NTNU
Norwegian University of Science and Technology

Faculty of Physics, Informatics and Mathematics

Department of Computer and Information Sciences

ENGLISH

Sensurfrist: 2012-06-19

Exam in
TDT 4242 Software Requirements and Testing

Saturday May 19, 2012
9:00 am – 1:00 pm

Aids allowed A:

Pocket calculators allowed
All printed and handwritten material is allowed

Contact person during the exam:

Professor Tor Stålhane, phone 73594484

Good Luck!

Introduction

In this exam you can score a maximum of 50 points. The remaining 50 points for the semester comes from the compulsory exercises.

If you feel that any of the problems require information that you do not find in the text, then you should

- Document the necessary assumptions
- Explain why you need them

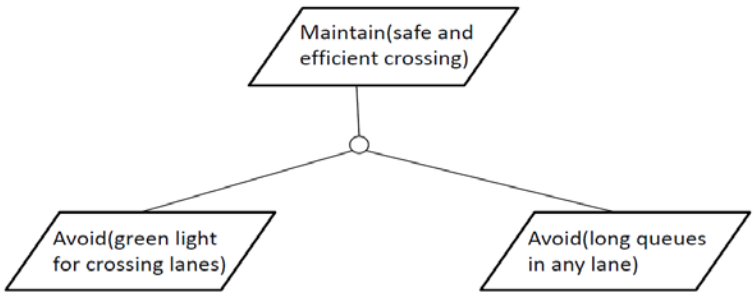
Your answers should be brief and to the point.

Problem 1 – Requirements engineering (20 points)

The case is a common traffic light with car sensors in all lanes leading into the crossing – see diagram in appendix 1.

1a – Temporal patterns – 10 points

Use the informal temporal patterns – see appendix 2 – to describe the two upper levels of the traffic light's control unit



It is important that the circle (and-symbol) is used in the goal decomposition.

1b – Textual use cases – 10 points

Write a textual use case for the following situation:

- Lanes C – D have green light.
- A car arrives in lane A and the lights shall change so that this car can pass through the crossing

State		***Lanes C – D has green light***
Triggering event		***A car arrives in lane A – priority lane***
Sequence number	***Input***	***System action***
1	***The lane A sensor is set to 1 (high)***	***Lights in lanes C and D are set to amber – empty the crossing*** ***Start timer 1 – 60 seconds*** ***Start control timer***
2		***Lights in lanes A and B are set to red / amber – get ready*** ***Start timer 2***
3	***Timer 1 releases***	***Lights in lanes C and D are set to red***

	or sensors in lanes C and D are set to 0 (low)	
4	***Timer 2 releases***	***Lights in lanes A and B are set to green***
5		***Stop control timer***
-	***-***	***End of use case***
6	***Control timer releases***	***Register that something is wrong***
7		***Set lights in all lanes to blinking amber***
8		***Report error to central computer*** ***Inhibit all lane sensors***
-	***-***	***End of use case***

Note: handling of a control timer and steps 6 – 8 are not needed to get a full score.

Problem 2 – Testing methods (15 points)

2a – Methods choice – 5 points

1. Describe strong and weak points for using random testing
    - **Strong points:**
        - **Can be done automatically**
        - **Can generate many tests in a short time**
    - **Weak points**
        - **Needs an oracle for automatic checking**
        - **Otherwise, only manual checking is possible**
2. Describe strong and weak points for using domain testing
    - ***Strong points***
        - ***Is efficient – need only one test-set per domain – on- and off-points***
    - ***Weak points***
        - ***It is difficult to identify domain borders for large pieces of software***

3. Give one example where we should use random testing and one examples of where we should use domain testing

    - ***Random testing – crash test. Generate random inputs, both correct and illegal ones. No oracle is needed except for crash – no crash.***
    - ***Domain testing. Small components – e.g., small classes or methods inside a class where it is easy to identify the domain borders***.

2b – Testing methods – 10 points

The most critical failure that can occur for a traffic light is that it gives a green light for two crossing lanes at the same time.

1. Explain why random testing is a good way to test this requirement

    It is simple to
    - ***generate inputs – only 1 or 0 from each sensor with random time intervals***
    - ***check the result – status of the two pairs of traffic lights***

2. Give a short description of a test strategy that can be used to test the requirement "The system shall not at the same time give a green light for two crossing lanes"

   A black box test will be sufficient. There are four inputs that can only be 1 (car present) or 0 (no car present). Thus we can easily construct four random input generators. The oracle is simple and can be automated – report an error if we get green lights for two crossing lanes.

   It is important to note that this test will only test the software. The rest – traffic lights plus lane sensors must be tested using e.g., scenario testing.

Problem 3 – Non-functional requirements (15 points)

The software in the traffic light system controller shall be easy to extend – e.g., with extra signal lights for pedestrians – and easy to maintain.

Write a set of requirements with corresponding tests for maintainability and extendability for the software of the traffic light control system.

We will take the definition of maintainability in ISO 9126 as our starting point. Note – extendability is not part of this standard and we will thus focus on maintainability. According to ISO 9126, maintainability consists of:

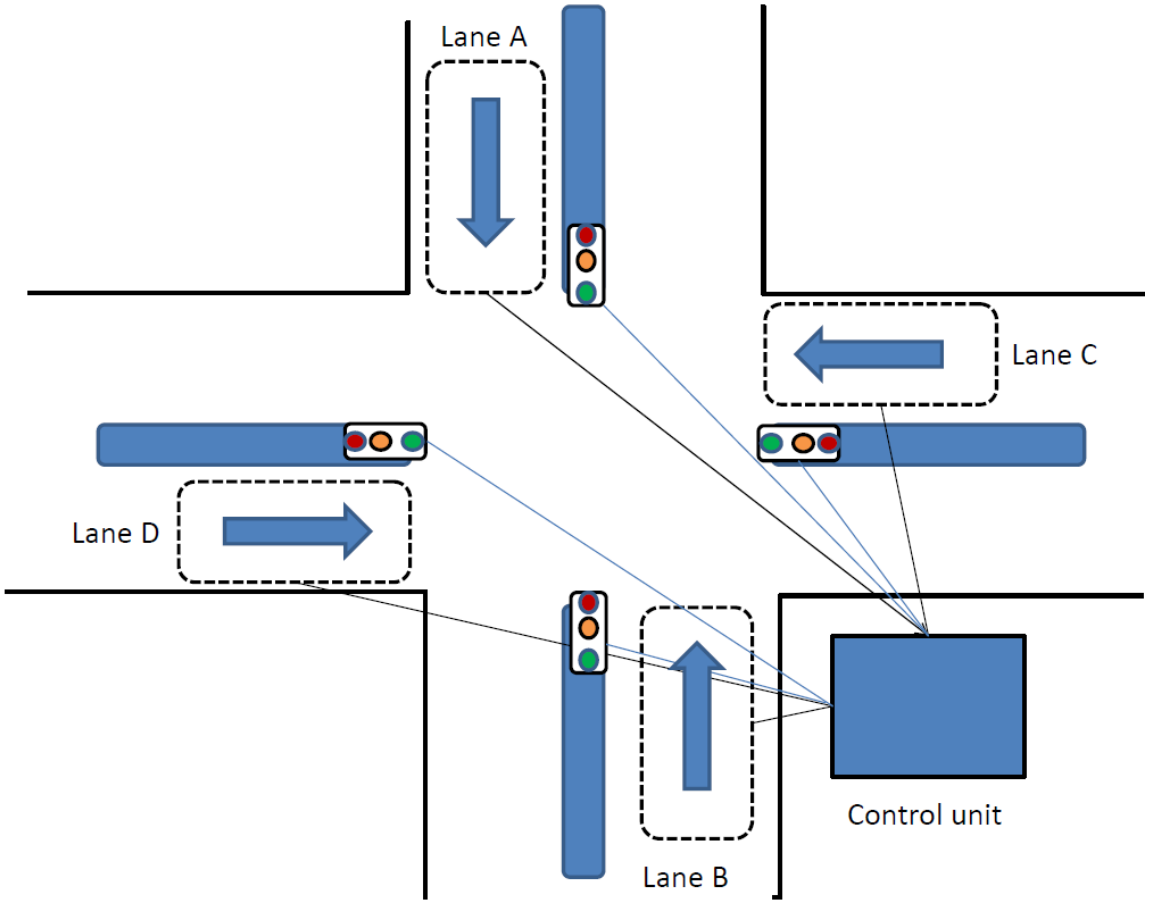
- ***Analysability***
  - ***Requirement: there should be a complete traceability matrix for functional requirement ⇔ code***
  - ***Test: manual spot check – the code traceability matrix is checked tor three requirements***
- ***Stability***
  - ***Requirement: no global variables are allowed and components that belongs together should be organized using a facade pattern***
  - ***Test:***
- ***Changeability***
  - ***Requirement: strict adherence to company naming conventions.***
  - ***Test: manual spot check – the naming used is checked in the four largest components.***
- ***Testability***
  - ***Requirement: complete traceability between functional requirement ⇔ test and strict adherence to analysability requirements***
  - ***Test: manual spot check – the test traceability matrix is checked tor three requirements***
- ***Maintainability – overall requirements and tests***
  - ***Requirements: small changes – estimated to be less than 50 statements – shall take less than two person-days. Medium changes – estimated to be 51 to 100 statements – shall take less than five person days. There are no requirements for large changes – estimated to be more than 100 statements.***
  - ***Tests: we need to test both insertion of new functionality (extendability) and the correction of a fault (maintainability).***
    - ***Fault: a fault that requires a small change is inserted into the code. The time needed to fix should be less than two person-days.***

- ***Extension: request a change in functionality that requires 50 – 100 statements of new or changed code. The needed time should be five person-days or less.***

Note: it is OK to give full score to students who have only used requirements and tests for maintainability. The number of statements used is not critical but it is unreasonable to set the same requirements of resources for all kinds of changes. It is not necessary to specify how the size of a change is estimated.

A simple alternative for all the requirements above could be to let a number of developers that have not been working on the project and let then assess either each component of maintainability or the total maintainability alone e.g., on a ten point scale. The requirement could then be that none of the participants should give a lower score than 6 or that the average score should be better than seven. This approach may replace both the detailed spot checks and the real tests of maintainability and extendability.

Appendix 1 – Traffic lights



The squares with rounded corners marked with dotted lines are the sensors. They consist of an energized loop. The permeability of the loop will increase strongly when there is a large mass of iron – e.g., a car – inside the loop. This increase is used by the sensor to give the signal "car present in lane" to the control unit.

The following rules apply:

- The route lane A – lane B is prioritized. Thus, when no cars are present in any of the sensor loops, the traffic lights for lane A and lane B are green and the lights in the other directions are red.
- If the traffic lights for lane A and lane B are red and the lights in the other directions are green, the traffic lights for lanes A and B will switch to green and the traffic lights for lanes C and D will switch to red within maximum one minute if a car is detected in lane A or B.
- The light cycle is the standardized cycle: green – amber – red – yellow – green
- You can only turn left or right if you have a green light. The rule used in e.g., the US that you can turn right with care, even if the light is red does ***not*** apply here.
- If the system detects hardware or software errors, all lights shall shift to blinking amber.

Appendix 2 – Temporal patterns for requirements