

# INFORME DEL PROYECTO FINAL

## GESTION DE MEMORIA COMBINADA “SEGMENTACIÓN PAGINADA”

Julián David Rojas Callejas

20182020153

Iván Darío López Díaz

20182020141

Universidad Distrital Francisco José de Caldas

Facultad de ingeniería, Ingeniería de sistemas

I.S., M.Sc. Oswaldo Alberto Romero Villalobos

Bogotá DC, Colombia

06 de junio de 2023

## INTRODUCCION

Actualmente la gestión de memoria es un aspecto crucial en los sistemas operativos pues permite manejar y optimizar los recursos y asimismo mejorar el rendimiento de las aplicaciones. Existen diferentes maneras de gestionar la memoria, sin embargo, hay unos más eficientes que otros. Dos de los más importantes son la segmentación y la paginación y estos a su vez fueron combinados para desarrollar lo que hoy conocemos como **segmentación paginada**. la segmentación paginada es un enfoque mucho más avanzado y sofisticado de la memoria en los sistemas operativos modernos porque aprovechan las ventajas de la segmentación y de la paginación para proporcionar un equilibrio óptimo entre la flexibilidad y la eficiencia en el uso de los recursos. para implementarse se necesita cuidado y experiencia porque de otra manera no se puede garantizar una gestión de memoria eficaz.

## **OBJETIVO GENERAL**

- Usando JS en el navegador, desarrollar un aplicativo que muestre como se gestiona un sistema de 16 MiB de RAM (24 bits) usando el modelo de gestión de memoria combinada SEGMENTACIÓN PAGINADA, donde los primeros 12 bits de la dirección virtual corresponden al número de segmento y los restantes para el desplazamiento.

## **OBJETIVOS ESPECÍFICOS**

- Implementar un algoritmo de gestión de memoria combinada segmentación paginada en JavaScript, considerando los diferentes items de memoria en segmentos y páginas, así como tablas de segmentos y páginas.
- Diseñar una interfaz gráfica utilizando HTML Y CSS para visualizar la simulación de la administración de memoria.
- Desarrollar las diferentes funcionalidades del simulador.
- Analizar y comparar los resultados obtenidos.

## **JUSTIFICACIÓN**

Ese proyecto tiene un aprendizaje práctico porque permite la simulación de la gestión de memoria combinada de una manera Diferente a que si se tratara solo de un tema teórico. Esto permite comprender mejor cómo funcionan los algoritmos y las estructuras de datos involucradas en la gestión de memoria. También es importante la realización de éste porque permite a nosotros los estudiantes experimentar con diferentes parámetros y variando los lenguajes de programación que usualmente usamos.

Por otra parte, la visualización y el análisis de los resultados facilita la comprensión de los procesos que hay tras los diferentes sistemas operativos, puede que existan muchísimos errores en el transcurso del proyecto, pero de esta manera es como se irán creando las bases.

## **MATERIALES**

1. Computadora con requisitos mínimos
2. Editor de texto o entorno de desarrollo, en nuestro caso Visual Studio Code con sus respectivas extensiones para JavaScript, HTML y CSS.
3. Bibliotecas de javascript

## **MÉTODOS**

1. Búsqueda de la información acerca de la segmentación paginada
2. Diseño de un bosquejo en Excel
3. Desarrollo de los mockups que se van a utilizar
4. Implementación del algoritmo de gestión de memoria
5. Desarrollo de la interfaz gráfica
6. Configuración de los parámetros
7. Pruebas y ajustes

## **MARCO TEÓRICO**

La gestión de memoria es muy importante porque se encarga de asignar y administrar de manera eficiente los recursos que tenemos nosotros disponibles en un sistema informático. El enfoque que estamos usando en nuestro laboratorio es el de combinación de memoria de segmentación y paginación o más conocido como segmentación paginada.

La segmentación es un método que divide la memoria en diferentes segmentos lógicos y cada uno está asociado a una parte específica del programa. a segmento está representando una unidad lógica que en otras palabras serían el código, los datos o la pila de un proceso. Esas divisiones proporcionan flexibilidad y permiten que los segmentos se puedan compartir en diferentes procesos y eso reduce la duplicación de memoria.

Por otro lado, la paginación divide la memoria en bloques de tamaño fijo y a esas las denominamos páginas y son básicamente unidades de asignación que se utilizan para dividir tanto a los segmentos como a la memoria física en trozos más pequeños y manejables. a la página se le asigna un marco de página en la memoria física lo que permite una gestión eficiente de los recursos. En la segmentación paginada cada segmento se divide en páginas del mismo tamaño y ofrece ventajas como que se mantiene la flexibilidad de la segmentación y a su vez permite que se maneje con mayor eficiencia la memoria física gracias a la paginación, ya que las páginas se pueden asignar y desasignar de forma independiente. El sistema operativo debe mantener una tabla con los segmentos y la información necesaria para traducir de direcciones de lógicas a físicas. esta tabla incluye la dirección base y el límite de cada segmento, así como las páginas asociadas a cada segmento las tablas de páginas a su vez contienen la información de traducción necesaria para cada página como la dirección base del marco de página.

## RESULTADOS

Después de investigar en muchas fuentes, logramos implementar lo que para nosotros es la segmentación paginada, tratamos de mantenernos al margen de las indicaciones que nos damos los libros y las clases y el resultado fue el siguiente:

1. Se implementó una interfaz donde inicialmente estamos mostrando una ventanilla para ingresar los diferentes datos de los procesos, seguido de una tabla que después de oprimir el botón agregar va adicionándoles procesos a ella misma y adicionalmente a la mano derecha encontramos la tabla de la cantidad de páginas necesarias para representar.

**Procesos**

nombre

.text:

.data:

.bss:

**Procesos agregados**

ID	Proceso	.text	.data	.bss	.stack	.heap	
P6	5644356	3434	3454	65536	131072		<input type="button" value="Eliminar"/>
P5	5434	344653	345344	65536	131072		<input type="button" value="Eliminar"/>
P1	54444	645644	565656	65536	131072		<input type="button" value="Eliminar"/>

**Cantidad de páginas necesarias**

ID	Proceso	.text	.data	.bss	.stack	.heap
P6	1379	1	1	16	32	
P5	2	85	85	16	32	
P1	14	158	139	16	32	

**Figura 1.** Interfaz gráfica desarrollada, se han agregado 3 procesos

2. Luego se presentan las diferentes indicaciones para acceder a las direcciones de memoria, en esta imagen podemos ver Cada uno de los segmentos los Marcos de página y las direcciones de memoria física. Al lado derecho, en color amarillo se puede observar la subdivisión del sistema operativo que en nuestro caso fue de 1 MiB.

Segmentos		Marco de página		Dirección de memoria		Diagrama
Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	
0	0	0	0	0	0	SO
1	1	1	1	4096	1000	SO
2	2	2	2	8192	2000	SO
3	3	3	3	12288	3000	SO
4	4	4	4	16384	4000	SO
5	5	5	5	20480	5000	SO
6	6	6	6	24576	6000	SO
7	7	7	7	28672	7000	SO
8	8	8	8	32768	8000	SO
9	9	9	9	36864	9000	SO
10	A	10	A	40960	A000	SO
11	B	11	B	45056	B000	SO
12	C	12	C	49152	C000	SO
13	D	13	D	53248	D000	SO
14	E	14	E	57344	E000	SO
15	F	15	F	61440	F000	SO
16	10	16	10	65536	10000	SO
17	11	17	11	69632	11000	SO
18	12	18	12	73728	12000	SO
19	13	19	13	77824	13000	SO
20	14	20	14	81920	14000	SO
21	15	21	15	86016	15000	SO
22	16	22	16	90112	16000	SO
23	17	23	17	94208	17000	SO
24	18	24	18	98304	18000	SO
25	19	25	19	102400	19000	SO
26	1A	26	1A	106496	1A000	SO
27	1B	27	1B	110592	1B000	SO
28	1C	28	1C	114688	1C000	SO
29	1D	29	1D	118784	1D000	SO
30	1E	30	1E	122880	1E000	SO
31	1F	31	1F	126976	1F000	SO
32	20	32	20	131072	20000	SO
33	21	33	21	135168	21000	SO
34	22	34	22	139264	22000	SO
35	23	35	23	143360	23000	SO
36	24	36	24	147456	24000	SO

**Figura 2.** Interfaz gráfica desarrollada, se evidencian los segmentos, marcos de página y dirección de memoria

3. Luego, podemos observar en color verde cuando empezamos a pintar los diferentes procesos en el orden de que se van asignando.



243	F3	243	F3	995328	F3000	SO
244	F4	244	F4	999424	F4000	SO
245	F5	245	F5	1003820	F5000	P6 text 4096
246	F6	246	F6	1007616	F6000	P6 text 4096
247	F7	247	F7	1011712	F7000	P6 text 4096
248	F8	248	F8	1015808	F8000	P6 text 4096
249	F9	249	F9	1019904	F9000	P6 text 4096
250	FA	250	FA	1024000	FA000	P6 text 4096
251	FB	251	FB	1028096	FB000	P6 text 4096
252	FC	252	FC	1032192	FC000	P6 text 4096
253	FD	253	FD	1036288	FD000	P6 text 4096
254	FE	254	FE	1040384	FE000	P6 text 4096
255	FF	255	FF	1044480	FF000	P6 text 4096
256	100	256	100	1048576	100000	P6 text 4096
257	101	257	101	1052672	101000	P6 text 4096
258	102	258	102	1056768	102000	P6 text 4096
259	103	259	103	1060864	103000	P6 text 4096
260	104	260	104	1064960	104000	P6 text 4096
261	105	261	105	1069056	105000	P6 text 4096
262	106	262	106	1073152	106000	P6 text 4096
263	107	263	107	1077248	107000	P6 text 4096
264	108	264	108	1081344	108000	P6 text 4096
265	109	265	109	1085440	109000	P6 text 4096
266	10A	266	10A	1089536	10A000	P6 text 4096
267	10B	267	10B	1093632	10B000	P6 text 4096
268	10C	268	10C	1097728	10C000	P6 text 4096
269	10D	269	10D	1101824	10D000	P6 text 4096
270	10E	270	10E	1105920	10E000	P6 text 4096
271	10F	271	10F	1110016	10F000	P6 text 4096
272	110	272	110	1114112	110000	P6 text 4096
273	111	273	111	1118208	111000	P6 text 4096
274	112	274	112	1122304	112000	P6 text 4096
275	113	275	113	1126400	113000	P6 text 4096
276	114	276	114	1130496	114000	P6 text 4096
277	115	277	115	1134592	115000	P6 text 4096
278	116	278	116	1138688	116000	P6 text 4096
279	117	279	117	1142784	117000	P6 text 4096
280	118	280	118	1146880	118000	P6 text 4096
281	119	281	119	1150976	119000	P6 text 4096
282	11A	282	11A	1155072	11A000	P6 text 4096
283	11B	283	11B	1159168	11B000	P6 text 4096
284	11C	284	11C	1163264	11C000	P6 text 4096
285	11D	285	11D	1167360	11D000	P6 text 4096

**Figura 3.** Interfaz gráfica desarrollada, se han empezado a pintar los procesos

4. También es posible ver un resumen de los procesos que hemos agregado de forma independiente mediante un combo box.

Mostrar resumen					
P6 ▼					
Resumen					
Numero segmento	Base Segmento	Limite Segmento	Marco página	Direccion	Proceso
0	100	4096	100	100000	PS text 4096
1	101	4096	101	101000	PS text 4096
2	102	4096	102	102000	PS text 4096
3	103	4096	103	103000	PS text 4096
4	104	4096	104	104000	PS text 4096
5	105	4096	105	105000	PS text 4096
6	106	4096	106	106000	PS text 4096
7	107	4096	107	107000	PS text 4096
8	108	4096	108	108000	PS text 4096
9	109	4096	109	109000	PS text 4096
10	10A	4096	10A	10A000	PS text 4096
11	10B	4096	10B	10B000	PS text 4096
12	10C	4096	10C	10C000	PS text 4096
13	10D	4096	10D	10D000	PS text 4096
14	10E	4096	10E	10E000	PS text 4096
15	10F	4096	10F	10F000	PS text 4096
16	110	4096	110	110000	PS text 4096
17	111	4096	111	111000	PS text 4096
18	112	4096	112	112000	PS text 4096
19	113	4096	113	113000	PS text 4096
20	114	4096	114	114000	PS text 4096
21	115	4096	115	115000	PS text 4096
22	116	4096	116	116000	PS text 4096
23	117	4096	117	117000	PS text 4096
24	118	4096	118	118000	PS text 4096
25	119	4096	119	119000	PS text 4096
26	11A	4096	11A	11A000	PS text 4096
27	11B	4096	11B	11B000	PS text 4096

**Figura 4.** Interfaz gráfica desarrollada, resumen de los procesos

5. También es posible eliminar los procesos, esto ocasiona que quede en pedazos sin llenar, por eso, cuando creamos el siguiente se van a llenar los espacios que están vacíos, si llegado el caso no alcanzan y aparece el otro proceso, pasaremos a la última posición y lo seguiremos agregando.

1526	5F6	1526	5F6	6250496	5F6000	P6 text 4096
1527	5F7	1527	5F7	6254592	5F7000	P6 text 4096
1528	5F8	1528	5F8	6258688	5F8000	P6 text 4096
1529	5F9	1529	5F9	6262784	5F9000	P6 text 4096
1530	5FA	1530	5FA	6266880	5FA000	P6 text 4096
1531	5FB	1531	5FB	6270976	5FB000	P6 text 4096
1532	5FC	1532	5FC	6275072	5FC000	P6 text 4096
1533	5FD	1533	5FD	6279168	5FD000	P6 text 4096
1534	5FE	1534	5FE	6283264	5FE000	P6 text 4096
1535	5FF	1535	5FF	6287360	5FF000	P6 text 4096
1536	600	1536	600	6291456	600000	P6 text 4096
1537	601	1537	601	6295552	601000	P6 text 4096
1538	602	1538	602	6299648	602000	P6 text 4096
1539	603	1539	603	6303744	603000	P6 text 4096
1540	604	1540	604	6307840	604000	P6 text 4096
1541	605	1541	605	6311936	605000	P6 text 4096
1542	606	1542	606	6316032	606000	P6 text 4096
1543	607	1543	607	6320128	607000	P6 text 4096
1544	608	1544	608	6324224	608000	P6 text 4096
1545	609	1545	609	6328320	609000	P6 text 4096
1546	60A	1546	60A	6332416	60A000	P6 text 4096
1547	60B	1547	60B	6336512	60B000	P6 text 4096
1548	60C	1548	60C	6340608	60C000	P6 text 4096
1549	60D	1549	60D	6344704	60D000	P6 text 4096
1550	60E	1550	60E	6348800	60E000	P6 text 4096
1551	60F	1551	60F	6352896	60F000	P6 text 4096
1552	610	1552	610	6356992	610000	P6 text 4096
1553	611	1553	611	6361088	611000	P6 text 4096
1554	612	1554	612	6365184	612000	P6 text 4096
1555	613	1555	613	6369280	613000	P6 text 4096
1556	614	1556	614	6373376	614000	P6 text 4096
1557	615	1557	615	6377472	615000	P6 text 4096
1558	616	1558	616	6381568	616000	P6 text 4096
1559	617	1559	617	6385664	617000	P6 text 4096
1560	618	1560	618	6389760	618000	P6 text 4096
1561	619	1561	619	6393856	619000	P6 text 4096
1562	61A	1562	61A	6397952	61A000	P6 text 4096
1563	61B	1563	61B	6402048	61B000	P6 text 4096
1564	61C	1564	61C	6406144	61C000	P6 text 4096
1565	61D	1565	61D	6410240	61D000	P6 text 4096
1566	61E	1566	61E	6414336	61E000	P6 text 4096

**Figura 5.** Interfaz gráfica desarrollada, eliminación e inserción procesos

## CONCLUSIONES

La gestión de memoria combinada segmentación paginada es una de las más eficientes para manejar los recursos que tenemos disponibles en los sistemas operativos, es un método que recopila las ventajas de la segmentación y la paginación y nos permite más flexibilidad y mejor administración.

La segmentación facilita la traducción de direcciones virtuales a direcciones físicas mediante las tablas de segmentos y las tablas de páginas. Y a pesar de las muchas ventajas que tiene, es importante tener en cuenta que la gestión de memoria de segmentación paginada puede traer

cierta complejidad para implementarse y en y en la traducción de las direcciones, se tiene que traducir correctamente para garantizar el funcionamiento correcto del sistema.

Es muy importante empezar a comprender y aplicar los conceptos aprendidos en la materia porque si nos damos cuenta, actualmente nuestra sociedad es dependiente de cualquier sistema operativo.

A pesar de que la experiencia en programación ya es suficiente, sí costó un poco más adaptarnos a la parte web y empezar a desarrollar el gestor de memoria. es verdad que javascript es un lenguaje de programación muchísimo más flexible, sin embargo, para la corrección de errores es un poco más complejo pues no se sabe exactamente dónde se ha situado.

## **REFERENCIAS**

Stallings, W. (2018). Operating Systems: Internals and Design Principles. Pearson.

Tanenbaum, A. S., & Bos, H. (2014). Modern Operating Systems. Pearson.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating System Concepts. Wiley.

Abraham, S. S. (2015). Operating Systems. PHI Learning Pvt. Ltd.