6
Sistema de Gerenciamento de Memória

Os sistemas de gerenciamento de memória baseiam-se no princípio de que a quantidade de dados necessária para realizar uma operação pode ser completamente armazenada na unidade de armazenamento desejada. Com base nesse princípio, foram criadas as operações de carregamento e liberação de dados. Com as operações de carregamento, o sistema carrega os dados requisitados pela aplicação para a unidade de armazenamento desejada. Quando essa unidade fica cheia, o sistema utiliza a operação de liberação para remover os dados que não são mais necessários.

Este trabalho desenvolveu um sistema de gerenciamento de memória com o objetivo de lidar com objetos gráficos 2D. Esses objetos devem ser definidos sobre um conjunto de suportes paramétricos retangulares. A arquitetura e as operações básicas do sistema foram inspiradas no modelo de memória virtual. Assim, o leitor deve estar familiarizado com o conceito, estrutura de dados e funcionamento desse modelo. O modelo de memória virtual é discutido em detalhes no Apêndice A.

O funcionamento do sistema consiste em alocar parte dos recursos de armazenamento e gerenciá-los de forma otimizada para as operações de visualização. Para isso, explora-se a estrutura de dados utilizada para representar o objeto gráfico e as operações de movimento de câmera realizadas durante o processo de visualização.

A arquitetura do sistema foi desenvolvida para lidar com vários estágios de armazenamento. A quantidade de estágios será determinada pelas características da aplicação, do hardware e pela da quantidade de dados. Os estágios de armazenamento podem ser locais ou remotos. Se a aplicação acessa os dados que estão armazenados remotamente, então é necessário que o estágio de rede seja adicionado no sistema. Caso contrário esse estágio não precisa ser adicionado. Os estágios locais como memórias de textura, memórias RAM e discos, são adicionados dependendo da quantidade de dados existente e da quantidade de espaço disponível em cada

estágio. Por exemplo, se é possível armazenar todos os dados na memória RAM, então não é necessário adicionar o estágio de disco no sistema.

6.1 Arquitetura do sistema

A arquitetura do sistema de gerenciamento de memória foi inspirada no modelo de memória virtual. O sistema de memória virtual foi escolhido como modelo devido às vantagens que ele oferece, veja o Apêndice A. Esse modelo sofreu modificações com objetivo de permitir que o sistema suportasse vários estágios de armazenamento.

A modificação é baseada na generalização do conceito de memória primária e secundária. De acordo com esse conceito, uma memória primária é todo tipo de estágio de armazenamento que tem taxa de transferência mais rápida e menor capacidade de armazenamento do que o estágio que está um nível abaixo na hierarquia. O estágio mais lento e com maior capacidade de armazenamento é considerado como a memória secundária, a Figura 6.1 mostra a relação entre os dispositivos de armazenamento.

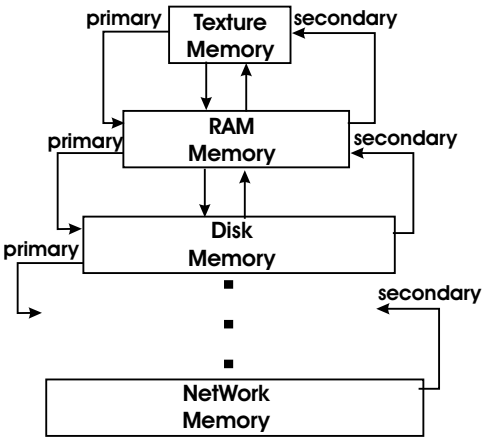

Figura 6.1: Hierarquia dos dispositivos de armazenamento e o conceito de dispositivo primário e secundário.

Os principais módulos do sistema são os seguintes:

– **Estágios de Armazenamento**: Representa os dispositivos de armazenamento. Esse módulo especifica uma interface comum para as operações de escrita, leitura e alocação de dados nos dispositivos. Essa interface permite que qualquer tipo de dispositivo de armazenamento seja inserido no sistema.

  - **Unidade de Transferência de Dados**: É responsável pela transferência de dados entre os estágios de armazenamento. Esse módulo define a forma como os dados são transferidos. Os dados podem ser transferidos de forma assíncrona ou síncrona. Na forma assíncrona o dispositivo mais rápido não espera que um pedido de carregamento seja atendido para fazer o novo pedido. Na forma síncrona ocorre o inverso, um novo pedido de carregamento só é feito quando o pedido anterior foi completado.

- **Sistema de Paginação**: Compreende as operações de carregamento, liberação e tratamento de ausência de páginas. Cada estágio de armazenamento possui seu próprio módulo de paginação. Isto permite associar algoritmos de paginação diferentes para cada estágio de armazenamento.
- **Interface**: Trata da comunicação entre a aplicação e o sistema de gerenciamento de memória. Essa interface permite realizar três operações: operação de sincronismo, operação de acesso aos dados e operações de troca de informações. Com a operação de sincronismo a aplicação informa ao sistema quando a operação de predição deve ser realizada e quanto tempo deve levar esta predição. A operação de acesso aos dados permite que a aplicação acesse os dados dos ladrilhos do objeto gráfico. A operação de troca de informação é utilizada para a aplicação fornecer dados que são utilizados na fase de predição.

A Figura 6.2 mostra os módulos que compõem o sistema e a forma como eles estão interligados.

A aplicação inicia o sistema de gerenciamento informando a quantidade de memória que deve ser alocada em cada estágio de armazenamento. Após o processo de alocação, o sistema divide essas regiões de armazenamento em blocos de mesmo tamanho. Os tamanhos de blocos de memória são determinados de acordo com o tamanho dos dados armazenados pelos ladrilhos. Uma página de dados é composta por um número finito de blocos. A página é a menor unidade utilizada nas operações de manipulação de dados dentro do sistema de gerenciamento de memória.

Os estágios de armazenamento possuem uma tabela de páginas. Essa tabela é utilizada para armazenar os mapeamentos entre endereços lógicos e os endereços físicos e o estado desses mapeamentos. O último estágio de armazenamento não possui tabela de páginas porque é capaz de armazenar todos os dados que definem o objeto gráfico.

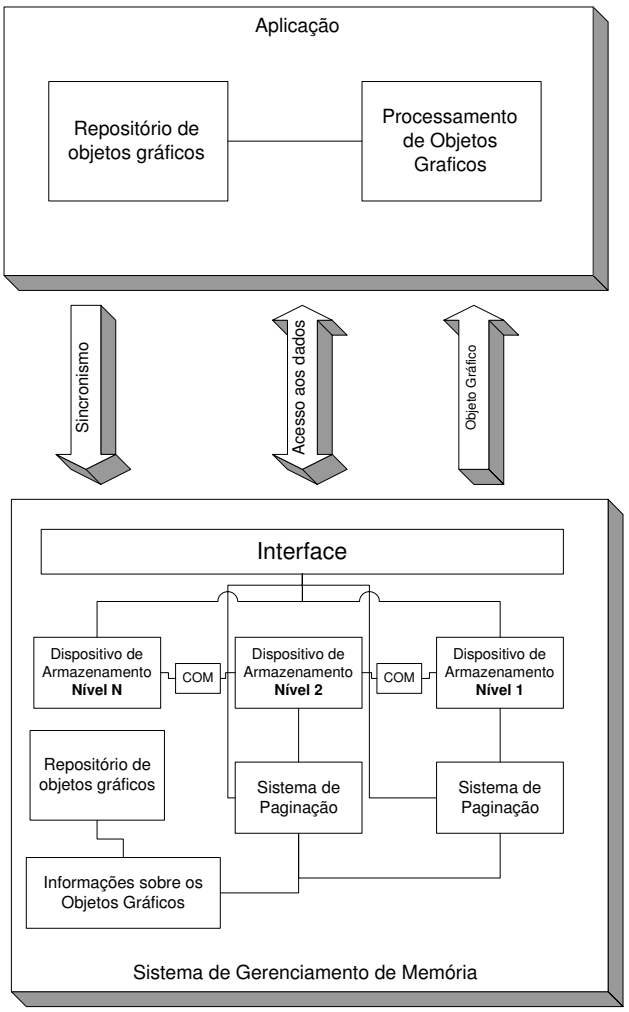

Figura 6.2: Arquitetura do sistema de gerenciamento de memória.

O processo de gerenciamento de memória é iniciado quando a aplicação pede ao sistema para acessar um objeto gráfico. Esse pedido é feito informando o nome do arquivo onde os dados estão armazenados. Após receber o pedido, o sistema constrói a estrutura em multi-resolução geométrica e de textura do objeto gráfico. Em seguida, o sistema cria um espaço de endereçamento lógico para o objeto gráfico e o passa para a aplicação. A aplicação acessa os dados de cada ladrilho a partir desse sistema de endereçamento. Na próxima seção será discutido com detalhes como é criado esse espaço de endereços.

O sistema de paginação funciona de acordo com o mesmo princípio do sistema de paginação sob-demanda. Ou seja, uma página de dados só é carregada quando ela for necessária e uma página só é liberada quando o estágio de armazenamento estiver cheio. A principal diferença

está nos algoritmos utilizados para o carregamento, liberação e tratamento de páginas ausentes. O sistema de gerenciamento de memória desenvolvido neste trabalho utiliza a técnica de paginação preditiva. Essa técnica desenvolve algoritmos de carregamento e liberação mais específicos, pois a sua implementação leva em consideração a natureza dos dados, como esses dados estão estruturados e como são acessados pela aplicação. Os sistemas que utilizam as técnicas de carregamento preditivo são chamados de *sistemas de gerenciamento de memória preditivos* ou simplesmente *sistemas preditivos*. A Figura 6.3 mostra como o sistema preditivo está estruturado.

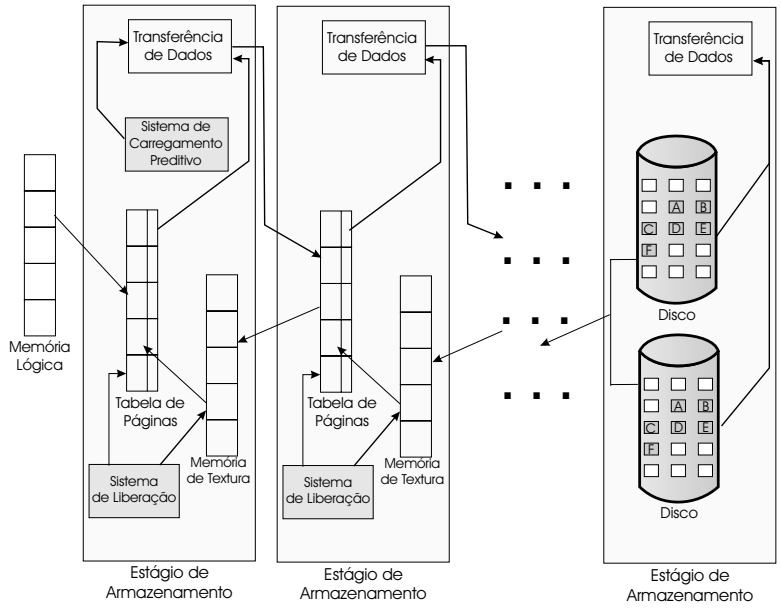

Figura 6.3: Mecanismo de paginação preditiva. Para facilitar a ilustração os sistemas de carregamento e liberação foram separados. No entanto, eles continuam fazendo parte de um sistema de paginação associado à um estágio de armazenamento.

Observe que só é necessário ter um sistema de carregamento de página. Esse sistema está ligado ao estágio mais alto de armazenamento de dados. Quando o sistema de carregamento pede para carregar uma página de dados, esse pedido é passado para os diversos estágios de armazenamento, em forma de cascata, até que a página requerida seja encontrada. Uma vez que a página tenha sido encontra em um determinado estágio, esta é carregada, também em forma de cascata, em todos os estágios acima deste, até estar acessível no estágio desejado (Figura 6.4). Já o sistema de liberação de página é específico para cada nível de armazenamento. Isto ocorre porque a necessidade de liberação de páginas em um estágio depende exclusivamente de sua capacidade de armazenamento, ou seja, quanto menor

for a capacidade de armazenamento do estágio maior será a necessidade de substituição de páginas. Assim, cada estágio de armazenamento pode ter sua própria política de liberação de páginas. De acordo com as regras dessa política, a complexidade do sistema pode aumentar ou diminuir.

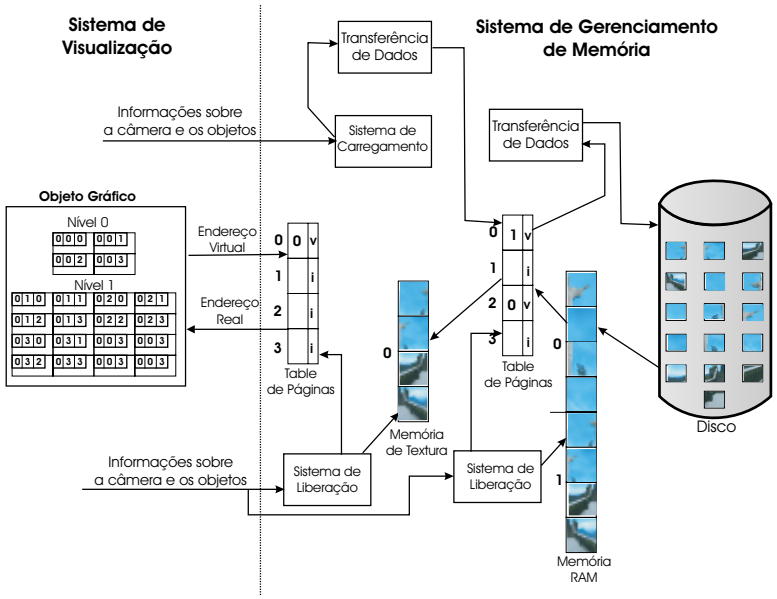

Figura 6.4: Comunicação entre os sistemas de visualização e de gerenciamento de memória durante o processo de visualização. O sistema de gerenciamento é ilustrado com três estágios de armazenamento.

Durante o processo de visualização, o sistema preditivo transforma os endereços virtuais em endereços reais onde os dados realmente estão armazenados. Esses endereços reais são passados para o sistema de visualização. O sistema sempre garantirá que um endereço virtual seja convertido em um endereço real. Para isso, é desenvolvido um sistema de paginação preditiva que explora a estrutura de dados dos objetos gráficos e a forma como esses objetos estão sendo visualizados. A Figura 6.4 mostra como o sistema de visualização troca informações com o sistema de gerenciamento de memória durante o processo de visualização. Observe o segundo campo das tabelas de mapeamento do estágio de textura e memória RAM. Se esse campo estiver com o símbolo "v" (válido) indica que o endereço virtual está carregado. Caso contrário, se o símbolo for "i" (inválido), então o endereço não está carregado. O primeiro campo da tabela armazena o identificador da página de dados que foi associada ao endereço virtual. Essa página armazena os endereços reais onde estão os dados referentes ao endereço virtual.

O sistema preditivo é implementado como uma camada acima do sistema operacional, veja a Figura 6.5. Observe que a aplicação utiliza os recursos oferecidos pelo sistema para acessar seus dados. O sistema preditivo, por sua vez, faz chamadas ao sistema operacional para acessar os dados e fornecê-los para a aplicação. O principal objetivo do sistema preditivo é permitir que uma grande quantidade de dados seja acessada pela aplicação sem a necessidade de utilizar o sistema de paginação do sistema operacional.

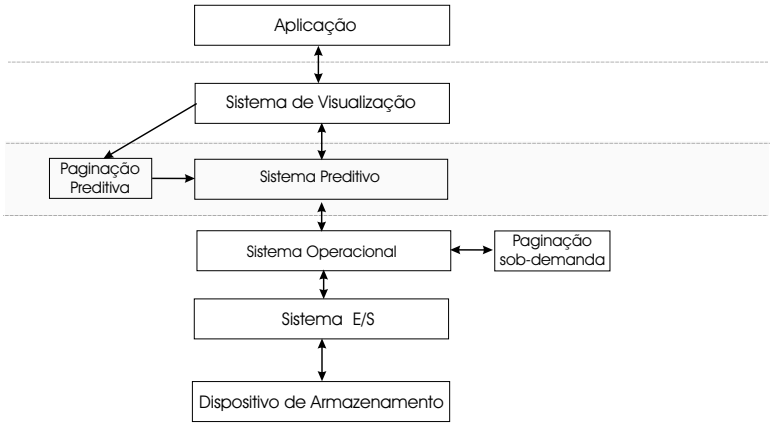

Figura 6.5: Organização do modelo de gerenciamento de memória para a visualização de objeto gráficos 2D.

6.1.1 Endereço Lógico

O tamanho do bloco de memória é configurado de acordo com as características da estrutura utilizada para representar os objetos gráficos. Como já foi discutido no Capítulo 4, os objetos gráficos são representados por ladrilhos e cada ladrilho é definido por $\mathcal{T}_{i,j,k} = (U_{i,j,k}, z_{i,j,k}, I_{i,j,k})$. O conjunto $U_{i,j,k}$ e o conjunto $I_{i,j,k}$ representam os dados armazenados pelos ladrilhos. Na representação em multi-resolução um ladrilho pai possui quatro ladrilhos filhos. Devido a essa característica, o tamanho de cada bloco de memória deve ser suficiente para armazenar os dados dos ladrilhos filhos. Desta forma, os ladrilhos filhos estão sempre armazenados no mesmo bloco de memória. Os dados de um ladrilho $\mathcal{T}_{i,j,k}$ são representados por um endereço lógico de quatro bytes. Esse endereço é definido da seguinte forma: dois bits para representar a posição dos dados na página, em seguida, vinte e dois bits para identificar a página, e finalmente, oito bits para identificar

o objeto gráfico 2D. A Figura 6.6 ilustra a estrutura de um endereço lógico. Os endereços lógicos também são chamados de *endereços virtuais*.

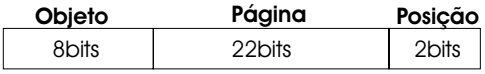

Figura 6.6: Cada endereço lógico identifica unicamente um ladrilho de um objeto gráfico 2D.

O espaço de endereços lógicos é criado após a construção da estrutura em multi-resolução. A atribuição dos endereços lógicos em cada ladrilho é realizada percorrendo os níveis da estrutura em multi-resolução geométrica. Isto é feito através da ligação existente entre os ladrilhos das duas estruturas em multi-resolução (veja a Seção 4.3). Durante essa fase podem surgir duas situações: o ladrilho geométrico está ligado a um ladrilho de textura que está no mesmo nível de resolução ou o ladrilho geométrico está ligado a um ladrilho de textura que está em um nível de resolução inferior. No primeiro caso, o ladrilho de textura é visitado e o campo `TexPointer` é iniciado com o endereço lógico. O segundo caso é ignorado porque os ladrilhos dos níveis inferiores já foram iniciados. A Figura 6.7 mostra como é formado o espaço de endereçamento lógico de um objeto gráfico.

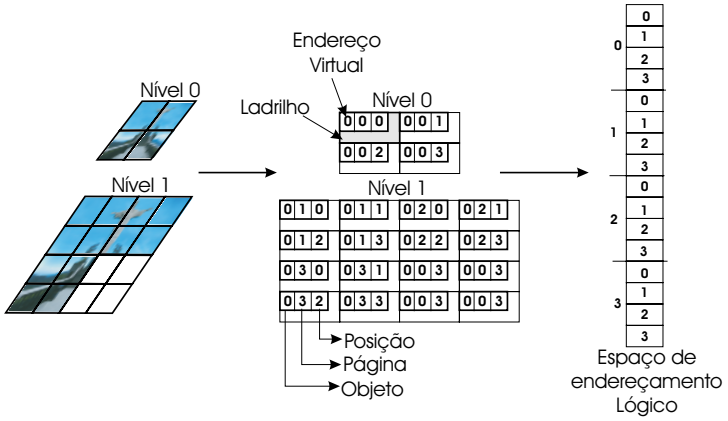

Figura 6.7: Construção do espaço de endereçamento lógico de um objeto gráfico 2D. O ladrilho sem textura recebe automaticamente o endereço da textura o nível inferior.

Para acessar os dados de um determinado ladrilho basta passar o endereço lógico para o sistema de gerenciamento de memória. Em resposta, o sistema retorna o endereço real onde a textura está armazenada.

6.2 Estágios de Armazenamento

Um estágio de armazenamento é definido como qualquer mecanismo que forneça operações de leitura, escrita e alocação de dados. Aqui, não importa se os dados são armazenados permanente ou temporariamente, ou se o dispositivo permite o armazenamento de dados gerais (memória RAM, HD, etc.) ou de dados específicos (memória de textura). Um estágio de armazenamento é composto pelos seguintes componentes:

- **Unidade de Controle**: Tem a finalidade de controlar as operações de carregamento, liberação e mapeamentos de páginas de forma consistente. A unidade de controle é responsável pela conversão de endereços virtuais em endereços reais.
- **Sistema de Paginação**: É formado pela união dos sistemas de carregamento, liberação e tratamento de ausência de páginas.
- **Sistema de Monitoração**: É utilizado para monitorar as operações de acesso aos dados efetuadas pela aplicação.
- **Unidade de Transferência de Dados**: Permite que o estágio de armazenamento possa trocar informações com outro estágio que está num nível abaixo do seu.
- **Memória Real**: Realiza os acessos diretos ao hardware que está associado ao estágio de armazenamento. O sistema pode criar vários espaços de memória dentro de um dispositivo.

Cada componente será discutido com mais detalhes nas próximas seções. A Figura 6.8 mostra a arquitetura de um estágio de armazenamento.

O estágio de armazenamento é uma abstração que fornece uma interface comum para operações de acesso de dados. O principal objetivo de um estágio de armazenamento é converter os endereços virtuais em endereços reais.

O principal componente de um estágio é a unidade de controle. A unidade de controle integra todos os outros componentes. A Figura 6.8 mostra todo o funcionamento de um estágio de armazenamento. Nessa ilustração foram feitas as seguintes considerações: o dado pedido não está carregado, o estágio está cheio e o sistema de tratamento de ausência de página faz pedidos de carregamento. Quando a aplicação passa um endereço virtual para um estágio, esse realiza os seguintes passos:

1. A primeira tarefa realizada pela unidade de controle é verificar se os dados correspondentes à este endereço está mapeado em algum

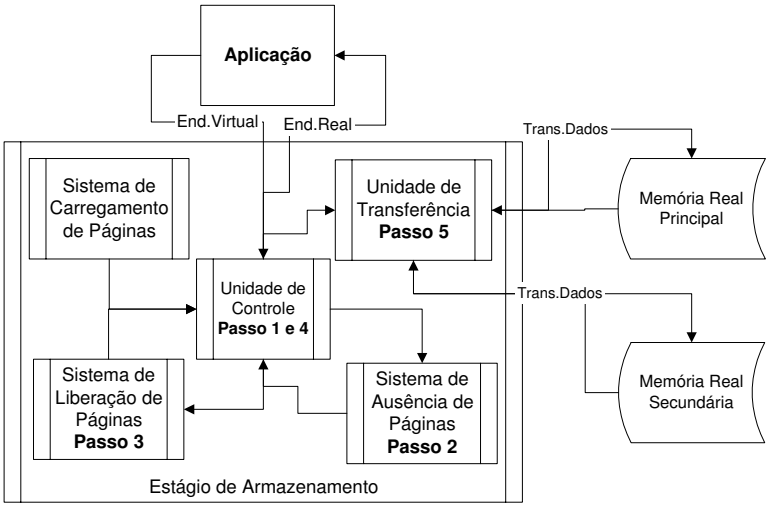

Figura 6.8: Arquitetura de um estágio de armazenamento.

endereço real, isto é, verifica se o dado requerido está carregado no dispositivo. Em caso afirmativo, o endereço real que contém os dados desejados é retornado, caso contrário é chamado o sistema de tratamento de ausência de páginas.

2. No sistema de ausência de páginas é feito o pedido de carregamento da página para a unidade de controle.

3. Para realizar o carregamento da página, a unidade de controle tem que alocar espaço no dispositivo. Se o dispositivo não estiver cheio, então a página é alocada. Caso contrário, a unidade de controle chama o sistema de liberação de páginas.

4. O sistema de liberação escolhe uma página que deve ser desalocada. Essa área é liberada e realocada pela unidade de controle.

5. A unidade de controle acessa a unidade de transferência e pede que os dados sejam carregados para a página alocada. Após a transferência, a unidade de controle retorna o endereço real da página com os dados requeridos.

Cada passo é ilustrado na Figura 6.8 e mais detalhadamente na Figura 6.12. Note que o sistema de carregamento de páginas em nenhum momento participa da tarefa de conversão entre endereços virtuais e reais. De fato, o sistema de carregamento só é utilizado para prever e carregar antecipadamente as páginas de dados. Deste modo, esse sistema só é

necessário estar presente no nível mais alto de armazenamento. Já nos outros níveis de armazenamento esse sistema é opcional.

6.2.1 Memória Real

A memória real é um nível de abstração utilizado para padronizar uma interface que permite realizar as operações de alocação, liberação, leitura e escrita de páginas. O objetivo dessa interface é permitir que qualquer hardware de armazenamento seja tratado da mesma forma. Esse nível de abstração também fornece uma interface para as operações de acesso aos dados. A Figura 6.9 mostra a estrutura de dados que constitui a memória real.

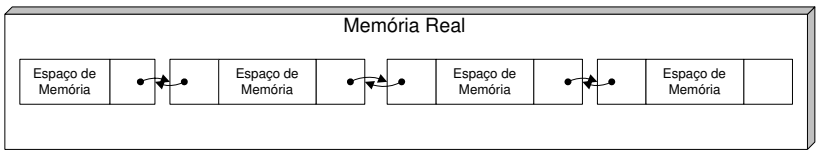

Figura 6.9: A memória real é o nível de abstração que permite a integração de qualquer dispositivo de armazenamento ao sistema.

Observe que a memória real pode ser formada por um ou vários espaços de memória. As operações de acesso, alocação e liberação de dados são efetivamente implementadas dentro dos espaços de memória. Cada espaço de memória possui uma tabela de páginas. Essa tabela armazena o estado (proteção, modificação, mapeamento), o endereço real e o endereço virtual das páginas. A Figura 6.10 ilustra a estrutura de uma tabela de páginas.

O campo que armazena o endereço virtual foi criado por uma questão de desempenho. Em geral, essa informação é utilizada pelos algoritmos de liberação de páginas. Isto foi necessário para evitar que o sistema de liberação varresse a tabela de mapeamento, pois essa tabela é bem maior do que a tabela de páginas. A Figura 6.11 mostra a arquitetura do espaço de memória e como se comunica com o resto do sistema preditivo.

A memória real fornece uma interface que permite ao sistema acessar todas as operações fornecidas pelo espaço de armazenamento. Para isso cada operação possui um parâmetro para especificar o espaço de memória que será acessado. A interface também permite operações de criação e remoção de espaços de memória. A criação de um espaço de memória é realizada através da especificação do espaço que será alocado no hardware, a quantidade e o tamanho dos blocos dentro de uma página. A partir

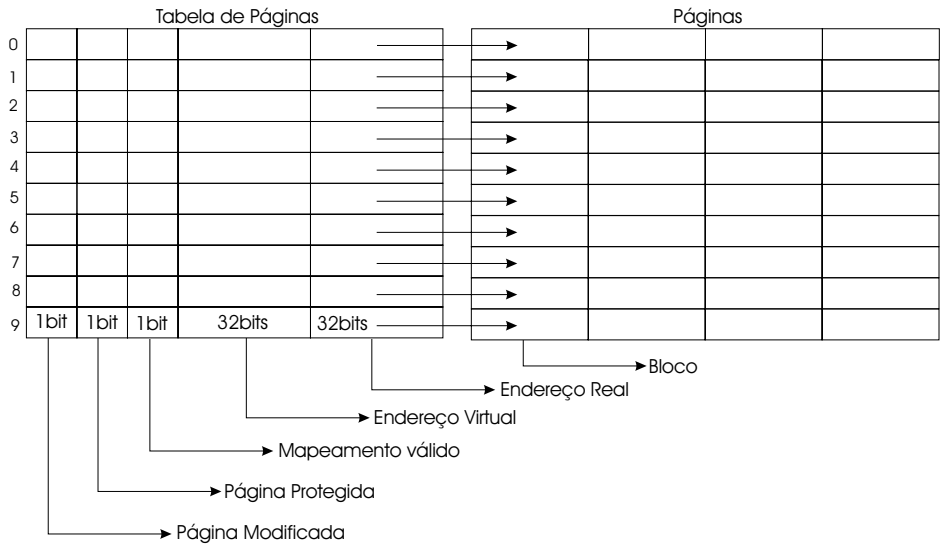

Figura 6.10: A tabela de páginas definida dentro de uma memória real. Neste exemplo uma página é divida em quadro blocos de dados.

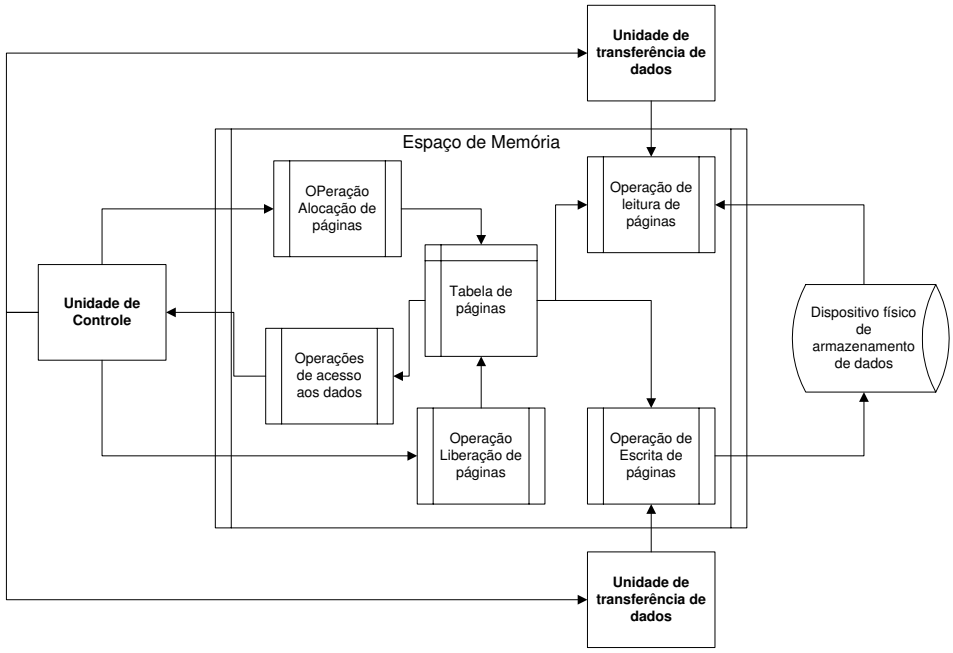

Figura 6.11: A arquitetura do espaço de memória. Todas as operações de acesso, alocação e liberação fazem consultas a tabela de páginas.

dessas informações é possível calcular o número de páginas que possuirá o espaço de memória e, assim, determinar o tamanho da tabela de páginas. Na operação de remoção basta fornecer o identificador do espaço de memória a ser removido.

O conceito de espaço de memória permite que objetos gráficos com mais de 4GBytes de dados sejam manipulados com ponteiro de 32 bits. Os dados do objeto gráfico devem ser particionados em arquivos contendo no máximo 4GBytes. No estágio de disco cada um desses arquivos é tratado como um espaço de memória. Quando a aplicação acessa uma página de dados o sistema determina qual espaço de memória deve ser acessado. Outra utilidade desse conceito é permitir que o sistema seja distribuído. Suponha que o nível de rede implemente uma arquitetura cliente-servidor. Neste caso, cada espaço de memória agiria como um cliente pedindo dados ao servidor. Assim, fazendo com que cada espaço de memória esteja associado à um servidor diferente obtém-se um sistema distribuído de dados.

6.2.2 Unidade de Controle

A unidade de controle tem o objetivo de controlar a execução de todas as operações de acesso aos dados e também controlar as operações de troca de dados com o estágio de armazenamento que está um nível abaixo. Ou seja, a execução de todas as operações necessárias para que uma página seja carregada ou liberada do estágio está implementado nessa unidade. Os sistemas de paginação apenas tomam a decisão de quais páginas devem ser carregadas ou liberadas. A Figura 6.12 mostra a arquitetura da unidade de controle. Dentro da unidade de controle são realizadas as conversões de endereços virtuais em endereços reais.

A unidade de controle cria uma estrutura de dados para cada objeto gráfico. Essa estrutura é composta por uma tabela de mapeamento e um identificador do espaço de memória associado ao objeto. A tabela de mapeamento armazena informações sobre as páginas de endereços virtuais. Essas informações são o estado de mapeamento de uma página virtual (Mapeada ou não) e o endereço da página real que está associada ao endereço da página virtual, veja a Figura 6.13. Essa tabela é utilizada para fazer as conversões entre endereço virtual e endereço real. O identificador do espaço de memória é utilizado para realizar as operações de carregamento e liberação de páginas. Os passos realizados para realizar a conversão de endereços são os seguintes:

1. Decodifica o endereço virtual para obter o identificador da página a que ele pertence. Esse identificador representa um índice da tabela de mapeamento.

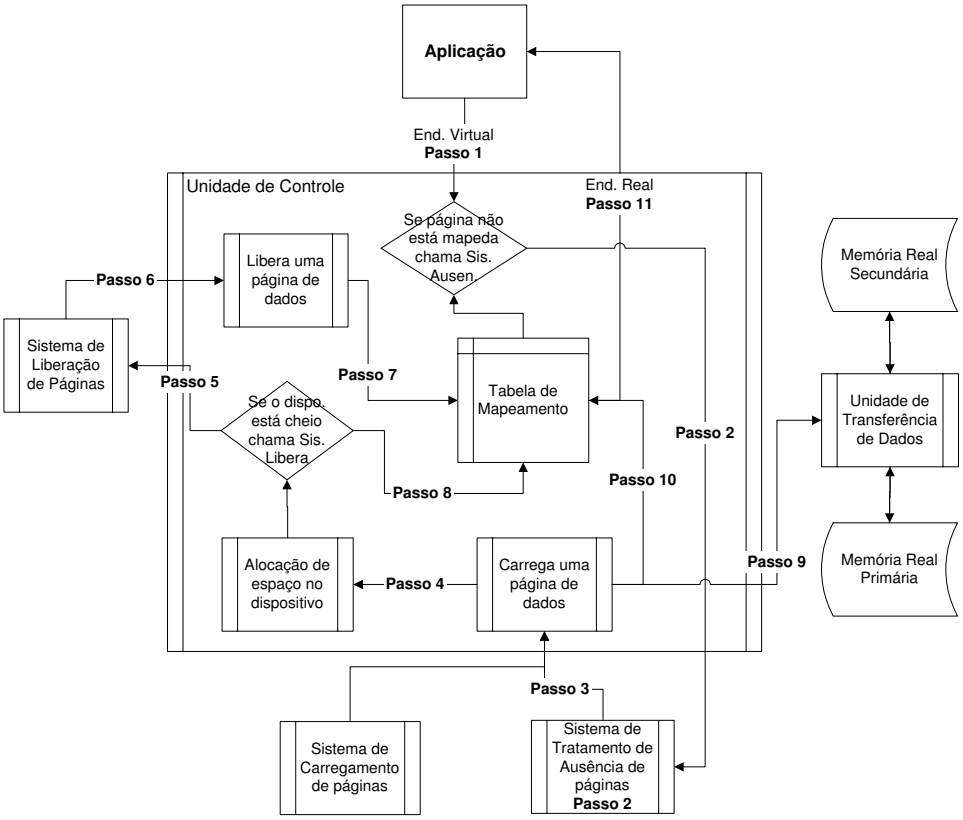

Figura 6.12: A arquitetura e o funcionamento da unidade de controle. Essa figura ilustra o pior caso de funcionamento do sistema.

2. Verifica se a página virtual está mapeada.

3. Se a página estiver mapeada, decodifica o offset do endereço virtual e retorna o endereço real que responde a esse offset.

As operações de carregamento de páginas são realizadas executando os seguintes passos:

1. Aloca um espaço no estágio de armazenamento para a página de dados.

2. Se o estágio estiver cheio, a unidade de controle chama o sistema de liberação de páginas para escolher uma página para ser liberada. A página escolhida é liberada e realocada pela unidade de controle.

3. A unidade de controle pede para a unidade de transferência de dados transferir os dados para a página alocada.

4. Após a unidade de transferência sinalizar que o dado foi carregado, a unidade de controle atualiza a tabela de mapeamento informando

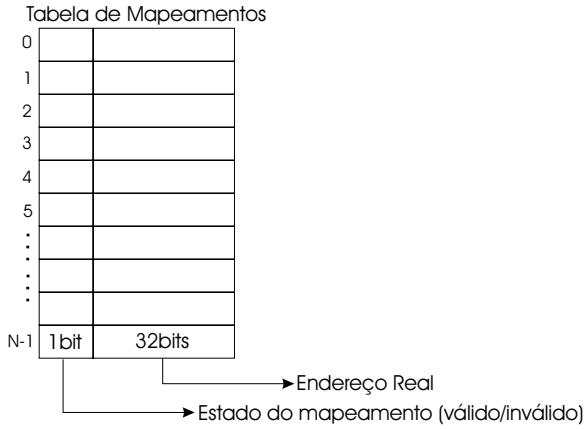

Figura 6.13: A tabela de mapeamento que é definida dentro de um estágio de armazenamento.

que a página virtual está associada à uma página real. O campo que armazena o endereço real é atualizado com o endereço da página alocada.

As operações de liberação de páginas são realizadas a partir dos seguintes passos:

1. Chama o sistema de liberação de páginas para escolher uma página para ser retirada.

2. Verifica se a página escolhida foi modificada pela aplicação.

3. Se a página foi modificada, a unidade de controle utiliza a unidade de transferência para transferir os dados desta página para o dispositivo que está um nível abaixo.

4. Após a unidade de controle sinalizar que os dados foram transferidos, a unidade de controle atualiza a tabela de mapeamento informando que a página virtual não está mais associada.

No caso das operações de ausência de páginas, a unidade de controle apenas chama o sistema de tratamento de ausência de páginas. A implementação mais simples consiste em pedir a unidade de controle para carregar a página ausente. Os sistemas operacionais utilizam este método para resolver esse problema.

6.2.3 Unidade de Transferência de Dados

A unidade de transferência de dados é o sistema utilizado para controlar a troca de dados entre dois estágios de armazenamento consecutivos. Este nível de abstração é necessário porque existem várias diferenças na forma como um tipo de dispositivo transfere ou recebe dados de outro dispositivo. Muitas vezes essas diferenças estão relacionadas com as interfaces fornecidas pelas bibliotecas que implementam as funções de leitura e escrita nos dispositivos de armazenamento. A operação de escrita é definida como a transferência de uma página do estágio mais alto para o estágio mais baixo. A operação de leitura representa a operação inversa, ou seja, realiza a transferência de uma página do estágio mais baixo para o estágio logo acima. A principal função da unidade de transferência é criar uma interface padrão para as operações de leitura e escrita entre os estágios de armazenamento. As operações de leitura e escrita podem ser realizadas de forma síncrona ou assíncrona.

Na forma síncrona a unidade de transferência só devolve o controle para o sistema quando a operação solicitada foi executada. No método assíncrono os pedidos de leitura/escrita são colocados em uma lista. Esses pedidos são executados paralelamente, ou seja, após o pedido ser colocado na lista a unidade de transferência retorna o controle para o sistema. Quando um pedido é executado, a unidade de controle é informada e executa o *Passo 4* mostrado na seção anterior.

Para facilitar o entendimento do processo de funcionamento da unidade de transferência, o estágio mais alto será chamado de estágio primário e o estágio mais baixo será chamado de estágio secundário. As unidades de controle desses estágios serão chamadas de unidades de controle primária e secundária, respectivamente.

As operações de transferência de dados são iniciadas quando a unidade de controle primária faz um pedido de leitura ou de escrita. Esse pedido é especificado a partir de um endereço virtual. Para atender um pedido de leitura a unidade de transferência executa os seguintes passos:

1. Acessa a unidade de controle primária para obter o endereço real da página e o espaço de memória referentes ao endereço virtual.

2. Acessa a unidade de controle secundária para obter o endereço real da página e o espaço de memória referentes ao endereço virtual. Se não existir uma página real associada ao endereço virtual, então a unidade

de controle secundária providenciará que a página seja carregada e envia a unidade de transferência.

3. Os dados contidos na página do estágio secundário são transferidos para a página do estágio primário.

A Figura 6.14 ilustra os passos realizados para a execução da operação de leitura. Observe que o *Passo 2* permite que um pedido de carregamento de uma página seja repassado para os outros estágios de armazenamento. Esta operação gera o efeito cascata de um pedido de carregamento. No caso

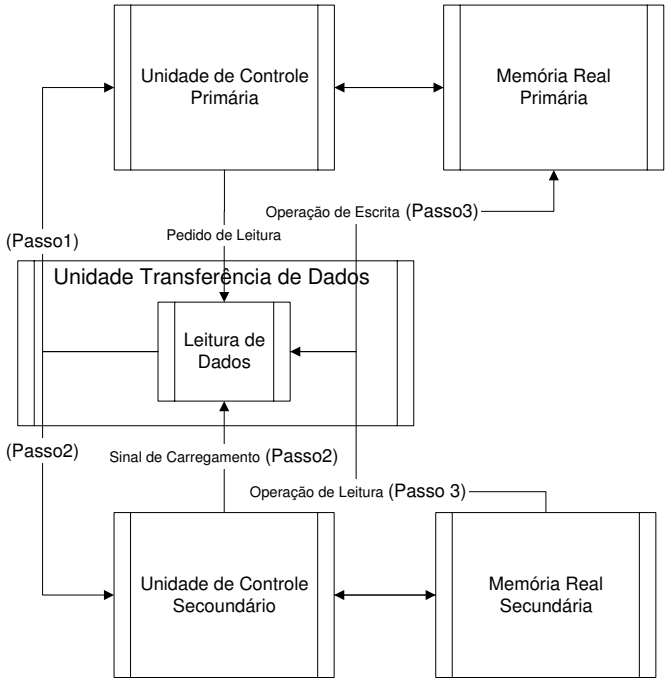

Figura 6.14: O funcionamento do sistema de transferência para realizar a operação de leitura. A operação de leitura é utilizada para transferir dados do estágio secundário para o estágio primário.

de uma operação de escrita, a unidade de transferência realiza os seguintes passos:

1. Acessa a unidade de controle primária para obter o endereço real da página e o espaço de memória referente ao endereço virtual.

2. Pede para a unidade de controle secundária reservar uma página real para o endereço virtual. Esse pedido só é feito se o endereço virtual não estiver associado a uma página real no estágio secundário. A unidade de controle envia um sinal informando o endereço de uma página real juntamente com o identificador do espaço de memória.

3. Os dados contidos na página do estágio primário são transferidos para a página do estágio secundário.

A Figura 6.15 mostra a execução da operação de escrita. A operação de escrita é necessária para que a aplicação possa modificar os dados. Ao modificar uma determinada página de dados, a aplicação deve informar ao sistema. O sistema liga o campo que indica que a página foi modificada. Antes da página modificada ser liberada do estágio primário, ela deve ser salva no estágio secundário. Isto é realizado a partir da operação de escrita. Essa operação se repetirá em todos os outros estágios até chegar ao estágio mais baixo de armazenamento. Observe que o sistema garante que a modificação sofrida nos dados da página sempre será encontrada em algum estágio de armazenamento. Do ponto de vista da aplicação, os dados modificados sempre serão acessados como se eles tivessem chegado ao estágio final. O sistema transfere todas as páginas modificadas para o estágio final quando encerra as suas operações. O sinal de término das operações deve ser enviado pela aplicação.

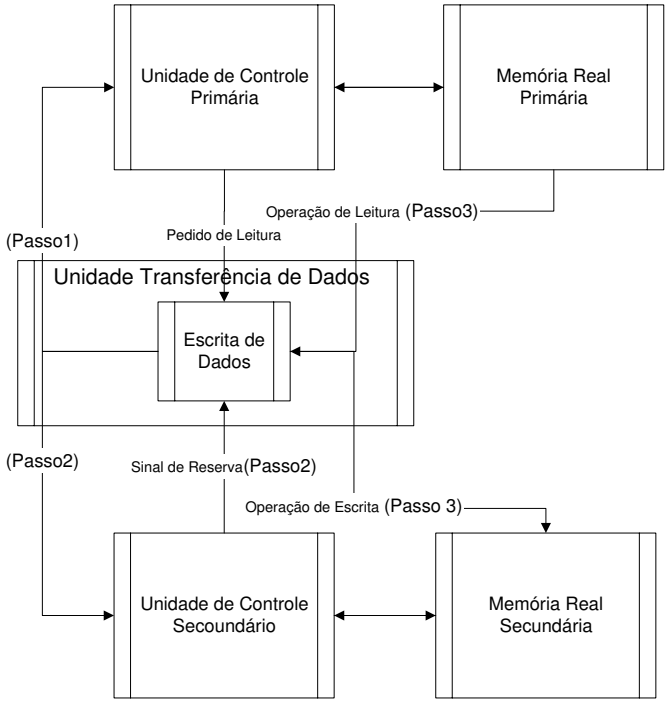

Figura 6.15: O funcionamento do sistema de transferência para realizar a operação de escrita. A operação de escrita é utilizada para transferir dados do estágio primário para o estágio secundário.

Na transmissão assíncrona, os *Passos 2 e 3* são executados em paralelo com as outras operações realizadas pelo sistema. Assim, o processamento do

sistema preditivo não é interrompido a cada pedido de carregamento. Esse método de transmissão é mais recomendado quando a diferença do tempo de acesso aos dados entre os estágios é grande. Por exemplo, o estágio de disco é muito mais lento que o estágio de memória RAM. Neste caso, a transmissão assíncrona de dados é mais adequada. A transmissão síncrona é mais adequada quando os estágios interligados possuem uma taxa de transferência rápida. Um exemplo disso ocorre nos estágios de textura e memória RAM. Ambos estágios possuem altas taxas de transferência, logo as operações de escrita ou leitura são realizadas rapidamente. A utilização do método assíncrono certamente resultaria em perda de desempenho, pois a execução desse método exige que uma estrutura de dados mais complexa seja gerenciada. Assim, a soma do tempo gasto para gerenciar essa estrutura mais o tempo para realizar a transferência da página pode ser maior do que simplesmente executar operação de transferência da página.

6.2.4 Monitoração de Tráfego de Dados

O sistema de monitoração é utilizado para monitorar as operações de acessos aos dados do estágio de armazenamento. A idéia é que cada operação de acesso aos dados resulte em um evento. Esse evento é chamado de *evento de acesso*. A Figura 6.16 mostra todos os eventos de acesso que podem ser monitorados pelo sistema. Os eventos de acesso são gerados pela unidade de controle de cada estágio de armazenamento. Os sistemas de monitoração podem ser utilizados nas seguintes situações:

- Em sistemas de liberação de páginas baseados em algoritmos que necessitam de informações sobre os acessos realizados no estágio. Dentre esses algoritmos podemos destacar o algoritmo NRU ("Not-Recently-Used"), FIFO ("First-In, Firt-Out"), LRU ("Least-Recently-Use"), entre outros [4].
- Em sistemas de visualização que mostram o que está acontecendo com os dados que estão armazenados nos estágios. Assim, é possível visualizar o efeito das operações de acesso e das operações realizadas nos sistemas de paginação.
- Em sistemas de depuração para verificar se os sistemas de paginação e até mesmo a aplicação estão funcionando como esperado.
- Em sistemas que estimam o tempo que uma página leva para sair do último estágio e chegar ao primeiro. Esse recurso será explorado no sistema de predição proposto neste trabalho.

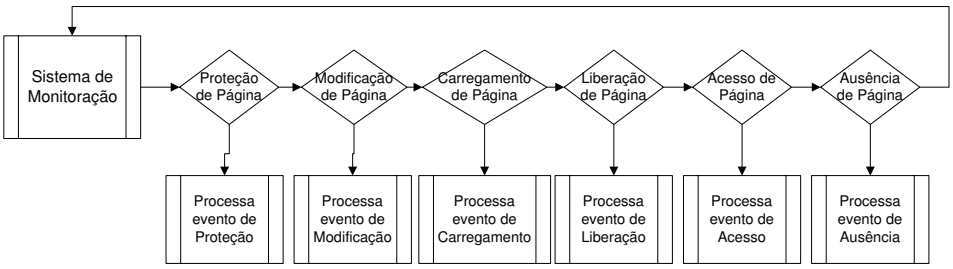

Figura 6.16: Arquitetura do sistema de monitoração das operações de acesso aos dados. Cada operação é convertida em um evento.

6.3 Sistema de Paginação

O sistema de paginação é formado por três sistemas: o sistema de carregamento, o sistema de liberação e o sistema de tratamento de ausência de páginas.

A função do sistema de carregamento de páginas é predizer e carregar as páginas de dados que serão utilizadas pela aplicação num futuro próximo. Essa tarefa só pode ser realizada se o sistema tiver informações sobre a forma como a aplicação acessa os seus dados. Por exemplo, nas aplicações de visualização, os dados são acessados de acordo com os movimentos da câmera virtual. Assim, é necessário o sistema saber como os parâmetros da câmera estão variando no tempo. Com essa informação é possível realizar a tarefa de predição. Essa característica torna a implementação do sistema de carregamento específica para um certo grupo de aplicações.

O sistema de carregamento é constituído por um sub-sistema chamado de *sistema de regras de carregamento*. Esse sistema tem a finalidade de definir um conjunto de dependências para que uma página seja carregada. Essas dependências são chamadas de *regras de carregamento*. O sistema de regras é a ponte que liga o sistema de carregamento à unidade de controle, veja a Figura 6.17.

Observe que o funcionamento de um sistema de carregamento é realizado em dois passos. O primeiro é decidir quais páginas de dados devem ser carregadas. Essa tarefa é realizada pelo algoritmo de predição, e depois, para cada página escolhida, chamar o sistema de regras de carregamento para executar os passos necessários para carregá-la. No Capítulo 7 são discutidas as regras de carregamento e o algoritmo de predição.

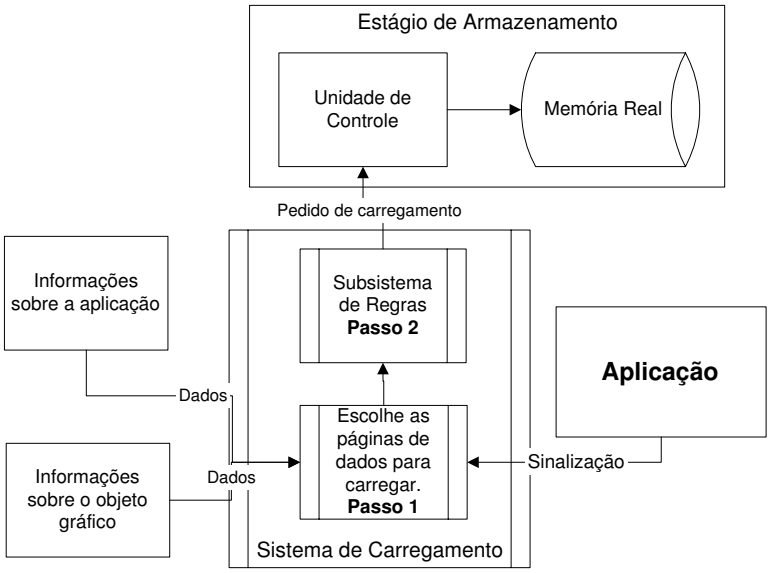

Figura 6.17: Organização e funcionamento básico de um sistema de carregamento de dados.

O sistema de liberação de páginas tem a função de escolher as melhores páginas para serem removidas do estágio de armazenamento. Em geral, esta tarefa só é realizada quando o estágio está cheio e é necessário liberar espaço para que uma nova página seja carregada.

A arquitetura do sistema de liberação é igual a do sistema de carregamento de páginas, veja a Figura 6.18. Observe que esse sistema também possui um sub-sistema de regras. Neste caso, o sub-sistema de regras é chamado de *sistemas de regras de liberação*. O objetivo desse sub-sistema é implementar os requisitos necessários para que uma página seja liberada. Esses requisitos definem as regras de liberação. As regras de liberação também são discutidas no Capítulo 7.

Apesar das arquiteturas dos sistemas serem iguais, o funcionamento do sistema de liberação é totalmente diferente do sistema de carregamento. As diferenças estão na forma como o sistema de liberação se comunica com o resto do sistema. O sistema de liberação se comunica com a unidade de controle de forma bi-lateral. De um lado, a unidade de controle envia uma mensagem para o sistema de liberação escolher as páginas que devem ser retiradas. Por outro, o sub-sistema de regras envia os endereços virtuais das páginas que foram escolhidas. A Figura 6.18 ilustra os dois passos executados pelo sistema de liberação para remover uma página do estágio de armazenamento.

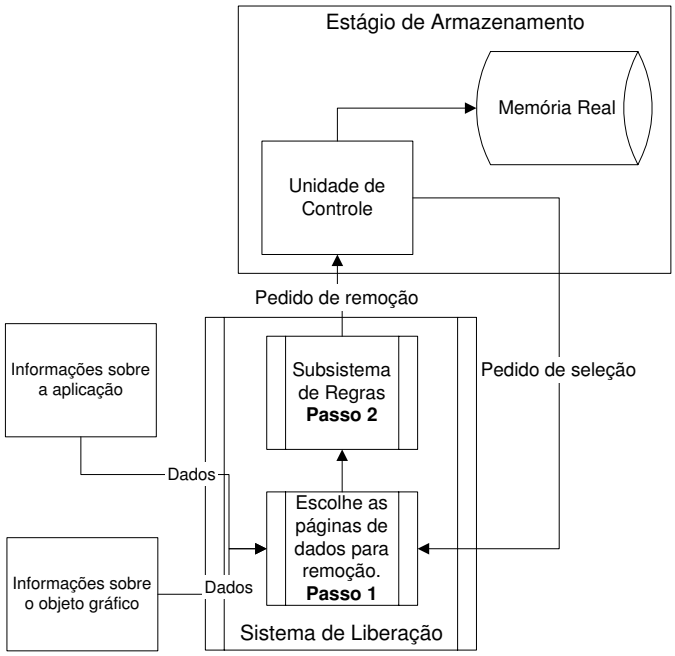

Figura 6.18: Organização e funcionamento básico de um sistema de liberação de páginas.

A melhor página a ser retirada é aquela que não será utilizada pela aplicação no maior espaço de tempo. Existem algoritmos, como NRU, LRU e outros, que podem ser utilizados para qualquer tipo de aplicação. Esses algoritmos são simples de implementar e podem ser utilizados em estágios que possuem um grande poder de armazenamento, como por exemplo, os discos. No entanto, se utilizados em estágios com pouco poder de armazenamento como, memórias de textura ou RAM, podem prejudicar o desempenho do sistema. Isto ocorre porque muitas vezes esses algoritmos podem escolher páginas que são necessárias para a aplicação. Assim, será necessário carregá-las novamente. Essa perda de desempenho pode ser minimizada projetando um sistema de liberação específico para o tipo de aplicação. No próximo capítulo é apresentada uma solução para as aplicações de visualização.

O sistema de tratamento de ausência de páginas tem a função de tratar o problema de acessos à páginas ausentes, isto é, páginas que não foram carregadas no estágio de armazenamento desejado. A interface do sistema fornece o operador *HandlePageFault* para lidar com esse problema. A Figura 6.19 mostra a arquitetura e o funcionamento desse sistema.

Note que existem dois eventos de falta de página que podem ocorrer no sistema. O primeiro ocorre quando um estágio primário pede uma página

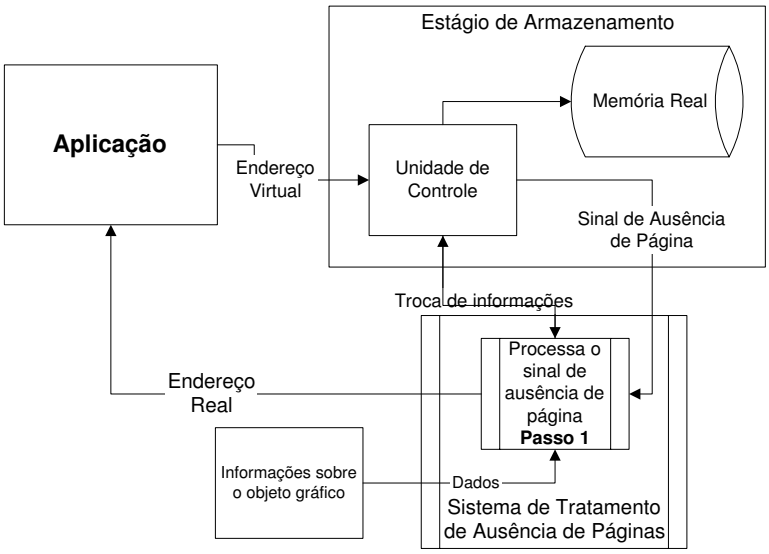

Figura 6.19: Arquitetura e funcionamento do sistema de tratamento de ausência de páginas. O sistema pode trocar informações com a unidade de controle. Essa troca de informações permite ao sistema realizar o pedido de carregamento da página. A unidade de controle responde ao pedido com o endereço real da página solicitada.

que não está carregada no estágio secundário e o outro ocorrre quando a aplicação pede um dado que não está armazenado no estágio desejado. No primeiro caso, a falta de página é imediatamente revertida em um pedido de carregamento. Nesse caso as operações de carregamento são realizadas no momento e dentro do tempo especificados pela aplicação. Desta forma, a aplicação não compete pelo processamento com o sistema e não perde desempenho.

O segundo caso também pode ser resolvido com o simples pedido de carregamento dos dados. A vantagem dessa solução é que pode ser utilizada em qualquer tipo aplicação, por isso que ela é implementada pelos sistemas operacionais. A principal desvantagem é a perda de desempenho da aplicação, pois é necessário esperar o carregamento dos dados para continuar o processamento.

Esse trabalho trata o problema de ausência de página sem a necessidade de carregar a página ausente. Para isso é necessário explorar a representação em multi-resolução dos objetos gráficos. A solução é baseada no princípio de que é possível substituir o dado desejado por outro de menor resolução que esteja carregado no estágio de armazenamento desejado. Os detalhes dessa técnica são apresentados no próximo capítulo.

6.4 Interface com as Aplicações

A interface define como a aplicação se comunica com o sistema de gerenciamento de memória. Ela foi dividida em três grupos. O primeiro grupo define as operações de acesso aos dados armazenados nos vários estágios de armazenamento. O segundo grupo define as operações com objetos gráficos. O terceiro grupo é formado por apenas uma operação que é utilizada para sincronismo. A tabela 6.4 mostra os três grupos que permitem acessar os recursos fornecidos pelo sistema.

Operadores de Comunicação		
Grupo	**Operações**	**Descrição**
Objeto Gráfico	OpenObject	Adiciona um objeto gráfico no sistema.
	CloseObject	Remove um objeto gráfico do sistema.
Acesso aos Dados	GetData	Retorna o endereço real que está associado a um endereço virtual.
	ModifiedData	Informa ao sistema que os dados foram modificados pela aplicação.
	LockData	Protege os dados que estão carregados em um dispositivo.
	IsDataLocked	Verifica se um endereço virtual está protegido.
	IsDataModified	Verifica se um endereço virtual foi modificado.
	IsDataLoaded	Verifica se um endereço virtual está carregado no dispositivo.
Sincronismo	Update	Informa ao sistema o fim de um ciclo de processamento.

Tabela 6.1: Grupo de operações que definem as funções básicas de um sistema de gerenciamento de memória.

As operações de acesso aos dados permitem que a aplicação realize operações como proteção de páginas, verificação dos possíveis estados de uma página de dados (protegida, carregada, modificada) e acesso aos dados, isto é, conversão entre endereço virtual para um endereço real.

As operações com objeto gráfico permitem a abertura e fechamento de arquivos. Esses arquivos contêm todos os dados que representam um objeto gráfico. Quando um arquivo é aberto, o sistema gera os endereços virtuais e cria as estruturas de dados com as informações que são necessárias durante as tarefas realizadas pelo sistema de gerenciamento.

As operações de sincronismo são representadas por um operador. A aplicação utiliza esse operador para sinalizar ao sistema de gerenciamento

que pode realizar suas tarefas. Esse sinal significa que a aplicação terminou um ciclo de processamento. Ao receber o sinal de sincronismo, os sistemas de paginação são ativados. Essa operação de sincronismo será discutida com mais detalhes na próxima seção.

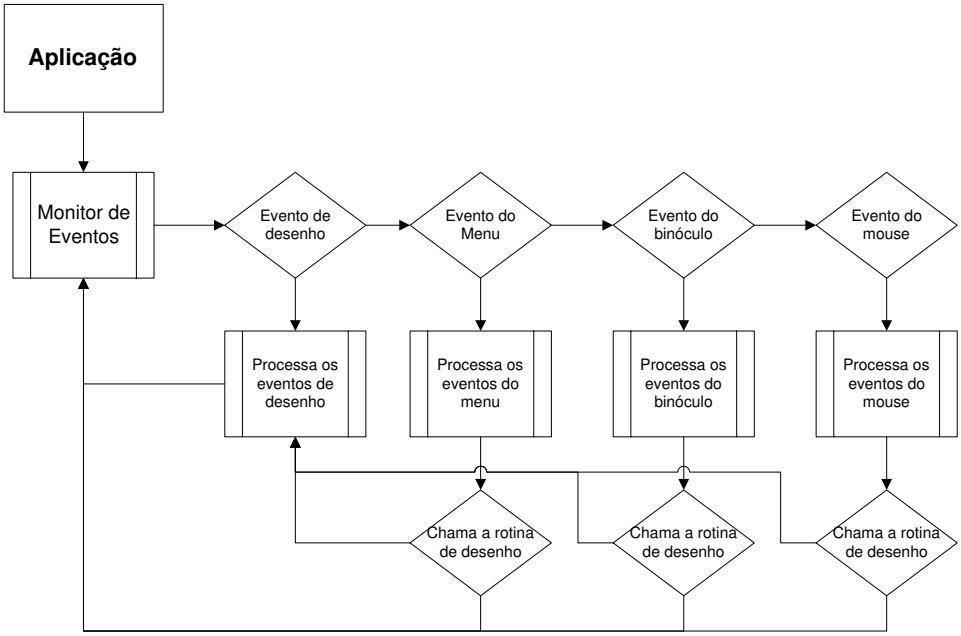

Figura 6.20: Fluxograma geral de uma aplicação de visualização (visualização de panoramas e imagens de satélite). Uma aplicação de visualização limita-se basicamente em processar os eventos vindos dos dispositivos de entrada e convertê-los em eventos de rendering.

Para mostrar como uma aplicação pode utilizar esse grupo de operações para trocar informações com o sistema de gerenciamento de memória, será utilizado como exemplo uma aplicação para visualização de imagens de satélite. A Figura 6.20 mostra o fluxograma geral do processamento realizado pela aplicação. Observe que a aplicação funciona num laço infinito que consiste em monitorar a fila de eventos e processar os eventos retirados desta fila. Os eventos processados são geralmente convertidos em eventos de visualização.

A aplicação é basicamente formada por duas fases de processamento. A primeira fase consiste em criar a estrutura em multi-resolução a partir de uma imagem de satélite. A construção dessa estrutura é feita conforme foi discutido no Capítulo 4. Após o objeto gráfico ser criado, ele é salvo em disco. No disco são criados dois arquivos, um contendo a estrutura em multi-resolução geométrica e o outro contendo a estrutura em multi-resolução de textura. Em seguida, a aplicação utiliza a operação *OpenObject* para

o sistema preditivo carregar o objeto gráfico. O sistema retorna o objeto gráfico para a aplicação. O objeto gráfico é imediatamente passado para o sistema de visualização. A Figura 6.21 mostra todos os passos executados pela aplicação e pelo sistema preditivo durante essa fase.

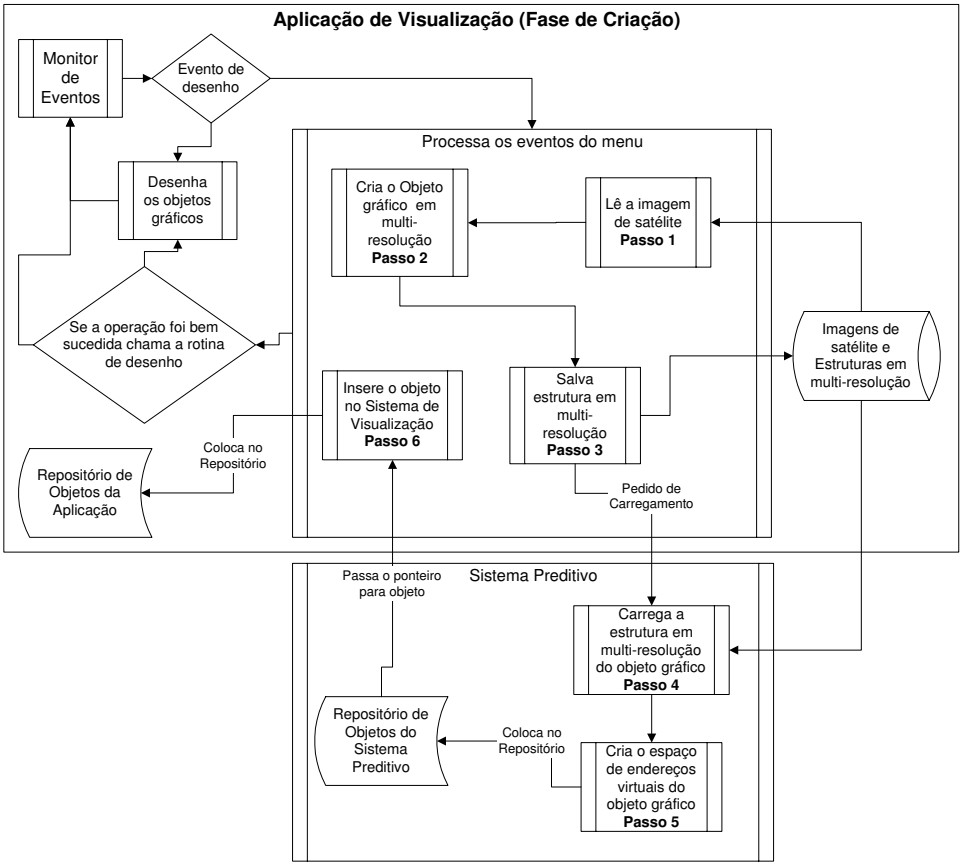

Figura 6.21: Processo de criação dos objetos gráficos.

No sistema preditivo cada objeto gráfico é associado a um espaço de memória dentro de cada estágio de armazenamento. Essas associações são realizadas considerando os tamanhos dos dados armazenados pelos ladrilhos. O espaço de memória escolhido é aquele que tiver o tamanho da página com capacidade de armazenar os quatro ladrilhos. Note que vários objetos gráficos podem compartilhar o mesmo espaço de memória contanto que os dados de seus ladrilhos possam ser armazenados. O conceito de espaço de memória será abordado com mais detalhes na seção 6.2. Por agora, considere que um espaço de memória é uma região do estágio de armazenamento reservada pelo sistema preditivo.

O sistema também cria uma estrutura de dados que é utilizada para trocar informações com o objeto gráfico e para acessar as estruturas em

multi-resolução. Essa estrutura de dados é chamada de `ObjectMemory`, veja a Figura 6.22. Observe que existe uma tabela de mapeamento e um ponteiro para o objeto gráfico. A tabela permite converter um endereço virtual em ponteiro para um ladrilho. Isto permite ao sistema acessar e percorrer a estrutura em multi-resolução. O ponteiro para o objeto gráfico permite ao sistema trocar informações com o mesmo. As informações fornecidas pelo objeto gráfico são necessárias para o processo de carregamento e liberação de páginas. A estrutura `ObjectMemory` será muito utilizada pelos algoritmos utilizados nos sistemas de paginação.

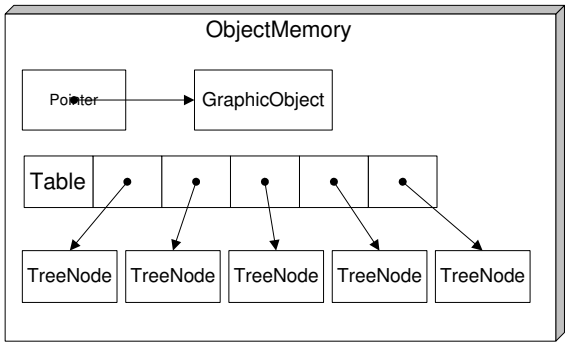

Figura 6.22: Estrutura de dados para cada objeto gráfico gerenciado pelo sistema.

A segunda fase de processamento consiste nas operações de visualização. As operações de acesso aos dados são utilizadas pela aplicação para visualizar a imagem de satélite. Durante o processo de visualização a aplicação utiliza o Algoritmo 5 para visualizar a imagem de satélite, veja a Capítulo 5. O Algoritmo 11 destaca a modificação realizada no algoritmo de visualização para utilizar os recursos do sistema. Observe que foi necessário acrescentar apenas uma linha de código. O operador utilizado foi *GetData*. Este operador recebe como parâmetro o estágio de armazenamento e o endereço virtual e a saída é o endereço real onde está o dado requisitado. Em seguida, a aplicação passa o endereço real da textura para a função `DrawTile`. Nesse exemplo, o estágio acessado é a memória de textura.

Note que a interface fornecida pelo sistema pode ser utilizada por qualquer tipo de aplicação. O sistema não tem nenhum conhecimento do tipo de objeto gráfico que está sendo visualizado. A outra vantagem é que essa interface torna transparente para a aplicação todos os problemas ligados com gerenciamento de memória, ou seja, são completamente transparentes todas as operações de transferência de dados e paginações realizadas pelo

Algoritmo 11 ViewSatelliteImage (**MultiRes** $\mathcal{MR}_{geo}$, **MultiRes** $\mathcal{MR}_{tex}$, **SatCamModel** $SatCam$, $ScreenW$, $ScreenH$)

```
CalculateLevel(MR_geo, MR_tex, SatCam, ScreenW, ScreenH, DrawLevel)
CalculateIndex(MR_geo, SatCam, DrawLevel, Index_min, Index_max)
for (i = Index_min.i ; i ≤ Index_max.i ; i = i + 1) do
  for (j = Index_min.j ; j ≤ Index_max.j ; j = i + 1) do
    VirtualAddress = MR_geo.Level[DrawLevel, i, j].T.MapTexture[0].TexPointer
    MemorySystem.GetData(VirtualAddress, &RealAddress)
    MR_geo.Level[DrawLevel, i, j].T.MapTexture[0].TexPointer = RealAddress
    DrawTile (MR_geo.Level[DrawLevel, i, j].T.GeoPointer, MR_geo.Level[DrawLevel, i, j].T.MapTexture[0])
    MR_geo.Level[DrawLevel, i, j].T.MapTexture[0].TexPointer = VirtualAddress
  end for
end for
MemorySystem.Update (TimeToUpdate)
end function
```

sistema. A aplicação acessa um dado no estágio de armazenamento desejado e utiliza esse dado.

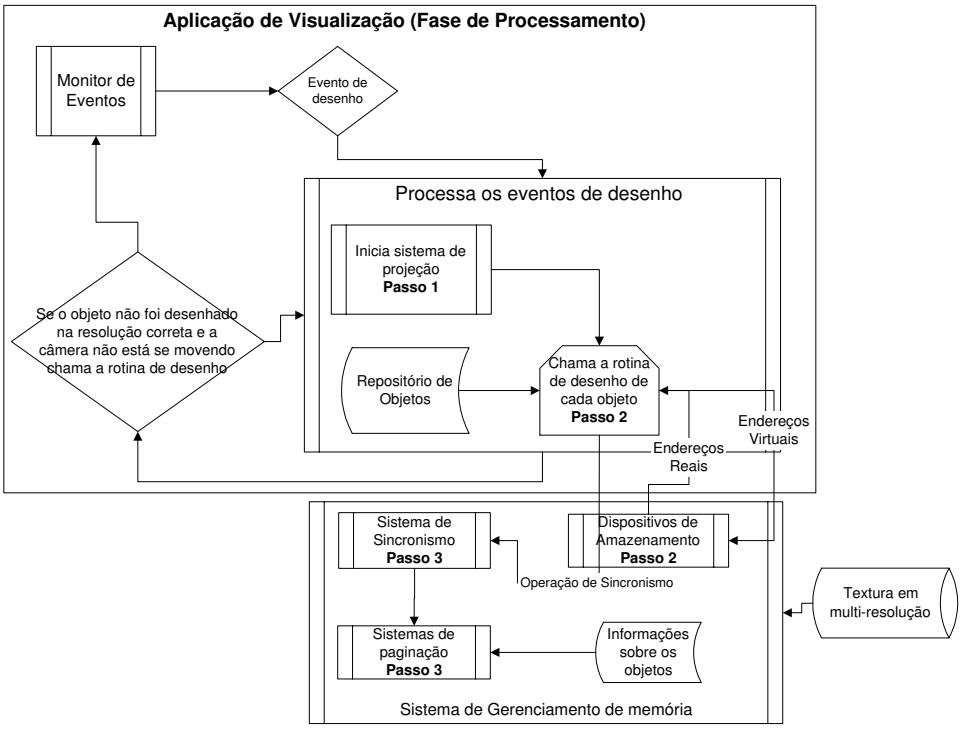

Figura 6.23: Processo de visualização dos objetos gráficos.

Após visualizar um quadro, a aplicação chama a operação de sincronismo do sistema. Ao chamar essa operação, a aplicação passa como parâmetro o tempo máximo (`TimeToUpdate`) que o sistema pode gastar para realizar as suas operações. O diagrama da Figura 6.23 mostra os passos dessa fase de processamento.

6.4.1 Sincronismo entre o Sistema e a Aplicação

O sistema de sincronismo é utilizado para informar ao sistema de gerenciamento de memória que um ciclo de processamento acabou de ser realizado. Esta sinalização é necessária devido a dois motivos:

- A sinalização evita a concorrência de recursos (CPU e acessos aos periféricos) entre a aplicação e o sistema de gerenciamento. Esse problema pode ser parcialmente resolvido se o sistema tiver mais de uma CPU.
- As tarefas realizadas pelo sistema de gerenciamento de memória são baseadas nos acessos realizados durante um certo intervalo de tempo ou ciclo de processamento. No caso dos sistemas operacionais esta sinalização é realizada por período de tempo. A desvantagem de utilizar a sinalização por tempo é que a aplicação pode ser interrompida antes de finalizar o seu processamento. Em aplicações de visualização em tempo-real a fase de síntese de quadros é crítica, ou seja, nesta fase a aplicação utiliza toda a capacidade de processamento disponível e não pode ser interrompida.

O sinal é passado para os sistemas de paginação de todos os níveis de armazenamento. Ao receber esse sinal, cada sistema realiza o processamento de suas tarefas. Por exemplo, o sistema de carregamento de páginas realiza a tarefa de predição e o sistema de liberação de páginas baseados no algoritmo NRU zera a tabela de páginas acessadas.

6.5 Interface com os Objetos Gráficos

A interface de comunicação entre o objeto gráfico e o sistema é utilizada com objetivo de isolar o sistema preditivo dos problemas geométricos e topológicos. Esses problemas são específicos para cada tipo de objeto gráfico. Essa camada de abstração permite que sistema possa lidar

com qualquer objeto gráfico de uma forma única. O operador utilizado para a troca de informações entre o sistema e os objetos gráficos é chamado de `ObjectQuery`.

O operador `ObjectQuery` recebe um parâmetro que contém três campos. O primeiro determina o tipo de informação que o sistema deseja obter do objeto gráfico. O segundo armazena os parâmetros de entrada para o objeto gráfico produzir a informação de saída. O terceiro campo contém a informação de saída. A Tabela 6.5 mostra os tipos de informação de busca que o sistema necessita para gerenciar as aplicações de visualização. A

Tipos de Informação	
ViewWindow	Retorna a janela de visão.
TileClassify	Classifição dos ladrilhos.
TileAddress	Ladrilhos que devem ser carregados.
BestLevel	Nível de resolução ideal para uma janela de visão.

Tabela 6.2: Operadores que definem a interface de comunicação com os objetos gráficos.

Figura 6.24 mostra os campos de entrada e saída de cada tipo de informação solicitada pelo sistema. As operações de busca `ViewWindow`, `TileAddress` e `BestLevel` são utilizadas apenas pelo sistema de carregamento de páginas. A operação `TileClassify` é utilizada tanto pelo sitema de carregamento quanto pelo sistema de liberação de páginas.

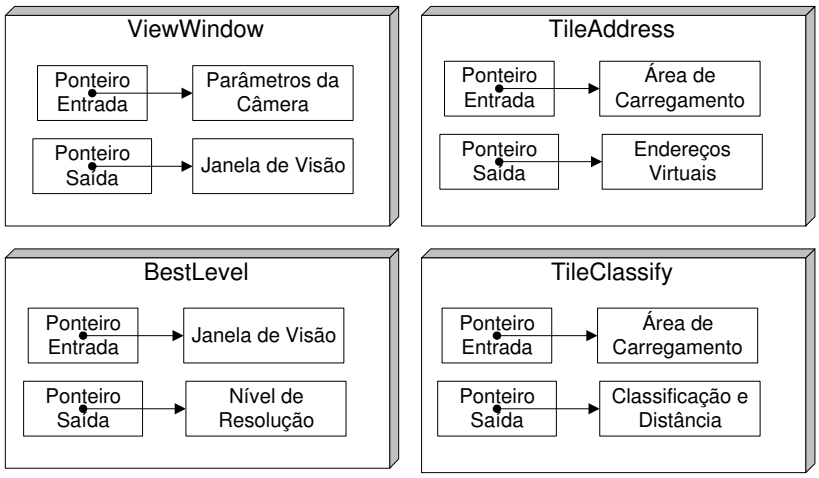

Figura 6.24: As estruturas de dados utilizadas nas quatro operações de busca definidas para as aplicações de visualização.

A informação `ViewWindow` é utilizada para converter os parâmetros da câmera virtual em uma janela de visualização. Essa informação é essencial

para o sistema de carregamento de páginas. O algoritmo de predição é implementado dentro desse sistema. O algoritmo de predição estima os parâmetros futuros da câmera e utilizada as informações de `ViewWindow` para determinar a região do objeto gráfico que deve ser carregada para a memória. Os parâmetros de entrada são os parâmetros da câmera e o parâmetro de saída são as coordenadas da janela de visualização.

A informação de busca `BestLevel` é utilizada para saber o melhor nível de resolução para uma determinada janela de visualização. Essa informação é importante para estimar o nível de resolução que deve ser carregado. O parâmetro de entrada são as coordenadas da janela de visualização e o parâmetro de saída é o nível de resolução.

A informação `TileAddress` retorna os endereços virtuais dos ladrilhos que devem ser carregados para o estágio mais alto de armazenamento. Os pâmetros de entrada são as coordenadas de uma região do objeto gráfico e o nível de resolução. O parâmetro de saída é um vetor com os endereços virtuais dos ladrilhos que estão dentro da região e que pertencem ao nível de resolução solicitado.

A informação de busca `TileClassify` informa se um ladrilho está dentro ou fora de uma determinada região e a distância que ele está dessa região. Essa informação é utilizada pelo sistema carregamento para poder cancelar os pedidos de carregamento que não são mais necessários. O sistema de liberação de páginas utiliza a mesma informação para estimar o prioridade de um ladrilho dentro do estágio de armazenamento. Os parâmetros de entrada são a região do objeto gráfico e um ladrilho. Os parâmetros de saída são a classificação (dentro, fora ou intersecta) e a distância do ladrilho para a região.

Observe que esse conjunto de informações de busca separa o sistema dos problemas relacionados com os cálculos de visibilidade. Esses cálculos são extremamente dependentes dos tipos de câmera virtual e de objeto gráfico utilizados no processo de visualização.