

Requirement analysis, design and verification

RADV, 2IW20

Jan Friso Groote (J.F.Groote@tue.nl, HG 6.75, 040-247 5010)

Mohammad Mousavi (M.R.Mousavi@tue.nl, HG 6.79, 040-247 2993).

Yaroslav Usenko (Y.S.Usenko@tue.nl, LaQuSo 040-247 3585).

<http://www.win.tue.nl/~jfg/educ/2IW20/2005/overzicht.html>

The purpose of this course is to learn means to specify behaviour of systems and to experience the design of a system, starting from behavioural requirements which must be proven to hold on the design.

The first five weeks of the trimester will be dedicated to learn the specification language μ CRL and to use it as a manual verification tool. After these five weeks there will be an examination. There is one possibility to retry this exam, at the end of the second trimester. The next five weeks are dedicated to an assignment. Here the goal is to apply the techniques and tools to the design of a small distributed and/or embedded system. The purpose is to design this system such that it is proven to comply with the requirements.

The marks for the exam and the assignment both contribute equally to the final score.

Literature

The course material consists of

- W.J. Fokkink, J.F. Groote and M.R. Reniers. Modelling Distributed Systems. Lecture notes. 2003.
- A.G. Wouters. Manual for the μ CRL toolset (version 2.8.2). Technical Report SEN-R0130, CWI, 2001.

The lecture notes by Fokkink, Groote and Reniers and the manual by Wouters can be obtained at the office of the secretary (Tineke van de Bosch, 247 5010, HG 6.74). The documents can also be downloaded from the website. The exam will cover the lecture notes on modelling distributed systems as well as everything said during the lectures. The manual for the μ CRL toolset will not be a part of the exam.

Assignment

The assignment consists of designing a controller for a small distributed and/or embedded system. Below a suggestion for such a system can be found, but, in consultation with the teachers, other systems can be chosen as well. The assignment can be carried out in groups of one to four persons.

Carrying out the assignment consists of the following steps:

1. Identify the global requirements on the whole system in natural language (i.e. English or Dutch). A typical requirement is that ‘the patient support system may never make a motorized horizontal movement while in emergency mode’.

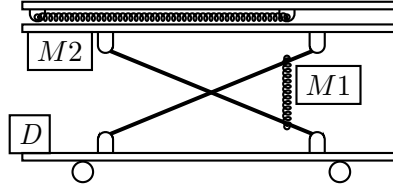


Figure 1: A movable patient support unit.

2. List the interactions of the control system with the outside world. These are for instance ‘turning a motor on’, ‘reading that the up button is released’ and ‘applying a break’. Describe clearly but compactly the meaning of each interaction in words.
3. Translate the global requirements in terms of these interactions.
4. Describe a compact architecture of the structure of the control system.
5. Describe the behaviour of all components in the architecture in terms of μ CRL.
6. Verify using the toolset that all requirements given in item 3 above are valid on the design in μ CRL.

The assignment must be documented in a technical report that covers all items above. This report must be a concise technical account of the system and must be written in such a way that from it the requirements, action interface, architecture and behaviour the design can be easily understood. It must also be clear how the requirements have been verified, in such a way that this can easily be redone without consulting any of the authors of the report.

The default assignment is the following, inspired by current developments at Philips Medical Systems where new, more flexibel, medical scanning machines are under construction. Current magnetic resonance scanners can be found at

<http://www.medical.philips.com/main/products/mri/products/>.

One of the extensions is a Movable Patient Support Platform, MPSP for short, which is basically a trolley on which a patient can lie. The trolley can either be uncoupled from the scanner or be docked. The advantage of this is that the patient can be prepared in a separate room, while another patient is being scanned. Currently patients are prepared while in or near the scanner, and during this time the scanner is idle.

There are two motors in the MPSP. Motor $M1$ controls the vertical and motor $M2$ controls the horizontal movement. Both motors have breaks that can be turned on and off separately from the motors. Horizontal movement is only allowed when the support unit is docked, i.e. connected to scanner. If the MPSP is not docked, the patient rest must always be in the rightmost position in figure 1, as otherwise there is a possibility for the MPSP to tumble over. When undocked the horizontal break must always be applied. The vertical break must always be applied, except when the vertical motor is on. If a motor is on, the breaks must not be used, as otherwise the motors will overheat possibly damaging motors.

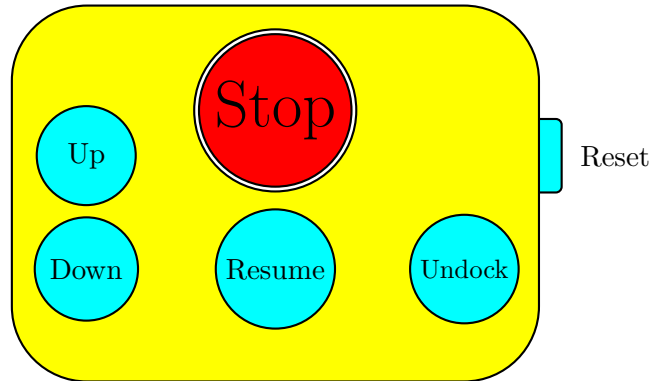


Figure 2: The user console on the patient support unit.

The movements are controlled via a console on the patient support unit, which is designed to be as simple as possible. There are up and down buttons moving the bed in both the vertical and horizontal direction. There is a stop button which puts the platform in emergency mode. When in emergency mode, the patient rest is (partly) inside the scanner, breaks must be released to allow medical staff to manually drag the patient outside the scanner. This may be useful when an emergency occurs (a heart-attack while scanning), but also when a system malfunction happens (break-down of the horizontal motor). The resume button puts the support unit back to normal operating mode. Finally, the Undock button can be used to disconnect the MPSP from the scanning device. The reset button is treated later.

Every scanners can have different height, generally dependent on how it is installed in the hospital. The patient rest can only be moved inside the scanner if scanner and rest are at the same height, which is called the standard height. Before use, the MPSP must be calibrated by setting the standard height. This is done as follows. The bed is docked. This is detected via a sensor in the docking unit D . Then using the up and down buttons the bed is moved to the correct height. By pressing the reset button once, this height is set to be the standard height. When pressing the reset button, while the bed is not docked, the standard height is forgotten and the bed goes into uncalibrated mode.

When the bed is undocked or uncalibrated, the up and down buttons can only be used to move the bed up and down. There are two detectors, detecting that the bed is in the uppermost and lowermost position. The patient rest is not allowed to move above the uppermost and below the lowermost position.

When the support unit is docked and calibrated, the patient rest will stop when it has reached standard height. If the up button is pressed at standard height, the bed moves into the scanner. In order to avoid unexpected movements it is important the the up button is released before the inward movement is commenced. This means that releasing the up and down buttons are important interface actions also.

When the platform is in the scanner and the down button is pressed, it moves outward, until the outward horizontal detector indicates that the bed is completely outside the scanner. By releasing and pressing the down button again the bed will subsequently move downwards.

Note that while the support unit is docked, it cannot be moved above standard height unless it was already above standard height when being docked (or when in uncalibrated mode).

When the undock button is pressed a message is sent to the scanner which will undock the platform. For this a gentle spring mechanism is used that pushes the platform away from the scanner. The undock message may never be sent to the scanner if the platform is not in outermost position, as otherwise there is a risk that the support unit will tumble over.

The assignment is to design a set of controllers that must operate the platform in such a way that no harm can ever be done to a patient or to the equipment. It is a strict requirement that at least two separate controllers are being used that communicate via an asynchronous network. The model must be such that qualified programmers can easily implement it. This assignment is underspecified. This means that in certain cases you have the freedom to make your own design decisions.

Note that the actual platform is much more complex than the one described here. For instance, there are at least three emergency modes, the platform can also be controlled via a console on the scanner, or via an operator on a host computer. In all cases the MPSP can respond differently. For instance, it is not allowed to move the patient up and down via the host computer. And if the platform is in uncalibrated mode, it may only be moved up and down via the console on the MPSP.

Tool sets

See the web page of this course for information about the tools and the toolset.

Global time schedule

Below a global time schedule is indicated. Furthermore, there are exercises indicated, which will be treated during the lectures. Students are supposed to make these exercises before the lectures, and compare their answers with those of the lecturer. The exercises can be found in the lecture notes.

28-11-2005. Abstract data types.

30-11-2005. Elementary process algebra, RSP, bisimulation. Exercises 2.5, 2.12, 2.14.

5-12-2005. Data in actions and processes, sum operator and conditionals. Exercises 3.3, 3.7, 3.10, 3.13, 4.3. Lecture will be given by Y. Usenko.

7-12-2005. Linear process equations, CL-RSP, Invariants (chapter 8). Exercises 3.17, 3.20, 3.21.

12-12-2005. Hidden actions. Exercises 8.8, 8.18.

14-12-2005. Protocol specifications. Exercises 5.1, 5.7, 5.11, 5.12.

19-12-2005. Linearisation of processes. Exercises 6.8, 6.11.

21-12-2005. Confluence and τ -priorisation. As exercises linearise the processes $X = a \cdot X \cdot X$, and $X = Y \cdot \delta$, $Y = a \cdot Y \cdot Y + c$.

- 9-1-2006. Cones and foci theorem. Exercises 7.9, 7.10 and 9.7.
- 11-1-2006. Questions and reserve. Exercises 9.4 and 9.5.
- 23-1-2006. Exam.
- 1-2-2006. Assignment, μ CRL toolset. Lecture will be given by M. Mousavi.
- 8-2-2006. Tools such as parelm, constelm, structelm, etc.
- 15-2-2006. Instantiator, confluence reduction.
- 22-2-2006. Prover. Modal logic. Muccheck.
- 13-3-2006. Exam (retry).
- 24-3-2006. Latest possible date to hand in the assignment.