Department of Computer Science	Exam **Distributed Systems**
Vrije Universiteit	Exam code: X_400130
Dr. T. Kielmann	18 – 12 – 2013

This is a *"closed book"* exam.
No printed materials or electronic devices are admitted for use during the exam.
You are supposed to answer the questions **in English**.

Wishing you lots of success with the exam!

Grading: The final grade is calculated by accumulating the scores per question (maximum: 90 points), and adding 10 bonus points. The maximum total is therefore 100 points. To pass the exam, it is sufficient to get at least 55 points.

1. Distribution Transparency and Scaling

1.a: Name five out of the seven forms of distribution transparency. 3pt
1.b: Give a compelling example why full distribution transparency can not be achieved (be brief!) 3pt
1.c: Using the example of a Web-based information system, explain which techniques can be used to deal with size scalability, geographical scalability, and administrative scalability. (be brief, one technique each!) 5pt

2. Architecture

Explain decoupling in *space* and in *time*. For the following three architectures, explain if/how they decouple components in space and/or time: RPC, a publish/subscribe system, a bulletin board. 5pt

3. RPC/RMI

3.a: How does RPC achieve location transparency? What are the limitations in this respect? 3pt
3.b: How does Java RMI overcome some of these limitations? 3pt
3.c: If your middleware only supports synchronous RPC, what can you do to improve geographical scalability? 3pt
3.d: In DCE RPC, how can a client find the endpoint serving a particular procedure? 2pt

4. DNS

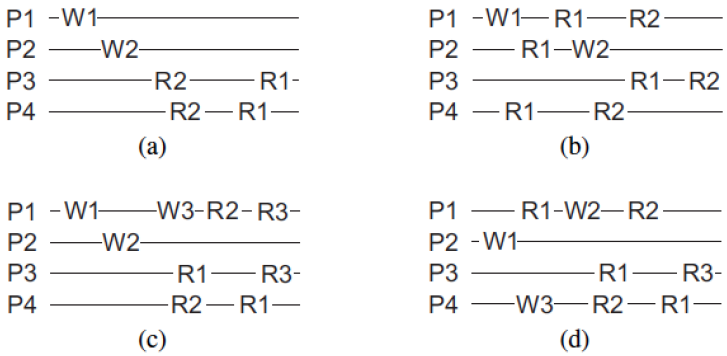
4.a: Explain how iterative name resolution in DNS works and why it may incur high latency costs. Give an example where recursive and iterative resolution is effectively combined. 6pt
4.b: When resolving the name *www.distributed-systems.net*, DNS will return an address, but not when trying to resolve *distributed-systems.net*. Explain why! 6pt

5. Logical clocks

5.a: Explain how Lamport's logical clocks work. 6pt
5.b: Explain how totally ordered multicast can be implemented with Lamport's logical clocks. 8pt
5.c: Provide an alternative implementation that realizes totally ordered multicasting. 6pt

6. Consistency

6.a: Consider the following four execution. We write $W1$ as an abbreviation for $W(x)1$, and likewise $R2$ for $R(x)2$, where x is the variable shared by the processes $P_1, \ldots, P_4$. Which of these four executions are sequentially consistent, and which ones are not? Explain your answer. Hint: think twice in cases (c) and (d). 6pt



6.b: Give an eample where using only client-centric consistency will lead to a conflict between update operations. 5pt

7. Fault tolerance

7.a: How large must a k-fault tolerant group be for halting failures and for arbitrary failures? 4pt

7.b: Consider a k-fault tolerant group, with $k > 1$. Assume that one process fails. Do we still have a k-fault tolerant group? Explain your answer! 4pt

7.c: For reaching consensus among a primary server P and backup servers $B_1, \ldots, B_{n-1}$ in the presence arbitrary failures, which two requirements must be met for *Byzantine agreement*? 2pt

7.d: For reaching Byzantine agreement, how many processes are needed for tolerating k failures? Show this by drawing diagrams of messages exchanged (for $k = 1$), for both a faulty primary and for a faulty backup process. 10pt