Architecture of Information Systems (192320111): First exam 2010/2011

June 29, 2011 13:45-15:45h, Room CR 2M

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 6 questions.

Distribution of points:

10 points for showing up;

Other questions as indicated.

Question 1 (10 points)

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

Question 2 (20 points, 5 for each of the four subquestions)

The figure below depicts a middleware hub. Consider applications A and C. Assume that A needs to request a service from C. Application A is designed to ~~use remote procedure calls~~ (MQ) to talk to other applications: to use a service, it sends a message to the other application and then waits for a message send by that other application that contains the return value. However, Application C is designed to use remote procedure calling (RPC).

a) The middleware hub is used to convert from MQ to RPC. Explain how this is done. Which message queue(s) is/are needed? Where are the stubs?

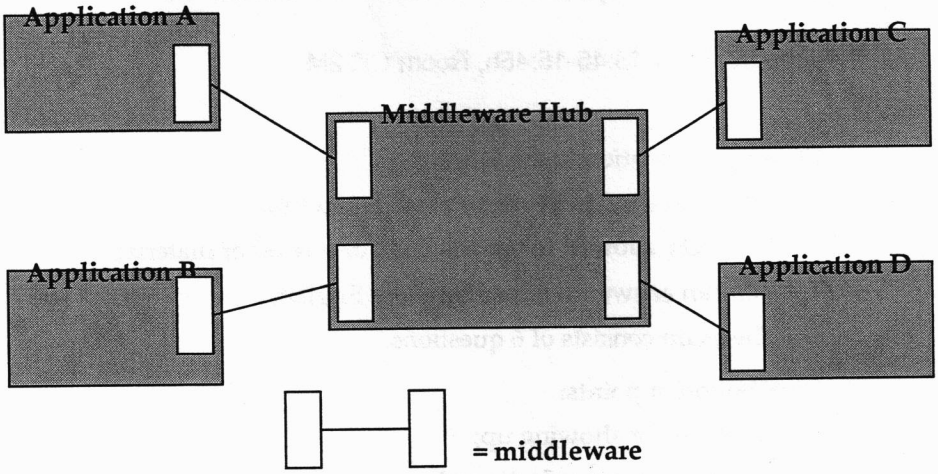
Suppose that C is the account management system of a bank; the service that A requests is to increase the balance of a certain account with x Euros. The middleware hub can choose between two types of MQ technology: one that guarantees at *least* once delivery and one that guarantees at *most* once delivery of each message.

b) In case of at *least* once delivery, what may go wrong? Can the hub and C avoid this? Same for at *most* once delivery.

Now suppose that C provides two additional services: request the current balance of an account, and set it to a certain amount.

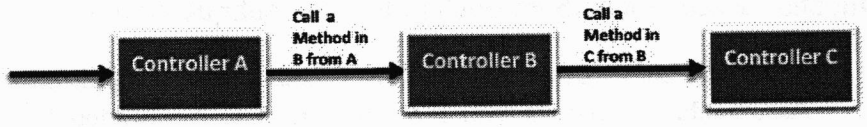
c) Using these additional services, can A by changed such that the implementation of the hub becomes simpler? Assume there is no concurrency.

d) Now assume that applications B and/or D may request C's services at the same time. Does this introduce a new risk? How can this be mitigated?

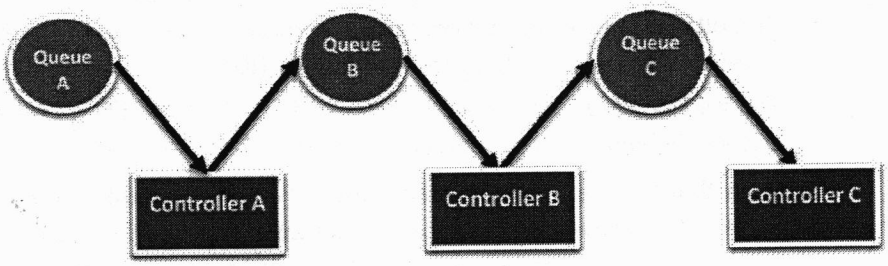


Question 3 (20 points, 10 for each subquestion)

In *Architecting for the Cloud: Best Practices*[1], the author uses the following figure to illustrate tight versus loose coupling:



Tight coupling (procedural programming)



Loose coupling (independent phases using queues)

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

[1] Amazon whitepaper by Jinesh Varia (2010), http://d36cz9buwru1tt.cloudfront.net/AWS_Cloud_Best_Practices.pdf.

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).

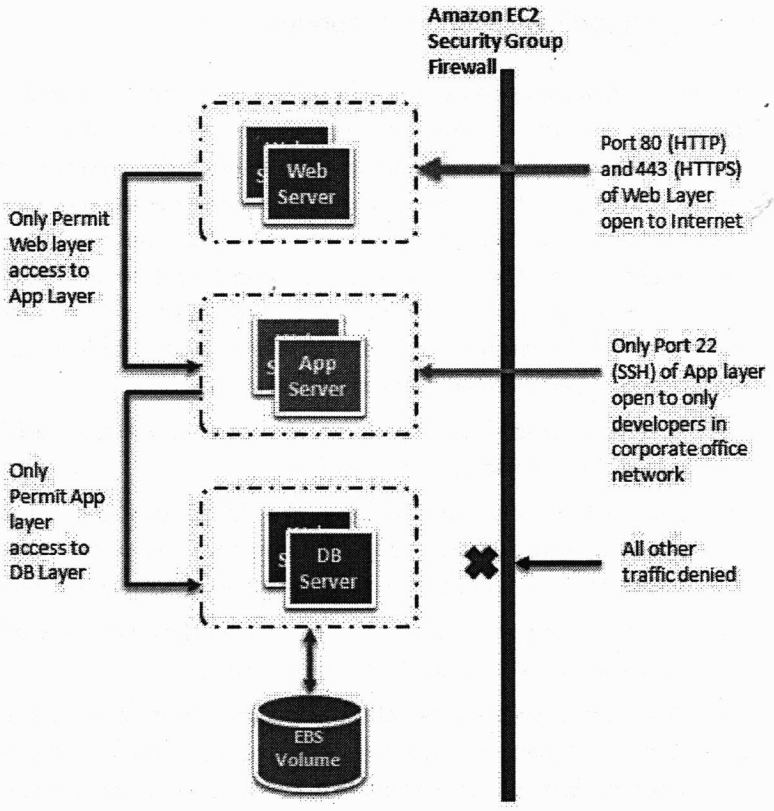
The author of the whitepapers is using the diagram to explain that, in his opinion, (remote) procedure calling between the controllers leads to tight coupling while using message queuing leads to loose coupling.

a) Describe 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)

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



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

rejected. 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 each of the three different kinds of servers is equipped with a component/function that can take authentication and authorization decisions. Re-draw the figure according to the onion model (indicating the access points and resources) and discuss which decision(s) each access point has to make.

Question 5 (10 points, 5 for each subquestion)

The book distinguishes between shared data and controlled duplication.

a) Describe one similarity and one difference between them from the perspective of security.

b) Describe one similarity and one difference between them from the perspective of fault tolerance.

Question 6 (20 points, 5 for each subquestion)

Assume we are designing an architecture for a website that is expecting a large number of visitors. For analysis of website visitors, a counter needs to be maintained that is incremented each time a visitor visits a page of the site (the counter is not shown on the website). To handle expected load, there are five webservers behind a load balancer that distributes incoming requests over those five webservers. The webservers are connected to a single database server which keeps the counter. To increment the counter, a webserver reads the current value of the counter from the database, adds 1 to it to get the new counter value, and saves the new value to the database.

a) The three actions needed to update the counter (read, add 1, write) need to be put in a transaction. Explain why.

b) Transactions are said to have the ACID properties (atomic, consistent, isolated, and durable). Explain these four properties in the context of this questions (i.e., in terms of webservers that need to increment a counter).

c) Incrementing the counter on the shared database server is a performance and scalability bottleneck. Explain why.

d) A common solution is called 'sharding' (not 'sharing'). In this case, additional database servers are installed and each of them keeps a partial counter. E.g., database server 1 keeps the counter for webservers 1 and 2, and database server 2 keeps the counter for webservers 3, 4 and 5. If the total number of visitors is needed, one can read the two partial counters and add them. Explain why this mitigates the performance/scalability bottleneck.