[Wersja polska na osobnej kartce]

Distributed Systems: Final Exam **February 3, 2012**
Exam duration: 2 h
Total points: 50

Question 1 (2p). Distributed systems are often heralded as *scalable*, where scalability refers to *size*.
a) Name at least two different sample units of the size metric when talking about scalability.
b) Name at least two other scalability types/metrics that a truly scalable distributed system needs to consider.

Question 2 (4p). *One-way hashing functions* are used in many areas of distributed systems.
a) Define a one-way hashing function (and explain its properties). (3p)
b) Give two examples of popular one-way hashing functions. (1p)

Question 3 (6p). *Remote procedure call (RPC)* is a communication paradigm aiming at distribution transparency.
a) Explain how RPC can be used to ensure migration transparency.
b) Explain how RPC can be used to ensure replication transparency.
c) Explain how RPC can be used to ensure failure transparency.

Question 4 (6p). Network Time Protocol (NTP) and other *time synchronization protocols* operate by exchanging local timestamps recorded at particular moments at different machines. Describe the NTP timestamp exchange algorithm for two machines: a client and a server. Make sure you explain how the exchanged timestamps can be used to synchronize time and what assumptions are made by the algorithm.

Question 5 (6p). Describe three different mutual algorithms for distributed systems. Discuss and compare their advantages and disadvantages.

Question 6 (13p). Suppose you are running a large distributed online shop with clients in different countries. To minimize the costs associated with hosting your system, you decided to adopt a cloud-based solution: you rent a number of machines from a global cloud service provider. The provider's machines can host arbitrary software that you supply. There are thousands of them, and they are located virtually all over the world. Moreover, the provider supports dynamic on-demand acquisition and relinquishment of machines, that is, at any time you can start additional instances of your software on arbitrary provider's machines that are currently available or you can dispose of any no longer needed running instances. Billing the additional resources is done automatically by the provider.

Normally, you rent a few machines in different parts of the world. However, occasionally the number of machines is insufficient to serve your clients' requests. Typically this happens when certain shop items become popular or before major holidays, but you are not always able to predict such situations. The great majority of requests during such bursts are requests for browsing your shop and particular items.

Your goal is to minimize the costs incurred by hosting your system and not to lose clients due to a degraded performance during request bursts.
a) Propose a logical organization of the software comprising your system.
b) Propose a physical distribution of the software across the provider's machines.
c) Describe the mechanisms and algorithms you would employ to handle request bursts.
d) Discuss a few potential problems with the solutions from a)-c).

Question 7 (13p). Consider a database that is replicated on *N* machines. To access the database, a

client accesses any of the replicas. The communication between clients and the replicas and between the replicas themselves is reliable, point-to-point, and FIFO-ordered. However, the communication delays can vary greatly. Consider the following variant of a totally-ordered multicast algorithm:

> *Each replica maintains Lamport's logical clock, and every inter-replica message is stamped with the value of the sender's clock upon transmission. Whenever a replica receives a database update message from a client, it broadcasts the update in a message to all replicas (including itself). Whenever a replica receives an update message from another replica (or from itself) it puts the message into a local queue, and acknowledges the reception of the update by broadcasting an acknowledgment message to all other replicas (including itself). The replica applies an update from its local queue to its local database if and only if:*
> *(1) the update message has the lowest timestamp among the messages in the replica's local queue, and*
> (2) *the update message has been acknowledged by a quorum of* $\lfloor N/2 \rfloor + 1$ *replicas.*
> *After the replica has applied the update to the local database, the update message and its acknowledgments are removed from the local queue. If later another acknowledgment arrives for the message, such a late acknowledgment is simply dropped from the queue.*

Prove that the above algorithm implements totally-ordered multicast or produce a counter example.

Good luck!

[English version on a separate sheet]

Systemy rozproszone: Egzamin końcowy **3 luty 2012**
Czas pisania: **2 h**
Liczba punktów: **50**

Pytanie 1 (2p). Systemy rozproszone są często reklamowane jako *skalowalne* (ang. *scalable*), gdzie skalowalność odnosi się do *rozmiaru*.
a) Wymień co najmniej dwie przykładowe jednostki rozmiaru używane przy definiowaniu skalowalności.
b) Wymień co najmniej dwa inne typy skalowalności, które prawdziwie skalowalny system rozproszony musi uwzględniać.

Pytanie 2 (4p). *Jednokierunkowe funkcje mieszające* (ang. *one-way hashing functions*) używane są w wielu dziedzinach systemów rozproszonych.
a) Podaj definicję jednokierunkowej funkcji mieszającej (oraz wyjaśnij jej własności). (3p)
b) Podaj dwa przykłady popularnych jednokierunkowych funkcji mieszających. (1p)

Pytanie 3 (6p). *Zdalne wołanie procedur* (ang. *remote procedure call (RPC)*) to paradygmat komunikacyjny mający zapewnić przeźroczystość rozproszenia (ang. *distribution transparency*).
a) Wyjaśnij, jak można użyć RPC do zapewnienia przeźroczystości migracji (ang. *migration transparency*).
b) Wyjaśnij, jak można użyć RPC do zapewnienia przeźroczystości replikacji (ang. *replication transparency*).
c) Wyjaśnij, jak można użyć RPC do zapewnienia przeźroczystości błędów (ang. *failure transparency*).

Pytanie 4 (6p). Network Time Protocol (NTP) i inne protokoły synchronizacji czasu opierają się na wymianie tzw. stempli czasowych (ang. *timestamp exchange*) zarejestrowanych przez różne maszyny w pewnych określonych momentach. Opisz algorytm wymiany stempli czasowych używany przez NTP do synchronizacji dwóch maszyn: klienta i serwera. Nie zapomnij wyjaśnić, w jaki sposób wymieniane stemple czasowe mogą być użyte do synchronizacji czasu oraz jakie założenia przyjmuje algorytm.

Pytanie 5 (6p). Opisz trzy różne algorytmy wzajemnego wykluczania (ang. *mutual exclusion*) zaprojektowane dla systemów rozproszonych. Omów i porównaj ich zalety i wady.

Pytanie 6 (13p). Wyobraź sobie, że jesteś odpowiedzialna/odpowiedzialny za działanie od strony technicznej dużego rozproszonego sklepu internetowego z klientami w różnych krajach. Aby zminimalizować koszty związane z utrzymywaniem infrastruktury informatycznej systemu, zdecydowałaś/zdecydowałeś się na rozwiązanie oparte na chmurze (ang. *cloud*): wynajmujesz maszyny od globalnego dostawcy. Na wynajętych maszynach może działać dowolne dostarczone przez Ciebie oprogramowanie. Maszyn do wynajęcia są tysiące i znajdują się praktycznie w dowolnym miejscu na świecie. Ponadto Twój dostawca wspiera dynamiczne (na żądanie) rezerwowanie i zwalnianie maszyn, to jest, w dowolnym momencie możesz uruchomić dodatkowe instancje swojego oprogramowania na dowolnych dostępnych maszynach dostawcy lub pozbyć się niepotrzebnych działających instancji. Należność finansowa za dodatkowe zasoby jest naliczana automatycznie przez dostawcę.

Zwykle wynajmujesz tylko kilka maszyn w różnych częściach świata. Jednakże od czasu do czasu ta liczba maszyn okazuje się niewystarczająca do obsłużenia żądań klientów. Sytuacja taka ma zwykle miejsce, gdy pewne sprzedawane artykuły stają się popularne lub tuż przed głównymi świętami w różnych krajach, lecz nie zawsze jesteś w stanie ją przewidzieć. Ogromna większość

żądań podczas takiej zwiększonej aktywności to żądania związane z przeglądaniem katalogu oferowanych produktów oraz przeglądaniem specyfikacji konkretnych produktów.

Twoim celem jest zminimalizowanie kosztu związanego z wynajmem maszyn oraz jednocześnie uniknięcie sytuacji, w których sklep traci klientów, ponieważ ci nie są zadowoleni z jego wydajności w okresach wzmożonej aktywności.

a) Zaproponuj logiczną organizację oprogramowania tworzącego Twój system.

b) Zaproponuj fizyczne rozmieszczenie i rozproszenie oprogramowania pomiędzy maszyny dostawcy.

c) Opisz mechanizmy i algorytmy, których użyłabyś/użyłbyś do obsługi okresów wzmożonej aktywności klientów.

d) Omów potencjalne problemy w rozwiązaniach z punktów a)-c).

Pytanie 7 (13p). Rozważmy bazę danych zreplikowaną na *N* maszynach. Aby korzystać z bazy, klient komunikuje się z którąkolwiek z replik. Komunikacja pomiędzy klientami i replikami oraz pomiędzy samymi replikami odbywa się punkt-do-punktu, jest niezawodna, oraz zachowuje kolejność komunikatów. Jednakże opóźnienia komunikacyjne mogą się istotnie zmieniać. Rozważmy następujący wariant algorytmu totalnie uporządkowanego rozgłaszania (ang. *totally-ordered multicast*):

> *Każda replika posiada zegar logiczny Lamporta a każdy komunikat przesyłany pomiędzy replikami niesie w sobie wartość zegara nadawcy podczas transmisji. Kiedy replika otrzymuje od klienta żądanie uaktualnienia bazy danych, rozgłasza komunikat zawierający to żądanie do wszystkich replik (włączając siebie). Kiedy replika otrzyma taki komunikat od innej repliki (lub od siebie), wstawia ten komunikat do lokalnej kolejki i potwierdza jego otrzymanie poprzez rozgłoszenie komunikatu potwierdzającego do wszystkich replik (włączając siebie). Replika aplikuje żądanie uaktualnienia bazy danych ze swojej lokalnej kolejki do swojej bazy danych wtedy i tylko wtedy, gdy:*
>
> *(1) komunikat zawierający żądanie ma najmniejszy stempel czasowy spośród komunikatów w lokalnej kolejce repliki oraz*
>
> (2) *komunikat zawierający żądanie został potwierdzony przez kworum* $\lfloor N/2 \rfloor + 1$ *replik.*
>
> *Po zaaplikowaniu przez replikę żądania uaktualnienia do lokalnej bazy danych, komunikat zawierający to żądanie oraz wszystkie komunikaty potwierdzeń dla tego komunikatu są usuwane z lokalnej kolejki. Jeśli później otrzymany zostanie kolejny komunikat potwierdzający dla usuniętego komunikatu żądania, takie spóźnione potwierdzenie jest od razu usuwane z lokalnej kolejki repliki.*

Udowodnij, że powyższy algorytm faktycznie implementuje totalnie uporządkowane rozgłaszanie lub podaj kontrprzykład.

Powodzenia!