**Brandon Hobbs**

**CS-230**

**Feb. 12, 2022**

**Module Six Assignment**

Draw It or Lose it will have a database of 200, 8-MB images. The gameplay “…consists of four rounds of play lasting one minute each. Drawings are rendered at a steady rate and are fully complete at the 30-second mark.” Because the game flow is dependent on a steady framerate, some amount of memory management must be considered for the game to be a success.

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

**A steady and predictable frame-rate will be necessary for the game to be successful. This suggests that any image being currently drawn needs to be loaded into the fastest type of memory, i.e., main memory/RAM. Moreover, it would be wise to over allocate memory to prevent the MMU from trying to have part of the image in virtual memory.**

**By making sure the current image is fully loaded into RAM this should help to prevent and lagging or poor performance. Subsequent images do not need to be loaded into RAM and could be left on disk until they are ready to be drawn.**

**For games that are fully installed on a desktop there may be no reason to worry about creating a queue or cache since loading data from disk to RAM happens quickly. This is in contrast to a server-based game where the data transfer rate may vary greatly and be much slower than disk read speeds. If the application is rendered on the client, then some amount of client storage may be warranted as part of a cache.**

**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?**

**At a minimum the disc storage needs to be able to hold the image files. RAM demands would depend on how the application was built and how much was written to the stack and heap – but at a minimum one image, at least, should be loaded into the main memory.**

**If this is a server-based deployment the client and the server will have different memory needs. The server will need to store the raw program and the images. RAM demands will depend on where the program is rendered, client or server.**

**If the application is to be rendered on the server, some knowledge of the number of concurrent games must be known to prevent heavy thrashing.**

**If the application is client rendered the client needs enough storage to contain any cached images, code deployed, e.g., JavaScript Applets, or maybe even some local settings such as user preferences.**

**The decision between client and server rendering will should come down to performance.**

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

In general, main memory should be sized for performance needs and storage for the physical holding of the images and application.

Memory, i.e., RAM, needs to be optimized so that there is a consistent, performant frame rate. This would entail ensuring that there is enough main memory allocated to meet these performance targets – with limited use of virtual memory. This amount of main memory would need to be able to scale with the number of sessions if the application was going to be rendered on the server.

Storage for the application on the server would be used to hold the images and the code. Storage on the client would only need to be considered if there was some sort of cache created for future game performance or to pre-load images.