Assignment 4 Jacob Pledger 001136676

1. When a filesystem is mounted, a vfs is created and initialized to point to that filesystem, along with the important information for that filesystem. Each vfs has a number of vnodes which are each initialized with the file handle of a remote file when it is opened, legal file operations, and other important file metadata. The device driver contains the functions that are legal for the mounted device/filesystem and a major and minor device number. These functions are loaded into a kernel-maintained table cdevsw. This table is initialized on start up and is used to access a device's functions given its major device number.

2. In a distributed system, since multiple clients will cache a file, there is a challenge to keep the cached files up to date. If multiple clients have a file open, and one client modifies it, then the other clients need to be updated at some point. Also, two clients may be editing the same file, and if one saves and closes the file first, the other client's is out of date. This creates a problem of which one the server should keep.

3. Unix semantics say that file writes by one client should be immediately available to all other clients, while session semantics say that the file is not updated for other clients until one client closes the file. The first client to close the file will have their changes sent to other clients first. Unix semantics are easily implemented in centralized systems, but difficult otherwise, since so many messages need to be sent out for each write made by any client. The benefit is that all clients always have the most up to date file. Session semantics save much more on network traffic, since updates only need to be sent after the file is closed. The problem with this is that clients may not always have the most up to date file, and if multiple clients are editing the file, then one client will have to update the file before being able to write.

4. Let e, e', v,v' be events of a process with vector clock V
   Assume $V(e) < V(e')$, so $\forall v \in V, v' \in V'$ $v < v'$
   An event $a \rightarrow b$ is represented in a vector clock as $V(a) < V(b)$. Therefore $e \rightarrow e'$ if and only if $V(e) < V(e')$

5. Since Vj[i] is an element of the vector clock of process Vj corresponding to the number of events of process Vi, Vj cannot change it, since it is only responsible for Vj[j]. Vj[i] can only be updated when Vi sends out a message to update the clock. Therefore, Vj[i] becomes equal to Vi[i] only after Vj receives a message from Vi. Vi will then update Vi[i] as it executes, making Vj[i] > Vi[i] until Vi sends out another message.

6. If the channel is not FIFO, then the sender of the marker message has no knowledge of which order the messages will be received in, and so can never know what state the receiving machine will be in when it gets the marker message. For example, if the sender sends a message and then a marker and the channel is not FIFO, the marker message could arrive before the normal message. This could cause the cut to either contain the state of the receiver before or after reception, depending on message reception order, which is inconsistent.