PAC 3

SISTEMES DISTRIBUITS AULA 2

MANUEL MARTÍNEZ DÍAZ

QUESTIONS

1. Explain what spatial and temporal coupling mean. Classify in the matrix of spatial/temporal coupling: RMI, JMS, JGroups, IP Multicast, mailbox, and Linda Tuple Space.

The spatial and temporal coupling is used in communication models to define the communication approach in a distributed system.

Spatial coupling is referred to what type communication is used; **direct**, the sender knows the receiver/receivers or **indirect** communication where the messages are directed towards a multicast group, space-couple and space-uncouple respectively.

Temporal coupling is the model communication considering when sender and receiver is available. **Time coupled** is when sender and receiver exist at the same moment and time uncoupled the sender and receiver can have independent lifetimes.

Matrix Spatial/temporal coupling

	Time-coupled	Time-uncoupled
Space coupling	RMI	Mailbox
Space uncoupling	IP multicast	JMS LindaTuple Space JGroups

2. Explain the differences between the four models of indirect communication: group communication, publish/subscribe, message queues and shared memory systems. Give real examples of each model.

All four models support space uncoupling, and time uncoupling is possible on group communication and publish/subscribe.

The main difference between the models is that message queues use a communication point to point and the rest use one-to-many. An example of message queues is WebSphere MQ with SSL that provide security, authentication and access control.

Publish/subscribe is other model that share with message queues, differ from the other two, a good scalability because of their independency between objects, the data and the processes are independent. In this model a publisher publish something and subscribers show interests by channel, topic or content for example. Is based on events and the architecture can be centralized or distributed with different nodes.

An example of publish /subscribe is Gryphon from IBM, Scribe or TERA used for monitoring applications and cooperative work.

In the *Shared memory systems* model there are two; Distributed shared memory (DSM) and Tuples space communication. DSM is at bytes level and Tuples offer a deeper level with semistructured data. DSM provides transparency to access data stored in different physical locations. You access that memory as a single memory but is distributed. With Tuples processes communicates through a tuple space. Tuples usually use a centralized solution, although is recommend a distributed one in order to provide fault tolerance and scalability. The most important different of DSM in front of the other models are that the system share a memory space that the other don't.

An example of DSM is the Open SSI, a single-system image clustering.

At last the model *group communication* is different in front the others because its goal is offer reliable multicast and ordered multicast (FIFO, total or partial ordering).
An example is the toolkit Jgroups.

3. Compare the Chord routing scheme with the Kademlia routing scheme using an example.
What similarities do Tapestry and Pastry routing mechanisms present?

In Chrod, each node has a successor and a predecessor. The successor to a node is the next node in the identifier circle in a clockwise direction. The predecessor is counter-clockwise. Each node has a finger table where keep a list of successors. When a node wants to look up a key k, it send a lookup(key) to query to the closest successor or predecessor (depending on the finger table) of k in its finger table (the "largest" one on the circle whose ID is smaller than k), until a node finds out the key is stored in its immediate successor. Then the IP address of the node is returned along reverse path.

Kademlia, assign each peer a Node ID of 160-bit. Is used a NodeID- base routing algorithm to locate peers near a destination key. Having a node p that wants to find an object o with key d, the routing process begins with a parallel lookup process. The node p creates k-candidate for key d with the closest nodes to key d that node p is aware of. On each iteration node p picks the first value greater than 0 in the k-candidate list that is has nor queried yet and sends each of them in parallel an asynchronous lookup query for key d. In response, each of these nodes r returns to node p the k closest nodes to key d that node r is aware of. When node p receive

each reply, it updates its k-candidates list to hold the k closest nodes to key d that node p knows about following the receipt of this reply. After each reply or a timeout from the different nodes r, node p sends a query to the first node in its k-candidates list that it has no contacted yet. The process ends when node p has finis the contacting all nodes in its k-candidates list.
It is used a XOR metis for distance between point.

Tapestry and Pastry routing algorithm are similar to the Plaxton's algorithm.

Tapestry and Pastry use a prefix/suffix address routing, and similar insertion/deletion algorithms, and similar storage overhead costs.

The routing mechanisms are based on matching the suffix in NodeID. The routing maps at each peer are organized into routing levels, where each level contains entries that point to a set of peers closest in distance that matches the suffix for that level. Also, each peers holds a list of pointer to peers referred to as neighbors.

The system assign a unique number as a key to every peer from each peer's routing table, whose key shares a longer prefix or suffice with the key used as target address.

4. Briefly describe the main design feature of BitTorrent. How does BitTorrent incentive the cooperation?

BitTorrent is a Peer-to-peer system to share files. Each file is divided in pieces (chunks) of the same size, usually 256 Kbytes. A peer that has a complete file is named seeder of that file and the that has an uncomplete files is a leecher.

A peer when connects to BitTorrent request to a central server named tracker to get a list of the peers that has that file, then the peer try to connect with all peers on the list. After the connection has establish, the peer will know which file's chunks has each connected peer. Using a rate first mechanism, the peer starts downloading the chunks that are less common, in order to populate them and equilibrate the file chunks availability. Not only the seeder uploads chunks, also leecher uploads chunks to server other peers, is only needed to have one chunk of the file.

When a leecher obtains all the chunks of a file, it changes to be a seeder.

BitTorrent use an incentive mechanism choking (not uploading) and unchoking (uploading). Peers also maintain the current download rates form all their links to determine which peer to unchoked. As a result, the peers with the highest download rate are unchoked, demanded to upload chunks, and the other peers are choked except for one that is decides by a mechanism called optimistic unchoking. This mechanism is used to find a peer with better download rate. When a better download rate peer is found the previous unchoked link is replaced by the new one. The other peers connected knows that a new unchoked link is in the unchoking list.

After a period of time that can be modified, usually 30 seconds, a new iteration of optimistic unchoking is performed. This method does not promote share because is focus finding peers with a better download rate.

In my opinion could be more efficient to determine a rule to reward the peers that upload or share more, for example letting faster downloads or prioritizing them in the peers connections.

5. Briefly explain the MapReduce model and how Google uses it.

Map Reduce is a programming model for wide distributed data processing, used generating search indexes, document clustering or data analytics in general. It's composed by three stages. First, the input data is split in pieces of the same length. The pieces are sent to different machines to be processed at the same time and create a result. This result is not the final one, but are used to combined them and get the output data.

To process the data is used an algorithm with two functions, one to process the data pieces and the other to combined and get the final data. This model is named by Google as MapReduce model and use the functions map and reduce.

The process engine has a Master node and Workers nodes. The master's control the process of the job and control the worker's status. It also assign the map and reduce tasks to the workers nodes based on data locality, network state and other considerations. In case of a worker node failed the master reassign the failed task to other worker node. The workers are instances of the MapReduce library that are executed in different clusters. The status of a worker could be idle, in-progress or completed.

The map function is applied to each input data element (key-value is provided to the function by a worker node) a user defined function to generate intermediate results, a list of key-values. The framework MapReduce process all the intermediate results and grouped by the same key and create a new list with key/value pair for each different key. This information is notified to the master node.

The list of value/key results of the map function will be the input data of the reduce function.

The reduce function is another user defined function that once all intermediate results are obtained from map workers node, is used in the workers assigns reduce tasks to merge the data by keys and all values with the same key are grouped together. Finally, the user-defined function is applied to each key/value pair, and the results are output to the distributed storage system (google use GFS).

Google use an interface independently of the MapReduce model, hidden the distributed computation, the fault tolerant and load balancers to focus in the data process. Google use MapReduce in web crawling, web indexing ore Google Maps. As an example when google count words in documents, the function map for every word in a document output (word, "1") and the reduce function sum all occurrences of words and output (word, total_count).

Fonts:

[1] BitTorrent, http://www.bittorrent.org/beps/bep_0003.html

[2] Paolo Di Francesco, Stefano Cucchiella. 2009. Selfish strategies affecting the BitTorrent protocol. http://www.idt.mdh.se/kurser/ct3340/ht09/ADMINISTRATION/IRCSE09-submissions/ircse09_submission_26.pdf

[3] BitTorrentSpecification - Theory.org Wiki

[4] Lachlan Aldred, Wil M.P. van der Aalst, Marlon Dumas and Arthur H.M. ter Hofstede. Dimensions of Coupling in Middleware. http://kodu.ut.ee/~dumas/pubs/ccmiddleware.pdf

[5] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnan. Chor

d: A Scalable Peer-to-peer Lookup Service for Internet Applications. ishttp://pdos.csail.mit.edu/papers/chord:sigcomm01/chord_sigcomm.pdf

[6] Li, F., Ooi, B-C., Özsu, M. T., Wu, S. 2013. Distributed Data Management Using MapReduce. ACM Comput. Surv. 0, 0, Article A (0), 41 pages.

[7] DSM, https://en.wikipedia.org/wiki/OpenSSI

[8] https://www.cl.cam.ac.uk/research/dtg/www/files/publications/public/mp431/ieee-survey.pdf

[9] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.7439&rep=rep1&type=pdf

[10] Kaleidoscope: Adding Colors to Kademlia, http://www.cs.technion.ac.il/~gilga/P2P2013_49.pdf

[11] Chord: A Scalable Peer-To-perr Lookup Protocol for Internet Aplicactions, http://es.slideshare.net/GertThijs/chord-presentation