1a Explain the difference between iterative and recursive name resolution in a distributed naming service such as DNS. *5pt*

See figures 4-8 and 4-9 in the textbook.

1b Some DNS host names can be resolved to several IP addresses. What is the purpose of such an approach? *5pt*

There may be several reasons for doing so. First, having multiple IP addresses means that the service that needs to be contacted under a certain name has been replicated. This replication can be done for fault tolerance or performance. The latter is particularly interesting: some DNS servers can return an IP address that is closest to the requesting client.

1c In contrast to naming service like DNS, general-purpose directory services such as X.500 and LDAP are much harder to scale. Why? *10pt*

The problem is that unlike DNS where each part of a pathname can be dealt with by a separate server, general-purpose directory services use arbitrary attribute/value pairs. To find what you are looking for, you essentially need to match your (potentially complex) query against a huge number of such pairs. These pairs may be distributed worldwide across many different servers. So far, directory services have shown to perform well only when implemented on a single machine.

2a Explain how using data distribution and replication as scaling techniques introduces new scalability problems. *5pt*

Distributing data worldwide without replication may lead to placing data far away from some clients. This approach will then lead to geographical scalability problems because we can not exploit locality. Data replication introduces a synchronization problem to keep all replicas consistent in the face of updates.

2b To improve the scalability of a Web site server clusters are often used. Why is it useful to apply content-aware request distribution in these cases, and what is the main drawback in doing so? *5pt*

With content-aware request distribution the content of a request is inspected and then forwarded to the most suitable server. This approach is useful because it allows the same request to be forwarded to the same server, in turn making server-side caching more effective. Also, it becomes possible to dedicate servers to special requests like images, audio, and so on.

2c Under which conditions is it useful to use mobile agents to alleviate scalability problems. Are there other reasons why mobile agents are useful? *10pt*

If it is cheaper in terms of response time to send an agent to a remote site, locally inspect the data needed to generate a response, and return the results back to the client. Other useful reasons have to do with programming convenience and perhaps security, but in general, performance improvement is the most important one.

3a What does an object adapter do? *5pt*

It accepts incoming invocation requests for the objects residing under its "regime." This means that the adapter decides how the invocation is to be carried out, for example, by using a separate thread. It is also responsible for deciding whether its objects are persistent or transient, making objects available to the outside world by providing (part of) a reference, and so on. In other words, an object adapter provides a wrapper around data and operations on that data that coincides with what you would expect from an object.

3b Explain the difference between request-level interceptors and message-level interceptors as used, for example, in CORBA. *5pt*

Request-level interceptors are special local components to which an invocation request is passed before passing it to the underlying ORB. Such an interceptor is invocation aware in the sense that it knows with which invocation it is dealing, and for which object it is intended. Typically, such interceptors can be used to implement replicated invocations. A message-level interceptor is a component that is logically placed between an ORB and the underlying operating system. It can thus handle only basic network messages, for example, by fragmenting them into smaller parts (and assembling these parts at the receiver side).

3c Consider a system supporting only remote objects such as CORBA. Sketch the design of an extension that uses request-level intereceptors to implement a primary-backup protocol. (Hint: do not use client-side interceptors.) *10pt*

The problem is that we need to replicate remote objects without affecting the original model. We can do so by using a server-side interceptor at each replica. If this interceptor intercepts a state-preserving invocation (like a read), it passes it on to the replica. A state-modifying invocation is treated differently. If the interceptor is associated to the primary replica, the backups are invoked using RMI, these invocations will be intercepted at every backup and passed on to the replica, the primary interceptor is notified, after which this interceptor passes the invocation to the primary. State-modifying invocations at a backup are intercepted and forwarded to the primary where they are processed as just explained.

4a How can we protect a certificate from being successfully used by an unauthorized user? (Hint: read the question carefully!) *5pt*

Bob needs to prove that he got the certificate from Alice. This can be done by having Alive pass a secret key K^- to Bob through a secure channel that can be used in conjunction with a public key $K+$ stored in the certificate. The server will ask Bob to decrypt the challenge $K^+(N)$ and return N. Yes, it's the same question as the one in the exam, but phrased differently. It is not my intention to tease you.

4b Public-key cryptosystems are claimed to scale better than shared-key systems. Does this claim really hold? *Hint:* think of certificate revocation. *5pt*

The problem with public-key systems is that certificate revocation requires that we check whether a certificate is still valid each time we use it. Checking the validity generally means contacting a global service that keeps track of, for example, revocation lists. Depending on the advertised lifetime of a certificate, we may introduce a scalability problem here.

5a Explain the principle of a publish-subscribe system. *5pt*

These systems use—at the very least—subject-based addressing, by which messages are tagged with a keyword describing the subject of the message. Publishers produce messages. A message on subject S is sent to only those processes that have subscribed to S.

5b IBM MQSeries and TIB/Rendezvous have two different ways of implementing a publish-subscribe system. What are they? *5pt*

MQSeries uses a centralized message broker, whereas TIB uses multicasting.

5c Sketch the principal working of a (possibly inefficient) publish-subscribe system with a centralized JavaSpace router. *10pt*

All messages have an associated tuple describing their contents. Messages are sent to the JavaSpace server where they are temporarily stored, for example, by means of a lease with a

short expiration time. Subscribers simply do a blocking read at the server waiting for messages to arrive. A separate read operation needs to be done for every type of message (i.e., a subscriber does a separate read for each subscription). After a read has been carried out, the subscriber simply does another blocking read. Although this may lead to reading the same message again, it is easy to detect duplicates and throw them away. Note, however, that this approach wastes resources.

Grading: *The final grade is calculated by accumulating the scores per question (maximum: 90 points), and adding 10 bonus points. The maximum total is therefore 100 points.*