DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

UNIVERSIDADE ESTADUAL DE CAMPINAS

Produção de Documentos para a World-Wide Web

Ivan Luiz Marques Ricarte

1997

Sumário

Introdução

A rede de informação mundial World-Wide Web (WWW) [4, 5] é a primeira concretização de propostas de integração de informação espalhada através de todo o mundo. Tais propostas existem há décadas, mas apenas agora existe a tecnologia para concretizá-las. A descrição oficial descreve WWW como uma "iniciativa de recuperação de informação hipermídia em área ampla objetivando dar acesso universal a um grande universo de documentos." Efetivamente, a WWW implementou — através de mecanismos uniformes de descrição e acesso — um espaço abstrato de conhecimento no topo da Internet, de forma que integra milhões de usuários espalhados por dezenas de países.

A WWW foi inicialmente proposta em março de 1989 [2,8], dentro do contexto de um centro de pesquisa em física de alta energia (CERN), que agrega membros de diversos países europeus. Esta iniciativa espalhou-se de tal forma que a própria WWW vem tendo taxas de crescimento superior a qualquer outro serviço suportado pela Internet.

Mosaic foi a primeira interface gráfica de interação com a WWW, tendo sido desenvolvida no Centro Nacional para Aplicações em Supercomputação (NCSA) dos Estados Unidos. A primeira versão deste programa tornou-se publicamente disponível no início de 1993. Desde então, diversas interfaces gráficas *(browsers)* para a apresentação de documentos da WWW estão disponíveis, suportando interações com o usuário (dirigidas por mouse), apresentação de documentos hipertexto e hipermídia, conexão com aplicativos externos para a apresentação de imagens, filmes e sons em diversos formatos, interação através de mecanismos de formulários (preencher campos, checar opções) e conexão com outros serviços Internet.

A WWW funciona segundo um modelo cliente-servidor. Um servidor WWW é um programa sendo executado em um computador cujo propósito principal é servir documentos a outros computadores que façam pedidos. Um cliente WWW é um programa que suporta a interface com o usuário e requisita documentos a servidores. Um servidor WWW não consome muitos recursos computacionais, pois ele não deve realizar nenhum cálculo e trabalha sob demanda. Prevê-se o suporte, a nível de servidores, de habilidades para criptografia e autenticação de usuários, o que permitirá estender WWW para aplicações comerciais.

Nesta parte do documento, serão apresentados os princípios que determinam a estrutura de documentos da WWW. Inicialmente haverá uma apresentação sobre aspectos gerais de representação de documentos em computador, seguida pela descrição da linguagem de descrição de documentos utilizada pela WWW, HTML. Extensões ao modelo básico que permitem um maior grau de interação com usuários, CGI e Java, serão apresentadas na sequência.

Este documento é uma reprodução parcial da apostila Internet Básico*, de Eleri Cardozo, Maurício Ferreira Magalhães e Ivan Luiz Marques Ricarte, DCA/FEEC/UNICAMP, 1997.*

Capítulo 1

Padrões de Representação

Há décadas que sistemas computacionais vêm sendo utilizados não só para processamento numérico, seu objetivo primário, mas também para armazenar documentos. Neste período houve uma evolução notável na forma de representação computacional destes documentos. Os primeiros textos eram simplesmente sequências de caracteres, criadas por programas *editores de texto*. Em uma segunda geração de documentos, criados por programas *processadores de texto*, além do texto em si é possível também armazenar caracteres de controle que direcionam a apresentação do documento, tais como espaçamento e mudanças de fonte.

De modo geral, documentos eletrônicos atuais ainda apresentam um caráter "monolítico", com mecanismos muito restritos de interação com usuários. Um documento multimídia, por outro lado, permite ampliar as formas de interação com usuário ao integrar diversas mídias. Por exemplo, a forma mais simples de integração de diversos tipos de dados é através da presença de ponteiros a partir de um "corpo básico" para outras fontes de informação. Esta noção de documentos multidimensionais teve início com o conceito de *hipertexto* [9, 16], nos quais ligações *(links)* conectavam distintos segmentos de texto (nós). Atualmente, utiliza-se o termo *hipermídia* para caracterizar esta não-linearidade na representação da informação quando as ligações incorporam não apenas textos mas também outras formas de representação de informação.

As atividades associadas à manipulação de documentos eletrônicos — edição voltada a caracteres e inclusão de figuras relativamente simples — eram plenamente satisfeitas por programas de processamento de texto. Documentos multimídia, por outro lado, irão requerer mecanismos mais complexos de manipulação, principalmente quando se considerar que, devido ao grande volume de dados envolvidos na representação, tais documentos estarão distribuídos entre diversos sistemas, não necessariamente homogêneos. A manutenção consistente destes dados representa um investimento significativo, e é fundamental que este conjunto de informação permaneça disponível aos usuários. Não se deve permitir que esta informação se perca em consequência de incompatibilidades entre estruturas e formatos suportados pelas aplicações. Neste sentido, o esforço em direção à padronização de estruturas de documentos multimídia e hipermídia é essencial para o sucesso de tais aplicações.

Na sequência, serão apresentados os princípios que permitem a representação de documentos de estruturas complexas. Serão também apresentados os aspectos relacionados a outras formas de representação de informação, como imagens, áudio e vídeo.

1.1 Texto

Claramente, os mecanismos utilizados por processadores de textos para a representação de documentos eletrônicos — essencialmente sequências de caracteres com informação de formatação agregada — não são adequados para representar documentos multimídia. É preciso estender mecanismos existentes ou até mesmo estabelecer novos mecanismos de representação de forma a satisfazer as necessidades da representação da informação multimídia. Para entender as propostas atuais de representação, será apresentado inicialmente o formato mais flexível de representação de documentos eletrônicos, que é usado pela terceira geração de ferramentas de manipulação de textos — os *compositores de texto*.

1.1.1 Linguagens de marcação

A abordagem mais utilizada atualmente para a descrição de documentos complexos é a representação através de linguagens de marcações *(markup languages)*. Esta abordagem é adotada por compositores de texto para distinguir o texto a ser impresso (a informação) das instruções de como o texto deve ser impresso (as marcações).

MARCAÇÃO

O termo marcação *(markup)* é oriundo da área de edição e publicação, denotando as instruções de uma pessoa (o editor) que eram escritas para o tipógrafo em um manuscrito ou cópia de um documento. Uma marcação é tudo que está presente em um documento que não é conteúdo, ou seja, que não é exibido na apresentação final do documento. Embora tradicionalmente a tarefa de incluir marcações no texto seja tarefa de um editor ou de um revisor, diversos programas de editoração eletrônica permitem a inclusão de marcações pelo próprio usuário que está criando o documento (o autor). Por exemplo, o programa `troff` (tradicional compositor de texto disponível em sistemas Unix) usa comandos de uma ou duas letras precedidos por um ponto, como o comando `.ce` para centralizar a linha seguinte de texto. O compositor de textos LaTeX (disponível para diversas plataformas computacionais e que foi utilizado para compor este material) permite a mesma funcionalidade através do comando `\centerline{...}`, com o texto a ser centralizado entre chaves.

MARCAÇÃO PROCEDIMENTAL

Marcações podem ser classificadas como procedimentais ou genéricas. Marcações procedimentais são aquelas contém informação sobre o que fazer no momento da impressão — centralizar o texto na linha, mudar o tipo ou tamanho de fonte, introduzir espaço extra no sentido horizontal ou vertical. Em outras palavras, o autor do documento atua diretamente sobre o *layout* final de apresentação do documento. A utilização de marcações procedimentais introduz completa flexibilidade sobre a forma de apresentação do documento. O autor do documento pode escolher qual a família do fonte usado para representar o título do documento, títulos de seções, posicionamento, espaçamento, etc.

Em documentos curtos, tais com uma carta ou um memorando, é fácil obter o resultado desejado com marcações procedimentais. Entretanto, esta flexibilidade pode trazer dificuldades quando se busca uma uniformidade de estilo de documentos gerados por uma empresa, por exemplo. Também torna-se difícil manter uma representação uniforme no estilo durante a manipulação de documentos longos e no trabalho de diversos autores de um mesmo documento. Além do mais, se a forma de apresentação tiver que ser alterada, o conteúdo do arquivo deve ser reeditado. Outra dificuldade é garantir que um recurso requerido — por exemplo, um fonte de estilo e tamanho específicos — estará disponível para a apresentação ou impressão do documento.

MARCAÇÃO GENÉRICA

Marcações genéricas (ou marcações descritivas) são indicações semânticas, que dão informação sobre o significado associado a um trecho de texto, tais como título e autor do documento, nome de capítulos e de seções. Neste caso, o autor não se preocupa com a apresentação e indica simplesmente o que deve ser impresso, sem entrar em detalhes de como isto deverá ser realizado. O *layout* de apresentação é responsabilidade do programa de composição de documentos. Uma marcação genérica é delimitada no texto por rótulos *(tags)*. O programa de composição, que traduz um texto com conteúdo e marcações genéricas para um texto apenas como conteúdo na forma final de apresentação, estabelece uma correlação entre estes rótulos e a aparência final do texto. Por exemplo, uma marcação de título de capítulo pode estabelecer para um estilo de apresentação final o início de uma nova página, uma mudança no tipo de letra usada, e espaço adicional após o texto do título antes de se restaurar o estilo de texto normal. Para outro estilo de apresentação final, a mesma marcação pode ser interpretada de forma diferente.

Marcações genéricas permitem que o autor do documento não se preocupe com os aspectos da apresentação, concentrando-se essencialmente no conteúdo. Uma das vantagens em sua utilização é que a representação do documento é independente de formatos de apresentação ou dos recursos disponíveis para a impressão ou visualização. Por exemplo, duas editoras com estilos totalmente distintos de impressão (fontes utilizados para textos e títulos, espaçamentos, formato de referências) poderiam utilizar exatamente a mesma representação de um documento com marcações genéricas. Outra vantagem na utilização de marcações genéricas é que algumas consultas podem ser realizadas sobre os documentos como se estes fossem um tipo de banco de dados. Por exemplo, se *autor*, *título do documento* e *título de capítulo* estivessem entre as marcações genéricas suportadas por um sistema de documentos, uma possível consulta seria "Mostre os autores e os títulos dos documentos cujos títulos de capítulos contenham as palavras *multimídia* e *documentos*." Este tipo de consulta já é possível em publicações eletrônicas em CD-ROM usualmente disponíveis em bibliotecas, embora muitos destes sistemas ainda realizem buscas sobre textos completos, que não permite especificar em que tipo de contexto se busca uma ocorrência.

LINGUAGEM DE MARCAÇÃO

A indicação de quais marcações são suportadas por um sistema de documentação eletrônica é dada pela linguagem de marcação utilizada. Informalmente, uma *linguagem de marcação* é um conjunto de construções usadas para expressar como o texto (isto é, tudo que não é marcação) deve ser processado ou manipulado. Uma característica particular de linguagens de marcações é a inclusão de regras para diferenciar as marcas dos dados (texto). Por exemplo, o programa compositor de documentos LaTeX define que toda instrução da linguagem (marcação) inicia-se com uma contrabarra, como `\centerline`. A maior parte destas linguagens de marcação oferece mecanismos para adicionar outras construções à linguagem através da definição de macros. Na sequência, uma estrutura padronizada internacionalmente para a definição de marcações genéricas é apresentada.

1.1.2 SGML

SGML *(Standard Generalized Markup Language)* [1, 15] é um padrão internacional (ISO 8879) adotado em 1986 com o propósito de atender a necessidades associadas à descrição de documentos em termos de sua estrutura. Este padrão deriva-se de GML, criada por Charles Goldfarb, Edward Mosher e Raymod Lorie em 1969 na IBM. SGML está principalmente voltada para a definição de famílias de documentos, mais do que para a representação de documentos individuais — embora possa ser utilizada para este fim. Assim, SGML não tem um conjunto fixo de rótulos de marcação,

mas permite criar marcações voltadas para virtualmente qualquer tipo de documento. SGML é frequentemente citada como sendo uma metalinguagem, pois ela é de fato uma linguagem que possibilita a criação de marcações genéricas específicas para uma dada aplicação — ou seja, é uma linguagem que permite criar outras linguagens de marcações que devem obedecer a formas e mecanismos padronizados.

SGML separa um documento em três camadas, que são estrutura, conteúdo e estilo. A *estrutura* de um documento é definida através de uma *Definição de Tipo de Documento* (DTD), que determina quais elementos podem estar presentes na descrição de um documento e qual o interrelacionamento entre estes elementos, de forma que a estrutura lógica do documento seja válida e consistente.

DTD

O *conteúdo* do documento SGML é a informação propriamente dita, composta por texto e por rótulos ao redor de partes do texto (as marcações). O conteúdo de um documento é estruturado na forma de *elementos*, tais como título e parágrafos. Um elemento é a menor parte de um documento manipulável por SGML, sendo o bloco básico de construção de tipos de documentos. Um elemento pode

ELEMENTO

1. conter dados, onde um dado de um elemento é simplesmente texto;
2. conter outros elementos, que são os *subelementos* componentes de cada elemento;
3. conter combinações de dados e subelementos; ou
4. ser vazio.

O *estilo* é uma característica do documento independente de SGML. O formato de apresentação de um documento (impresso ou visualizado na tela de um computador) é função de um programa de composição que interpreta os elementos de um documento e define como cada elemento deve ser apresentado. Há um outro padrão internacional (ISO 10179), DSSSL *(Document Style Semantics and Specification Language)* que aborda a padronização de estilos de documentos.

A tarefa de um projetista de tipos de documentos é identificar os elementos que constituirão um documento e definir uma estrutura hierárquica destes elementos através de outros elementos básicos. Uma definição de elemento consiste de um nome (um identificador genérico) que será utilizado em rótulos, uma descrição de conteúdo (usando um "modelo de conteúdo") e uma indicação sobre o que deve estar presente e o que pode ser omitido na marcação do elemento. Um elemento no documento instância é geralmente indicado por um rótulo de início, conteúdo e um rótulo de final. O conjunto destas definições de elementos, a DTD, não tem por objetivo ser interpretada por seres humanos, mas sim por um programa de computador chamado *parser*.

O conceito de elemento, juntamente com a noção de modelos de conteúdo, fornece uma abstração poderosa sobre os diferentes tipos de texto que podem ser encontrados em um documento. Por exemplo, texto ordinário é apenas um conjunto de caracteres que será formatado de algum modo para saída. Outros elementos podem receber tratamento diferenciado.

Considere uma aplicação onde os documentos fossem cadastros de clientes de uma empresa, onde o telefone do cliente poderia ser um texto com um tratamento especial. Neste caso, faria sentido que o projetista do tipo de documento *cadastro* incluísse na linguagem de marcação o elemento *telefone*, que poderia ser descrito no documento como

```
<fone><codp>55</codp><area>19</area><pref>239</pref>4313</fone>
```


A partir desta representação, seria possível por exemplo localizar clientes com telefones de contato no Brasil (código de país 55) ou na região de Campinas (código de país 55 e código de área 19). Ainda mais, seria possível definir uma notação associada a este elemento de forma a adaptar a apresentação de acordo com necessidades particulares. Por exemplo, este mesmo elemento poderia ser apresentado na forma +55-19-239 4313 em um documento voltado para distribuição internacional ou como (019) 239 4313 em um documento voltado para distribuição nacional.

ENTIDADE

Uma outra noção importante em SGML é o conceito de entidades. Uma *entidade* é, basicamente, um segmento de texto que pode ser referenciado por um nome. Entidades podem ser internas (abreviaturas para construções da linguagem, texto que não deve ser interpretado) ou externas (texto, entidades em uma notação especial a ser interpretada por um programa específico, etc).

SGML separa a informação (o conteúdo de um documento) da apresentação, o que permite que um mesmo documento possa ser apresentado de diversas maneiras, mesmo entre sistemas com características de hardware muito diferentes. Esta abordagem é essencialmente oposta à abordagem WYSIWYG (*what you see is what you get*, isto é, o que você vê é o que você obtém). Em SGML, sabe-se o que um segmento de texto é (por exemplo, um título de seção) mas não se sabe como este segmento será apresentado; em WYSIWYG, por outro lado, conhece-se a aparência de um segmento de texto, mas não é possível determinar o que é aquele segmento. Outra vantagem de SGML é sua flexibilidade, suficiente para permitir a representação de documentos com estrutura complexa e permitir ligações entre elementos de documentos e outras aplicações, tais como bases de dados.

Há diversas aplicações de SGML em uso atualmente, mas certamente uma delas tem recebido destaque muito maior recentemente — a linguagem HTML, usada para descrever os documentos na WWW.

1.2 Imagens

Uma das formas tradicionais de incorporação de imagens a documentos multimídia é através de um processo de digitalização da imagem, realizado com o auxílio de um *scanner* ou câmera digital. Uma imagem digitalizada é representada por um conjunto de *pixels*, sendo cada pixel associado a um ponto da imagem digital com a correspondente informação sobre a cor.

RESOLUÇÃO

Uma das características fundamentais de uma imagem digital é sua resolução, ou seja, qual a quantidade de pixels que será utilizada para representar a imagem. Normalmente, a resolução é expressa em termos da unidade *dpi (dots per inch)*, ou pontos por polegada, assumindo usualmente valores da ordem de 300 ou 600 dpi. Quanto maior o valor da resolução, maior será o tamanho do arquivo contendo a imagem digitalizada.

CORES

Outra característica importante é a quantidade de cores representáveis em uma imagem digitalizada. Atualmente, o padrão mais fiel de representação de cores é o chamado *true color*, onde a cada pixel a informação de cor é representada por 24 bits. Desta forma, é possível representar até 2^{24} cores, ou pouco mais de 16 milhões de cores diferentes. O padrão mais simples utiliza apenas um bit por pixel, representando apenas preto ou branco. Há também previsão no uso de até 48 bits por pixel para a representação de cores.

Finalmente, a imagem digitalizada deve ser salva em um arquivo para acesso posterior. O formato deste arquivo, como cada parte da informação deve ser salva, pode variar. Muitas vezes, estes formatos incorporam informação de forma comprimida com o fim de reduzir o tamanho do

arquivo com a imagem. Há diversos formatos disponíveis para a representação de imagens digitais.

GIF *(Graphics Interchange Format)* foi um dos formatos mais utilizados para a transferência eletrônica de imagens por muito tempo. É aceito por grande parte de visualizadores de documentos. Este formato utiliza oito bits para descrever cada pixel, de modo que suporta até 256 cores distintas apenas, o que implica em perca de qualidade para imagens realísticas (fotografias, por exemplo). Dois padrões GIF são normalmente referenciados, GIF87 (original) e GIF89a. Em GIF89a, o fundo da imagem pode ser definido como transparente, o que permite mesclar de forma natural uma imagem em um documento com outro padrão de fundo qualquer. GIF

Atualmente, o uso de GIF não tem sido recomendado. Além da baixa resolução, GIF utiliza um padrão de compressão proprietário, patenteado pela Unisys. Assim, se alguém quiser desenvolver software usando este padrão deve pagar taxas, o que vem desencorajando seu uso em muitas aplicações mais recentes.

JPEG *(Joint Photographic Experts Group)* é um padrão de compressão para imagens estáticas que é adequado para imagens originais em *true color*. A partir de uma mesma imagem original de alta qualidade, JPEG resulta em arquivos até dez vezes menores que arquivos GIF e ainda com melhor apresentação em monitores capazes de mostrar imagens *true color*. No entanto, o algoritmo de compressão usado em JPEG apresenta perdas — ou seja, não é possível restaurar a imagem com a qualidade original. É um algoritmo excelente para trabalhar com imagens realísticas, mas aplicar a compressão JPEG a uma imagem já em baixa qualidade — GIF, ou um desenho de linhas, ou a uma imagem simples como um logotipo — não apresentará bons resultados. O formato de arquivo usado para armazenar uma imagem JPEG é chamado de JFIF *(JPEG File Interchange Format)* [11]. JPEG apresenta algumas características opcionais que dependem de patentes proprietárias — estas características não são normalmente utilizadas em software público ou na World-Wide Web. JPEG

A recomendação do consórcio WWW para a representação de imagens é o formato PNG[1] *(Portable Network Graphics)* [7, 14]. O objetivo é substituir gradualmente o uso de GIF por PNG, que tem como principais características: PNG

- representação eficiente (indexada) de até 256 cores, mas com capacidade para suportar até 48 bits por pixel;
- possibilidade de apresentação progressiva, começando por uma imagem de baixa qualidade que pode ter sua resolução melhorada progressivamente;
- qualquer parte da imagem pode ser definida como transparente, criando um efeito de imagens não retangulares;
- mecanismo de compressão (público) sem perdas.

Outros formatos relacionados a imagens incluem BMP *(Microsoft BitMap File)*, PBM *(Portable Bitmap File)*, PNM *(Portable aNy Map File)*, PPM *(Portable Pixel Map File)*, TIFF *(Tagged Interchange File Format)* e PostScript. TIFF foi inicialmente registrado como um formato para imagens de fac-símiles, mas tem sido estendido para diversos outros tipos de imagens. Por muito tempo, arquivos em formato TIFF foram utilizados para representar imagens em 24 bits por pixel sem perdas, mas atualmente PNG assume este papel. PostScript é na verdade uma linguagem para descrição de páginas, normalmente usado por diversas impressoras. Assim, um arquivo PostScript TIFF POSTSCRIPT

[1] Pronuncia-se *ping*.

é na verdade um programa que será interpretado pelo dispositivo de apresentação. Apesar de ser um formato proprietário (Adobe), seu uso é livre e há diversos aplicativos publicamente disponíveis para visualizar o conteúdo de arquivos PostScript. Atualmente, PostScript vem sendo gradualmente substituido por outro formato da Adobe mais flexível e compacto, PDF *(Portable Document Format)*.

1.3 Áudio

A digitalização de sinais de áudio, da mesma forma que imagens, também passa por um processo que implica em alguma perda de qualidade. O processo de digitalização é resultado de uma amostragem seguida por quantização e codificação. A amostragem é determinada por uma frequência que especifica qual a distância (em tempo) entre duas amostras consecutivas no sinal original (analógico) que está sendo digitalizado. Quanto maior a frequência de amostragem, melhor será a qualidade resultante e maior será o arquivo gerado. Na quantização, os valores associados a estas amostras são mapeados a um conjunto de valores pré-determinados. Os valores distintos neste conjunto são denominados níveis de quantização, e o número de níveis depende de quantos bits serão utilizados para representar cada valor da amostra — valores usuais são 8 e 16 bits por amostra. Finalmente, na codificação se especifica como cada amostra deve ser representada neste conjunto de bits.

Embora a possibilidade de combinações de parâmetros seja virtualmente infinita, existe uma tendência no sentido de buscar uma uniformização nas combinações de frequências de amostragem e número de bits usados na codificação das amostras [17]. As combinações mais utilizadas são 8 KHz com 8 bits (um padrão usado em telefonia digital nos Estados Unidos, U-LAW), 22,05 KHz com 8 bits (“meio CD”) e 44,1 KHz com 16 bits (valores utilizados em CDs de áudio).

Alguns dos formatos mais utilizados para representação de áudio são AU (Next/Sun), RIFF WAVE (Microsoft/IBM) e AIFF (Apple). O formato da Next e da Sun, AU, é usualmente utilizado em estações de trabalho Unix, em arquivos com extensão `.au` ou `.snd`. Neste formato, o arquivo contém um cabeçalho com a seguinte estrutura (em sintaxe C):

AU

```
typedef struct {
    int magic;              /* magic number SND_MAGIC */
    int dataLocation;       /* offset or pointer to the data */
    int dataSize;           /* number of bytes of data */
    int dataFormat;         /* the data format code */
    int samplingRate;       /* the sampling rate */
    int channelCount;       /* the number of channels */
    char info[4];           /* optional text information */
} SNDSoundStruct;
```

O primeiro campo, `magic`, é um número inteiro que identifica o tipo de arquivo e deve ser igual à sequência hexadecimal `2e736e64`, que corresponde à codificação ASCII da string “.snd”. O segundo campo, `dataLocation`, é um inteiro que indica em que posição do arquivo (em bytes a partir do início do arquivo, com valor mínimo de 28 — o menor tamanho possível para este cabeçalho) começa efetivamente a sequência de áudio, sendo que o terceiro campo descreve o tamanho desta sequência. O campo `dataFormat` é um código que identifica o tipo de

sons — normalmente, o tipo de quantização que foi utilizado. Os valores mais comuns para este campo são 1 (U-LAW, 8 bits) e 3 (amostras lineares de 16 bits). A taxa de amostragem é indicada por um código no campo `samplingRate`, podendo assumir valores representados pelas constantes `SND_RATE_CODEC` (8012,821 Hz, CODEC input), `SND_RATE_LOW` (22050 Hz, low sampling rate output) ou `SND_RATE_HIGH` (44100 Hz, high sampling rate output). O campo `channelCount` é uma indicação do número de canais utilizados, normalmente 1 (mono) ou dois (estéreo). Finalmente, o cabeçalho do arquivo termina com uma sequência de caracteres em padrão C (terminada pelo caracter NUL) contendo um texto que pode ser usado para documentar o conetúdo do arquivo. Mesmo que esta sequência não esteja presente, quatro bytes são reservados para ela no cabeçalho. Os dados de áudio vêm na sequência, logo após o fim do cabeçalho.

O formato RIFF WAVE *(Waveform Audio File Format)* [12] foi desenvolvido pela Microsoft e pela IBM, sendo de uso mais amplo na plataforma Microsoft Windows. O arquivo começa com a sequência de caracteres `WAVE`, seguida por um cabeçalho, o *WAVE Format Chunk*, que especifica o formato dos dados de áudio na sequência. O cabeçalho permite especificar a categoria do formato utilizado, o número de canais (1 ou 2), a taxa de amostragem (expressa em amostras por segundo) e o número médio de bytes por segundo (para fins de estimativa do tamanho de buffer necessário para receber estes dados). As categorias de formatos mais comuns são Microsoft Pulse Code Modulation (sequência hexadecimal 0001), IBM mu-law (0101), a-law (0102) e AVC Adaptive Differential Pulse Code Modulation format (0103). Campos específicos para cada um destes formatos podem vir na sequência do cabeçalho — por exemplo, descrevendo o número de bits por amostra. Os dados de áudio vêm na sequência. Estes dados são precedidos por uma string `wavl` *(wave list)*, sendo cada amostra representada por um valor inteiro ou por uma especificação de quantidade de amostras equivalente a "silêncio." É possível agregar também documentação ao arquivo de áudio.

RIFF WAVE

O formato AIFF *(Audio Interchange File Format)* foi desenvolvido pela Apple para a representação de músicas e sons, sendo de uso mais amplo em plataformas Macintosh e em máquinas Silicon Graphics. AIFF-C ou AIFC é uma extensão de AIFF que suporta compressão de dados. Há também arquivos que usam o formato MOD, que foi desenvolvido para representar músicas na plataforma Amiga mas é atualmente suportado para diversas outras plataformas.

1.4 Vídeo

A representação de vídeo combina aspectos da representação de imagens (um vídeo é uma sequência de imagens) com aspectos da representação de áudio (é preciso ser apresentado em tempo real para manter significado). Características típicas de arquivos de vídeo são resolução e número de quadros que devem ser exibidos por segundo (a amostragem). Um filme em cinema tem altíssima resolução (qualidade fotográfica em cada quadro) com vinte e quatro quadros exibidos a cada segundo; televisão normal tem uma resolução muito mais baixa (próxima a 600 pixels na dimensão vertical) com exibição a 30 quadros por segundo; e a chamada televisão de alta definição tem propostas em torno de 2048×1024 pixels a 30 quadros por segundo.

Não surpreendentemente, alguns vídeos em formato digital são representados como sequências de imagens paradas — normalmente comprimidas em formato JPEG, resultando em um formato de vídeo tipicamente referenciado como M-JPEG *(motion JPEG)* [6]. Este formato não é padrão, apesar de muito utilizado devido à facilidade de se obter hardware para comprimir e descomprimir arquivos em JPEG. Uma das desvantagens deste formato é que ele não detecta a semelhança entre

M-JPEG

duas imagens consecutivas, enviando sempre a imagem completa para cada quadro.

MPEG

O formato MPEG *(Motion Picture Experts Group)* [10] busca exatamente se aproveitar desta semelhança entre quadros consecutivos (a chamada redundância temporal) para reduzir o volume de dados associado ao vídeo. MPEG obtém uma boa taxa de compressão usando, além de quadros codificados com a informação sobre a imagem completa, dois tipos de quadros contendo apenas informação parcial: quadros preditivos (com diferenças em relação a um quadro futuro) e quadros interpolados (com diferenças relativas a um quadro passado e a outro futuro).

NÍVEIS MPEG

O padrão MPEG estabelece quatro níveis diferentes de aplicações. MPEG-1 é o padrão voltado para aplicações típicas em CD-ROMs. MPEG-2 é uma extensão de MPEG-1 para uma classe mais ampla de aplicações, suportando de forma eficiente a codificação de vídeos em padrão de televisão (imagens entrelaçadas). MPEG-3 era a proposta de extensão de MPEG-2 para televisão de alta definição, mas atualmente esta proposta foi incorporada a MPEG-2, de forma que este nível foi desativado. Finalmente, MPEG-4 é uma proposta ainda em andamento para vídeos de baixa resolução, usando taxas de transmissão de até 64 Kbits por segundo, de aplicação típica em sistemas de vídeo-conferência.

Outros formatos usuais de vídeo digital incluem H.261 (recomendação ITU-T para vídeo-conferência), QuickTime (formato desenvolvido pela Apple) e AVI (formato da plataforma Microsoft Windows similar a QuickTime).

Capítulo 2

HTML

HTML *(HyperText Markup Language)* [3] é a linguagem adotada para a definição de documentos multimídia e hipermídia acessados através da WWW. Especificamente, HTML é definida através de uma DTD-SGML, ou seja, é uma coleção de elementos definidos em SGML usados para descrever documentos da WWW.

Um documento HTML permite integrar diversos tipos de mídia, embora sua representação contenha apenas texto. Assim, um documento HTML pode ser criado com qualquer editor de textos ASCII. Um *browser* ou visualizador WWW (por exemplo, Mosaic, Netscape, Cello, WebExplorer, Lynx) é um programa utilizado para apresentar o documento em uma dada plataforma computacional, devendo suportar a apresentação do texto dos documentos e, opcionalmente, de suas imagens (internas ou externas ao documento), filmes, animações e sons, de acordo com os recursos disponíveis no sistema onde a apresentação é realizada.

Atualmente, a recomendação oficial de HTML está na versão 3.2 [13], que foi desenvolvida em 1996. A proposta anterior, 3.0, não conseguiu ser padronizada dentro do tempo previsto por conter diversas características adicionais em relação ao padrão anterior (2.0) sobre as quais não se conseguiu atingir um consenso — por exemplo, a representação de fórmulas matemáticas. Assim, a versão 3.2 foi proposta com um número menor de adições em relação ao padrão HTML 2.0, mas com maior probabilidade de aceitação, e de fato tornou-se uma recomendação oficial em janeiro de 1997.

2.1 Estrutura de documentos

Como todo documento SGML, um documento HTML é composto por diversos elementos demarcados por rótulos. Um elemento é composto por um rótulo de início de elemento, pelo conteúdo do elemento e por um rótulo de fim de elemento. Seguindo o padrão SGML, cada rótulo é denotado pelo símbolo “menor-que” (<), seguido por algum texto (a diretiva que indica o nome do elemento) e encerrado pelo símbolo “maior-que” (>). Após o rótulo de início de elemento, o conteúdo do elemento é apresentado. O final do elemento é determinado pelo rótulo de finalização, similar ao rótulo de inicialização mas com a diferença que o nome do elemento é precedido por uma barra (/). Por exemplo,

```
<P>Algum texto</P>
```

é um elemento *parágrafo* cujo conteúdo é *Algum texto*. Para HTML, não há diferença entre letras maiúsculas e minúsculas no nome do rótulo, de forma que o exemplo anterior é equivalente a

```
<p>Algum texto</p>
```

Um rótulo pode conter conter atributos, que são descritos juntamente com o rótulo de início de elemento, como em

```
<p align=center>texto centralizado</p>
```

A recomendação HTML 3.2 determina que um documento é composto por um elemento HTML precedido por uma declaração DOCTYPE, que identifica o tipo de documento. Por sua vez, o elemento HTML é composto por dois elementos, HEAD e BODY, como em: HTML DOCTYPE

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
```

. . . sub-elementos do cabeçalho do documento

```
</HEAD>
<BODY>
```

. . . sub-elementos do corpo do documento

```
</BODY>
</HTML>
```

Na prática, a declaração de tipo e os rótulos HTML, HEAD e BODY podem ser omitidos, uma vez que a maior parte dos programas de interpretação HTML conseguem deduzir estes rótulos a partir dos demais elementos. Entretanto, a recomendação HTML 3.2 determina que todo documento HTML que está de acordo com a especificação deve conter pelo menos o título do documento e a declaração de tipo que permite distinguir documentos nesta versão de outros documentos. Assim, na recomendação HTML 3.2 um documento mínimo tem a seguinte forma:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<TITLE>Um exemplo muito simples de documento HTML</TITLE>
```

2.2 Cabeçalho

O elemento HEAD tem por conteúdo o cabeçalho do documento, sendo que os rótulos de início e de final deste elemento podem ser omitidos. O conteúdo do cabeçalho é uma coleção de outros elementos, descritos na sequência, e não aparece no conteúdo do documento na sua apresentação final. HEAD

O elemento TITLE define o título do documento, devendo estar sempre presente. Requer os rótulos de início e de final de elemento. Na visualização de um documento HTML por um browser como Netscape, o título do documento aparece no topo da janela (e não no interior da janela, cujo conteúdo é totalmente descrito no elemento BODY). TITLE

O elemento `STYLE` é reservado para uso futuro[1], estando relacionado ao conceito de folhas de estilos [18], conjuntos de definições relacionadas com a aparência do documento. STYLE

O elemento `SCRIPT` é reservado para uso futuro relacionado com linguagens de script. SCRIPT

O elemento `ISINDEX` é um elemento que indica que o usuário (leitor) do documento deverá entrar uma linha de texto que será utilizada como uma *string* de busca. O processamento desta *string* é realizado pelo servidor que contém o documento, normalmente através de CGI. Este elemento aceita o atributo `prompt`, como em ISINDEX

```
<ISINDEX PROMPT="Procure por">
```

O elemento `BASE` determina o URL base para a resolução de URLs relativos no documento. Um URL base é uma especificação completa do endereço de um documento, como em BASE

```
<BASE href="http://www.dca.fee.unicamp.br/Welcome.html">
```

enquanto que uma URL relativa contém referências parciais a arquivos. O elemento `BASE` permite que estas referências sejam resolvidas corretamente, mesmo quando o documento que contém as referências relativas tiver sido copiado para outra localização.

O elemento `META` é utilizado para descrever informação adicional sobre o documento (ou *metainformação*) na forma de pares de atributos `name` e `content`, como em META

```
<META NAME="Author" CONTENT="Ivan L. M. Ricarte">
```

O elemento `LINK` é um mecanismo para estabelecer conexões entre o documento corrente e outros documentos ou recursos. Este elemento aceita os atributos LINK

`href` o URL especificando o recurso relacionado;

`rel` o relacionamento direto, também denominado de *tipo de ligação*. Embora não padronizado, algumas convenções são usadas para valor deste atributo, tais como *top* (referência para o documento topo de uma hierarquia), *contents* (para o documento que contém uma tabela de conteúdos para um grupo de documentos), *index* (para um documento de índice), *glossary* (para um glossário de termos relevantes para o documento corrente), *copyright* (para uma mensagem de *copyright*), *next* (para um próximo documento em uma sequência), *previous* (para um documento anterior na sequência), *help* (para um documento de ajuda) e *search* (para um documento que permitirá a busca de material relacionado ao documento corrente);

`rev` estabelece o relacionamento inverso, ou seja, se há um link direto (`rel`) de um documento A para um documento B com valor *xyz*, o mesmo relacionamento pode ser expresso como um relacionamento inverso `rev=xyz` do documento B para o documento A. Algumas vezes, o relacionamento inverso `rev=made` é usado para identificar o autor do documento, com referência para a sua página pessoal ou e-mail;

`title` estabelece um título indicativo para o recurso relacionado.

O elemento `LINK` ainda é ignorado por diversos *browsers*.

[1] Alguns *browsers*, como Netscape Navigator 4.0, já suportam o conceito de estilo.

2.3 Corpo

O corpo de um documento contém tudo o que será apresentado do documento. Ele é composto por um elemento `BODY`, o qual por sua vez é composto por diversos outros elementos. Os elementos que podem fazer parte do elemento `BODY` podem ser do tipo *cabeçalho de seção*, de nível *bloco* (que causam uma quebra de parágrafo), de nível texto (que não causam quebra de parágrafos) ou o elemento `ADDRESS`.

2.3.1 O Elemento BODY

O elemento `BODY` contém o corpo do documento, sendo que os rótulos de início e fim do elemento podem ser omitidos. Quando presente, este elemento pode conter os seguintes atributos opcionais: BODY

`bgcolor` especifica a cor de fundo para o corpo do documento;

`text` especifica a cor do texto;

`link` especifica a cor usada para indicar *links* de hipertexto ainda não visitados;

`vlink` especifica a cor usada para indicar *links* de hipertexto já visitados;

`alink` especifica a cor usada para indicar *links* de hipertexto ativos, ou seja, a cor do texto destes links no momento em que são selecionados;

`background` permite especificar o URL de uma imagem que será utilizada (de forma repetitiva) como imagem de fundo do documento ao invés de uma cor sólida.

Cores podem ser descritas por nomes no padrão VGA — *black*, *silver*, *gray*, *white*, *maroon*, *red*, *purple*, *fuchsia*, *green*, *lime*, *olive*, *yellow*, *navy*, *blue*, *teal* e *aqua* — ou por níveis de RGB (vermelho, verde e azul) expressos como três pares de números hexadecimais, como em *#008080* para uma cor verde-azulada *(teal)*.

2.3.2 Elementos de cabeçalho de seção

Os elementos `H1`, `H2`, `H3`, `H4`, `H5` e `H6` definem cabeçalhos de seções no documento, sendo que `H1` é o cabeçalho mais importante (de maior destaque) enquanto que `H6` é o cabeçalho menos importante (de menor destaque). Elementos de cabeçalho devem sempre ter presentes os rótulos de início e de fim do elemento. H1...H6

Cabeçalhos de seção aceitam o atributo `align`, que pode assumir os valores *left* (o padrão), *center* ou *right*. Por exemplo,

```
<H1 align=center>Titulo</H1>
```

irá apresentar `Titulo` centralizado entre as margens esquerda e direita da janela do *browser*. Um cabeçalho é um elemento com comportamento de nível bloco, ou seja, ele estabelece uma quebra de parágrafo na apresentação do documento.

2.3.3 Elemento de endereço

O elemento de endereço ADDRESS especifica informação sobre a autoria e detalhes de contato para o documento corrente. Este elemento requer os rótulos de início e de fim de elemento. Um exemplo de elemento ADDRESS é: ADDRESS

```
<address>
Ivan L. M. Ricarte<br>
DCA/FEEC/UNICAMP<br>
<a href="mailto:ricarte@dca.fee.unicamp.br">
ricarte@dca.fee.unicamp.br</A>
</address>
```

2.3.4 Elementos de nível bloco

O elemento P define parágrafos. Este elemento requer o rótulo de início de elemento, mas o rótulo de final de elemento é opcional[2]. A apresentação do elemento depende do *browser*, mas normalmente o texto de um parágrafo preenche a largura disponível na janela, alinhado à esquerda e não justificado à direita. O atributo `align` pode receber valores *left* (padrão), *center* e *right* para definir alinhamentos à esquerda, centralizado ou à direita, respectivamente. P

HTML suporta três tipos de listas, que são a lista numerada, a lista não numerada, e a lista descritiva. Cada lista é composta por vários itens, sendo que os itens da lista podem ser compostos de vários parágrafos ou por sublistas.

O elemento UL *(Unordered List)* define uma lista não numerada, requerendo os rótulos de início e de fim de elemento. O conteúdo (itens) da lista é representado por um conjunto de elementos LI *(List Item)*. O rótulo de fim de item de lista pode ser omitido. Os elementos UL e LI podem ter o atributo `type`, que indica o tipo de símbolo que irá preceder cada item da lista. Os valores válidos para este atributo são *disc* (círculo preenchido), *square* (um quadrado preenchido) e *circle* (círculo vazado). O padrão depende do nível de aninhamento da lista. O elemento UL pode também ter o atributo `compact`, que não recebe nenhum valor e funciona como uma indicação para o *browser* realizar uma apresentação da lista em um estilo mais compacto. UL LI

O elemento OL *(Ordered List)* define uma lista cujos itens são numerados, requerendo os rótulos de início e de fim de elemento. Os atributos aceitos para este elemento são `compact` (como em UL), `type` e `start`. O atributo `type` pode assumir os valores *1* para representar numeração arábica, *a* para representar ordenação alfabética em letras minúsculas, *A* para ordenação alfabética em letras maiúsculas, *i* para ordenação em algarismos romanos expressos em letras minúsculas e *I* para ordenação em algarismos romanos expressos em letras maiúsculas. O atributo `start` recebe como valor um número que indica o valor inicial da sequência de numeração (1, por padrão). Os itens da lista também são representados por elementos LI, que podem receber os atributos `type` e `value`, sendo que este último permite especificar um valor específico para um item da lista. Os atributos `start` e `value` são os únicos mecanismos para estabelecer continuação de lista anterior e para pular itens de uma lista. OL

O terceiro tipo de lista é definida pelo elemento DL *(Definition List)*, que aceita apenas o atributo DL

[2] Alguns *browsers* ainda permitem omitir completamente este rótulo, considerando como parágrafos sequências de caracteres separadas por linhas em branco. Esta prática, no entanto, não é recomendada.

`compact`. Cada item desta lista é composto pelo par de elementos `DT` e `DD`, que contém o nome da definição e a descrição da definição, respectivamente. Estes subelementos não aceitam atributos. `DT` pode conter apenas elementos de nível texto, enquanto `DD` também aceita elementos de nível bloco (mas não elementos cabeçalhos de seção ou elementos de endereço). DT, DD

Os elementos `DIR` e `MENU` estavam previstos como formas alternativas de listas não numeradas, mas na prática estes elementos têm apresentação idêntica ao elemento `UL`. DIR, MENU

Normalmente o *browser* tem a liberdade de rearranjar o texto para a apresentação. Para apresentar um segmento de texto com preservação de espaços e quebras de linha originais, este segmento deve ser incluído em um elemento `PRE` *(Preformatted)*. O conteúdo deste elemento é apresentado em fonte de largura fixa, conservando a quantidade de espaços entre as palavras e as mudanças de linha[3]. O elemento `PRE` aceita o atributo `width` com um valor numérico, que pode ser usado pelo *browser* como uma indicação da largura necessária (em quantidade de caracteres) para a apresentação do elemento, de modo que seria possível escolher o melhor tamanho de fonte ou posicionar melhor o texto. PRE

Para mudar o alinhamento (por padrão, à esquerda) de um segmento do documento, o elemento `DIV` é suportado. Este elemento aceita o atributo `align`, que pode assumir os valores *left*, *center* ou *right*. O elemento `CENTER` também é suportado, e equivale ao elemento `DIV` com atributo `align=center`. Para ambos elementos os rótulos de início e de fim de elemento são requeridos. DIV CENTER

O elemento `BLOCKQUOTE` indica que seu conteúdo deve ser apresentado como uma citação, o que em geral é implementado como um bloco de texto indentado em relação à margem esquerda. Os rótulos de início e de fim de elemento são requeridos. Este elemento não tem atributos. BLOCKQUOTE

O elemento `HR` *(Horizontal Rule)* permite estabelecer uma mudança de contexto na apresentação por meio de uma linha horizontal. Os atributos válidos para este elemento são: HR

`align` determina a posição da linha entre as margens esquerda e direita. Pode receber os valores *left*, *right* ou *center* (o padrão);

`noshade` determina que a linha deve ser apresentada com uma única cor — por padrão, duas cores são usadas para dar impressão de volume. Não recebe valores;

`size` determina a altura da linha em pixels;

`width` determina a largura da linha em pixels (como em `width=90`) ou em porcentagem da largura atual da janela (como em `width="33%"`).

2.3.5 Elementos de tabela

Tabelas são elementos de nível bloco que podem ser criadas com o elemento `TABLE`, que também pode ser utilizado para arranjos mais elaborados de texto na apresentação. O elemento `TABLE` aceita os seguintes atributos opcionais: TABLE

`align` pode assumir os valores *left* (por padrão), *center* ou *right*. Especifica o posicionamento horizontal da tabela com relação às margens da janela;

[3] O uso do carácter de tabulação é desencorajado, visto que muitos editores redifinem o número de espaços associados a um *tab* e a apresentação final pode então ser diferente do que esperado.

`width` determina a largura da tabela em pixels ou em termos de uma porcentagem da largura da janela, como descrito para o elemento `HR`. Por padrão, a largura de uma tabela é estabelecida automaticamente pela largura de seu conteúdo;

`border` pode ser usado para especificar a largura em pixels da borda a ser desenhada ao redor da tabela. O valor 0 (padrão) indica que a borda não deve estar presente;

`cellspacing` é o valor em pixels entre as bordas de células adjacentes da tabela;

`cellpadding` é o valor em pixels entre a borda e o conteúdo de cada célula.

O conteúdo do elemento `TABLE` pode conter elementos de legenda, de definição de linha da tabela e de definição de células (conteúdo, dados) da tabela.

CAPTION

O elemento de legenda `CAPTION` define um texto de descrição ou identificação para a tabela, tendo um atributo, `align`, que pode receber os valores *top* ou *bottom*, indicando respectivamente se a legenda será posicionada antes (no topo de) ou após (no fundo de) a tabela. A convenção é posicionamento no topo da tabela, por padrão. Os rótulos de início e de fim de elemento são requeridos para este elemento.

TR

O elemento `TR` *(Table Row)* define uma linha da tabela que deverá conter elementos com definição de conteúdo (as células da tabela). O rótulo de início de elemento é requerido, mas o rótulo de fim de elemento é opcional. Os dois atributos válidos para este elemento são `align`, que define o alinhamento horizontal, podendo assumir os valores *left* (padrão), *center* ou *right*; e `valign`, que define o alinhamento da linha na vertical, podendo assumir os valores *top*, *middle* ou *bottom*.

TH, TD

O conteúdo de cada linha da tabela é descrito por elementos de células `TH` *(Table Header)* e `TD` *(Table Data)*. Os rótulos de início são sempre necessários, mas os de fim de elemento podem ser omitidos. Estes elementos podem receber os seguintes atributos:

`nowrap` inibe a troca automática de linhas no conteúdo desta célula;

`rowspan` recebe um valor positivo (1, por padrão) que indica o número de linhas da tabela ocupadas por esta célula;

`colspan` recebe um valor positivo (1, por padrão) que indica o número de colunas da tabela ocupadas por esta célula;

`align` especifica o posicionamento horizontal do conteúdo na célula, podendo assumir os valores *left* (padrão para `TD`), *center* (padrão para `TH`) ou *right*;

`valign` especifica o posicionamento vertical do conteúdo na célula, podendo assumir os valores *top*, *middle* (o padrão) ou *bottom*;

`width` especifica a largura sugerida para o conteúdo da célula (sem incluir o *cellpadding*);

`height` especifica a altura sugerida para o conteúdo da célula (sem incluir o *cellpadding*).

2.3.6 Elementos de nível texto

Estes elementos não provocam quebras de parágrafos, não podendo incluir em seu conteúdo elementos de nível bloco. Eles podem ser agrupados em elementos de estilo de fonte, de frase, de campos de formulários, de âncora, de imagens inseridas, de *applets*, de mudança de fonte, de quebra de linha e de mapas de imagem. Formulários e *applets* serão descritos adiante, nas Seções 2.4.3 e 2.4.4, respectivamente.

MUDANÇA DE FONTE

Elementos de estilo de fonte determinam modificações na aparência do texto, sempre requerendo os rótulos de início e de fim de elemento. Podem ser aninhados, mas a sequência de emparelhamento deve ser obedecida. Os elementos são:

`TT` teletipo, fonte com espaçamento fixo;

`I` itálico;

`B` negrito *(boldface)*;

`U` sublinhado *(underlined)*;

`STRIKE` riscado;

`BIG` fonte maior;

`SMALL` fonte menor;

`SUB` índice *(subscript)*; e

`SUP` expoente *(superscript)*.

MUDANÇA DE ESTILO

Os elementos de frase também alteram a aparência do texto apresentado, mas usam indicações não diretamente relacionadas ao tipo de fonte a ser usado — esta é uma decisão do implementador do *browser*. Estes elementos sempre requerem os rótulos de início e de fim de elemento:

`EM` ênfase básica (tipicamente itálico);

`STRONG` ênfase forte (tipicamente negrito);

`DFN` definição;

`CODE` segmentos de programas;

`SAMP` amostras de saídas de programas;

`KBD` texto a ser teclado pelo usuário;

`VAR` nomes de variáveis ou argumentos; e

`CITE` citações ou referências a outras fontes.

O elemento FONT modifica o tamanho e/ou a cor do texto entre os rótulos de início e de fim de elemento. Este elemento aceita dois atributos. O atributo size pode receber um valor inteiro entre *1* e *7*, respectivamente do menor para o maior tamanho de fonte. O tamanho também pode ser expresso relativo ao tamanho corrente, como *+2* ou *-1* para aumentar em dois ou diminuir de um o tamanho do fonte. O atributo color define a cor do texto, sendo esta definida como descrita para o elemento BODY. FONT

O elemento BASEFONT define o tamanho de fonte a ser usado como base, definido no atributo size. É um elemento vazio, e portanto tem apenas o rótulo de início de elemento — o conteúdo e o rótulo de fim de elemento não existem. BASEFONT

O elemento BR estabelece uma mudança de linha, mas não de parágrafo. É também um elemento vazio, e portanto tem apenas o rótulo de início de elemento (o rótulo de fim de elemento é proibido). Aceita um atributo, clear, que pode assumir os valores *left*, *right* ou *all* indicando se a quebra de linha deve passar além de imagens à esquerda, à direita ou em ambas as margens, respectivamente. BR

2.4 Elementos especiais

Até o momento, os comandos descritos de HTML limitam-se essencialmente à descrição de texto e não diferem muito da funcionalidade encontrada em processadores de textos. A componente de hipertexto de HTML é suportada através do conceito de *âncoras*, o que diferencia HTML de padrões de representação tradicional de textos. Nesta seção também serão descritos os mecanismos para manipulação de imagens em HTML. Finalmente, será descrito os elementos usados para a criação de formulários e inclusão de *applets*.

2.4.1 Elemento de âncora

O elemento de âncora A *(anchor)* representa uma das ferramentas mais poderosas de HTML, pois é ela que permite o estabelecimento de ligações entre documentos. Âncoras não podem ser aninhadas, e sempre requerem os rótulos de início e de fim de elemento. A

A ligação entre dois documentos requer o estabelecimento de um ponto de partida e um ponto de chegada. O elemento âncora permite a definição de ambos. O ponto de partida é uma *referência hipertexto*, sendo definida por um elemento A com o atributo href indicando (através de URLs) o nome do recurso apontado. O recurso destino pode ser outro documento HTML ou um arquivo em outro formato (PostScript, PDF, uma imagem, etc). Por exemplo, HREF

```
<A href="http://www.dca.fee.unicamp.br/Welcome.html">DCA</A>
```

estabelece no texto uma ligação para o início do documento Welcome.html no servidor WWW www.dca.fee.unicamp.br, sendo que o texto DCA aparecerá em destaque para indicar que este é uma ligação de hipertexto.

O ponto de chegada é, por padrão, o início do documento. Entretanto, pontos de chegada em locais arbitrários de um documento podem ser definidos por um elemento A com o atributo name definido. O valor de name deve ser uma sequência de caracteres que define de forma unívoca no escopo do documento este ponto de chegada. Por exemplo, supondo que no meio de um documento HTML de nome amostra.html seja definida a seguinte entidade, NAME


```
<H2><A name=meio>Muita coisa já passou...</A></H2>
```

este ponto do documento poderia ser acessado diretamente a partir de outro documento pela referência

```
<A href="amostra.html#meio">Para o meio!</A>
```

ou a partir de outro ponto do mesmo documento como

```
<A href="#meio">Para o meio!</A>
```

2.4.2 Imagens

Imagens externas ao documento podem ser normalmente referenciadas através de âncoras, onde o destino é um arquivo com a imagem. Nestes casos, a imagem ligada será apresentada através de algum programa externo ao *browser*.

Imagens podem também ser inseridas como parte do conteúdo do documento através do elemento `IMG`. Este é um elemento vazio, portanto sem conteúdo e sem rótulo de fim de elemento. IMG
Elementos `IMG` suportam os seguintes atributos:

`src` é atributo obrigatório que especifica o URL do arquivo (o recurso) contendo a imagem, que deve ser em algum formato suportado pelo *browser* — em geral, GIF, JPEG ou PNG;

`alt` é uma descrição textual alternativa à imagem, para ser usada quando a imagem não pode ser apresentada — em *browsers* usando apenas texto ou baseados em apresentação auditiva, por exemplo;

`align` indica a posição da imagem com relação à linha corrente de texto, podendo assumir os valores *top* (topo da imagem alinhado com o topo da linha de texto), *middle*, *bottom* (base da imagem alinhada com a base da linha de texto, o padrão), *left* (imagem à esquerda, com texto acompanhando a imagem à direita) ou *right*;

`width` especifica a largura da imagem em pixels e, juntamente com a altura da imagem (atributo `height`) permite que o *browser* reserve o espaço correto para a imagem antes que sua transferência esteja completa;

`height` especifica a altura da imagem em pixels;

`border` indica a largura da borda ao redor da imagem em pixels. Esta característica é particularmente útil quando a imagem é parte de um link de hipertexto, de forma que esta borda possa aparecer em cor diferente para indicar esta condição. Um valor de 0 para este atributo elimina a borda;

`hspace` determina um espaço em branco (afastamento) ao redor da imagem, à esquerda e à direita, em pixels;

`vspace` é similar a `hspace` para o espaçamento na vertical;

`ismap` é o mecanismo usado para indicar que a imagem é um mapa de links. Quando um clique do mouse é detectado sobre esta imagem, as coordenadas (x, y) onde este evento ocorreu são enviadas para o servidor de onde o documento é originário para interpretação e devido processamento;

`usemap` é usado em conjunção com o elemento `MAP` (veja abaixo) para permitir associar links distintos a partes da imagem. Este mecanismo suporta a implementação de mapas de links no lado do cliente (sem intervenção do servidor original do arquivo).

O elemento `MAP` suporta mapas gráficos de links sem a intervenção do servidor. Neste caso, o mapa é incluído no mesmo arquivo que a imagem — há previsão para especificar mapas em arquivos a parte, mas esta característica não é amplamente suportada. Os rótulos de início e de fim de elemento são necessários para este elemento. O elemento `MAP` tem um atributo, `name`, que é usado para estabelecer a ligação com o elemento `IMG` que faz referência a este mapa. MAP

O elemento `MAP` contém um ou mais elementos `AREA`, que indicam que regiões da figura são "clicáveis" e que links estas regiões estabelecem. O elemento `AREA` é vazio, de forma que não tem conteúdo ou rótulo de fim de elemento. Toda a informação necessária para suportar sua funcionalidade é descrita através de seus atributos, que são: AREA

`shape` indica o formato da região. Possíveis valores são *rect* (região retangular, o padrão), *circle* (região circular) ou *poly* (polígono arbitrário);

`coords` determinam as coordenadas da região. Os valores assumidos dependem do formato da região. Para regiões retangulares, `coords` deve receber uma sequência de quatro valores que indicam, respectivamente, a coordenada x à esquerda, a coordenada y ao topo, a coordenada x à direita e a coordenada y à base. Para regiões circulares, são esperados três valores que indicam respectivamente as coordenadas x e y do centro e o raio do círculo. Para polígonos arbitrários, os valores são pares x e y determinando as coordenadas de cada vértice do polígono. Estes valores podem ser expressos de forma absoluta (medidos em pixels) ou em termos relativos às dimensões da imagem (expressos em porcentagem);

`href` especifica o alvo do hiperlink estabelecido pela região;

`nohref` permite especificar "buracos" que não estabelecem hiperlinks em uma imagem;

`alt` estabelece textos que devem ser utilizados como alternativas quando o *browser* não estiver mostrando gráficos. O texto também pode ser utilizado pelos *browsers* para fornecer informação na linha de *status* quando o mouse passar sobre a região do hiperlink.

Quando há sobreposição de regiões no mapa de links, aquelas que são definidas primeiro têm precedência sobre as demais — assim, regiões `nohref` devem vir antes da definição de links.

2.4.3 Formulários

Formulários em HTML são elementos que recebem entradas através do *browser* e enviam estas entradas para o processamento no servidor de documentos. O tipo mais simples de interação, com apenas um campo de uma linha de texto no documento, é suportada através do elemento `ISINDEX`.

O elemento `FORM` permite definir formulários mais flexíveis, contendo também campos de FORM

múltiplas linhas, botões e menus. Os rótulos de início e de fim de elemento são requeridos, e o elemento recebe os seguintes atributos:

`action` indica, por um URL, o que será feito com o conteúdo do formulário. Possíveis ações a tomar podem incluir o envio por e-mail, como em

```
action="mailto:alguem@algum.lugar"
```

ou indicar um programa ou script no servidor que está preparado para receber e processar este conteúdo, como em

```
action="http://www.laranja.br/cgi-bin/precatorio.pl"
```

onde `precatorio.pl` é o nome de um script Perl executável no servidor indicado, `www.laranja.br`;

`method` pode ser especificado quando `action` especifica um URL, indicando que método HTTP será utilizado para enviar o conteúdo do formulário para o servidor. Valores possíveis são *get* (o padrão), usado quando a operação realizada é apenas de consulta a dados existentes, e *post*, usado quando o conteúdo do formulário irá atualizar alguma informação (conteúdo de páginas ou bases de dados) no servidor;

`enctype` determina o mecanismo utilizado para codificar o conteúdo do formulário para o envio. Por padrão, um formato de codificação *www-form-urlencoded* é utilizado.

O conteúdo de um elemento formulário é constituído por elementos de campos de formulários, que podem ser de entrada de dados, de seleção de opções ou de áreas de texto.

INPUT

O elemento `INPUT` pode ser utilizado para criar campos de entrada de textos de uma linha, campos de senhas, botões de opções e de seleção, botões para conclusão (submissão) e para reinício *(reset)* do preenchimento do formulário, campos escondidos e botões de imagens. Este elemento é vazio, de modo que apenas o rótulo de início de elemento deve estar presente. O seu comportamento é definido por seus atributos — `type, name, value, checked, size, maxlength, src, align` — que assumem valores de acordo com o tipo de campo sendo definido.

LINHA DE TEXTO

Para definir um campo que receberá uma única linha de texto, `type` deve assumir o valor *text* (o padrão). O tamanho visível do campo é determinado pelo valor do atributo `size`, sendo possível entrar textos mais longos que este valor. Caso se deseje limitar o tamanho do texto sendo digitado, este limite deve ser especificado com o atributo `maxlength`. O atributo `name` permite especificar uma string de nome para o campo, enquanto que o atributo `value` permite especificar uma string com um valor inicial já presente no campo.

Quando `type` receber o valor *password*, o campo é similar a um campo de texto mas o que for digitado não irá aparecer na tela do usuário (por exemplo, aparecem apenas `*` no campo).

OPÇÕES

O atributo `type` com valor *checkbox* permite trabalhar com atributos booleanos, ou seja, que recebem valores do tipo sim ou não. O atributo `checked` não recebe valores, especificando apenas que inicialmente a variável vai estar no estado *checked*.

ALTERNATIVAS

Variáveis que podem assumir um valor entre um grupo de alternativas são manipuladas por campos com atributo `type` com valor *radio*. Cada valor possível corresponde a um elemento `INPUT`, sendo que os elementos são agrupados tendo o mesmo valor para o atributo `name`.

Há basicamente dois tipos de botões que podem ser criados para um formulário, para confirmação (submit) ou para apagar tudo que foi modificado pelo usuário (reset). Um elemento `INPUT` com `type` recebendo o valor *submit* define um botão que, quando selecionado pelo usuário, enviará o conteúdo do formulário para seu destino. O rótulo visível no botão é determinado pelo atributo `value`. Se o atributo `name` estiver também definido, a informação nome/valor será incluída na codificação, o que permite ter várias alternativas de submissão de um formulário com tratamento diferenciado. Botões de submissão podem conter imagens ao invés de texto quando `type` tiver o valor *image*. Neste caso, o atributo `src` deve ser utiliado para indicar o URL do arquivo com a imagem a ser usada, e o atributo `align` indica o alinhamento da imagem com relação ao texto, como para elementos `IMG`. A informação sobre a posição onde a imagem foi selecionada também é passada no conteúdo do formulário, usando o nome do campo com sufixos `.x` e `.y` agregados ao nome. Quando `type` tem o valor *reset*, o elemento definido é um botão que, quando ativado, restaura o formulário ao estado inicial. O atributo `value` permite definir o rótulo para o botão. Botões de restauração nunca são enviados como parte do formulário.

BOTÃO CONFIRMAÇÃO

BOTÃO RESTAURAÇÃO

É possível anexar o conteúdo de um arquivo ao conteúdo enviado de um formulário usando um tipo de entrada com valor *file*. Em geral, a apresentação deste tipo de entrada inclui um campo de linha de texto, onde o nome do arquivo pode ser especificado diretamente, e um botão associado à funcionalidade de seleção interativa entre os arquivos disponíveis *(file browsing)*. Como para entradas do tipo linhas de texto, os atributos `size` e `maxlength` podem ser definidos para este elemento de entrada. O atributo `accept` permite restringir os tipos de arquivos que podem ser anexados ao formulário, recebendo valores na forma de tipos de conteúdo MIME.

INCLUSÃO DE ARQUIVO

O elemento `INPUT` pode conter elementos que não são apresentados ao usuário, mas com informação que será recebida pelo servidor juntamente com o conteúdo do formulário. Uma possível aplicação deste tipo de informação é validar a resposta do formulário ou associar estados ao formulário. Para tanto, o atributo `type` recebe o valor *hidden*, com atributos adicionais `name` e `value`. O valor do atributo `value` é pré-definido e, como não haverá entrada do usuário (que não vê este campo), não será alterado, sendo enviado de volta ao servidor com o valor original.

Menus de opções, onde as opções são apresentados em *drop-down*[4] são suportados pelo elemento `SELECT`. Os rótulos de início e de fim de elemento são requiridos para este elemento. Os atributos de `SELECT` são:

SELECT

`name` define o nome da propriedade que irá identificar a opção no envio do formulário;

`size` define o número de opções vísiveis em menus múltiplos;

`multiple` identifica que usuários podem escolher múltiplas opções. Por padrão, apenas uma opção pode ser selecionada.

Cada opção do menu é definida por um elemento `OPTION`, cujo conteúdo é o texto a ser apresentado na opção. Os atributos aceitos por este elemento são:

OPTION

`selected` indica, quando presente, que esta opção deve estar selecionada na apresentação inicial do formulário. Apenas uma opção pode conter este atributo;

[4]Onde só aparece a primeira opção, a apresentação das outras opções só ocorre quando se clica com o mouse na caixa do menu.

`value` indica qual o valor que deve ser enviado no conteúdo do formulário quando esta opção é selecionada do menu. O nome da propriedade para este valor é obtido do atributo `name` do elemento `SELECT` correspondente a esta opção.

Finalmente, é possível definir áreas de entrada para diversas linhas de texto com o elemento `TEXTAREA`, para o qual os rótulos de início e de fim de elemento são requeridos. O conteúdo deste elemento pode ser apenas texto, que quando presente será utilizado como valor inicial para o conteúdo do formulário. Os atributos válidos para este elemento são: TEXTAREA

`name` é o nome da propriedade que identifica a área de texto no formulário;

`rows` especifica o número visível de linhas na área de texto;

`cols` especifica a largura visível em termos de (largura média de) caracteres.

Os *browsers* devem ser capazes de suportar mecanismos de *scrolling* para permitir entradas de linhas mais longas e de mais linhas que aquelas especificadas em `cols` e `rows`, respectivamente.

2.4.4 Applets

Um *applet* é um programa usualmente escrito em Java que pode ser carregado de um servidor e executado localmente, contanto que o *browser* suporte esta execução (seja um *Java-enabled browser*). O elemento `APPLET` é o elemento de HTML que permite embutir um *applet* em uma página WWW. APPLET

O elemento `APPLET` requer os rótulos de início e de fim de elemento, sendo que o conteúdo deste elemento pode ter elementos `PARAM` (veja abaixo) e uma alternativa que pode ser exibida quando o *browser* não souber o que é este elemento. Os atributos aceitos por este elemento são:

`codebase` especifica (através de um URL) o diretório de onde o *applet* será carregado. Caso não seja especificado, o URL do documento corrente é usado como base;

`code` é o atributo obrigatório que indica o nome do arquivo com o código (compilado) do *applet*. Este nome deve ser relativo ao URL básico do *applet*, não podendo ser absoluto;

`alt` contém texto que deve ser exibido pelo *browser* caso ele consiga entender o elemento `APPLET` mas não pode executar o código Java;

`name` permite associar uma identificação a um *applet*, permitindo encontrar e trocar informação entre *applets* na mesma página;

`width` determina a largura inicial da janela na página para execução do *applet*, em pixels;

`height` determina a altura inicial da janela na página para execução do *applet*, em pixels;

`align` determina o alinhamento da *applet* em relação ao texto, exatamente como para o elemento `IMG`;

`vspace` como no elemento `IMG`, especifica um espaço em pixels ao redor do topo e da base da janela do `applet`;

`hspace` especifica um espaço em pixels ao redor das margens esquerda e direita da janela do `applet`.

O elemento PARAM é usado para passar parâmetros para a execução do *applet*, podendo receber dois atributos: PARAM

`name` o nome do parâmetro do *applet*;

`value` o valor para o parâmetro.

Os elementos de parâmetros *devem* ser colocados no começo do conteúdo do elemento APPLET, antes de qualquer outro elemento de texto.

Referências Bibliográficas

[1] ArborText, Inc. Getting started with SGML. http://www.sgmlopen.org/sgml/docs/getstart.htm, 1995.

[2] Tim Berners-Lee. Information management: A proposal. http://www.w3.org/pub/WWW/History/1989/proposal.html, March 1989.

[3] Tim Berners-Lee. Hypertext markup language (HTML): Working and background materials. http://www.w3.org/pub/WWW/MarkUp/, 1990.

[4] Tim Berners-Lee. About the World-Wide Web consortium. http://www.w3.org/pub/WWW/Consortium/, January 1997.

[5] Tim Berners-Lee, Robert Cailliau, et al. The World-Wide Web. *Communications of the ACM*, 37(8):76–82, August 1994.

[6] Jean Bolot and Philipp Hochska. Sound and video on the Web. In *5th WWW Conference*, Paris, France, May 1996. http://www.inria.fr/rodeo/personnel/hoschka/WWW5tutorial.ps.gz.

[7] Thomas Boutell. PNG (Portable Network Graphics) specification. http://www.w3.org/pub/WWW/TR/REC-png-multi.html, October 1996. Version 1.0.

[8] Robert Cailliau. A little history of the World Wide Web. http://www.w3.org/pub/WWW/History.html, October 1995.

[9] Jeff Conklin. Hypertext: An introduction and survey. *IEEE Computer*, pages 17–41, September 1987.

[10] Luigi Filippini. Moving Picture Expert Group information page. http://www.vol.it/MPEG/, May 1996.

[11] Eric Hamilton. JPEG File Interchange Format. http://www.w3.org/pub/WWW/Graphics/JPEG/jfif.txt, September 1992.

[12] Microsoft. Multimedia programming interface and data specification. Online documentation (RTF). file://ftp.cwi.nl/pub/audio/RIFF-format.

[13] Dave Raggett. *HTML 3.2 Reference Specification*, January 1997. http://www.w3.org/pub/WWW/TR/REC-html32.html.

[14] Greg Roelofs. Portable Network Graphics page. `http://www.wco.com/~png/`, April 1997.

[15] SGML Open. SGML Open home page. `http://www.sgmlopen.org/`, 1997.

[16] John B. Smith and Stephen F. Weiss. Hypertext. *Communications of the ACM*, 31(7):816–819, July 1988.

[17] Guido van Rossum. The CWI audio file formats guide. `http://cuiwww.unige.ch/OSG/AudioFormats/`, November 1991.

[18] World Wide Web Consortium. Web style sheets. `http://www.w3.org/pub/WWW/Style/`, March 1997.

