merknad [LJ10]: Chapter 17 Component based

Merknad [LJ11]: It should be Provid

Merknad [LJ12]: Should be Reliabilit

Merknad [LJ13]: It should be 99,99

Merknad [LJ14]: 1.Testing is alwa important – you cannot have agile without testing.

Merknad [LJ15]: Ch8 testing slides

Merknad [LJ16]: ✧Whenever possible, unit testing should be automated so that tests are run and checked without manual interventior

Merknad [LJ17]: It describes also th interaction between the components inside the system.

Part 2 Total 30 p.

Consider the software to be embedded in a nebulizer pump used by children who receive asthma treatment at hospitals or day care centers. The software system is connected to the internet and it can access parameters such as day temperature.

Figure: A child receives nebulizer treatment.

Before use, the software system must be set up by personnel with parameters about the child, such as weight, age, seriousness of sickness. The system calculates the amount of medicine required to be delivered and the duration of the treatment based on information about the child and the temperature of the day. It sends signals to a micro-pump to deliver the correct dose of medicine. During the treatment an adult must be with the child.

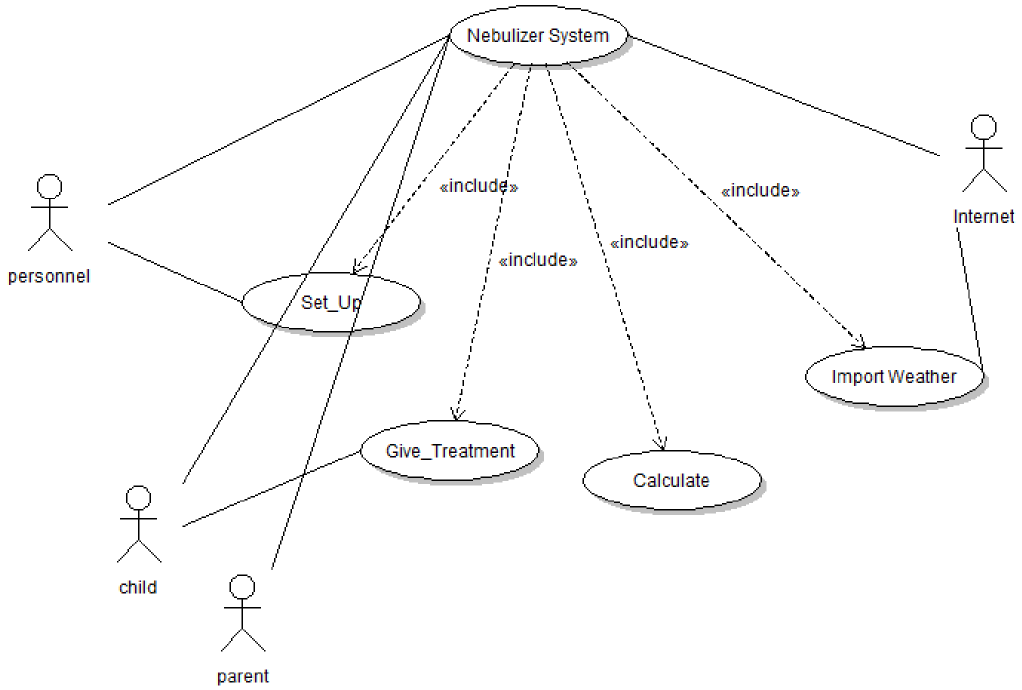
a) Identify the main three stakeholders

1. child

2. personnel

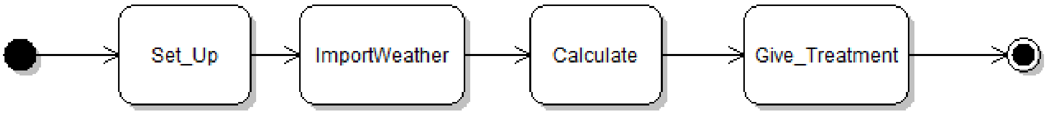
3. parent

b) Define the main Use Case Diagram for the system.



It is important that there is match between the stakeholders at 2a) and the stakeholders at 2b). Internet is not strictly required.

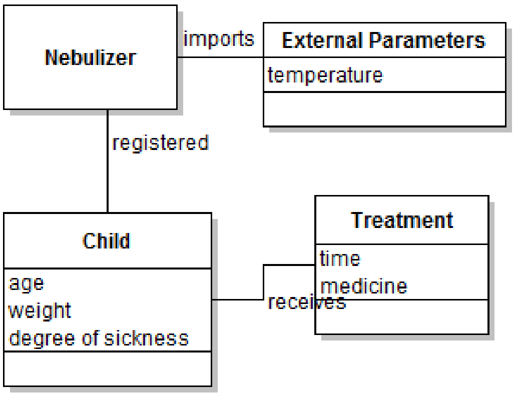
c) Define the main activity diagram for the system



Even if it is not required, we accept solutions which models conditions and loops, like for example after Calculate and after Give_Treatment.

d) Define the main classes and the main class diagram.

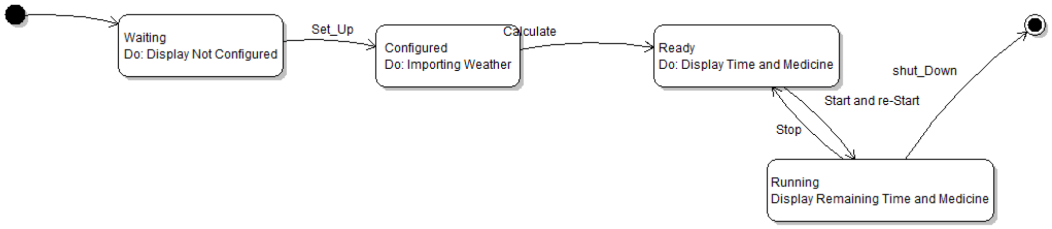
Merknad [LJ18]: Nebulizer, child, treatment, external-paramenters (temperature)


Nebulizer
imports
External Parameters
temperature
registered
Child
age
weight
degree of sickness
receives
Treatment
time
medicine

e) Define the main methods of class Nebulizer (one of these methods is **set up**).

- Set up
- calculate
- Start
- Stop
- Re-start
- Shut-down

f) Define the state diagram for class Nebulizer.



Testing Total 20 p.

Consider the Nebulizer system

a) define the main usage scenario for the system. See Figure 8.10 page 226 in the Text Book for an example.

Before the scheduled treatment the nurse in charge set it up with the information about the child. She provides the personal number of the child, its weight, age, seriousness of treatment. She also refills the pump with medicine. She then issues the command “calculate”. The time and amount of medicine will be displayed. The parent issues the command “give_treatment” and assists the child during treatment. The parent can, if needed “start” and “stop” the pump.

b) starting from the scenario defined at point a) define the main features of the system to be tested. Set_up is one of this feature.

Merknad [LJ19]: See page 226

- Set_up of the information about the child
- Calculation of the required time and medicine dose
- Visualization of progress
- Start, Stop, Restart

c) Define at least 4 Test Cases for Unit Testing for method **set up** of class **Nebulizer**

Function name	Input	Pre-Condition	Expected Output
set up	Child_Id, Age, Weight, Seriousness With Age in Range AND Weight in Range AND Seriousness in Range	Child_Id in DataBase State = Waiting	OK
Set up	Child_Id, Age, Weight, Seriousness With Age in Range Weight in Range AND Seriousness in Range	Child_Id NOT in DataBase	ERROR Child_Id NOT in DataBase
Set up	Child_Id, Age, Weight, Seriousness With Age NOT in Range Weight in Range AND Seriousness in Range	Child_Id NOT in DataBase	ERROR Child_Id NOT in DataBase AND Age NOT in Range
Set up	Child_Id, Age, Weight, Seriousness With Age NOT in Range Weight NOT in Range AND Seriousness in Range	Child_Id NOT in DataBase	ERROR Child_Id NOT in DataBase AND Age NOT in Range AND Weight NOT in Range

d) Using the state model (See Part 2.f above), identify sequences of state transitions to be tested and the event sequences to cause these transitions

[1.] Start -> Waiting -> Set_Up -> Configured -> Calculate -> Ready

[2.] Ready -> Start -> Running ->Shutdown

[3.] Ready -> Start -> Running –>Stop -> Ready

See Page 212 of the text Book.

Process and Project Total 20 p.

Given the following simple process description for company A:

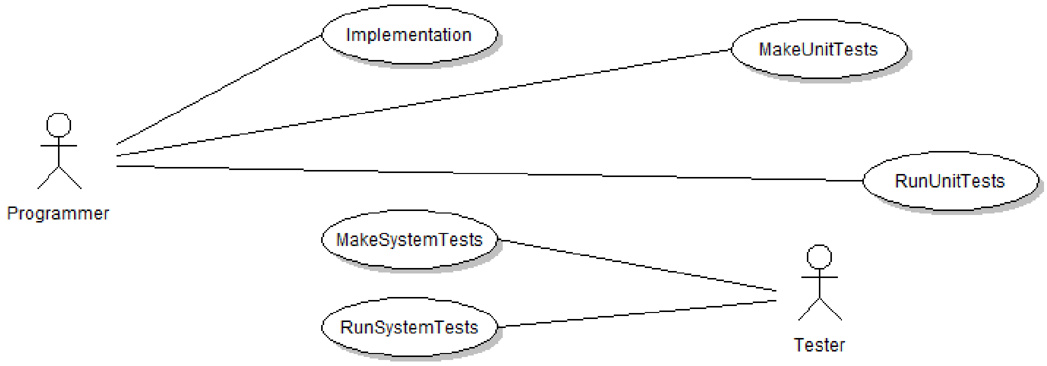
System testing involves a separate testing team with no involvement from programmers

System test cases are developed before the system is implemented

Unit test cases are developed and run by the programmers

a) Define the process description above as one main UML use case diagram

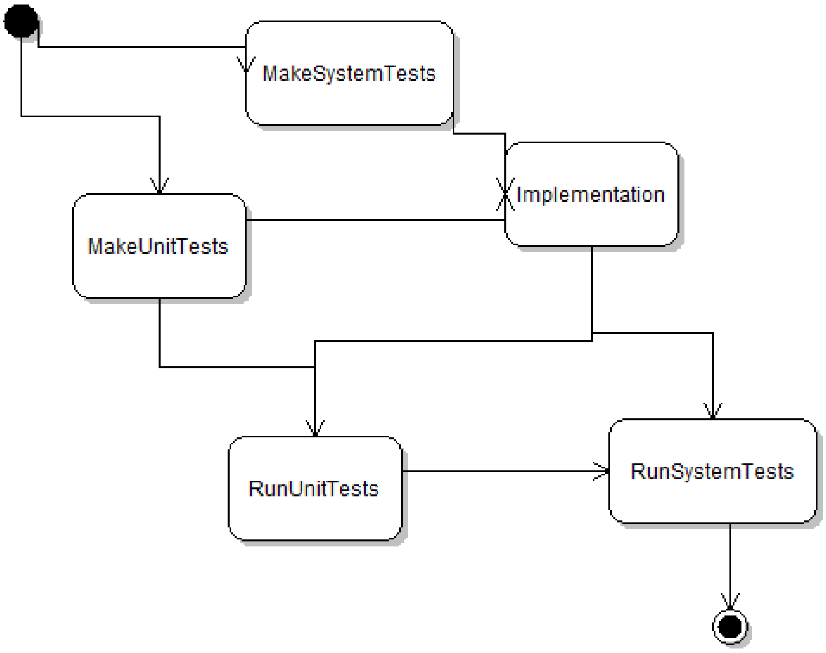
In text form it is (I will make the diagram when we are sure it is the right solution. If the student does not have Implementation we take off 1 point.



b) Define the process description above as one main UML Activity Diagram

Merknad [LJ20]: See for example figure 8.9

It is important that the student understand that one cannot run system tests before implementation nor run system tests before unit tests.



Given the following constraints for a project that follows the process description for company A

The project starts first week of January 2014 and should finish as soon as possible
Mary, Jo, Luise og Pål shall be allocated to the project. Each one costs to the company 1000 NOK per hour (this includes salary, social costs, and overhead).
Mary is an analyst and testing expert
Jo and Luise are skilled programmers
Pål is a project manager and analyst
Each task lasts between two and four weeks.
The estimated budget is 1.4 million NOK

For each task, estimate duration in days, effort (person day), dependencies, and resource allocation. Each person can work on a task a percentage of her time. See figure 23.5 text book as inspiration but use the table provided here.

task #	task name	start (week)	end (week)	duration (days)	effort (person days)	dependencies	resources
t0	project management	1	16	80	16		pål(20%)
t2	MakeSystemTests	1	4	20	34		mary (100%) pål (70%)
t3	MakeUnitTests	2	4	15	30		jo (100%) luise (100%)
t4	Implementation	5	8	20	40	t2,t3	jo (100%) luise (100%)
t5	RunUnitTests	9	12	20	34	t4	mary (100%) pål (70%)
t6	RunSystemTests	13	16	20	20	t5	mary 100%
total (days)					174		
budget					1392000		