A

This exam consists of 20 multiple choice questions. In every case only one of the answers is correct. You should answer in a separate sheet. If correctly answered, they contribute 0,5 points to the final grade. If incorrectly answered, the contribution is negative: -0.167. So, think carefully your answers.

THEORY

1. Containers are useful for deploying distributed service components because...

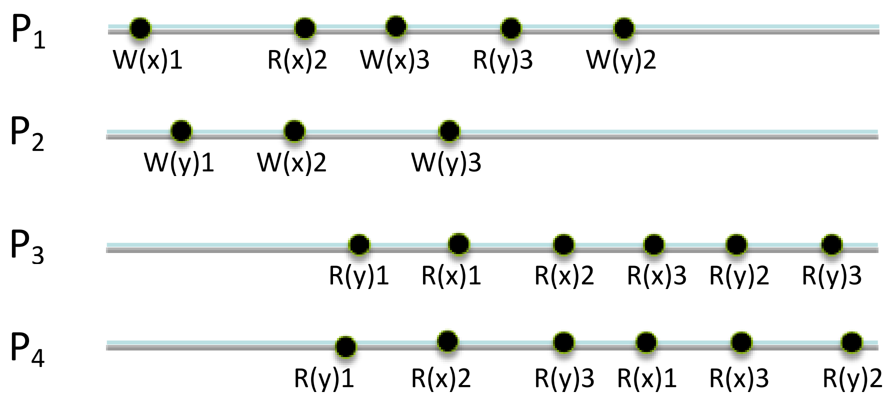
a	They allow that every other component see the internals of the installed component image. False. The internals of the installed components in an image remain hidden to external agents.
b	They facilitate dependency resolution (or injection) and provide commands for dealing with several component life cycle actions. True. This is the feature of containers that makes them convenient for deploying service components.
c	They automate many security-related tasks and are able to implement all security policies. False. We haven't considered any security aspect in our description of containers.
d	They ensure inter-component consistency in case of network partitions. False. Containers are unable to guarantee inter-component communications when the network that connects them is broken. Indeed, nothing can be done (by containers or by any other mechanisms) in that case.

2. In the deployment model supported by Windows Azure...

a	Component instances may be placed in independent failure domains. True. Windows Azure distinguishes independent failure domains at deployment time, allowing that component instances were placed in different domains in order to improve their availability in case of failures.
b	A precise sequencing plan may be settled in the initial component deployment (i.e., when the components are initially installed). False. No sequenced deployment plan may be stated using the current Windows Azure tools. The components themselves should take care of the existing dependences and of the sequence of actions to be performed in the deployment stage.
c	It provides the functionality being needed for upgrading stateful components. False. The upgrading mechanisms in that platform are simple and focused on stateless components.
d	There is no component monitoring once those components have been deployed. False. Component monitoring is currently used in Windows Azure for detecting component failures.

A

3. Considering the following execution...



We can say that the strongest consistency model it complies with is...

a	Sequential. False. Sequential consistency demands, among other things, that all processes see the same sequence of events. In this case this is not respected by all processes. For instance, P3 and P4 have seen the R(y)3 and R(y)2 events in the opposite order.
b	Causal. False. P1 has read value 3 on "y" before writing value 2 on that variable. According to the causal memory model, this compels every other reader to see value 3 before value 2, but P3 does not respect that constraint.
c	Cache. False. Cache consistency requires that every process sees the same sequence of values in each variable, considering each variable independently on the others. However, P3 and P4 haven't seen the same sequence of values on variable "y". Note that P3 has seen 1, 2, 3 and P4 has seen 1, 3, 2.
d	FIFO. True. FIFO consistency demands that what has been written by a process is seen in writing order by the remaining processes. P1 wrote x=1, x=3, y=2 and this has been respected by readers P3 and P4. P2 wrote y=1, x=2, y=3 and this order has been respected by readers P1, P3 and P4. Since P2 did not read any value, the resulting system is FIFO.

4. The "primary partition" model for dealing with network partitions...

a	...allows that every process goes on once a network partition has occurred. False. This may be tolerated in the "partitionable" model but not in the "primary partition" one.
b	...ensures that all processes in the primary partition accept new client requests maintaining a consistency as strong as required in that subgroup. True. To this end, all processes placed in the other subgroups are blocked and cannot continue.
c	...always guarantees that a primary subgroup or primary partition exists when a network partition occurs. False. Unfortunately, this cannot be guaranteed by this model. It depends on the connectivity among nodes once the partition has arisen. In some cases, the greatest subgroup does not contain at least a half of the system nodes.
d	...clearly violates the constraints of the CAP theorem since once a network partition arises the system is still kept consistent and highly available. False. Availability is not maintained in minor subgroups. Therefore, CAP is respected.

A

5. Considering the CAP theorem, we can say that nobody may implement a distributed system with replicated components that ensures...

a	...eventual consistency and that assumes a partitionable model when a network partition arises. False. This is tolerated by the CAP theorem. This sacrifices strong consistency.
b	...eventual consistency and that assumes a primary partition model when a network partition arises. False. This is tolerated by the CAP theorem. In this case, we are renouncing to both strong consistency and high availability.
c	...sequential consistency and that assumes a partitionable model when a network partition arises. True. In a partitionable model every network subgroup may continue with its execution. As a result, all system nodes remain available. When the system network is partitioned this would imply that strong (i.e., sequential) consistency would have been maintained, with high availability and network partition tolerance. This is impossible since when every subgroup continues in case of network partitions there is no way to propagate all state updates to the remaining subgroups. As a result of this, sequential consistency will not be guaranteed in a system-wide scope.
d	...sequential consistency and that assumes a primary partition model when a network partition arises. False. This is tolerated by the CAP theorem. This sacrifices service availability in the minor subgroups.

6. Transactions in persistent datastores...

a	...ensure atomicity and isolation for the sequence of updating actions contained in them. True. Regular transactions guarantee atomicity and isolation (plus semantic consistency and durability) for their encompassed updating actions.
b	...are fully supported by NoSQL datastores and highly enhance their scalability. False. They are not supported entirely in NoSQL datastores. Besides this, the embedded concurrency-control support in transactions prevents them from scaling in the regular case.
c	...are used for ensuring both availability and replica consistency in partitionable distributed systems. False. Distributed transactions cannot overcome a network partition. They demand network connectivity among the agents that have participated in a distributed transaction.
d	...may only be provided in scalable systems when a fast consistency model is being used. False. In the regular case, ACID transactions are assuming a sequential consistency model among the agents that share a distributed transaction. Since the sequential model is not fast, this statement cannot be true.

7. Regarding some horizontal scalability mechanisms...

a	Decentralised algorithms are always more appropriate than centralised algorithms, independently on the characteristics of the task to be solved.

	False. The choice of the algorithm type depends on the task to be solved. For instance, if a problem demands that all participating processes complete different agreements in multiple execution stages, it will be better to concentrate the agreement efforts in the election of a coordinator (i.e., central) process that will subsequently impose its criterion in all those future agreement-related stages. In this way, the coordination required by agreement stages is limited to an initial step of the algorithm (coordinator election) and the amount of messages needed in subsequent stages will be strongly reduced. On the other hand, when the algorithm can be decomposed into multiple independent tasks with little or no coordination, each of those tasks may be executed by an independent process using a decentralised algorithm.
b	Data distribution criteria should be known by every process, making possible that every process was responsible of a subset of data. True. Data distribution is convenient for achieving good scalability following a horizontal approach. Distribution criteria should be shared (i.e., known) by all participating processes in order to reduce their coordination needs.
c	Replication is always welcome since it doesn't introduce any problem in the management of read and write accesses to data. False. Replication may introduce problems in case of write accesses since they need to be applied in every replica and this might introduce important delays in some cases; e.g., in case of operations that modify a lot of state.
d	The usage of caches is optimal since it guarantees automatically the consistency among cache data and server data. False. Unfortunately, client caches are unable to guarantee their consistency in an automatic way with the state being managed by servers.

8. A service is elastic when it is...

a	...scalable and able to dynamically adapt the amount of assigned resources to its current workload. True. That is definition of an elastic service.
b	...reliable, available and safe. False. Reliability, availability and safety are different aspects of dependability. They are not directly related with elasticity.
c	...network-partition tolerant, available and consistent. False. These are the three dimensions of the CAP theorem.
d	...atomic, consistent, isolated and durable. False. These are the four properties of transactions in relational database management systems.

9. Regarding the risks that might compromise security in a distributed system...

a	An attack is the set of actions that have been applied when a threat has been exercised. True. This is the definition of attack that we have seen in Unit 8.

b	A threat is an inherent weakness in a process or in a communication channel. False. That is the definition of vulnerability. Vulnerabilities and attacks are not the same thing.
c	All vulnerabilities may be avoided using isolation strategies (e.g., virtual machines, sandboxes, isolated networks...). False. Isolation strategies are a coarse defensive strategy but, unfortunately, independently on their granularity, there is no single perfect defensive strategy. For instance, with isolation strategies we are trying to set a boundary that prevents all the agents that are external to our system from seeing our isolated components. Unfortunately, that isolation cannot be complete since the deployed elements, sooner or later, will need to interact with any external agent and that interaction will be vulnerable; i.e., it is almost impossible to set that boundary.
d	All vulnerabilities may be avoided using exclusion strategies (e.g., firewalls, passwords...). False. Exclusion strategies are a medium-grained defensive strategy but, unfortunately, independently on their granularity, there is no single perfect defensive strategy. For instance, with exclusion strategies we are trying to keep all potential threats out of our system. Unfortunately, that exclusion cannot be complete since the mechanisms being used for implementing exclusion aren't perfect; e.g., passwords are an exclusion approach but not all the passwords being used by enterprise employees are strong enough. Sooner or later, some of them may be known by other people.

10. Security mechanisms are used for...

a	...specifying a set of security rules. False. What specifies a set of security rules is a policy, not a mechanism.
b	...ensuring or assessing the correctness of a security system. False. The tool that ensures or assesses the correctness of a security system is "assurance".
c	...stating who (i.e., which principals) may apply what actions on which objects. False. Again, this is another (more refined) definition of what is a security policy.
d	...implementing security policies in a system. True. That is the goal of security mechanisms: to implement policies.

SEMINARS

11. Assuming that we are using an Ubuntu Linux as our local operating system where Docker has been successfully installed then, in order to run the "node" interpreter in a container with the latest Fedora Linux distribution (whose name is "fedora" in the Docker repositories), we should use:

a	This command line: **docker run –i –t fedora node** False. The "node" interpreter is not installed in the default "fedora" Docker image.

b	Nothing can be done, since we cannot run a Fedora image on an Ubuntu host. False. A Fedora image can be run without problems using an Ubuntu host.
[✓ c]	We must create an extended Fedora image with the "nodejs" package installed on it (e.g., "fnode") and use this command line: **docker run –i –t fnode node** True. If the image being used has the "node" interpreter installed in it, the syntax and arguments of the needing docker command are those mentioned in this sentence.
d	Nothing can be done with Docker. Instead of Docker, we need VirtualBox to run the virtual images mentioned in this question. False. Docker can manage these requirements. We saw several examples in Seminar 4.

12. On Docker images...

[✓ a]	New images consist of multiple auFS "layers" that have been added on top of other base Docker images. True. Each time we add or remove any file from a Docker image a new auFS layer is placed on top of the layers already contained in that base image.
b	Docker images may be run in non-Docker containers; e.g., in VirtualBox. False. Docker images need to be run in Docker containers using a Docker daemon to this end.
c	Docker images may be run in all hosts, independently on their local operating system or architecture; e.g., in a PC host running Windows 8. False. Unfortunately, Docker daemons need to run on top of any Linux system, but not (yet) in other operating systems.
d	The same Docker container may be run in multiple Docker images at the same time. False. The reverse is true: "The same Docker image may be run in multiple Docker containers".

13. These rules must be respected in order to implement a replication protocol that provides a fast consistency model among the replicas of a given service.

a	Write actions are never propagated to the other replicas. False. Write actions, sooner or later, must be propagated to other replicas. Otherwise, those replicas will remain inconsistent.
b	Write actions aren't completed in the writer process A until the sequencer propagates to A the written value. False. External processes must not participate in any action management in order to implement a fast consistency model.
c	All update operations on variables should be expressed as commutative operations (instead of the regular assignments being used in other protocols). False. Commutativity was an optional aspect to be considered for implementing eventual consistency, but it is not needed for defining fast consistency models.
[✓ d]	Every action (i.e., reads and writes) should be locally considered complete without needing any message propagation or acknowledgement. True. That is the condition to be respected by fast consistency models.

14. In order to implement eventual consistency, we should take care of...

a	Using FIFO channels between all system processes.

	False. FIFO communication was not a requirement for implementing eventual consistency.
b	Ensuring that all written values are delivered to all participating processes and that they may be applied in any order by every receiver. True. All updating actions must reach every system process (although that propagation may be lazy) in order to allow those processes to converge. State convergence was the main characteristic of eventual consistency. If those updating actions are commutative, then there is a total freedom in the order they may be received and applied.
c	Using a sequencer for ensuring an agreement on the total ordering of all write events. False. In the TSR seminars, we have only used sequencer processes for implementing the sequential consistency model. Sequencers are not required for implementing eventual consistency.
d	Using synchronous channels in all memory-related communications. False. Synchronous channels are not needed for implementing eventual consistency.

15. Seminar 5 has shown three possible implementations of memory consistency models. In those implementations...

a	None of them was able to support causal consistency. False. The second implementation (based on a sequencer) was able to support the sequential consistency model. Since sequential consistency implies causal consistency, then that implementation was supporting causal consistency. Therefore, one of them respected (i.e., implemented) causal consistency.
b	Sequencer-based protocols were implemented using the "cluster" module. False. The sequencer implementation discussed in Seminar 5 was not based on the "cluster" node.js module.
c	When a new process P is added to an eventually-consistent system some synchronisation is needed to accept P and transfer to it a copy of the shared variables. True. In the last question related to the third implementation (that, indeed, supported eventual consistency) we were asking for a protocol to transfer the initial state to an emerging new replica. In order to achieve this, we need to temporarily stop the activity in the remaining replicas.
d	None of them implements a fast consistency model. False. The most relaxed consistency models (e.g., FIFO) are fast. The first implementation was implementing FIFO consistency and it was using a protocol that complied with all fast consistency conditions to this end.

16. The "cluster" module in Node.js ...

a	...uses by default a ROUTER-DEALER communication pattern. False. ROUTER-DEALER is a ZeroMQ communication pattern. The "cluster" module need not require the "zmq" module by default. Therefore, that communication pattern is not used by the "cluster" module.

[✓ b]	...balances the incoming workload among all worker processes created with its support. True. This is one of its goals and it is already implemented by default in that module.
c	...provides support for managing distributed services running on clusters of workstations. False. In spite of its name, the “cluster” module is intended for supporting a set of worker processes that are all run in the same computer.
d	...inherently supports distributed services with primary-backup replication. False. The “cluster” module does not support the primary-backup replication model in a direct way. Indeed, there is no state transference between master and workers. All worker agents are active processes. They do not behave as backup replicas of any other agent.

17. When MongoDB is deployed onto multiple servers, it needs 3 *configuration servers* (CS) because...

[✓ a]	The information maintained by the CS is critical and needs strong consistency. Thus, the failure of 1 configuration server may be overcome. True. The information being maintained by these agents is critical. They need to be 3 replicas in order to overcome partition failures following the primary partition model (or process failures, overcoming a single failure).
b	MongoDB assumes the “stop” failure model and in this way it can overcome that all these servers fail simultaneously. False. It does not assume the “stop” failure model but, even in the hypothetical case of assuming it, it could not tolerate that all replicas fail at once. At least one of them needs to be alive in the “stop” model.
c	Every client request should be filtered by these servers and at least three of them are needed for appropriately balancing their workload. False. Configuration servers do not filter client requests. Configuration servers are contacted by “mongos” agents when those agents are not able to find with their cached configuration data where (i.e., in which “mongod” server in case of using database sharding) a given document ID is placed.
d	Configuration servers are replicas of the *mongos* agents. False. There are three different types of servers in MongoDB: configuration server, mongod and mongos. Therefore, mongos agents and configuration servers are not the same kind of agent and one of them cannot be the replica of the other.

18. MongoDB is, among other things, ...

a	...a node.js database server that internally uses the “cluster” module for enhancing its scalability. False. MongoDB is an example of NoSQL datastore but it is not implemented in node.js nor uses its “cluster” module.

b	...a relational database management system. False. It is not relational.
c	...a database management system that manages its replicas ensuring strict consistency. False. It does not respect strict consistency among its replicas. The strict consistency model is too strong for being scalable and scalability is one of the goals of MongoDB.
d	...a distributed service that may follow the primary-backup replication model. True. When replication is used in MongoDB, we may replace a "mongod" agent with a replica set. MongoDB replica sets follow the primary-backup replication model.

19. Some preventive actions for managing vulnerabilities are...

a	At the software development stage: consider that all users are trustworthy and reliable, providing always a correct and valid input. False. Unfortunately, not all users will be able to behave in that so benign way. We should forecast which could be their worst behaviour and be ready to manage it.
b	Regarding human staff: monitor their behaviour in order to prevent internal sabotages. True. One of the recommendations mentioned in Seminar 8 was that. Note that security threats may be internal to the company and some internal threats may be prevented in this way.
c	Regarding system administration: provide as many public services as possible. False. The amount of public services should be as low as possible since each available service provides a potential door for incoming attacks. Therefore, the lower, the better.
d	Regarding system administration: ignore the security alert reports since most of them are confusing and may introduce new vulnerabilities. False. Security alerts should be considered. If, in any case, any of them seems to be confusing, we should look for additional information in order to understand it, but its fixes should be applied as soon as possible.

20. In order to avoid security vulnerabilities, a company should compel their employees to...

a	Choose very strong passwords, keeping them in a spreadsheet on their smartphone in order to guarantee that passwords will never be lost. False. Passwords should not be kept in that way.
b	Take care of the input they are providing to the computing applications and its expected output, reporting to the system administrators any application malfunction. True. Any application misbehaviour should be reported as soon as possible.
c	Take (part of) their work home, providing as many on-line services as possible to complete those pending tasks since this will improve employee's performance. False. Take a look at the justification given in part "c" of question 19.
d	Ignore everything about social engineering techniques since the more we know about them, the more we'll be tempted to use them against our own company. False. Social techniques should be well-known in order to avoid that attackers use them against our company or against our employees. In this way, the company staff will take care of their access procedures in order to avoid those vulnerabilities.