Zachary Meisner

2/12/22

6-1 Assignment: Memory and Storage Management

Operating Platforms

Professor Fredericks

What considerations and specific approaches would it take to ensure that memory is effectively managed in the software application, Draw It or Lose It?

When it comes to making sure memory is efficiently managed in a software application, there are many different implementations and considerations needed to make the application the most efficient. Firstly, during the development phase, we must think of the best types of patterns we can implement into the design to make everything work properly, and to make sure that we are allowing our systems to run smoothly. One way this can be done is by choosing and implementing the correct file systems. "File systems provide efficient and convenient access to the disk by allowing data to be stored, located, and retrieved easily." (Silberschatz et al., *11.1 File-System Structure* 2008) In order to get efficient and convenient access to the disk, we must solve two design problems presented by the file system within implementation during the use-case. We must think of defining the files, their attributes, and operations allowed on a file, in addition to the directory structure for organizing files, and we must implement the correct algorithms and data structures, mapping the logical file system into secondary storage devices. This can be challenging because every type of operating system is different, which poses an additional challenge for developers. One potential way to solve this issue is with the implementation of Cloud, specifically Serverless Architecture.

What considerations and specific approaches would you take to determine how much storage is needed and how to manage storage for your client's application, Draw It or Lose It?

In order to determine the amount of storage needed, we can look at the actual files and their size. In addition to this, we can consider the number of files that will need to be stored, and used or implemented within the system created, over a linear period, in order to understand the growth rate of the amount of data we may need to prepare to store in the future. What is most important is that we are managing storage per client, meaning that the way that we store the files, and the implementation of the application during runtime may change all together. One real life example can be seen within the implementation of microservice architecture. "Basically, this architecture type is a particular way of developing software, web or mobile applications as suites of independent services." (Arsov, K Microservices vs. SOA – is there any difference at all? 2021) On a case-by-case basis in this instance, we can have an elaborate number of services offered for any user to be able to access and use. This way, we can allocate enough memory based off the required, requested services, in addition to the files they create and save within the context of the runtime of the application. This way we can determine when something needs to be accessed, for example if the customer logs into the application, then we can retrieve the allotted clusters of files from storage and bring them into memory for the customer to interact with and have no issues during the session.

What are the differences in how memory and storage are used in terms of the game application functionality?

The main differences in how memory and storage are used in terms of game application functionality come with the implementations of memory and storage on a case-by-case basis. I think what is most important to consider is the architecture and design of the game, and how

players will end up using the game, in order to determine what should be memory, and what should be stored. In addition to this, one of the main distinguishing factors between memory and storage are the allocation methods, and algorithms used within the structure of the program in order to interact with the different types of data. “Three major methods of allocating disk space are in wide use: contiguous, linked, and indexed.” (Silberschatz et al., *11.4 Allocation Methods* 2008) What is most important here is the use case of each type of data in the spectrum of allocation during runtime. In the case we need to implement memory and storage working together, we need to understand how both algorithms used to filter through each case-by-case basis are going to communicate with one another. We also must implement different types of quality scans during this system so we can make sure that the output is not correct, and that the overall quality of the accessing of the data does not corrupt any of our files or interfere with either of the systems in their day-to-day processes. One example of this that I can think of is in queues while waiting for games to play with other people. When the lobby is loaded, there is often a starter map you get to run around and play in while waiting for the actual game to load. I imagine what is happening here is that the application is requesting storage to be put into memory for the number of players within the lobby, in addition to creating a network for each player to interact with one another. This helps create an efficient application where we can isolate instances of the game that will not interfere with other players in different maps, as we have loaded into our own personal server and loaded the storage into that server's memory.

Citations

Silberschatz, A., Gagne, G., & Galvin, P. B. (2008). 11.4 Allocation Methods. In *Operating System Concepts, 8th edition*. essay, John Wiley & Sons.

Silberschatz, A., Gagne, G., & Galvin, P. B. (2008). 11.1 File-System Structure. In *Operating System Concepts, 8th edition*. essay, John Wiley & Sons.

Arsov, K. (2021, May 18). Microservices vs. SOA – is there any difference at all? Medium, Retrieved February 12, 2022, from https://medium.com/microtica/microservices-vs-soa-is-there-any-difference-at-all-2ale3b66elbe