Distributed Systems: Concepts and Design

Edition 5

By George Coulouris, Jean Dollimore, Tim Kindberg and Gordon Blair

Chapter 10 Exercise Solutions

10.1 Explain the characteristics of a peer-to-peer service. What are some examples of peer-to- peer middleware? *pages 424, 425*

10.1 Ans.

Peer-to-peer systems share these characteristics:

- Their design ensures that each user contributes resources to the system.
- Although they may differ in the resources that they contribute, all the nodes in a peer-to-peer system have the same functional capabilities and responsibilities.
- Their correct operation does not depend on the existence of any centrally-administered systems.
- They can be designed to offer a limited degree of anonymity to the providers and users of resources.
- A key issue for their efficient operation is the choice of an algorithm for the placement of data across many hosts and subsequent access to it in a manner that balances the workload and ensures availability without adding undue overheads.

The best-known and most fully developed examples of peer-to- peer middleware include Pastry [Rowstron and Druschel 2001], Tapestry [Zhao *et al.* 2004], CAN [Ratnasamy *et al*. 2001], Chord [Stoica *et al*. 2001] and Kademlia [Maymounkov and Mazieres 2002].

10.2 The problem of maintaining indexes of available resources is application-dependent. Consider the suitability of each of your answers to Exercise 10.1 for

(i) music and media file sharing,

(ii) long-term storage of archived material such as journal or newspaper content,

(iii) network storage of general-purpose read-write files.

10.2 Ans.

a) Solution (i) is suitable for slowly - changing collections of resources such as media files. It is used by BitTorrent, supplemented by a *tracker* service for each resource that holds up-to-date information about the current locations of available replicas.

b) Solution (ii) is suitable; the updates can be disseminated in a slow algorithm that provides eventual consistency.

c) This is difficult to achieve in a totally scalable manner. Some partitioning of the files as in Ivy and Oceanstore is needed. Clients must hold the GUIDs or other reference to the group of hosts responsible for the portions of the store they are interested in.

10.3 What are the non-functional requirements of peer-to-peer middleware? *page 431*

10.3 Ans.

Peer-to-peer middleware must address the following non-functional requirements:

- Global scalability
- Load balancing
- Optimization for local interactions between neighboring peers
- Accommodating to highly dynamic host availability
- Security of data in an environment with heterogeneous trust
- Anonymity, deniability and resistance to censorship

10.4 The guarantees offered by conventional servers may be violated by:

i) physical damage to the host;

ii) Errors or inconsistencies by system adminstrators or their managers;

iii) successful attacks on the security of the system software;

iv) hardware or software errors.

Give two examples of possible incidents for each type of violation. Which of them could be described as a breach of trust or a criminal act? Would they be breaches of trust if they occurred on a personal computer that was contributing some resources to a peer-to-peer service? Why is this relevant for peer-to-peer systems?

10.4 Ans.

i) Power failure, earthquake, flood, act of war or sabotage, owner throws the computer away. The last two are breeches of trust for servers, but the owner of a PC is free to throw it away.

ii) Accidental deletion of a file, permission failure, there are many possible errors in system administration. Maybe not a breach of trust, but repeated occurrences are a serious matter in a service, though not on a PC.

iii) The attacks described in Section 7.1.1 are always a breach of trust or a criminal attack for servers. But for a PC the perpetrator may be the owner who may attack a user who is sharing the resources of their computer with impunity.

iv) Hard disk failures, network failures, program bugs. These are not normally due to breaches of trust or criminal acts. Servers are configured to 'fail over' to use alternative hardware and backup copies of data. The the software they run is probably more carefully validated before it is put into use.

The differences in what is 'trusted behaviour' for servers and PCs is relevant because peer-to-peer system must be designed to cope with the looser interpretation of trust for PCs.

10.5 Peer-to-peer systems typically depend on *untrusted* and *volatile* computer systems for most of their resources. Trust is a social phenomenon with technical consequences. Volatility (i.e. unpredicatable availability) also is often due to human actions. Elaborate your answers to Exercise 10.4 by discussing the possible ways in which each of them is likely to differ according to the following attributes of the computers used:

i) ownership

ii) geographic location

iii) network connectivity

iv) country or legal jurisdiction

What does this suggest about policies for the placement of data objects in a peer-to-peer storage service?

10.5 Ans.

ownership The owner of a computer is likely to act in a manner that maximizes his benefit, regardless of the fact that some of its resources are shared by others. The notion of trust is relative to ownership, so actions of type (b) or even (c) may be classified as acceptable.

geographic location Computers are subject to events of type (a) according to their geographic location.

network connectivity Portions of a network may become separated, making communication between them impossible. This might enable the owners in a disconnected portion to act against the interest of the majority.

country or jurisdiction The affects the 'freedom of speech' issue since governments or courts may persecute the owners of information or order the deletion of data. The latter can be addressed by ensuring that there are replicas in several countries/jurisdictions.

10.6 Assess the availability and trustworthiness of the personal computers in your environment. You should estimate:

Uptime: hours per day when the computer is operating and connected to the Internet.

Software consistency: is the software managed by a competent technician?

Security: is the computer fully protected against tampering by its users or others?

Based on your assessment, discuss the feasibility of running a data sharng service on the set of computers you have assessed and outline the problems that must be addressed in a peer-to-peer data sharing service.

10.6 Ans.

Answer is dependent on your local system environment.

10.7 What are the tasks of a routing overlay? Give a comparative study between structured and unstructured peer-to-peer systems.

10.7 Ans.

The tasks of a routing overlay are as follows:

1. A client wishing to invoke an operation on an object submits a request including the object's GUID to the routing overlay, which routes the request to a node at which a replica of the object resides.
2. A node wishing to make a new object available to a peer-to-peer service computes a GUID for the object and announces it to the routing overlay, which then ensures that the object is reachable by all other clients.
3. When clients request the removal of objects from the service the routing overlay must make them unavailable.
4. Nodes (i.e. computers) may join and leave the service. When a node joins the service, the routing overlay arranges for it to assume some of the responsibilities of other nodes. When a node leaves (either voluntarily or as a result of a system or network fault), its responsibilities are distributed amongst the other nodes.

In structured approaches, there is an overall global policy governing the topology of the network, the placement of objects in the network, and the routing os search functions used to locate objects in the network. In other words, there is a specific (distributed) data structure underpinning the associated overlay and a set of algorithms operating over that data structure.

Because of this maintenance argument, unstructured peer-to-peer approaches have also been developed. In unstructured approaches, there is no overall control over the topology or the placement of objects within the network.

10.8 The design of Ivy resolves several previously unresolved issues arising from the need to host files in partially trusted or unreliable machines. Explain these issues. *pages 455, 456*

10.8 Ans.

- The maintenance of consistent file metadata with potentially concurrent file updates at different nodes.
- Partial trust between participants and vulnerability to attacks of participants' machines. Recovery from integrity failures caused by such attacks is based on the notion of views of the file system.
- Continued operation during partitions in the network which can result in conflicting updates to shared files.

10.9 Routing algorithms choose a next hop according to an estimate of distance in some addressing space. Pastry and Tapestry both use circular linear address spaces in which a function based on the approximate numerical difference between GUIDs determines their separation. Kademlia uses the XOR of the GUIDs. How does this help in the maintenance of routing tables? Does the XOR operation provide appropriate properties for a distance metric?

10.9 Ans.

The Kademlia paper states that the symmetry of the XOR operation ensures a simple and efficient routing mechanism. It results in symmetric routing tables being which isn't true for some other routing algorithms. (It is for Pastry, but not for Chord). Symmetry means that nodes can learn new routing information from incoming messages, since the routes they have taken are reversible. In Kademlia the XOR of the GUIDs for the source and destination nodes is treated as a numeric distance metric and this results in a sensibly-behaved distance metric (see Kademlia paper Section 2.1).

10.10 When the Squirrel peer-to-peer web caching service was evaluated by simulation, 4.11 hops were required on average to route a request for a cache entry when simulating the Redmond traffic, whereas only 1.8 were required for the Cambridge traffic. Explain this and show that it supports the theoretical performance claimed for Pastry.

10.10 Ans.

The answer is simply that the number of routing hops required in Pastry is O(log N) where N is the number of nodes participating in the overlay. The Cambridge data was based on 105 nodes whereas the Redmond data inclided 36000 nodes. The simulated performance is almost exactly the same as the theoretical: ln(36000)/ ln(105) = 2.26; 4.11/1.8 = 2.28.

10.11 In unstructured peer-to-peer systems, significant improvements on search results can be provided by the adoption of particular search strategies. Compare and contrast expanded ring search and random walk strategies, highlighting when each approach is likely to be effective.

10.11 Ans.

An expanded ring search is a variant of a flooding based strategy whereby flood messages are sent out with increasing time-to-live values until a particular item is found. The approach is therefore quite heavyweight in terms of the message traffic generated but can be very effective if items are found locally, for example with popular items which are likely to be heavily replicated. A random walk strategy is one where a walker is set out on a random path until an item is found. This can greatly decrease the number of messages generated but at the expense of significant increases in the average time to find an item. This can be reduced by having multiple walkers (changing the the granularity of coverage).

In summary, expanded ring search can be used when minimizing delay is more important than the number of messages generated and can be very effective if items are likely to be local. Random walk strategies can be used when the number of messages should be kept under control and this can be fine-tuned in multiple walker variants.

