TECHNISCHE UNIVERSITEIT EINDHOVEN
Faculteit Wiskunde en Informatica

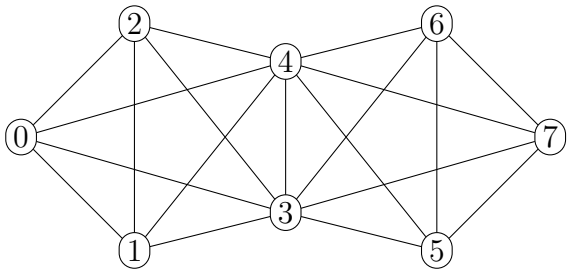
Examination Architecture of Distributed Systems (2IMN10),
on Thursday, January 25, 2018, from 18.00 to 21.00 hours.

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) Give the basic ingredients of an architectural description as specified by the ISO/IEC/IEEE 42010 standard. Illustrate your discussion with an appropriate UML model.

   Answer. For a UML-diagram see the architectural description (meta-)model (slide 31 of the introduction slide set). Of that diagram at least the boxes labelled System of Interest, Stakeholder, Concern, Viewpoint, View, Model, and Architectural Description should be present, and of course the relationships that hold between them.

2. 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. Consider a replicated distributed data store with 8 nodes connected in the following way:



   (a) (0.5 point) Determine respectively the maximum *read-resilience* and the maximum *write-resilience* (maximum values of t), in case this data store uses a quorum-based protocol with $R = 1$ and $W = 8$. Assume that in every operation a client communicates with only a single node of the data store, which is therefore responsible for assembling the quorum. Also beware that a faulty node is *incapable* of performing routing actions necessary to assist a correct node in assembling a quorum.

(b) (0.5 point) The same question for $R = 2$ and $W = 7$.

(c) (0.5 point) The same question for $R = 3$ and $W = 6$.

(d) (0.5 point) The same question for $R = 4$ and $W = 5$.

Answer. Due to its topology the data store has the following properties:

- In case there is a single faulty node, the data store remains connected. Hence, each correct node can assemble a quorum of maximum size 7, consisting of all correct nodes.
- In case there are two faulty nodes, and they happen to be nodes 3 and 4, the network partitions into two parts, each of which is a ring of three nodes. Hence, each correct node can assemble a quorum of at most size 3, consisting of the nodes on its ring. In all other cases, the data store again remains connected and each correct node can assemble a quorum of maximum size 6.
- In case there are three faulty nodes, containing nodes 3 and 4 and any other node, say 0, then nodes 1 and 2 can assemble a quorum of at most size 2, consisting of the pair of them. All other correct nodes can assemble a quorum of size 3.
- Finally, since there are nodes with 4 neighbors, four or more faulty nodes may cause a correct node to become isolated, in which case the maximum quorum size is 1.

From these observations it follows that

(a) In case $R = 1$ and $W = 8$, the protocol is 7-*read-resilient* and 0-*write-resilient*.

(b) In case $R = 2$ and $W = 7$, the protocol is 3-*read-resilient* and 1-*write-resilient*.

(c) In case $R = 3$ and $W = 6$, the protocol is 2-*read-resilient* and 1-*write-resilient*.

(d) In case $R = 4$ and $W = 5$, the protocol is 1-*read-resilient* and 1-*write-resilient*.

3. Both Amdahl's law and Gustafson's law address scalability by looking at the speedup that can be achieved by the replication of processing elements.

   (a) (0.5 point) Give a definition of this scalability metric (i.e., speed-up).

   Answer. The speed-up S is given by the formula

   $$S(P, N) = \frac{T(1, N)}{T(P, N)}$$

   where $T(P, N)$ is the execution time a problem of size N using P processing elements.

(b) (0.5) Explain why, in general, using P processing elements does not result in a speed-up of size P.

Answer. In general, not all parts of a computation benefit from the presence of multiple PEs. For instance, for a loop the individual iterations of the body can be divided over the PES, but each of them still needs its own copy of the loop control. Moreover, there are additional costs, because the PEs need to communicate to realize the entire computation.

(c) (1.0 point) Explain the difference in scalability perspective offered by the two laws. In particular, indicate the underlying assumption that makes Gustafson's law more optimistic, i.e., explain why it considers more systems scalable than Amdahl's law.

Answer. Amdahl's law considers the situation of solving a *fixed-size* instance of a problem with an increasing amount of PEs. In this case, the ratio between the size of sequential part of the instance and the part of the instance amenable to parallelization places an upper bound on the attainable speed-up. Gustafson's law, on the other hand, considers the case where the *instance size is increased* with the number of PEs, and states that speed-up proportional to the number of PEs is obtained as long as the instance size is chosen such that the ratio of the two parts is kept constant. As long as this does not lead to overwhelming instance sizes, scalability is maintained.

4. 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 (3^{rd} ed.) Figures 5.17 and 5.18 or TvS (2^{nd} 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 (3^{rd} ed.) Fig. 5.20, or TvS (2^{nd} ed.) Fig. 5.18.

5. Consider the Chord scheme for DHTs. Assume a 6-bit identifier space, and assume that the node set N is given by $id(N) = \{6, 19, 35, 38, 54, 61\}$.

(a) (0.5 point) Give the finger table of node 35.

Answer. For a 6-bit identifier space all finger tables have 6 entries. Table FT_{35} is given by:

$FT_{35}[1] = FT_{35}[2] = 38$, $FT_{35}[3] = FT_{35}[4] = FT_{35}[5] = 54$, and $FT_{35}[6] = 6$.

(b) (0.5 point) Describe in detail how key 63 is resolved from node 19. Assume that all nodes are aware of the identity of their predecessor.

Answer. To resolve key k, a node n first determines whether it is responsible for that key, i.e., it checks whether $pred(n) < k \leq n$. If this is not the case, it looks up a node p in its finger table to which it forwards the key for resolution. Node p is the largest node found in FT_n that does not *overshoot* node $succ(k)$ that is responsible for k, i.e., $p = FT_n[1]$ if $n < k < FT_n[1]$, or $p = \max\{FT_n[i] \mid 1 \leq i \leq 6 \wedge FT_n[i] \leq k\}$ otherwise. Here max means *clockwise furthest from n*. Note that, in general, $succ(k)$ is not known to node n, but $k \leq succ(k)$. Thus, to resolve key 63 from node 19, we obtain the path $19 \rightarrow 54 \rightarrow 61 \rightarrow 6 = succ(63)$.

(c) (1.0 point) Assume that node 8 is added to the set of nodes N. Describe in detail all modifications to the DHT required by this addition, i.e., indicate all modified or new entries of finger tables and predecessor values, and all keys whose associated resources, if present, have to be reallocated.

Answer. Node 8 is inserted between nodes 6 and 19. Hence, $pred(8) = 6$ and $pred(19) = 8$. Moreover, all resources associated with keys 7 and 8 are transferred from node 19 to node 8. The finger table of the new node becomes $FT_8[1] = FT_8[2] = FT_8[3] = FT_8[4] = 19$, $FT_8[5] = 35$, and $FT_8[6] = 54$. Finally, for $1 \leq i \leq 6$ and $p \in N$, entry $FT_p[i]$ becomes 8 when $6 < p + 2^{i-1} \leq 8$, where computation is done modulo 64, if necessary. For node 6, this means $0 < 2^{i-1} \leq 2$, so $FT_6[1] = FT_6[2] = 8$. For node 61 this means $9 < 2^{i-1} \leq 11$, so its finger table remains unaltered. For node 54, this means $16 < 2^{i-1} \leq 18$, so also its finger table remains unaltered. For the other nodes, their forward distance to node 6 along the ring is at least $32 = 2^5$. Hence, also their finger tables remain unaltered.

6. (2 points) Describe the service-oriented architectural style (SOA) 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 13 of the slide set on architectural styles. A weak point would be security, e.g. access control, which is easier to realize in a centralized architecture.

7. There exist various primary-based protocols for (data) consistency.

(a) (1 point) Describe the local-read, primary-write variant (aka as primary-backup protocol). Illustrate your discussion with a diagram.

Answer. See section Remote-Write protocols, vST (3rd ed.) pp 399-400, or TvS (2nd ed.), pp. 308-309. Your answer must discuss the order in which

the communications between the primary and backup servers take place. In particular, it should be clear that a write operation blocks until replicas have updated their value.

(b) (0.5 point) What type of consistency is obtained with this protocol?

Answer. Sequential consistency. Operations originating from the same client, which arrive at the same replica, are executed in the order issued by the client.

(c) (0.5 point) In general, a read operation on the store need not provide the most recent version of a data object. For the primary-backup protocol, how many versions can a read operation lack behind in the worst case?

Answer. The object read from a local replica can be at most 1 version behind, because the write operations are sequentialized by the primary. Hence, all replicas are updated before the next write can start.

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

(a) (0.75 point) Availability is a quality attribute that is measured by means of the MTTF (mean time to failure) metric.

Answer. False.
Availability is defined as the probability, at any moment in time, to find a service (device) ready for correct response. Hence, to measure availability, knowledge both about the time when a service is functioning correctly and about the time when the service fails is needed. The first is captured by the MTTF-metric, the latter by the MTTR-(mean time to repair)metric, and availability is given in terms of both metrics by the quotient $\frac{MTTF}{MTTF+MTTR}$.

(b) (0.75 point) The fact that DNS uses anycasting to contact its root servers contributes to its scalability.

Answer. True.
Anycasting offers several advantages. One of these is that the network can route a request to the nearest root server. Thus, by appropriate placement of these servers, load balancing can be achieved, which contributes to the scalability of the system. It also decreases response time for clients and reduces network traffic for network providers.

(c) (0.75 point) Gossiping can be used to obtain eventually-consistent replicas.

Answer. True.
Provided the selection of gossip partners is such that eventually each peer has been engaged in a gossip session to exchange state information with all of its neighbors. Random selection of gossip partners such that each neighbor has a positive probability to be selected will do so.

(d) (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.

(e) (0.75 point) Top-down system design is an important aspect of component-based software engineering.

Answer. False.
In component-based software engineering, systems and applications are, whenever possible, built from predefined components that interoperate according to contractually specified interfaces. This is a bottom-up approach.

(f) (0.75 point) Marshalling is a tactic to increase interoperability.

Answer. True.
Marshalling relies on independent formats for transferring data between two parties. Thus, it facilitates communication between parties (components) developed using different programming languages, and deployed on different operating systems and hardware platforms. Each party, instead of being aware of a lot of possible combinations for other parties, only needs to understand the marshalling formats.

(g) (0.75 point) A collaboration diagram that describes the interactions between the various architectural elements of an RPC-architecture, such as the user program, stubs, OS primitives, etc., belongs primarily to the logical view.

Answer. False.
The logical view is primarily concerned with externally (to the user) visible behavior. It is the process view that primarily defines the interactions between all kinds of architectural elements, such as for instance invocation of stubs, which are transparent to the user.

(h) (0.75 point) A reverse proxy is an architectural element that is useful to address security concerns.

Answer. True.
For instance, in a demilitarized zone (DMZ-)architecture, requests from clients to origin servers are rerouted via reverse proxies, which are the only entities that may access these origin servers. Hence, reverse proxies can be used to implement all kinds of access control policies, thus enhancing the security of a service provider's system.