Architecture of Information Systems (192320111): First exam 2012/2013

July 3, 2013 13:45-15:45h

Please pay attention to the following:

This exam has to be completed in 2 hours.

It is **NOT** allowed to use the book or any other material.

You can answer in either Dutch or English.

The exam consists of 7 questions.

Distribution of points:

10 points for showing up;

Other questions as indicated.

Question 1 (10 points)

We have seen several definitions of the notion of 'architecture' in the course, e.g., the one present in the book:

> "IT architecture is a solution to the problem 'How do I structure my IT applications to best suit the business?'",

and the one presented at the guest lecture by Sogeti:

> "Architecture is a consistent set of rules and models that guides the design and implementation of processes, organizational structures, information flows and the technical infrastructure within an organization."

Describe one similarity and one difference between these two definitions.

Question 2 (20 points, 10 for each subquestion)

Google App Engine (GAE) is a well-known "Platform-as-a-Service" (PaaS) provider: for GAE customers, GAE hosts a Python environment and a Java environment on Google's servers that customers use to run their (Python or Java) programs without having to run their own servers or maintain virtualized operating systems. Google runs user-submitted programs on a large number of servers; it may happen that different requests for the same program are handled by different servers, without the user knowing. Part of the Python and Java environments is the Google Data Store: a non-relational database that can be used to store objects (which Google calls 'entities'). Entities are identified by a unique identifier: either a string set by the user, or an integer. For

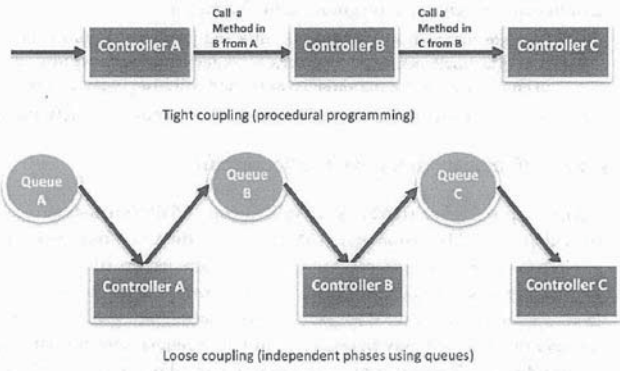
integer keys, the Google App Engine provides an additional feature called 'auto-increment': if this feature is enabled, whenever a user program creates an entity, the environment automatically generates a key that is guaranteed to be unique. GAE actually offers two variants: the 'legacy strategy', in which auto-generated keys are small integers, and 'scattered ids', in which auto-generated keys are very large random numbers that are approximately uniformly distributed over their range.
Recently[1], Google announced that for performance reasons, 'scattered ids' would become the default strategy, and the 'legacy strategy' would eventually be phased out.

a) Explain **how** using very large random numbers as keys may help in improving performance, assuming that many concurrent entity creations can be expected.

b) As stated before, the Google Data Store is not a relational database system. In fact, it is hierarchical: each entity may have a `parent'. Entities related by ancestry (i.e., an entity, its parent, its parents' parent, etc.) are called an 'entity group'. In the Google Data Store, transactions are in principle only possible with entities from the same entity group. Explain how this restriction may help to keep the Google Data Store scalable.

Question 3 (20 points, 10 for each subquestion)

In *Architecting for the Cloud: Best Practices*[2], the author uses the following figure to illustrate tight versus loose coupling:



[1] Cloud Platform Blog, May 31, 2013. http://googlecloudplatform.blogspot.nl/2013/05/update-on-datastore-auto-ids.html
[2] Amazon whitepaper by Jinesh Varia (2010), http://d36cz9buwru1tt.cloudfront.net/AWS_Cloud_Best_Practices.pdf.

Assume that the controllers depicted are applications that each run on a separate server. Tight coupling means that there are many dependencies between components (thus, in the figure, there are many dependencies between Controller A and Controller B, etc.), where a dependency of A on B means that the designer of B needs to know details of A. Loose coupling means that there are few dependencies. Dependencies mentioned in the whitepaper include runtime dependencies (e.g., due to a controller that is not responding) as well as development-time dependencies (e.g., Java interface specifications).

The author of the whitepapers is using the diagram to explain that, in his opinion, (remote) procedure calling between the controllers leads to tight coupling while using message queuing leads to loose coupling.

a) Explain why the author is right.

b) Although (according to the author) using message queuing leads to looser coupling, there are still remaining dependencies. Give an example of a dependency that is not taken away by using message queuing instead of RPC.

Question 4 (10 points, 5 for each subquestion)

TLB (Three-Letter Bank) is a large retail bank where millions of customers have payment and savings accounts. Every customer with a savings account also has a payment account, but not the other way around. Payment accounts have a small credit limit, meaning that the balance of a payment account can be negative up to this limit (say EUR 500). Savings accounts can never be negative. Savings accounts are protected by the following limit: a withdrawal via Internet can only be made to the payment account of the same customer (i.e., it is not possible to directly transfer an amount from a savings account to an account of someone else without first transferring it to the payment account).

Consider the following scenario:

- On a Friday evening, when her payment account has a small positive balance, using the web interface of TLB, a customer transfers an amount (substantially larger than the payment credit limit and current account balance combined) from her savings account to her payments account. The web interface immediately shows the new balance: the savings account balance is decreased, the payment account is increased;
- In line with TLB policies, the actual withdrawal is deferred until bank opening ours, the following Monday. The customer doesn't know this;
- Convinced that the withdrawal was successful as indicated in the web interface, a few minutes after the withdrawal, the customer transfers the money that has just been put in her payment account to another account of someone else at a different bank. This payment is executed

immediately, and the web interface shows the same balance as in the beginning of the scenario for the payment account;
- A few weeks later the customer, much to her surprise, notices that TLB has automatically taken a few Euros interest because the customer's payment account was negative over the weekend.

This seems to have something to do with transactions and ACID (atomic, consistent, isolated, durable) properties of transactions.

a) The withdrawal at Friday evening seems to have left the account record of the customer in an inconsistent state (in the ACID-sense). Explain.

TLB is notified of this scenario and decides that it is unacceptable. TLB decides to fix the situation by reconfiguring their systems such that the update of the web interface and the actual withdrawal is one atomic transaction.

b) Explain the impact of this on the scenario described above. What would the customer see and what who she be (dis)allowed to do?

Question 5 (10 points)

Imagine we have two applications A and B which communicate using Message Queuing. A sends a request to B and B sends one response back for each request. Because processing the requests is computationally expensive, there are three servers running the application B. For reasons of resiliency, there are two servers running application A. There are no databases involved in the applications.

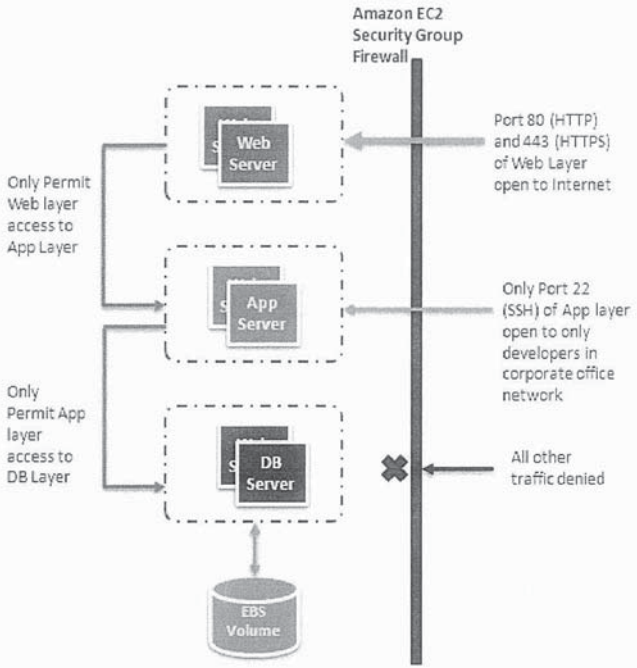
Draw a picture with both applications, all servers and all queues needed in this setup. Mark all arrows with "request", "response" or "both".

Question 6 (10 points)

In security, there are two concepts with similar-sounding names: authentication and authorization. Define each of them and explain the difference.

Question 7 (10 points)

In *Architecting for the Cloud: Best Practices*[1], the author discusses a security setup using the following figure (see next page):



A "security group" is an Amazon term for a set of servers on the same local area network protected by a firewall. The firewall is configured as depicted on the right hand side of this figure: traffic (that is: all traffic, e.g. requests coming from any user) for ports 22, 80 and 443 is allowed to pass, all other traffic is denied. In addition, traffic for port 22 can only pass if it originates from particular computers (those in the "corporate office network").

The Web Servers, App Servers and DB Servers depicted are obviously connected to one another, but this is left implicit.

Assume that the Web Servers, App Servers and DB Servers all delegate authentication and authorization decisions to a central identity management system (not depicted in the figure). Re-draw the figure according to the onion model.

(end)