UML for Embedded Systems

Exam FALL 2021
Software of a space-based embedded system

Ludovic Apvrille

ludovic.apvrille@telecom-paris.fr

http://soc.eurecom.fr/UMLEmb/

During an exam, you are not supposed to talk with anyone else, by any means (including mobile phones, chat, etc.), see question I.

A grade is provided for each question. 1 bonus point is awarded for writing quality (report and models).

1 Objective

Your objective is to model the **software** of a space-based embedded system.

You have exactly 3 hours to model this system and answer various questions: the time is very short. This means that **you have to make modeling assumptions**. **Keep your diagrams simple and readable**, in particular the analysis diagrams.

Your grade takes into account your report and your models. At the end of the exam, **reports** (in pdf format) and **models** (in TTool format) **must be sent to me by email**, with Alexia Cepero in cc. The report should contain explanations concerning your models, as well as relevant screen captures of models (e.g., interesting simulation traces, formal verification results).

2 System specification

Again, the system to model is the software running aboard the space-based system described below.

2.1 Description

2.1.1 Overall description

A ground station needs to regularly monitor the safety data of a space-based system: 3D position, temperature, battery level, fuel quantity. For this, a ground station can send, via radio-frequencies, a TC (*TeleCommand*) to the space-based system. Once received by the RF receiver, the software of the space-based system gets the request for information. Data of TCs are ciphered. Once the software has deciphered data, it stores data in an intermediate buffer, and a task to handle this request is triggered. This task builds the answer by reading requested values from sensors. Once the answer packet has been built, it is first enciphered and then sent via a TM (*TeleMetry*) to the ground station, using the RF transmitter.
To ensure that the system does not crash, a microcontroller of this system is dedicated to execute a software task that checks, every 10ms, that all other software tasks of the space-

based embedded system are still responsive. For this, a signal is sent to each task. If some of the tasks have not responded to this signal, then the whole system is restarted, apart from the watchdog: the latter is not expected to crash, apart if the battery is too low to power the microcontroller. Obviously, this watchdog task is of prime importance for this reliability of the system.
Sometimes, while the software system is computing a TM, another TC is received. To avoid redundancy, the TM under construction is canceled: a new TM corresponding to the latest TC is computed and sent.
Last but not least, space-based systems are not well protected against high-energy particles. Such a particle can provoke a bit flip from 0 to 1, or the opposite. The memory is the most sensitive elements of the platform. Therefore, for each block of data the software writes into memory, an error correction code (CRC) of this block has to be computed by the software and stored into memory along with the data block. When this block is read, the corresponding CRC must also be read and checked.

3 Assignments

I. Personal work

1. Recopy the following text at the beginning of your report (this is mandatory)

```
I pledge on my honor that I will not
receive any unauthorized help on this
exam, that I will not help others in any
way on this exam, and that all my
answers will be my own personal work.
```

II. Assumptions

1. Your assumptions should be clear. Do list them in the report: that list might evolve according to the models you make afterwards. Make a clear separation between environment and system assumptions. [2 points]

III. Requirements

1. Create a requirement diagram. [3 points]

IV. Analysis

1. Make a use case diagram. [3 points]

2. Continue the analysis in the form you want: activity diagrams, nominal scenario, error scenarios, . . . : you are free to use the diagrams you want. Of course, the idea here is to show important points of the specification. [3 points]

V. Design and validation

1. Make a block diagram. Put the emphasis on which blocks are used to model the system being designed, and which ones are used either to model the environment, or to prove properties (observers). [2 points]

2. Draw state machines, and provide a nominal simulation trace, as well as an error trace. [3 points]

3. Prove that a TC always result in a TM, apart if another TC is received before sending TM. Last, from requirements, define a property of your choice, and prove whether it is satisfied (or not!). And obviously, explain how you have modeled these two properties [3 points]

Good luck!