Universidad de Guadalajara.

Centro Universitario de Ciencias Exactas e Ingenierías.

Programación de Sistemas

Maestro: Valentín Martínez López

Sección D06

Arenas de la Cruz Jesús Antonio.

302098601

CARGADOR - LIGADOR

CLASES DE CARGADOR.

Un cargador es la parte de un sistema operativo que es responsable de cargar programas en memoria, usualmente es una parte del núcleo del sistema operativo y es cargado al iniciar el sistema. En algunos sistemas operativos que tiene núcleo paginable pueden tener cargador en una parte paginable de la memoria.

Existen sistemas operativos especializados que no contiene cargador, y el manejador actúa como tal. Algunos sistemas utilizan cargadores relocalizadles, los cuales ajustan direcciones de memoria en un ejecutable para compensar las variaciones en la cual la memoria disponible de la aplicación comienza.

Muchos cargadores permiten al usuario especificar opciones, que modificar el procesamiento estándar descrito, en ciertos sistemas el programador puede incluso introducir proposiciones de control del cargador en el programa fuente, y el ensamblador o el compilador retienen esos mandatos como parte del programa objeto.

CARGADOR

Un cargador es un programa del sistema que realiza la función de carga, es normalmente un programa pequeño que permite al usuario entrar directamente las palabras de instrucción y de datos a direcciones concretas de la memoria. Consiste en un juego de instrucciones que permiten al dispositivo de entrada asignar la dirección de inicio de la memoria y asegurar que el computador leerá el programa y lo cargara byte a byte. La ligadura de tales recursos puede ser estática o dinámica.

Esto nos lleva al siguiente proceso:

Carga: Lleva el programa objeto a la memoria para su ejecución.

Relocalización: Modifica el programa objeto de forma que pueda cargarse en una dirección diferente de la localidad especificada originalmente.

Ligado: Combina dos o más programas objeto independientes y proporciona la información necesaria para realizar referencias a ellos.

LIGADOR

Realiza la operación de enlazar programas objetos independientes. Los editores de ligado pueden efectuar funciones además de la simple preparación de un programa objeto para su ejecución, también para construir paquetes de subrutinas u otras secciones de control. En general tienden a ofrecer mayor flexibilidad y control con el correspondiente incremento de complejidad y sobrecarga. La tarea primaria del enlazador es resolver referencias externas lleva a cabo la siguiente etapa del proceso de traducción enlazando los módulos ensambladores y los acervos para formar un programa completo.

Sus funciones son:

- Enlaza r el código intermedio compilado independiente en un solo modulo resolviendo las diferencias entre Tokens.
- Reunir procedimientos traducidos por separado y enlazarlos para que se ejecuten como una unidad llamada programa binario ejecutable.

Tipos de cargadores.

ARRANQUE.

Algunos computadores requerían que el operador introdujera en la memoria el código objeto de un cargador absoluto, utilizando los interruptores en la consola del computador, este proceso era incomodo y propenso a errores.

El cargador de arranque proporciona la posibilidad de cargarse las rutinas solo cuando y si se necesiten. Si las subrutinas son grandes o tienen muchas referencias externas se pueden conseguir ahorros considerables de tiempo y espacio de memoria.

BOOTSTRAP

El cargador se cargará en la memoria por los siguientes métodos:

Entrada manual: Mediante el teclado el usuario tecle el cargador BOOTSTRAP.

Entrada por ROM: Las instrucciones de inicialización están almacenadas en la ROM, Cuando se requiere el BOOTSTRAP, el operador ejecuta las instrucciones memorizadas en ROM, así se elimina también la posibilidad de borrados accidentales.

CARGADORES CON REUBICACÓN.

Un mismo programa necesita ejecutarse en diferentes posiciones de memoria. Por esto la traducción debe estar realizada de forma correcta utilizando referencias de una dirección especial llamada de reubicación.

CARGADORES ABSOLUTOS.

El cargador pone en memoria las instrucciones sin dichas instrucciones se almacena siempre en el mismo espacio de memoria, se dice que es un cargador absoluto.

CARGADORES INICIALES.

Indican a la computadora la forma de poner, dentro de la memoria principal unos datos que están guardados en un periférico de memoria externa, cargan en la memoria pequeños programas que inician el funcionamiento de una computadora.

CARGADORTES LIGADORES (IINKER).

Consiste en añadir al programa objeto obtenido a la traducción las rutinas externas a las que hace referencia dicho programa. El ensamblador debe permitir dichas referencias y las rutinas deben estar a su vez en lenguaje máquina guardadas en algún elemento accesible para el montador.

DIFERENCIA FUNDAMENTAL ENTRE UN EDITOR DE LIGADO Y UN CARGADOR LIGADOR.

Un cargador tiene como función principal la de subir un programa objeto que se encuentra en almacenamiento secundario a la memoria para que pueda ser ejecutado, si el programa tiene definidas algunas referencias externas, entonces es cuando entra el proceso de liga. El proceso de carga puede ser absoluto o relocalizadle y el proceso de liga puede ser estático o dinámico.

Un cargador ligador realiza todas las operaciones de ligado y relocalización, y carga el programa ligado directamente en la memoria primaria para su ejecución. Un editor de ligado produce una versión ligada del programa que se escribe en un archivo o biblioteca para su ejecución posterior.

El editor de ligado realiza la relocalización de todas las secciones de control al inicio del programa ligado. Los editores de ligado se pueden utilizar para construir paquetes de subrutinas u otras secciones de control.

LIGADO DINAMICO

Consiste en enlazar en tiempo de ejecución los módulos que contienen las subrutinas, carga las rutinas solo si son necesarias, evita la necesidad de cargar la biblioteca completa para cada ejecución, hace innecesarios que el programa conozca el conjunto de subrutinas que se pueden utilizar.

Este método también podría considerarse como una solicitud a una parte del cargador que se mantiene en la memoria durante la ejecución del programa. La asociación de dirección real y el nombre simbólico de la rutina llamada no se hace hasta que se ejecuta la proposición llamada.

Tablas y lógica del cargador ligador.

La entrada de este cargador consta de un conjunto de programas objeto que se van a ligar. Una sección puede hacer una referencia externa a un símbolo cuya definición aparece más delante en este flujo de entrada.

La principal estructura de datos necesaria para el cargador ligador es una tabla de símbolos externos TABSE (análoga TAMBISIM), se usa para almacenar el nombre y la dirección de los

símbolos externos, también suele indicar en qué sección de control se define el símbolo. Entre las variables importantes podemos encontrar DIRPROG (dirección y carga del programa) y DIRSC (dirección de la sección de control).

Cuando se utiliza ligado dinámico, la asociación de una dirección real y el nombre simbólico de la rutina llamada no se hace hasta que se ejecuta la proposición llamada.

CARGADOR CYBER

Suelen contener mucho más valores re localizables que los programas de VAX o del sistema /370. Una palabra de CYBER puede contener más de una instrucción, por lo que no es posible usar un solo bit de relocalización por palabra. A causa de que en CYBER sólo las instrucciones de 30 bits pueden contener direcciones de memoria, existen cinco posibles de valores relocalizables dentro de una palabra.

Sin relocalización.

Valor re localizable sólo en la mitad superior de la palabra.

Valor re localizable sólo en la mitad inferior de la palabra.

Valores re localizables en las mitades superior e inferior de la palabra.

Valor re localizable en la mitad de los 30 bits de la palabra.

Cuando se usa la técnica de la máscara de bits, hay un campo de cuatro bits asociado a cada palabra de código objeto. Esos cuatro bits se utilizan para codificar las posibles antes listadas, y también para especificar si la dirección base del programa se suma o resta a cada valor relocalizable.

El cargador de CYBER puede utilizar programas de superposiciones de un tipo más restringido que el descrito. Una estructura de superposiciones está limitada a un máximo de tres niveles. Cada segmento está identificado por un par ordenado de enteros, y un segmento solamente puede tener un punto de entrada.

Cada segmento, excepto el raíz, debe cargarse por medio de una solicitud explicita; no existe la carga automática de segmentos. El programa de aplicación puede solicitar la carga de un segmento por medio de una llamada de servicio al sistema operativo. Como alternativa, en el segmento raíz puede haber un pequeño cargador residente para manejar el proceso de superposición.

Hay también una opción más poderosa que las superposiciones: la segmentación. Un programa segmentado también usa una estructura de árbol; sin embargo, puede haber más de tres niveles, y cada segmento puede tener varios puntos de entrada.

EJEMPLO DE EDITOR DE LIGADO 370

El formato de los programas objeto manejado por el editor de ligado del Sistema/370 es muy parecido al analizado para SIC/XE.

El programa de salida del editor de ligado se llama módulo de carga, y puede cargarse en la memoria para su ejecución, y suele contener suficiente información para permitir que el editor de ligado los reprocese. El usuario tiene la posibilidad de especificar que un módulo de carga sea “no editable”, en cuyo caso puede omitirse gran parte de la información de control, para producir un módulo de carga más pequeño.

El editor de ligado 370 puede manejar programas de superposiciones, con diversas características extendidas. En cada segmento puede haber varios puntos de entrada. Para manejar esta característica, el editor de ligado crea automáticamente tablas de entradas en todos los segmentos que contienen una llamada que puede causar superposiciones.

El proceso de superposiciones tiene lugar automáticamente al llamar a un segmento.

La mayoría de los sistemas 370 disponen de un cargador ligador, así como de un editor de ligado. El cargador ofrece menos opciones y posibilidades que las que presenta el editor de ligado. Si no se necesitan las funciones especializadas del editor de ligado, por razones de eficiencia se recomienda utilizar el cargador.

EJEMPLOS DE EDITOR DE LIGADO VAX.

El ligador VAX es un editor de ligado que realiza las mismas funciones básicas alcanzadas con anterioridad. La acción del ligador en la creación de las secciones de imagen está controlada por ensamblador o compilador por medio de una secuencia de mandatos que forman parte del programa objeto.

El lenguaje de mandatos ofrece una gran diversidad de posibilidades: hay más de 50 códigos de mandatos posibles. El ligador VAX puede generar tres tipos de imágenes. Una imagen ejecutable es aquella adecuada para la carga y ejecución; sin embargo, el ligador no puede reprocesar este tipo de imagen. Una imagen compartible no es ejecutable, pero el ligador puede reprocesarla, y se puede utilizar, por ejemplo como tapa intermedia en el ligado de un programa muy grande. Las imágenes compartibles también hacen posibles que diferentes

Programas compartan la misma copia de ciertas instrucciones o área de datos. El tercer tipo de imagen que puede crear el ligado es una imagen de sistema, concebida para ser ejecutada directamente en la máquina VAX.

RELACION CARGADOR –SISTEMA OPERATIVO.

La memoria es un recurso de tipo apropiable, que debe ser administrado para obtener su mejor provecho. En esta Unidad se verá la administración de memoria en sistemas multiprogramados, es decir, sistema en donde coexisten varios procesos en memoria.

En general, podemos distinguir dos esquemas de administración de memoria para sistemas multiprogramados, uno dice que los procesos deben estar cargados completamente en memoria para poder ejecutarse y el otro esquema no requiere que el proceso completo este cargado en memoria.

Un conjunto de procesos comparte el procesador y este es administrado con el fin de mejorar su utilización, para conseguir esto, es necesario mantener varios procesos en memoria, es decir, compartir la memoria (principal, primaria, real, RAM, rwm).

El "administrador de la memoria" es la parte del sistema operativo que se encarga de la memoria. Los sistemas de administración se pueden dividir en dos clases:

1. Aquellos que sólo trabajan con la memoria principal.

2. Aquellos que mueven procesos entre memoria principal y el disco, durante la ejecución.

Las estrategias de la administración del almacenamiento se dividen en las siguientes categorías.

- Estrategias de obtención
- Por demanda
- Anticipada
- Estrategias de colocación
- Estrategias de reemplazo

La memoria es un recurso utilizado para el almacenamiento de las instrucciones que forman un proceso. El ciclo típico de ejecución de instrucciones incluye la transferencia de instrucciones de memoria a CPU, decodificación de instrucciones, búsqueda de operandos y ejecución efectiva de las instrucciones, posteriormente los resultados también pueden ser almacenados en memoria.

Cuando se ejecuta un proceso, este debe ser cargado en memoria previamente. Anteriormente, cuando es un programa binario ejecutable está almacenado en memoria secundaria, típicamente disco.

Cuando un proceso entra en ejecución se utiliza un proceso del sistema que se denomina cargador, que lo transfiere a memoria. En Unix por ejemplo, el cargador almacena el proceso en memoria distinguiendo en él tres segmentos distintos:

- Segmento de código
- Segmento de Datos
- Segmento de Pila

Para ejecutar un proceso, este debe cargarse en memoria. Generalmente la cola de entrada es un conjunto de procesos que están en disco esperando entrar en memoria, uno de los procesos es seleccionado desde esta cola de entrada y es cargado en memoria. En ocasiones se debe relocalizar las direcciones o los enlaces de referencia externas a puntos de entrada. Al finalizar la ejecución, el área de memoria que ocupaba se declara disponible.

La asignación de memoria en los segmentos es responsabilidad de diferentes etapas en el sistema.

Compilación

Enlazado (linker)

Carga (loader)

Asignación de memoria en tiempo de ejecución (run time)

Antes de hablar de cada una de estas etapas menciones algunos aspectos generales que son necesarios de tener presente. Un programa fuente puede estar formado por uno o un conjunto de archivos en donde se definen indistintamente.

Las variables, funciones y/o procedimientos. Por ejemplo, en C, se usa el encabezado externo para indicar que una función definida en un archivo puede ser utilizada por otro archivo después de su declaración correspondiente.

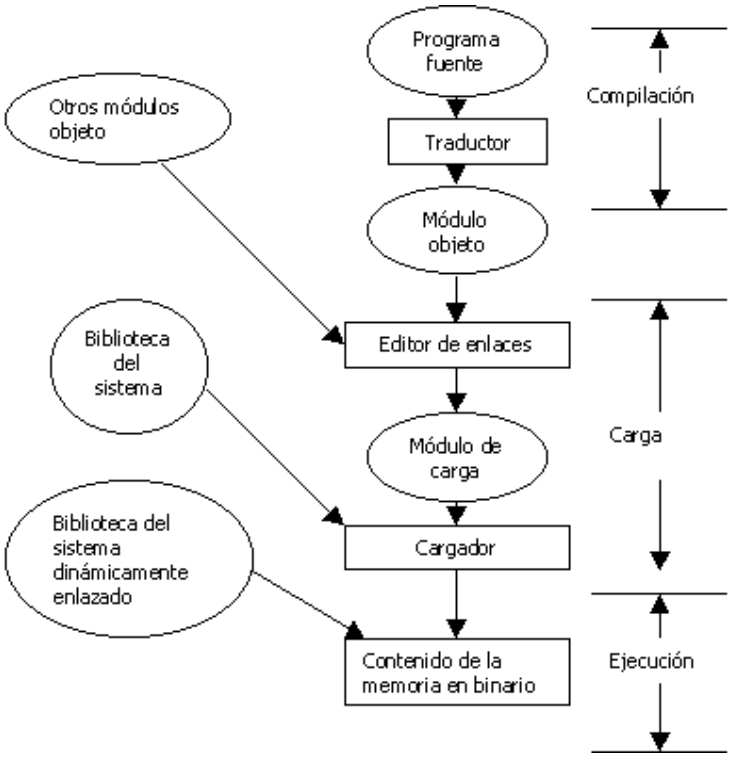
En el caso que un programa este definido mediante un conjunto de archivos fuentes, el compilador genera el código objeto de cada uno de los archivos correspondientes. Luego es el enlazador el encargado de generar un solo archivo ejecutable a partir de todos los módulos objetos. Finalmente el cargador es el encargado de cargar en memoria el archivo ejecutable.

Compilación: Durante el proceso de compilación se generan los archivos objetos de cada uno de los archivos fuentes del programa. Las referencias externas no son resueltas en esta etapa.

Enlazado: En esta etapa se reagrupan todos los archivos objetos pertenecientes al programa y se genera un solo archivo ejecutable. Las referencias externas son resueltas así como las llamadas a funciones de biblioteca del sistema.

Carga: Durante este proceso se carga en memoria el programa ejecutable. Este proceso se realiza cuando se ejecuta el programa. Esta carga puede permitir la compartición de código ejecutable entre diferentes procesos.

Ejecución: Durante la ejecución el proceso se moverá de un lugar a otro de la memoria, el enlace final se deberá postergar hasta el momento de la ejecución o run-time. Para que este esquema funcione, se debe tener hardware especial.



SISTEMAS OPERATIVOS

El sistema operativo es el programa que controla todos los recursos del computador y ofrece el soporte básico sobre el que pueden escribirse los programas de aplicación, realiza dos funciones básicamente no relacionadas extendiendo la maquina y gestionando los recursos.

La clasificación de los sistemas operativos se basa en el número de usuarios que el sistema puede atender al mismo tiempo. También se puede tomar en cuenta el tiempo de acceso que proporciona al usuario (ejemplo: por lotes y multiusuario.)

FUNCIONES DEL EDITOR DE ENLACE.

Los programas suelen componerse de más de un procedimiento, los cargadores y compiladores cargan el resultado de la traducción de un procedimiento en la memoria secundaria. Posteriormente para que este programa pueda ejecutarse todos estos resultados tienen que ligarse y recuperarse de forma correcta. Los programas que realizan la función de ligado se les conoce como cargador, cargador montado y editor de enlace.

La traducción de un programa objeto es en dos pasos:

Compilación y ensamblaje de los procedimientos fuente.

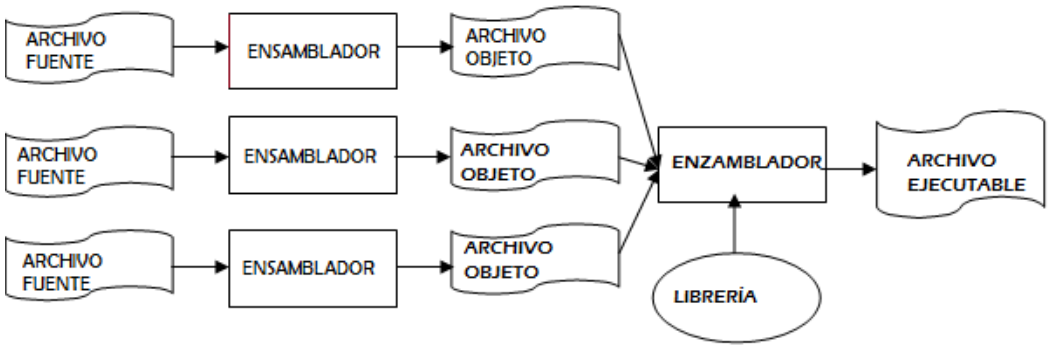
Encadenamiento o montaje de los módulos objeto.

La tarea principal del enlazador es resolver las referencias, establecer la correspondencia entre símbolos definidos en un archivo objeto y su empleo en otro archivo.

Los pasos necesarios para ejecutar el editor de enlace son:

1. Probar directrices: directivas e instrucciones.
2. Referencias externas: Establecer la correspondencia entre símbolos definidos de un archivo objeto y su empleo en otro archivo.
3. Abrir archivos objeto y formatear datos.
4. Combinar archivos objeto.
5. Enlazar código compilado.
6. Incorpora rutinas con librerías en caso de solicitarlo el propio programa.
7. Reducir procedimientos traducidos por separado y enlazarlos para su ejecución.

En general tienden a ofrecer una gran flexibilidad y control con el correspondiente incremento de complejidad y sobrecarga.



EDITOR DE LIGADO

Ligadura de una subrutina: El enlace de una subrutina es la estructura con que se comparte la información del involucrado, el involucrado proporciona la dirección como parte del enlace de subrutina. Estas se encuentran almacenadas en librerías, es aquí donde acude el programa a buscarlas y la adjunta al programa objeto, a este proceso se le llama ligadura de rutinas y subrutinas.

Las razones por las cuales se debe dividir un programa en subprogramas son las siguientes:

- Vinculación de lenguajes
- Facilitar el desarrollo de proyectos largos.
- Incrustar partes de un programa durante su ejecución a causa del gran tamaño del programa.

PROBLEMAS DE LIGA DE OBJETOS Y MÉTODOS DE SOLUCION.

El problema que presenta es el de la liga de referencias adelantadas externas el cuan consiste en que el cargador haga referencia a un símbolo de un modulo externo cuya definición aparece más adelante en el flujo de entrada. Por tal motivo la operación de ligado no se puede hacer hasta haber asignado una dirección al símbolo externo implicado en la liga, y la solución son los siguientes pasos:

1. Combinación de bloques de memoria de código y de datos: En la memoria estarán intercalados bloques de datos y de código.
2. Construcción de un Acervo: Una tabla que contenga todos los símbolos requeridos para la ejecución de un programa.
3. Lista de partes exportables públicas y privadas incluyendo un desplazamiento: Se especifican las partes de un modulo que podrán exportarse y que partes son públicas y privadas.

Gracias al enlazado el programa se puede dividir en módulos, que pueden ensamblarse por separado y enlazarse en una ocasión posterior, puede ser de naturaleza dinámica o estática. El estático da como resultado, un archivo ejecutable con todos los símbolos y módulos.

El cargador dinámico se utiliza para cargar dinámicamente bibliotecas compartidas durante el inicio ejecutable. Un cargador dinámico es útil cuando no es posible asignar en memoria un programa completo, dicho cargador se basa en el binder para operar de esta forma. Cada estructura se coloca dinámicamente en memoria. En este esquema cada módulo se va intercalando en memoria conforme se requiere, ese es el concepto de cargador dinámico.

El ligado dinámico ofrece algunas ventajas sobre los otros tipos de ligado. Proporciona la posibilidad de cargar las rutinas sólo cuando y si se necesitan. SI las subrutinas son grandes o tienen muchas referencias externas, se pueden conseguir ahorros considerables de tiempo y espacio de memoria. De forma similar, supóngase que en cualquier ejecución un programa usa sólo pocas de una gran cantidad de subrutinas posibles, pero el número exacto de rutinas necesarias no puede predecirse hasta que el programa examina su entrada. Esta situación podría presentarse, con un programa que permita al usuario llamar interactivamente a cualquiera de las subrutinas de una gran biblioteca matemática y estadística. El usuario podría suministrar la entrada de datos desde un terminal de tiempo compartido, y los resultados podrían exhibirse en el terminal. En este caso podrían ser necesarias todas las subrutinas de la biblioteca, pero en cualquier sesión de terminal solo se usarían unas cuantas. El ligado dinámico evita la necesidad de cargar la biblioteca completa para cada ejecución. El ligado dinámico puede incluso hacer innecesario que el programa conozca el conjunto de subrutinas que se podría utilizar. El nombre de la subrutina se trataría simplemente como otro elemento de entrada. Para realizar la carga de ligado de una subrutina llamada se puede utilizar varios mecanismos distintos. En el método que se analiza aquí, las rutinas que se carguen dinámicamente deben llamarse por medio de una solicitud de servicio al sistema operativo. Este método también podría considerarse como una solicitud a una parte del cargador que se mantiene en la memoria durante la ejecución del programa. Cuando se utiliza ligado dinámico, la asociación de una dirección real y el nombre simbólico de la rutina llamada no se hace hasta que se ejecuta la proposición llamada.

DISEÑO Y PROGRAMACION DE UN CARGADOR.

Muchos cargadores permiten al usuario especificar opciones que modificar el procesamiento estándar descrito. Muchos cargadores tienen un lenguaje especial de mandatos que se utiliza para especificar opciones. Algunas veces existe un archivo independiente de entrada al cargador que contiene esas proposiciones de control. En ocasiones esas mismas proposiciones también pueden estar intercaladas en el flujo primario de entrada entre los programas objeto. En ciertos sistemas el programador puede incluso introducir proposiciones de control del cargador en el programa fuente, y el ensamblador o el compilador retienen esos mandatos como parte del programa objeto.

Una opción del cargador permite la selección de fuentes alternativas de entrada, por ejemplo el mandato INCLUDE, puede indicar al cargador que lea el programa objeto designado en una biblioteca y que lo trate como si fuera parte de la entrada primaria del cargador. Otros mandatos permiten al usuario eliminar símbolos externos o secciones de control completas. También es posible cambiar referencias externas dentro del programa que se está cargando y ligando. Por ejemplo, el mandato DELETE, puede indicar al cargador que elimine la sección de control nombrada del conjunto de programas que se está cargando. El mandato CHANGE puede hacer que el símbolo externo nombre1 se cambie a nombre2 siempre que aparezca en los programas objeto. Otra opción común para el cargador implica la inclusión automática de

rutinas de biblioteca para satisfacer referencias externas. La mayoría de los cargadores permiten al usuario especificar bibliotecas alternativas para búsqueda, por medio de una proposición del tipo LIBRARY MILIB. Suele buscar en esas bibliotecas especificadas por el usuario antes que en las bibliotecas estándar del sistema. Esto permite al usuario utilizar versiones especiales de esas rutinas estándar.

Los cargadores que realizan la búsqueda automática en bibliotecas para satisfacer referencias externas, a menudo permiten al usuario especificar que no se resuelvan de esa forma algunas referencias. Si se sabe que el análisis estadístico no se va a realizar en una ejecución determinada de este programa, el usuario puede incluir un mandato como NOCALL DEVSTD, PLOT, CORREL para indicar al cargador que no se resuelvan esas referencias externas, evitando así el trabajo extra de cargar y ligar rutinas innecesarias, con el consiguiente ahorro del espacio de memoria que se requeriría.

También se puede especificar que no se resuelva ninguna referencia externa por búsqueda en biblioteca, aunque eso daría como resultado un error si el programa intenta hacer esa referencia externa durante la ejecución. Esta opción es más útil cuando se van a ligar programas, pero no se van a ejecutar de inmediato. En tales casos suele ser conveniente posponer la resolución de referencias externas.

Los cargadores a menudo incluyen otras opciones. Una de las tales opciones es la posibilidad de especificar la localidad donde se inicia la ejecución. Otra es la posibilidad de controlar si el cargador debe intentar o no la ejecución del programa si se detectaran errores durante la carga.

ALGORITMO DE CARGADOR.

1. Colocar un programa objeto en la memoria.
2. Iniciar su ejecución.
3. Si tenemos un cargador que no necesita realizar las funciones de ligado y relocalización de programas, su operación es simple pues todas las funciones se realizan en un solo paso.
4. Se revisa el registro de encabezamiento para comprobar si se ha presentado el programa correcto para la carga (entrando en la memoria disponible).
5. A medida de que lee cada registro de texto, el código objeto que contiene pasa a dirección de la memoria indicada.
6. Cuando se encuentra el registro de fin, el cargador salta a la dirección especificada para iniciar la ejecución del programa cargado.

Cargador absoluto.

Existes dos variables importantes en el cargador absoluto, dir_byte que se declara como la dirección de un carácter, contiene la dirección de un byte de memoria. A esta variable en cada iteración se le asigna el siguiente byte del archivo ejecutable.

Cargador relocalizable.

Un espacio lógico de direcciones representa el espacio desde la perspectiva del programa, el ensamblador y el enlazador usan el espacio lógico de direcciones para completar sus etapas del proceso de traducción. El espacio físico de direcciones representa la perspectiva de la memoria de máquina, el cargador completa el proceso de traducción estableciendo la correspondencia de las direcciones del programa entre el espacio lógico de direcciones ye l espacio físico

Cuando se utiliza la relocalización dinámica los programas relocalizar en cualquier momento incluso después de haber iniciado su ejecución. Para la relocalización estática se requiere que el enlazador proporcione información de relocalización adicional en el archivo ejecutable, consiste en una lista de parches que indican cuales son las posiciones del programa. La relocalización dinámica es más rápida y flexible que la estática. Esta establece la correspondencia entre las direcciones usadas en el programa y direcciones físicas cada vez que se utilizan durante la ejecución del programa.

RELOCALIZACION.

Los cargadores que permiten relocalización de programas se denomina cargadores re localizadores o relativos. Se utiliza un registro de modificación para describir cada parte del código que se ha de cambiar al relocalizar el programa. Depende en gran cantidad de las características de la maquina.

Bibliografía.

- ANÁLISIS, DISEÑO Y PROGRAMACIÓN DE SISTEMAS. PRINCIPIOS DE FUNCIONAMIENTO DEL SOFTWARE DE SISTEMAS
- SISTEMAS OPERATIVOS WILLIAM STALLINGS PDF
- SISTEMAS OPERATIVOS MODERNOS TANENBAUN- ESPAÑOL PDF
- http://www.mitecnologico.com/Main/LigadoresDinamicos
- http://www.monografias.com/trabajos5/carli/carli.shtml#cyber
- http://balvi.galeon.com/#_Toc137619486
- williambader.com/museum/vax/vaxhistory.html
- http://148.202.148.5/cursos/cc206/programaciondesistemas_3_2.htm