

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).

Michel Reniers (M.A.Reniers@tue.nl, HG 6.76, 040-247 2999).

<http://www.win.tue.nl/~jfg/educ/2IW20/winter2007/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.

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, also another system can be chosen. The assignment can be carried out in groups of one to four persons.

Carrying out the assignment consists of the following steps:

1. Identify in words global requirements on the whole system. Typical requirements are ‘both doors of a lock leading to the vacuum chamber can never be open at the same time’. These requirements can be described in natural language.
2. Identify the interactions of the control system to the whole system. 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.
7. Address what will happen in case of faulty behaviour of the whole system. This can either be done by modelling faulty behaviour explicitly, or by arguing that the design of the control system is such that it properly deals with faulty behaviour.

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 such that from it the requirements, action interface, architecture and behavioural design can be easily understood. It must also be clear how the requirements are verified, in such a way that this can easily be redone without consulting any of the authors of the report.

At ASML (www.asml.nl), the world leading wafer stepper manufacturer, situated in Veldhoven, near Eindhoven, a new generation of wafer steppers is currently under construction. Wafers are being used to produce integrated circuits (ICs). Transistors and other components are etched using a mask and a light beam on the wafer. The components on IC’s become increasingly smaller. This leads to the need for light with a higher frequency, i.e. in the ultraviolet bandwidth. But as the atmosphere absorbs ultraviolet light, the whole production process must take place in vacuum.

Therefore, wafers must enter the machine via air locks. In our setup there are two locks. Robots are used to move the wafers around. They put the wafers on special stands, from which other robots can pick the wafers up again. In the vacuum chamber there is a separate unit that is responsible for projecting an image onto the wafers. This unit must be told that a wafer to be processed is ready, and it will indicate that a processed wafer can be moved back to the outside of the waferstepper. The whole machine is depicted in figure 1.

The assignment is to design a controller for the movement of the wafers in and out the waferstepper. The controller must control and check the status of doors, pumps and vacuum sensors. As you are the designer of the system, you are free to add extra functionality, such as extra buffer capacity, extra sensors or extra robots. Of course there is a trade-off between additional cost and additional benefit that must be respected. Most important however is that the controller will function as desired. This means it will not violate any of the requirements that you have posed on it. The design of the controller must also be such that qualified programmers can easily implement it.

Typical requirement breaches that can be expected are:

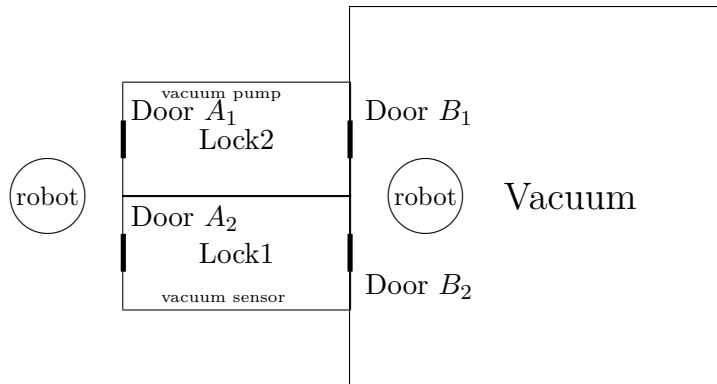


Figure 1: A simplified UV wafer stepper, with two air locks and two robot arms

- More than one wafer is put in any position.
- Robots try to move wafers through closed doors.
- The doors are simultaneously being opened.

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.

27-11-2006. Abstract data types.

29-11-2006. Elementary process algebra, RSP, bisimulation. Exercises 2.5, 2.12, 2.14.

4-12-2006. Data in actions and processes, sum operator and conditionals. Exercises 3.3, 3.7, 3.10, 3.13, 4.3.

6-12-2006. Linear process equations, CL-RSP, Invariants (chapter 8). Exercises 3.17, 3.20, 3.21.

11-12-2006. Hidden actions. Exercises 8.8, 8.18.

13-12-2006. Protocol specifications. Exercises 5.1, 5.7, 5.11, 5.12.

18-12-2006. Linearisation of processes. Exercises 6.8, 6.11.

20-12-2006. Confluence and τ -prioritisation. As exercises linearise the processes $X = a \cdot X \cdot X$, and $X = Y \cdot \delta$, $Y = a \cdot Y \cdot Y + c$.

- 8-1-2007. Cones and foci theorem. Exercises 7.9, 7.10 and 9.7.
- 10-1-2006. Questions and reserve. Exercises 9.4 and 9.5.
- 30-1-2007. Exam (9:00-12:00).
- 7-2-2007. Assignment, μ CRL toolset.
- 14-2-2007. Tools such as parelm, constelm, structelm, etc.
- 28-2-2007. Instantiator, confluence reduction.
- 7-3-2007. Prover. Modal logic. Muccheck.
- 22-3-2007. Exam (retry, 9:00-12:00).
- 16-3-2007. Latest possible date to hand in the assignment.