Distributed Systems

Chapter 3 – Questions

WI. How is the security property of a program defined (in a sentence)?

W2. What characterizes an immutable object? What is a stateless method?

W3. When is an object fully synced? What guidelines must an object follow to be safe?

W4. How is the liveliness of a program defined? What is meant by the extended concept of vitality?

W5. What is a deadlock?

w6. What possibilities are there to ensure mutual exclusion?

W7. What is a semaphore? How is the P() operation defined as the V() operation? What is a fair semaphore?

w8. What is a monitor? How does the wait() signal work? How does the signal() signal work? Which methods are used for signaling on monitors in Java?

W9. Why not suspend threads from execution by calling suspend()?

Wl0. How do the new lock classes help deadlock avoidance?

Exercises

ex 1 : Modify program 3.12 to use monitors instead of semaphores.

ex2. Extend the class Leaf from Exercise 14 (Chapter 2) so that a static variable counts how many leaf objects are created.

ex3. Program a seven-suspendable thread using semaphores. The suspendable thread should be suspended from the main thread in an endless loop and woken up again after two seconds. After waking up, the main thread should do without its time slice. The suspendable thread is supposed to increment a variable and output it in each increment.

ex4. Write a program that reads in a number range from the user, i.e. upper and lower limit, and a number of calculation threads to be started. A load distribution should take place, i.e. each arithmetic thread should fetch a number from the number range that has not yet been checked and check whether this number is a prime number. Then it outputs the number with the test result (true or false). The process is repeated until all numbers in the number range have been checked.

ex5. Modify the semaphore from Program 3.16 to use locks instead of the synchronized keyword.

End of exercise and question paper