The UNIX fork() function is used by the operating system to take an existing process and create a child of said process. This child is an exact copy of the parent process but is stored in its own unique memory space. The reason for creating an exact duplicate is to minimize the amount of work the operating system has to do when creating a new process. The child process can either run the same code as the parent or have its code changed to execute entirely new code while still maintaining access to all the resources the parent process has. By having a parent child relationship, a standard and easy to understand hierarchical structure is established that can be easily traversed if necessary. This leads into another benefit of the fork() function and that is multiprocessing.

The parent process has the ability to create as many children as it requires (or wants to create) allowing for multiple programs to run concurrently and finish tasks in a faster manner. This can have both benefits and disadvantages. A benefit of sharing resources between the parent and child processes in multiprocessing, is that resources do not need to be duplicated and therefore memory is saved and can be utilized elsewhere. A disadvantage would be if the parent and child are trying to utilize the same resource at the same time this could lead to a battle for resources and potential corruption of those resources.

The child process has access to all the resources allocated to the parent process and includes all the environment variables, signals, counters and the current state of the parent process. These states are the following: new, running, blocked, read, suspended ready / blocked, and or terminated. After the fork() function is called whatever state the parent process is in the child will inherit that state. For example, if the parent process were in the blocked state, the child process would also be in the blocked state. Is it key to remember though that the parent process can be in any of these seven states and the child process will always mimic that state.