TECHNISCHE UNIVERSITEIT EINDHOVEN
Faculteit Wiskunde en Informatica

Examination Architecture of Distributed Systems (2IMN10),
on Thursday, January 30, 2020, 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 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.

   Answer. See slide 23 of the slide set on architectural styles.

2. For bringing a replica of a data store up to date, a pushed-based or a pull-based protocol can be used.

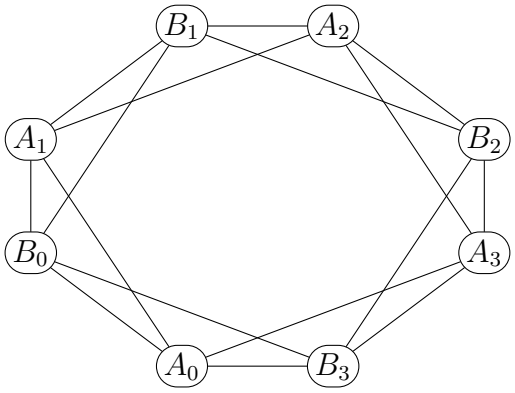
   (a) (1.5 point) Name three aspects for which these protocols perform differently and indicate these differences.

   Answer. See slide 5 of the slideset on architectural issues of replication or vST pp 390–392.

   (b) (0.5 point) Explain the use of leases to obtain a protocol that is a mix of push and pull. Indicate the benefits of such an approach.

   Answer. Leases can be used with client-initiated replicas. When a client does not possess, or has a stale copy of, an object, it can either pull the object from the server or request a lease. When the server obtains a new value for an object, it pushes that value to those clients that have an outstanding lease for that object. In this way, clients with a lease will always have the latest value of an object. By this protocol, clients can reduce response times by requesting leases and servers can minimize the size of the state they have to store by keeping the number of leases limited. Thus this protocol can adapt at run-time to changing client and server needs and to changing read-over-write ratios for objects.

3. Consider a replicated distributed data store with 8 replicas each managed by an individual replica manager (RM). For communication within the data store, the RMs are connected according to the following network topology.



Clients access the data store via an RM. Each client always contacts 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.

(b) (0.5 point) 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. Determine both the maximum read-resilience and the maximum write-resilience (maximum values of t), in case this data store uses a quorum-based protocol with $NR = 1$ and $NW = 8$. Assume that a faulty node is also incapable of performing routing actions necessary to assist a correct node in assembling a quorum.

(c) (0.5 point) The same question for $NR = 2$ and $NW = 7$.

(d) (0.5 point) The same question for $NR = 3$ and $NW = 6$.

(e) (0.5 point) The same question for $NR = 4$ and $NW = 5$.

Answer. To answer these questions, we observe that

i. If there are at least t faulty nodes, any correct node can assemble a quorum of at most $8-t$ nodes, where the maximum is only obtained when the network consisting of the remaining correct nodes is not partitioned.

ii. If there are at most 3 faulty nodes, the remaining network of correct nodes stays connected. To show this, consider two cases: all faulty nodes are on a single cycle (A or B), or they are divided over the two cycles. In case all faulty nodes are on the same cycle, say A, the remaining correct A-node has two neighbors on the B-cycle, hence the remaining network is connected. In case the faulty nodes are divided over the two cycles, assume w.l.o.g. that there is 1 faulty A-node and 2 faulty B-nodes. Since, originally, the A-nodes formed a cycle, the remaining correct A-nodes form a chain. Since each B-node originally had two A-neighbors, the remaining correct B-nodes have at least one neighbor on the A-chain. Hence the remaining network is connected.

iii. If there are at least 4 faulty nodes, it becomes possible to isolate a correct node, because originally each node has 4 neighbors.

From these observations it follows that for $t \leq 3$ each node can assemble a quorum of $8 - t$, and for $t \geq 4$ the faulty nodes may be such that there exists a correct node that can assemble a quorum of only 1. From this we conclude that for

$NR = 1$, $NW = 8$ the protocol is at most 7-read-resilient and 0-write-resilient,
$NR = 2$, $NW = 7$ the protocol is at most 3-read-resilient and 1-write-resilient,
$NR = 3$, $NW = 6$ the protocol is at most 3-read-resilient and 2-write-resilient,
$NR = 4$, $NW = 5$ the protocol is at most 3-read-resilient and 3-write-resilient.

4. (1 point) Informally, any consensus protocol decides on a value v based on proposals made by each of its participants. More formally, a consensus protocol needs to realize the following four properties: termination, validity, integrity, and agreement. Give the definition of each of these properties.

Answer.

Termination Every correct process eventually decides some value.

Validity If a process decides v, then v was proposed by some process.

Integrity No process decides twice.

Agreement No two correct processes decide differently.

5. (2 points) Explain how, for RPC-based applications, the IDL-compiler of an RPC framework (e.g. DCE RPC) is used to develop client and server binaries from client and server code. For the sake of explanation, you may assume that both client and server code are given in C.

Answer. The client binary is generated from client C-code that contains RPCs.

Therefore, in order to compile the client code, a (set of) header (#include) file containing the interfaces of these remote procedures is needed. In an RPC-framework, the interfaces of RPs are described in a separate interface definition language (IDL) that is independent from both the language of the client and the server code (in this case just the single language C) and the IDL compiler is used to generate the C-specific header file(s) from this description. Given these header files, the client code can be translated into client object files using a standard C-compiler. At runtime, upon calling an RP, a stub is invoked that performs marshalling the data involved in the RPCs into an external data representation (XDR), which is specified as part of the IDL. This piece of code is identical for all client applications, since it only depends on the interfaces of the RPCs and needs to be generated only once. To that end, the IDL-compiler is used to generate client stub C-code, from the IDL-file, which subsequently is translated into a client stub object file using a standard C-compiler. Finally, the client binary is obtained by linking the client object file, the client stub object file, and possible runtime library routines into a single binary using a standard linker. To obtain the server binary an analoguous process is performed. Of course, the server stub should take care of demarshalling, but its code can likewise be generated by the IDL-compiler from the IDL- file. For a graphical illustration of this development process, see slide 39 of the slideset on interaction styles.

6. Consider the Chord scheme for DHTs. Assume a 6-bit identifier space, and assume that the node set N is given by $id(N) = \{1, 7, 18, 23, 36, 55\}$.

   (a) (0.5 point) Give the finger tables for all nodes.

   Answer. For a 6-bit identifier space, all finger tables have 6 entries. Recall that $FT_p[i] = succ(p + 2^{i-1})$, for $1 \leq i \leq 6$. Hence the tables are given by:

$FT_p[i]$		p					
		1	7	18	23	36	55
i	1	7	18	23	36	55	1
	2	7	18	23	36	55	1
	3	7	18	23	36	55	1
	4	18	18	36	36	55	1
	5	18	23	36	55	55	7
	6	36	55	55	55	7	23

   (b) Next, consider the situation in which node set N is extended with node 39.

   - (0.5 point) For all $p \in N$, indicate which entries of FT_p are changed and what the new values are.
   - (0.5 point) Give a key k, $0 \leq k < 64$ and a node $p \in N$ for which the addition of node 39 increases the number of steps necessary for the resolution of key k starting in node p.

- (0.5 point) Give a key k, $0 \leq k < 64$ and a node $p \in N$ for which the addition of node 39 reduces the number of steps necessary for the resolution of key k starting in node p.

Answers. The only way an entry $FT_p[i]$ can change due to the addition of node 39 is that its value becomes 39. For this to happen, it must be the case that $36 < (p+2^{i-1}) \mod 64 \leq 39$. The instances for which this condition holds are $(p,i) \in \{(7,6),(23,5),(36,1),(36,2)\}$.

Since $FT_7[6]$ has changed from 55 to 39, it follows that key $k = 55$ is now resolved in 2 steps from node $p = 7$ ($7 \rightarrow 39 \rightarrow 55$), whereas it used to be 1 step. Note that it is not necessary to compute FT_{39} to see this, because 55 is the successor of 39 and is therefore reachable in 1 step from 39.

As a consequence of the same change, it follows that key $k = 39$ is now resolved in a single step from node $p = 7$, whereas it used to be resolved in 3 steps ($7 \rightarrow 23 \rightarrow 36 \rightarrow 55$).

7. 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 (PEs).

   (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 time penalties, because the PEs need to communicate and synchronize 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.

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

   (a) (0.75 point) Anycasting is only used in local area networks.
   Answer. False.
   Anycasting is, e.g., used to contact a DNS root server. It is implemented in the network layer by the Border Gateway Protocol that routes traffic *between* networks.

   (b) (0.75 point) Absence of client-oriented consistency breaks replication transparency.
   Answer. True.
   Replication transparency means that users of a resource or service cannot detect that it is replicated. In the absence of client-oriented consistency, users (clients) of a replicated data store may observe an unexpected state. Assuming that the data store operates functionally correct and other clients do not intentionally provide misinformation, such as for instance replying to non-existing messages, the most plausible explanation for the observed state then is that they currently interact with a replica different from the one with which they interacted in the past.

   (c) (0.75 point) A scalable solution for partitioning a key-value store over a number of identical nodes should allow multiple partitions per node.

   Answer. True.
   Each partition covers a range of keys and scalable solutions should keep the number of queries per range roughly equal (given that nodes are identical). Hot key ranges can be eliminated by hashing the keys with a uniform hash-function, but to avoid hot keys rebalancing is necessary. By allowing multiple partitions per node, it becomes possible to balance the query load by splitting, merging and redistributing partitions over the nodes, instead of redistributing the keys over partitions which requires modification of the hash function and thereby redistributing all key-value pairs.

   (d) (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.

   (e) (0.75 point) Buffering is a QoS-tactic for stream-oriented communication that reduces jitter.

Answer. True.
By first capturing packets transmitted at constant rate, but arriving with fluctuating inter-arrival times, i.e. exhibiting jitter, due to fluctuating end-to-end delay in a buffer and subsequently reading from this buffer at a constant rate equal to the transmission rate, jitter can to a large extend be eliminated.

(f) (0.75 point) In an architectural description each model SHOULD identify its stakeholders.
Answer. False.
In an architectural description it are the views or rather the viewpoints to which the views conform that should identify the stakeholders. Of course, it does no harm if this information is reiterated in the models, but as long as it is clear to which views the model belongs its stakeholders can be retrieved.

(g) (0.75 point) Peer-to-peer architectures are not reliable.
Answer. False.
Reliability is concerned with the continuity of correct service (expressed as the expected time to failure). Since all peers provide the same functionality, the service is only discontinued when the last peer leaves the community. Since this is a rare situation, peer-to-peer architectures are extremely reliable. Of course, service degradation and subsequent restoration may occur frequently.

(h) (0.75 point) Triple modular redundancy is a *passive* redundancy tactic.
Answer. False.
In triple modular redundance, each service (computation) is performed by three independent components, whereafter the outcome is determined by majority vote. If the outcome is input to a subsequent computation also the voting elements are replicated three times. Since replicas compute in parallel this is a form of active redundancy.