TECHNISCHE UNIVERSITEIT EINDHOVEN
Faculteit Wiskunde en Informatica

Examination Architecture of Distributed Systems (2IMN10),
on Thursday, January 31, 2019, from 18.00 to 21.00 hours.

Before you start, read the entire exam carefully. Answers to all questions must be motivated and stated clearly. For each question the maximum obtainable score is indicated between parentheses. The total score sums up to 20 points. This is a closed book exam, i.e., you are not allowed to use books or other lecture material when answering the questions.

1. (2 points) Describe the Client-Server architectural style using the appropriate vocabulary. Name the concepts and rules involved, give a motivation for its usage, and mention typical behavior and its weak points.

   Answer. See slide 11 of the slide set on architectural styles.

2. For name spaces that are distributed across multiple name servers one distinguishes between iterative and recursive name resolution.

   (a) (1.0 point) Describe in some detail how each mechanism works.

   Answer. See vST (3rd ed.) Figures 5.17 and 5.18 or TvS (2nd ed.) plus accompanying text.

   (b) (0.5 point) Give an argument in favor for iterative resolution.

   Answer. In general, iterative resolution puts a lower load on non-leaf servers than recursive resolution. This is especially important for servers in the global layer, that are most frequently contacted. For root-servers, this is so important that these servers do not support recursive resolution.

   (c) (0.5 point) Give an argument in favor for recursive resolution.

   Answer. With recursive resolution caching can be more effective. Resolution requests ending on the same suffix may share usage of a cached entry for that suffix. Also recursive caching leads to cheaper communication, because it involves fewer long-distance communications. See vST (3rd ed.) Fig. 5.20, or TvS (2nd ed.) Fig. 5.18.

3. Component-based software engineering is an approach to system design that creates software systems from independently developed components.

   (a) (1 point) Give the (generic) definition of a component.

   Answer. See slide 10 of the slide set on component-based software engineering (CBSE).

(b) (1 point) What is the role of a component-framework in this approach?

Answer. See slides 12 and 13 of the slide set on component-based software engineering (CBSE).

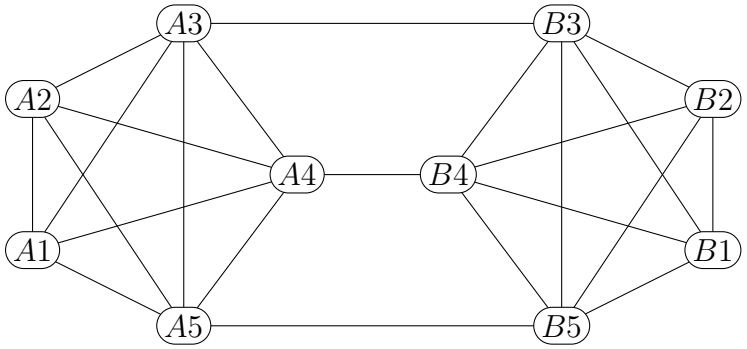
4. Consider a replicated distributed data store with N replicas each managed by an individual replica manager (RM). Each client always accesses the data store via a single RM, but may contact distinct RMs in successive operations. To increase availability under network partitions, while maintaining eventual consistency, the data store uses Gifford's quorum protocol, with read quorum size NR and write quorum size NW.

   (a) (1.0 point) Explain how Gifford's quorum protocol works and indicate the constraints that need to be imposed on the quorum sizes in order for the protocol to achieve consistency.

   Answer. To execute an operation the data store needs to establish a subset of RMs, called a quorum, that is capable to engage in the operation, i.e., is reachable from the RM where the operation is submitted to the store. For read operations, the size of the set is given by NR, and for write operations by NW. Upon writing, the data object is updated in all replicas of the write quorum and is given a unique time stamp (version number) that is more recent than any time stamp handed out in an earlier update. Upon reading, the value returned to the clients is the value from the replica that holds the most recent time stamp. For a data store with N replicas, the quorum sizes need to satisfy two constraints:

   - $NR + NW > N$, to prevent read-write conflicts,
   - $NW > N/2$, to prevent write-write conflicts.

Let $N = 10$, and assume that, for communication within the data store, the RMs are connected according to the following network topology.



Moreover, we define that a protocol for accessing a given replicated distributed data store with at least one correct node is *t-read-resilient* (*t-write-resilient*), when, in the

presence of at most t faulty nodes, all clients that contact a correct node can perform a read (write) operation.

(b) (0.5 point) Determine respectively the maximum *read-resilience* t_r and the maximum *write-resilience* t_w (maximum values of t) in case Gifford's quorum-based protocol uses $NR = 2$ and $NW = 9$. Beware that a faulty node is *incapable* of performing routing actions necessary to assist a correct node in assembling a quorum!

(c) (0.5 point) The same question for $NR = 3$ and $NW = 8$.

Answers to (b) and (c). The network consists of two fully connected subgraphs (cliques A and B) of size 5 that are connected through 3 node disjoint edges $E3 = (A3, B3)$, $E4 = (A4, B4)$, and $E5 = (A5, B5)$. Hence, a single or two arbitrary failing nodes will result in a network in which each correct RM can assemble a quorum of size 9 or 8 respectively, since the network remains fully connected. When three nodes fail in the worst case, namely when either all A- or all B-endpoints of the three edges $E3, E4, E5$ fail, the network partitions into two cliques, one of size 2 and the other of size 5. In that case, the RMs in the smaller clique can only assemble a quorum of size two. Finally, nodes $A1, B1, A2$, and $B2$ all have 4 neighbors, so failing of their neighbors results in an isolated RM that can only assemble a quorum of size 1. Thus, we find

- $t_r = 3$ and $t_w = 1$ in case $NR = 2$ and $NW = 9$
- $t_r = 2$ and $t_w = 2$ in case $NR = 3$ and $NW = 8$

5. Often, replication is applied in a system architecture to meet quality requirements.

(a) (1 point) Name four quality drivers for the usage of replication.

Answer. Four important drivers are: *availability, reliability, performance and scalability.*

(b) (1 point) For each of the named qualities indicate a concrete context in which it is used, and explicitly identify what is replicated.

Answer.

availability and reliability. Redundancy allows switching to a backup server in case the primary server fails. This increases both availability and reliability. In this case both the primary server and its state need to be replicated. Another reason to temporarily switch to a backup server is to do periodic maintenance or update servers. So maintainability, updatability and serviceability can also be seen as quality drivers.

performance. By replication of processing elements and distributing, but not necessarily replicating, the data to be processed, performance, in particular throughput, is increased by concurrency. As another example, sending a query for domain name resolution to the geographically nearest DNS root

server (replica), diminishes the round trip communication time and therefore response time.

scalability. In any P2P system, each peer, in its role as client, increases system load, but also, in its role as server, provides resources. Hence the balance between system load and system capacity (i.e., amount of resources) is maintained, and thus scalability is realized.

6. (2 points) Name at least two viewpoints (in Kruchten parlance a.k.a. library views). For each viewpoint you mention, indicate its principal stakeholders, their concerns and the architectural issues addressed by the viewpoint.

   Answer. See slide 38 of the introductory slide set for Kruchten views and slides 41-44 for viewpoints from the Rozanski-Woods viewpoint library.

7. Consider the Chord scheme for DHTs. Assume a 6-bit identifier space, and assume that the node set N is given by $id(N) = \{5, 16, 31, 38, 57, 63\}$.

   (a) (0.5 point) Give the finger table of node 31.

   Answer. For a 6-bit identifier space all finger tables have 6 entries. Table FT_{31} is given by $FT_{31}[i] = succ(31 + 2^{i-1})$, i.e.,

   $FT_{31}[1] = FT_{31}[2] = FT_{31}[3] = 38$, $FT_{31}[4] = FT_{31}[5] = 57$, and $FT_{31}[6] = 63$.

   (b) (1.0 point) Assume that node 5 is removed. This requires modification of the finger tables of the remaining nodes. Indicate for each of these nodes which entries of its finger table need to be modified and what their new values should be.

   Answer. Since the next node on the ring after node 5 is node 16, all finger table entries whose value is 5 have to be changed to 16. No other entries need to be modified. For node p and index i with $1 \leq i \leq 6$, entry $FT_p[i] = 5$ if and only if $63 < (p + 2^{i-1}) \mod 64 \leq 5$, or equivalently $63 - p < 2^{i-1} \leq 69 - p$. For nodes $16, 31, 38$, there is no index i that satisfies this criterion. For node 57 the criterion becomes $6 < 2^{i-1} \leq 12$ which has the single solution $i = 4$ and finally for node 63, the criterion becomes $0 < 2^{i-1} \leq 6$ which has solutions $1 \leq i \leq 3$.

   (c) (0.5 point) Indicate a key k for which resolution starting at node 31 requires a different number of steps before and after removal of node 5. Also give the resolution sequence before and after removal. You may assume that every node is aware of the identity of its predecessor.

   Answer. Assuming that node 31 is aware that node 16 is its predecessor, it "knows"that it is responsible for keys k with $16 < k \leq 31$ itself, and resolves those keys in zero steps. This does not change when node 5 is removed. For keys k with $31 < k \leq 63$, the resolution process will only consult finger table entries on nodes $31, 38$ or 57 that are unmodified. Hence, for these keys, the resolution stays the same. So, the only keys for which the number of resolution steps changes are the ones for which resolution changes from $31 \to 63 \to 5 \to 16$

before removal of node 5 to $31 \rightarrow 63 \rightarrow 16$ after removal. It follows that the key should be larger than 5 but smaller than 16. For any key in this range, the number of resolution steps decreases by one. Note that for $63 < k \leq 5$ the number of resolution steps stays the same, although resolution changes from $31 \rightarrow 63 \rightarrow 5$ to $31 \rightarrow 63 \rightarrow 16$.

8. Indicate for the following statements whether they are true or false. Motivate your answer with a short argument.

   (a) (0.75 point) UPnP supports third-party composition.
   Answer. True.
   This is one of the activities of a UPnP control point, e.g. in case of video streaming in which a video-source is connected to a displaying device.

   (b) (0.75 point) Sue has designed a distributed system that maintains network newsgroups. Her implementation guarantees *monotonic-writes* consistency. Thus, she guarantees that users of her system will never see a reaction that refers to an article which they have not seen before.
   Answer. False.
   Monotonic-writes consistency only guarantees maintaining the order of writes originating from the same process (user of the newsgroup system). In the above scenario, the author of the reaction is usually not the author of the original article, hence the system need not guarantee the order of the two writes. In particular, there may be replicas that have already stored the reaction, but not the original article. Reading from such a replica will be confusing. What is needed is *writes follow reads* consistency.

   (c) (0.75 point) Using message queues to deliver tasks to worker processes that can perform them, will guarantee that each task is executed precisely once.
   Answer. False.
   Message queues guarantee reliable delivery, i.e., each task will eventually be delivered at its destination queue (once). Removing the task from the queue and executing it, however, is the responsibility of the worker processes associated with the queue.

   (d) (0.75 point) In Round-Robin DNS, the authoritative name server answers successive queries to the same domain name by cycling through a set of IP-addresses. Therefore, Round-Robin DNS can be used for application level load balancing.
   Answer. True.
   The IP-addresses can stand for a cluster of machines that each host an instance of a web-service known under a single domain name. By returning these IP addresses in a cyclic fashion, the query load for this web-service is spread over multiple machines. Moreover, this is an application level solution, because DNS is an application level protocol, and also web-services are accessed by application level protocols. The approach requires stateless servers and client-side caching may reduce its effectiveness.

(e) (0.75 point) Flat names can be used for distributed resolution.
Answer. True.
A resolution mechanism based on DHTs, such as used by Chord, in general, relies on several peers to resolve a key. These keys are obtained by hashing and are flat names.

(f) (0.75 point) Thin clients minimize client-server communication.
Answer. False.
Thin clients pass user data to servers without or with a minimal amount of processing. As a consequence, erroneous data is usually only detected at the server side. This invokes additional communication in the form of an error message from the server to the client followed by corrected user input from the client to server. So both the number of communication actions and the total amount of data communicated would have been smaller when error detection would have been done at the client side by a thicker client.

(g) (0.75 point) A closure mechanism describes how to finalize the resolution process.
Answer. False.
A closure mechanism defines where and how to start the resolution process.

(h) (0.75 point) Software maintainability issues are best addressed in the deployment view.

Answer. False.
Software maintenance is made more easy by software modules with clear and limited responsibilities and few dependencies on other modules. This are typically issues that are addressed in the development view.