DEPARTAMENTO DE

ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Sistemas de Autoria para CD-i

Desenvolvimento de um Protótipo em MediaMogul

Rui Manuel Dias Ferreira de Lima

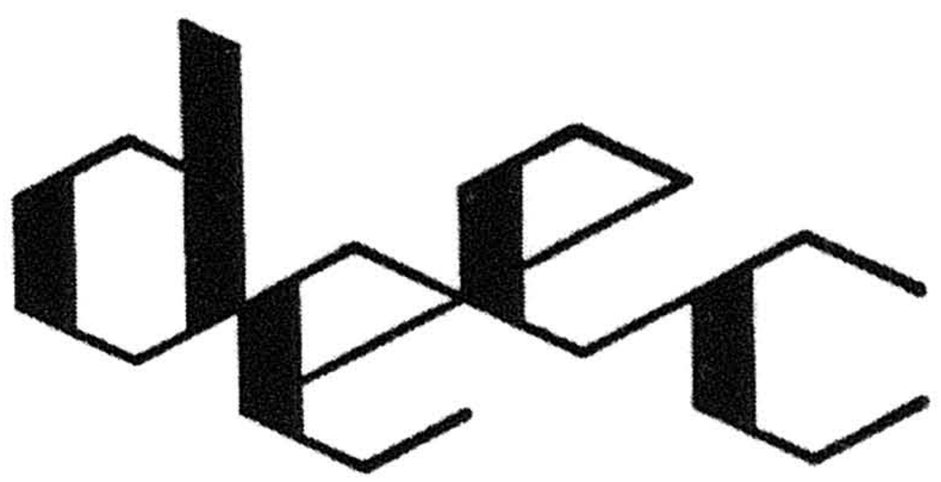
FACULDADE DE ENGENHARIA

UNIVERSIDADE DO PORTO

Rua dos Bragas, 4099 Porto Codex – PORTUGAL

FEUP

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
Mestrado em Engenharia Electrotécnica e de Computadores



Departamento de
Engenharia de Electrotécnica e de Computadores

cat

DEEC

FEUP

Sistemas de Autoria para CD-i

Desenvolvimento de um Protótipo em MediaMogul

Rui Manuel Dias Ferreira de Lima

Licenciado em Engenharia Electrotécnica e de Computadores
pela Faculdade de Engenharia da Universidade do Porto

Dissertação submetida para satisfação parcial
dos requisitos do programa do curso de
Mestrado em Engenharia Electrotécnica e de Computadores
(área de especialização em Telecomunicações)

81.3(043) LIMR/SIS
UNIVERSIDADE DO PORTO
Faculdade de Engenharia
BIBLIOTECA
N.º 21701
CDU
Data 16 5 96
N.N.: 2142

PORTO, JULHO 1995

Dissertação realizada sob supervisão do

Professor Eurico Manuel Elias de Morais Carrapatoso

Professor Auxiliar do Departamento de Engenharia Electrotécnica e
de Computadores da Faculdade de Engenharia da Universidade do Porto

Dissertação realizada por

Rui Manuel Dias Ferreira de Lima

CMUP Telef: +351-2-325713 Fax:+351-2-2008258

Projecto LEONARDO

Centro Multimédia da Universidade do Porto

Resumo

Este início de década está a ser marcado pelas tecnologias multimédia. As redes de comunicação de dados (*Internet*) contribuem bastante para a disseminação da informação. As aplicações surgem dotadas de interfaces homem-máquina com tratamento gráfico e sonoro apurado. As plataformas multimédia baseadas em CDs dispõem de uma boa capacidade de armazenamento de dados e constituem um meio seguro de guardar informação. O CD-i é a primeira plataforma multimédia destinada ao grande público, que dispensa qualquer familiaridade com os computadores.

O presente trabalho sintetiza as características da plataforma CD-i, fazendo algumas observações sobre a programação de aplicações interactivas e analisando alguns dos sistemas de autoria na perspectiva da construção de protótipos.

Descreve também a implementação de um protótipo, utilizando o sistema de autoria MediaMogul. É ainda apresentado o modo como recorrendo a técnicas de programação se desenvolveram subrotinas usando a linguagem C, para permitir a extensão das capacidades do MediaMogul.

Palavras Chave

Multimédia

Interactivo

Sistemas de Autoria

Disco Compacto

Entrelaçamento

Subrotinas

Abstract

Multimedia technologies mark the beginning of the current decade. Communication networks (Internet) have contributed for the wide dissemination of information. The emergent applications have man-machine interfaces with a fine graphic and audio treatment. Multimedia platforms based on CDs have a large data storing capacity and are a safe way to store information. CD-i is the first multimedia technology aimed at a mass audience, without any computing knowledge.

This dissertation surveys the main features and capabilities offered by CD-i platforms, makes some remarks about programming interactive applications and analyses some authoring systems having in mind the implementation of prototypes.

It also describes the implementation of a prototype, developed using the MediaMogul authoring system. The extension of MediaMogul features with some subroutines implemented in C language is also presented.

Keywords

Multimedia

Interactive

Authoring Systems

Compact Disc

Interleaving

Plug-Ins

Agradecimentos

Agradeço particularmente ao meu orientador, Professor Eurico Manuel de Elias Morais Carrapatoso, a disponibilidade, o cuidado e o espírito crítico que muito beneficiaram esta dissertação.

Reitero o meu reconhecimento aos Engenheiros Nuno Ribeiro e Rui Moreira, pelas opiniões prestadas e franca colaboração, que sempre demonstraram no decorrer do Mestrado, com os quais foram estudadas as matérias abordadas nesta dissertação. Constituiram um grupo de trabalho de grande valor humano, onde se estabeleceram laços de estreita amizade.

Ao Professor Raul Moreira Vidal manifesto a minha gratidão pela motivação com que me sensibilizou para a frequência deste curso de Mestrado.

O trabalho realizado só foi possível graças à colaboração do CMUP - Centro Multimédia da Universidade do Porto, que, sob a direcção do Professor Duarte Costa Pereira e seus assessores Dra. Assunção Costa Lima e Eng. José C. Guedes Vaz, contribuiram com comentários e sugestões que traçaram o desenvolvimento deste trabalho. Todo o equipamento do CMUP foi colocado à disposição.

Parte dos elementos reais utilizados neste trabalho foram gentilmente cedidos pelos autores do protótipo Leonardo, aos quais agradeço a boa vontade demonstrada e que passo a nomear: Albertina Afonso, José C. Guedes Vaz, Maria José Ferreira, Maria Teresa Abrantes e Arq. António A. Madureira.

Para a Sílvia uma palavra de apreço pela força e postura que adoptou.

Enleia-se por gosto a liberdade;
E depois a paixão na alma se apura,
Alguns então lhe chamam desventura,
Chamam-lhe alguns então felicidade.

Aos familiares, por todo o apoio prestado, e pela compreensão e paciência demonstrada especialmente na fase conclusiva do trabalho.

Finalmente apresento os mais profundos e sinceros agradecimentos a todas as pessoas que tornaram possível a realização desta dissertação e que contribuíram de alguma forma para a sua realização.

Aos meus pais:

Manuel Lima e Augusta Dias

Índice

Resumo ..... **i**

Abstract ..... **ii**

Agradecimentos ..... **iii**

Índice ..... **v**

Abreviaturas ..... **ix**

Lista de Figuras ..... **xi**

Lista de Tabelas ..... **xii**

I Introdução **1**

1.1 Objectivos 2

1.2 Estrutura da Dissertação 2

II Sistemas Multimédia **4**

2.1 Terminologia 5

2.2 História dos Sistemas Multimédia 6

2.3 Plataformas Multimédia Baseadas em CDs 7

2.4 A Plataforma CD-i 11

2.4.1 Imagens e Som em CD-i 13

2.4.2 Software e Hardware do CD-i 14

2.5 O Disco CD-i 15
2.5.1 Modos e Formatos do CD 17
2.5.2 Detecção e Correcção de Erros 19
2.5.3 Canais de Transferência de Dados 19

2.6 O Futuro 20

III O Sistema CD-i 22

3.1 Arquitectura do Leitor de CD-i 23

3.2 Sistema Operativo CD-RTOS 25
3.2.1 *Kernel* 26
3.2.2 Gestores de Ficheiros 27
3.2.3 Controladores (*Drivers*) 30

3.3 Entrelaçamento de Áudio e Vídeo 31

3.4 Construção do Disco CD-i 34

3.5 Codificação de Imagens no Sistema CD-i 37
3.5.1 Características Técnicas das Imagens em CD-i 37
3.5.1.1 Texto 40
3.5.1.2 Métodos de Codificação de Imagens Paradas 40
3.5.2 Efeitos com Imagens em CD-i 42
3.5.2.1 Tipos de Planos 42
3.5.2.2 Criação de Efeitos Especiais em CD-i 43
3.5.3 Sistema FMV (*Full-Motion Video*) em CD-i 45
3.5.3.1 Aplicação da Norma MPEG ao FMV 45
3.5.3.2 Codificação FMV (*Full-Motion Video*) 47
3.5.3.3 Áudio e Vídeo no Formato MPEG 48
3.5.3.4 Sincronização com a Leitura (*Playback*) 49

3.6 Áudio no Sistema CD-i 50

IV Programação no Sistema CD-i 53

4.1 Programação de Aplicações Interactivas 54

4.2 Programação em CD-RTOS 55
4.2.1 *Compact Disc File Manager* (CDFM) 56
4.2.2 *User Communications Manager* (UCM) 59

4.3 Ambientes para Produção de Aplicações CD-i 60
4.3.1 Linguagens de Programação 62
4.3.2 Programação com Sistemas de Autoria 63
4.3.3 Bibliotecas de Funções 65
4.3.4 Motores de Software 65

4.4 O Estúdio CD-i 66

V Sistemas de Autoria 68

5.1 O que é um Sistema de Autoria para CD-i 69

5.2 ASLAN 70
5.2.1 Sequências 71
5.2.2 Menus 72

5.3 MediaMogul 73
5.3.1 Configuração do Sistema MediaMogul 74
5.3.2 Módulos do MediaMogul 75
5.3.3 Características e Limitações do MediaMogul 80
5.3.4 Ferramentas Auxiliares para Animações e Janelas de Vídeo 81
5.3.5 Vídeo Digital (DV) em MediaMogul 83
5.3.6 Script-to-Disk (S2D) 84
5.3.6.1 Espaço de Armazenamento em Disco 85
5.3.6.2 Plataformas de Desenvolvimento 86
5.3.6.3 Construção de um Disco CD-i 87
5.3.7 *Mac Imagination Pack* 88

5.4 HAL-90 88

5.5 Griffin Tools 90
5.5.1 Programação Concorrente 91
5.5.2 Exemplo: Visualização de Imagens 91

5.6 Balboa 92

5.7 CORE 94

5.8 Escolha do Sistema de Autoria 95

VI Subrotinas para Estender as Capacidades do MediaMogul 96

6.1 Como Chamar uma Subrotina 97

6.2 MediaMogul *Plug-Ins* 98

6.3 Gestão de Memória nas Subrotinas 99

6.4 Limitações das *Plug-Ins* do MediaMogul 101

6.5 Utilização de Variáveis em Subrotinas 101

6.6 Desenvolvimento de Subrotinas em C 102
6.6.1 *Input Environment* 102
6.6.2 *Video Environment* 104
6.6.3 Compilação das Subrotinas 105

VII Desenvolvimento de um Protótipo em MediaMogul 107

7.1 Estrutura do Protótipo 108

7.2 Desenvolvimento do Protótipo 109
7.2.1 Implementação dos Segmentos 111
7.2.2 Implementação do Jogo 114

7.3 Teste e Gravação do CD-i 115

VIII Conclusões 116

8.1 Contributo Principal 118

8.2 Resultados Práticos 118

8.3 Limitações 119

8.4 Sugestões para Desenvolvimentos Futuros 120

Bibliografia 121

Apêndice A - Formatos de Codificação de Imagens A-1

Apêndice B - Exemplo de uma subrotina: videosc B-1

Abreviaturas

ADPCM	*Adaptive Differential Pulse Code Modulation*
APU	*Audio Processing Unit*
ASCII	*American Standard Code for Information Interchange*
CBT	*Computer-Based Training*
CD	*Compact Disc*
CD+G	*Compact Disc Plus Graphics*
CD-DA	*Compact Disc Digital Audio*
CD-i	*Compact Disc Interactive*
CD-ROM	*Compact Disc Read Only Memory*
CD-ROM XA	*Compact Disc Read Only Memory Extended Architecture*
CD-RTOS	*Compact Disc Real-Time Operating System*
CDFM	*Compact Disc File Manager*
CDTV	*Commodore Dynamic Total Vision*
CIRC	*Cross-Interleaved Reed-Solomon Code*
CLUT	*Colour Look-Up Table*
CMUP	Centro Multimédia da Universidade do Porto
CPU	*Central Processing Unit*
CSD	*Configuration Status Descriptor*
CSPS	*Constrained System Parameter Stream*
DAT	*Digital Audio Tape*
DCP	*Display Control Program*
DVI	*Digital Video Interactive*
FCT	*Field Control Table*
FMV	*Full-Motion Video*
fps	*frames per second*
HD-CD	*High Density Compact Disc*
Hi-Fi	Alta-Fidelidade
ISO	*International Standards Organisation*
IV	*Interactive Video*

JPEG	*Joint Photographic Experts Group*
Kbps	*Kbit per second*
LCT	*Line Control Table*
MAC	*Macintosh Computer*
MPEG	*Motion Picture Experts Group*
NFM	*Network File Manager*
NRFM	*Non-Volatile RAM File Manager*
NTSC	*(American) National Television Standards Committee*
OS-9	*Operating System written for Motorola's 6809*
PAL	*Phase Alternate Line*
PC	*Personal Computer*
PCB	*Play Control Block*
PCL	*Play Control Lists*
PCM	*Pulse Code Modulation*
PSC	*Play System Call*
RAM	*Random Access Memory*
RBFM	*Random Block File Manager*
RGB	*Red Green Blue*
RLE	*Run-Length Encoding*
ROM	*Read-Only Memory*
RTCA	*Real-Time Control Area*
RTF	*Real-Time File* (ficheiro de tempo-real)
RTRI	*Real-Time Record Interpreter*
S2D	*Script-to-Disc*
SCFM	*Sequential Character File Manager*
SCSI	*Standard Computer System Interface*
SECAM	*Séquence Couleur à Mémoire*
UCM	*User Communication Manager*
VGA	*Visual Graphics Adapter*
VHS	*Video Home System*
WORM	*Write Once, Read Many*

Lista de Figuras

Figura 2.1 - Multimédia 6
Figura 2.2 - Plataformas multimédia baseadas em CDs 8
Figura 2.3 - Leitor de CD-i 12
Figura 2.4 - Organização das pistas de um CD-i. 16
Figura 3.1 - Arquitectura do leitor CD-i 25
Figura 3.2 - *Start-up Screen* 27
Figura 3.3 - Camadas do Sistema Operativo. 30
Figura 3.4 - Entrelaçamento de vídeo com áudio. 31
Figura 3.5 - Ficheiros de tempo-real 32
Figura 3.6 - Zona de Visualização 39
Figura 3.7 - Armazenamento de uma stream MPEG num dado meio 46
Figura 3.8 - Sistema genérico de codificação MPEG 46
Figura 3.9 - Digitalização de áudio (Digidesign) 51
Figura 3.10 - Conversor de áudio PCM em ADPCM 52
Figura 4.1 - Sistema de Autoria para CD-i 64
Figura 5.1 - Uma sequência ASLAN com 2 pontos de entrada 71
Figura 5.2 - Módulos do MediaMogul 74
Figura 5.3 - Gestor de ficheiros do MediaMogul 76
Figura 5.4 - Editor de scripts do MediaMogul 76
Figura 5.5 - Editor de botões do MediaMogul 78
Figura 5.6 - Janela de informações dos botões 78
Figura 5.7 - Ferramenta de desenho do MediaMogul 79
Figura 5.8 - Shell do CD-RTOS 81
Figura 5.9 - Criar uma aplicação DV em MediaMogul 84
Figura 5.10 - Ligações do Emulador e do WORM 87
Figura 5.11 - Realização de um título em HAL-90 89
Figura 5.12 - Arquitectura do Balboa 93
Figura 6.1 - Chamada de uma subrotina 97
Figura 7.1 - Estrutura do protótipo Leonardo 108
Figura 7.2 - Desenvolvimento de um protótipo em MediaMogul 110
Figura 7.3 - Visualização de uma imagem 384x560 113
Figura 7.4 - Tabuleiro do Jogo da Glória 114

Lista de Tabelas

Tabela 3.1 - Extensões dos ficheiros gerados pelo Master 36
Tabela 3.2 - Qualidade do áudio em CD-i 50
Tabela 5.1 - Janelas de vídeo sem áudio 82
Tabela 5.2 - Janelas de vídeo com áudio C-mono 82
Tabela 5.3 - Janelas de vídeo com áudio C-stereo ou B-mono 82
Tabela 5.4 - Janelas de vídeo com áudio B-stereo ou A-mono 83
Tabela 5.5 - Janelas de vídeo com áudio A-stereo 83

I

INTRODUÇÃO

Os sistemas analógicos de armazenamento da informação são habitualmente implementados sobre plataformas cuja natureza tecnológica garante uma sincronização perfeita entre as imagens e o som. No mundo digital da programação multimédia é diferente.

Uma aplicação multimédia, quer em CD-i ou noutro meio digital, é uma mistura de diferentes elementos, que, por controlo de um programa de computador, são sincronizados com vista a ter uma apresentação idêntica à da televisão. Dado que os ficheiros digitais são muito grandes, a largura de banda do processador é limitada e a capacidade de armazenamento é finita, a produção de aplicações multimédia é um exercício de gestão cujo objectivo é garantir que os ficheiros digitais estão devidamente organizados para que o seu conjunto se transforme numa aplicação multimédia.

Com vista a diminuir o esforço informático necessário para a realização de aplicações multimédia, foram desenvolvidos sistemas de autoria que, de uma forma rápida e eficaz, permitem ao autor (programador) ter acesso a um conjunto de procedimentos optimizados para a realização de aplicações multimédia específicas.

Apesar desta complexidade ser reduzida pelos sistemas de autoria, pode-se afirmar que o desenvolvimento de um projecto CD-i requer uma equipa de produção multidisciplinar, nomeadamente a experiência de especialistas em software.

1.1 OBJECTIVOS

Reconhecendo a importância que a tecnologia CD-i tem e que virá a ser ainda maior a curto prazo no domínio da multimédia definiram-se para este trabalho um conjunto de objectivos:

- Estudar em pormenor o sistema CD-i: hardware, sistema operativo, animações, vídeo digital, formatos de codificação de imagem e som.
- Estudar as plataformas baseadas em CDs, enquadrando o CD-i no universo dos sistemas multimédia e contribuir para uma clarificação dos conceitos fundamentais que estão na base destas tecnologias.
- Averiguar as potencialidades da programação de aplicações interactivas sobre a plataforma CD-i.
- Familiarização com os sistemas de autoria existentes, apresentar as suas características principais e as limitações que impõem às aplicações.
- Descrever o processo de criação de um protótipo de uma aplicação CD-i, usando o sistema de autoria MediaMogul.

1.2 ESTRUTURA DA DISSERTAÇÃO

O presente trabalho enquadra-se globalmente na área dos sistemas multimédia, incidindo particularmente sobre a plataforma CD-i.

A estrutura desta dissertação pretende enquadrar a plataforma CD-i no universo dos sistemas multimédia, caracterizando aspectos técnicos do sistema CD-i (em especial o hardware, software e a programação) que é necessário dominar nas diferentes fases do desenvolvimento de aplicações multimédia interactivas sobre esta plataforma.

Esta dissertação está organizada em oito capítulos; o capítulo I é uma introdução onde se apresenta o enquadramento geral do tema de dissertação e os objectivos específicos; os capítulos II, III, IV e V tem um carácter mais descritivo quando comparados com os dois capítulos seguintes que são de natureza prática; o capítulo VIII apresenta as conclusões tecendo algumas considerações sobre a plataforma CD-i e as ferramentas usadas na programação de aplicações.

O capítulo II contém uma breve introdução aos sistemas multimédia digitais, realçando aspectos do desenvolvimento de aplicações e estabelece o contexto em que se enquadram as plataformas baseadas em discos compactos digitais. Faz uma apresentação sumária do sistema CD-i e do modo de armazenamento dos dados no CD. Por último apresenta as linhas do desenvolvimento futuro dos sistemas multimédia.

O capítulo III é dedicado a uma descrição detalhada do sistema CD-i, realçando os aspectos relacionados com a arquitectura do sistema, o sistema operativo, a codificação de imagens, vídeo e áudio. São ainda discriminadas as fases de construção de um disco CD-i, dando particular destaque ao entrelaçamento dos dados.

O capítulo IV aborda a programação no sistema CD-i, apresentando as características principais do sistema operativo, descrevendo o modo como são interpretados os ficheiros de tempo-real e como é realizado o controlo dos sinais I/O. Caracteriza os ambientes para produção de aplicações CD-i, apresentando a evolução das linguagens de programação e das ferramentas que auxiliam a produção de aplicações CD-i. Por fim fornece uma configuração possível para um estúdio de CD-i.

O capítulo V concentra-se nos sistemas de autoria para CD-i; ASLAN, HAL-90, Griffin Tools, Balboa, CORE e MediaMogul ao qual é dado especial relevo. São apresentadas as características principais destes sistemas e o tipo de utilizadores a que se destinam.

O capítulo VI tem um natureza prática e mostra como é possível estender as capacidades do MediaMogul, com a inclusão de funcionalidades com carácter específico (*Plug-Ins*), adicionando-lhe subrotinas escritas na linguagem de programação C.

O capítulo VII apresenta o desenvolvimento de um protótipo CD-i, usando o sistema de autoria MediaMogul. O protótipo é sobre a vida e obra de *Leonardo da Vinci*.

II

SISTEMAS MULTIMÉDIA

Os conceitos teóricos que levaram à especificação dos primeiros sistemas multimédia foram estabelecidos nos anos 40; no entanto, o desenvolvimento de sistemas multimédia evoluiu muito lentamente ao longo de 40 anos, porque foi necessário esperar pelos avanços tecnológicos de suporte à sua implementação prática. Só recentemente (últimos 10 anos) é que as novas tecnologias de informação permitiram a disseminação dos sistemas multimédia numa escala comercial. O desenvolvimento dos sistemas multimédia está intimamente ligado ao aparecimento de produtos que melhoram a comunicação entre o utilizador e a máquina, tais como o rato (*mouse*), o controlo à distância, as teclas de funções especiais e ecrãs sensíveis ao tacto.

A modelo **multimédia** não se encontra limitado ao domínio da informação (ex: bases de dados, gráficos digitais, áudio, simulação), ou aos meios usados para estruturar grupos de documentos dentro de um determinado contexto. Potencialmente, trata-se de um **paradigma unificador** para abordar a presente diversidade de documentos, em que cada tarefa ou material requer uma ferramenta independente e especializada. O ambiente multimédia pode ser comparado a um gestor de aplicações, cuja funcionalidade persiste entre as aplicações, fazendo a integração dos diferentes meios de apresentação da informação. Este modelo permite aumentar a qualidade do sistema de informação, dada a variedade de meios de informação a que pode recorrer e ao facto de poder melhorar o modo de como esta informação pode ser usada, através de ferramentas consistentes para a sua apresentação e manipulação [**WOODHEAD90**].

2.1 TERMINOLOGIA

Para se compreender a evolução das tecnologias de informação multimédia considera-se importante clarificar alguns dos conceitos básicos.

Meios (*Media*)

Conjunto básico das **tecnologias de comunicação** e **armazenamento de informação**. Genericamente, os meios abrangem texto, gráficos, áudio, imagens e vídeo. Também incluem o código formal em que são transcritos, os materiais onde são gravados e as convenções pelas quais são organizados em colecções.

Normalmente, um investigador necessita de consultar diferentes meios (ex: papel, livros, bibliotecas) para obter a resposta adequada ao seu problema, de acordo com a estratégia adoptada. A existência de diferentes modos de armazenamento da informação constitui uma barreira à pesquisa de dados específicos e para aliviar o utilizador deste esforço suplementar é conveniente que a localização física da informação esteja escondida por uma interface de navegação.

Hipermédia

O prefixo **hiper** é análogo ao prefixo **meta** no contexto computacional e pretende salientar que se está a falar de algo que se encontra num nível superior de conteúdo ou semântico.

O modelo hipermédia caracteriza-se pela construção de **aplicações não-lineares**, tendo por base um sistema computacional. Este conceito também é usado para descrever as próprias aplicações. Utiliza-se o termo **hiperdocumento** para catalogar dados que são manuseados por aplicações multimédia, para deste modo mostrar uma separação entre o programa e os dados.

Hipertexto

Palavra originalmente atribuída a Ted Nelson (1960). Trata-se de um subconjunto do modelo hipermédia, no qual a informação está organizada sobre a forma de texto, ficando visíveis ao leitor um conjunto de ligações que lhe permitem chegar a outros locais no texto. Estas **ligações** (*links*) são projectadas para permitir uma **leitura não linear do texto**, mais rápida e assegurando uma coerência de contexto.

Multimédia:

Trata-se do conceito genérico englobando hipermédia e outros modelos **interactivos**. A palavra multimédia serve para designar um sistema constituído por diferentes meios ou métodos, que se encontravam previamente dispersos por diferentes objectos ou disciplinas. Tipicamente, no contexto computacional, um sistema multimédia tem por base um sistema **digital** de armazenamento ou de processamento de material audiovisual.

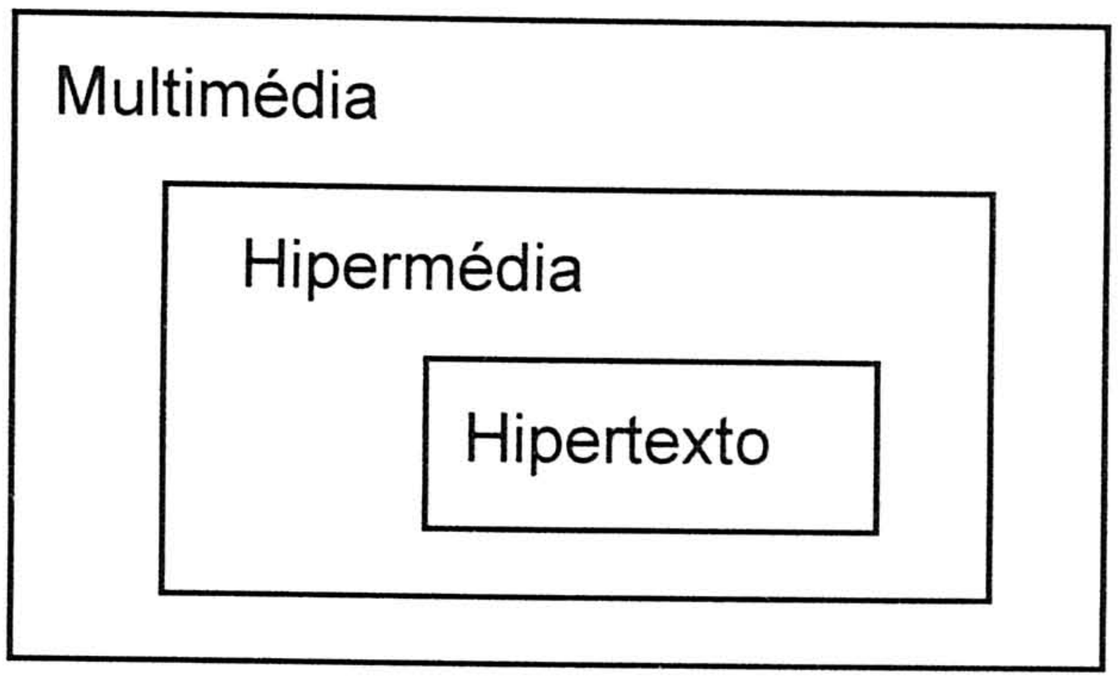


Figura 2.1 - Multimédia

A definição conceptual de um sistema multimédia não é uma questão pacífica. O termo multimédia surge nebuloso, revestido de ambiguidades que são função das arbitrariedades da designação linguística e das configurações propostas pelos construtores. O conceito multimédia é aplicado em diferentes áreas (ex: publicidade) mas neste trabalho considera-se a sua aplicação na área dos sistemas informáticos. A construção de uma unidade multimédia tem a ver com o entrelaçamento dos tipos de informação e as aplicações têm características e modos de procedimento que na maioria dos casos não são compatíveis ou não podem ser combinados entre si. Assim, processamento de texto, base de dados, folha de cálculo, desenho vectorizado, paginação, digitalização de imagens e sons, são entidades que existem de forma autónoma. A sua **integração** num unidade multimédia reveste-se de dificuldades que só agora começaram a ser superadas. Para além do problema de integração, deve acrescentar-se que o conceito multimédia tem subjacente um outro que o justifica, em boa medida. O conceito de **interactividade** não pode ser deixado de levar em linha de conta, porque está ligado a problemas que se prendem com a programação recorrendo a ferramentas de inteligência artificial ou sistemas periciais [**CORREIA92**].

2.2 História dos Sistemas Multimédia

Vannevar Bush (1945) foi uma das primeiras pessoas a propor aquilo que hoje se pode englobar no conceito multimédia. Apresentou o sistema **Memex** que foi desenvolvido à volta de conceitos simples baseados em associações, descrevendo-o como "*algo parecido com uma biblioteca mecânica de documentos privados*". Os utilizadores podiam gravar ou seleccionar itens, fazendo a sua ligação a nomes que eram posteriormente armazenados, permitindo a selecção da informação por associação em vez da indexação normal [**BUSH45**].

Aplicações, como bases de dados, processadores de texto, folhas de cálculo e outras, atingiram um nível de desenvolvimento tal que uma abordagem baseada simplesmente no processamento de dados não é suficiente para satisfazer as necessidades dos utilizadores. O conceito de informação é

qualitativamente superior ao conceito de dados, pelo que foi necessário criar novos modelos para as aplicações.

Com o surgimento de novas tecnologias da computação a partir de 1960 (Ted Nelson) começaram a ser realizados os primeiros protótipos de sistemas com **hipertexto**. Engelbart desenvolveu o primeiro sistema *on-line* de bases de dados, em que o texto foi fragmentado em vários níveis hierárquicos, permitindo vistas gerais e de pormenor em janelas de maior ou de menor nível [ENGELBART63]. Foram desenvolvidos novos sistemas de interacção entre o utilizador a as aplicações, sendo o rato e as teclas de funções específicas os mais importantes, que permitiram que se desenvolvessem aplicações em que o **acesso aos dados era sensível ao contexto** do utilizador (ex: Ajuda / *Help*).

O paradigma proposto por **Randall Trigg** na sua tese Textnet [TRIGG86] é o mais utilizado pelos sistemas hipermédia comerciais. A sua tese é baseada no conceito de **ligações de tipo alargado** (*extensible typed links*), ou seja, quando são definidas as ligações são-lhes associados nomes descrevendo as relações entre as diferentes hierarquias. Um exemplo típico é o sistema **ZOG**, uma base de dados de documentos em texto hierarquicamente ligados, funcionando sobre um ambiente com uma interface consistente e possibilitando vários utilizadores. Este sistema veio a ser aplicado na área da teleconferência e no treino por computador.

Na década de 80 os computadores pessoais sofreram um aumento drástico da capacidade de processamento, sendo possível a execução de aplicações multimédia no domínio doméstico. Os principais impulsionadores foram os pacotes de software **Guide** (PC) e **HyperCard** (Mac), que permitem ao utilizador estabelecer ligações entre texto, gráficos e som. A capacidade de integrar meios de origens diversas, tais como vídeo e som, resultou da aplicação de modelos análogos aos usados nas bases de dados relacionais. Surgiram finalmente ambientes de alto-nível de programação (ex: **HyperTalk**) fornecendo ao utilizador uma interface verdadeiramente amigável.

A difusão pelas vias comerciais de software catalogado como sendo de hipertexto, hipermédia, sistemas de ensino inteligentes e sistemas multimédia interactivos, destinado a funcionar sobre sistemas baseados em micro-computadores com algum hardware específico (CDROM, Placa de Som, Placa de Vídeo e Sintetisadores) fazia com que se caminhasse a passos largos para a visão de Bush (Memex - um estação multimédia interactiva portátil).

2.3 Plataformas Multimédia Baseadas em CDs

Um sistema multimédia tem de aceder a elevados volumes de dados, o que se traduz num problema grave, pois é necessário encontrar uma plataforma de armazenamento da informação que seja eficiente e fácil de manusear pelo utilizador. A tecnologia de gravação laser permitiu o desenvolvimento de **discos compactos** (CDs), discos de reflexão óptica com 12 cm de diâmetro, que se podem definir genericamente como um meio de armazenamento de dados digitais de alta densidade.

A plataforma multimédia CD-i resultou de uma evolução da tecnologia associada às plataformas baseadas em discos compactos (CDs), havendo uma sequência cronológica no surgimento de diferentes plataformas, que são introdutoriamente aqui apresentadas.

As plataformas multimédia baseadas em CDs são muito similares, porque têm uma natureza comum:

- Capacidade de armazenar a quantidade e variedade de informação sob a forma de som, imagens, texto, gráficos, dados e o programa de controlo;
- Permitir a transferência eficiente de toda a informação do meio de armazenamento (CD) para os módulos apropriados do sistema de saída (ex: televisão, colunas, PC).

As várias plataformas existentes oferecem um vasto leque de características cujo objectivo é responder às diferentes necessidades dos vários sectores do mercado. Em seguida apresenta-se uma breve descrição das plataformas multimédia baseadas em CDs, identificando as principais similaridades e diferenças. A figura 2.2 ilustra a proximidade das plataformas baseadas em CD.

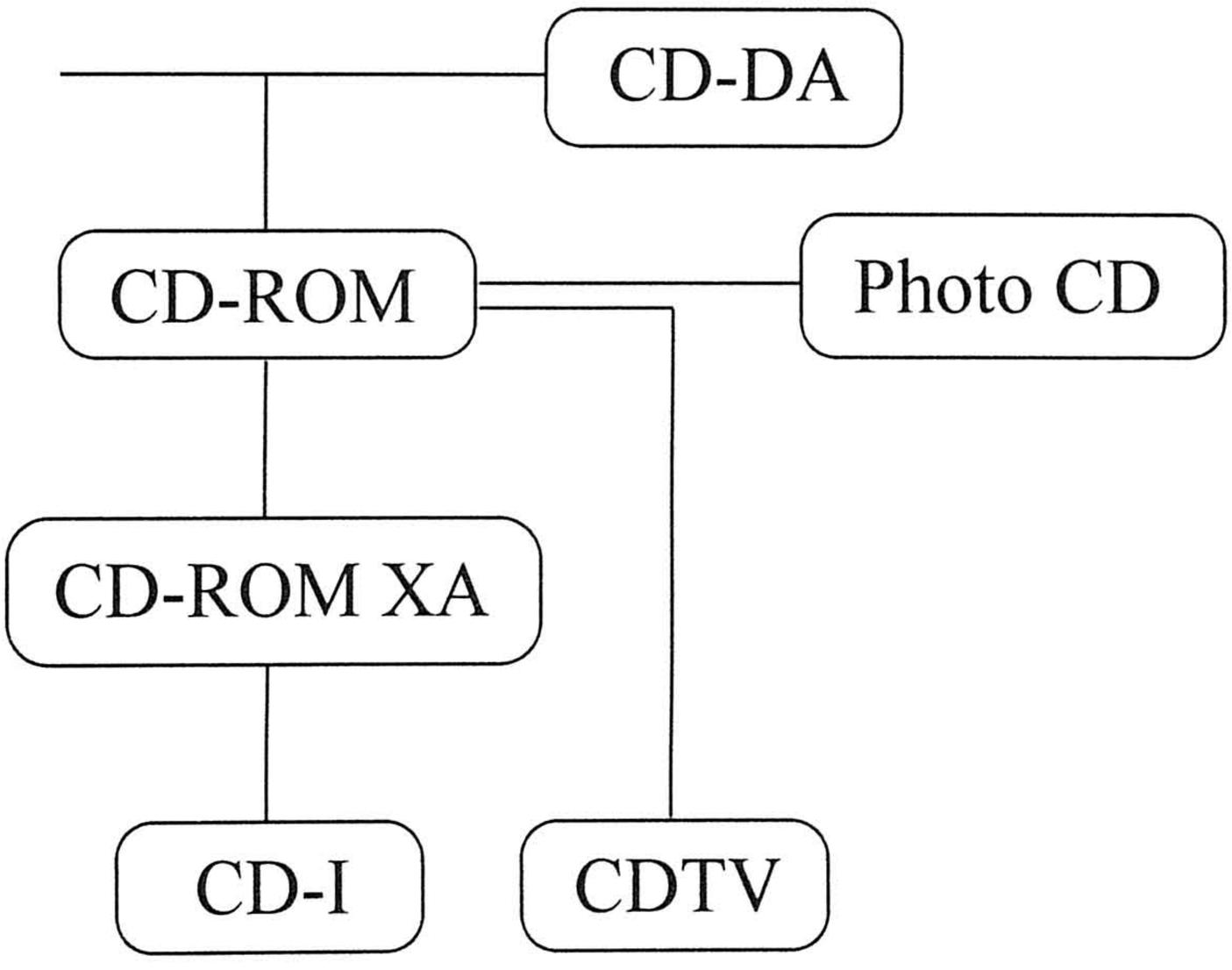


Figura 2.2 - Plataformas multimédia baseadas em CDs

CD-DA

O CD-DA foi desenvolvido pela Philips e pela Sony, tendo sido lançado no mercado em 1982 e constitui a primeira plataforma digital de uma série de novos produtos.

A plataforma CD-DA foi largamente aceite no mercado de consumidores de **música**. Ainda hoje ao CD está associado um conceito de grande qualidade de som, mas com o surgimento de jogos (ex: SEGA) de computador noutros formatos de CD esta conotação está-se a perder. O CD-DA utiliza um sistema de codificação de alta qualidade PCM (*Pulse Code Modulation*), que

pode ocupar todo o espaço de armazenamento disponível, a largura de banda e a capacidade de processamento do leitor para reproduzir som com a máxima qualidade possível.

Não é correcto afirmar-se que o CD-DA seja de uma plataforma multimédia, mas recentemente, algumas das editoras musicais estão a produzir discos **CD+G** (*Compact Disc Plus Graphics*) que podem ser reproduzidos por um qualquer leitor e que oferecem novas funcionalidades para as pessoas que actualizem os seus leitores de discos compactos, permitindo a exibição de textos ou gráficos, que são gravados juntamente com o áudio no disco compacto.

Um disco CD-DA pode armazenar até **72 minutos** de som digital de elevada qualidade, sendo o acesso às pistas de áudio quase instantâneo. Os leitores de CD-DA mais avançados são programáveis, ou seja o ouvinte pode seleccionar a sequência das músicas ou, alternativamente, pode saltar sobre uma pista que não goste de ouvir. O programa de controlo está residente na memória do leitor de CD-DA, pelo que o disco não sofre qualquer alteração, podendo-se afirmar que virtualmente todo o espaço do disco pode ser usado para gravação de áudio.

CD-ROM

A plataforma CD-ROM foi desenvolvida para oferecer um meio de **armazenamento de grandes volumes de dados**, para sistemas baseados em computador. Este formato surgiu imediatamente a seguir ao CD-DA, em 1985. O disco CD-ROM tem uma capacidade armazenamento de **650 MB** de dados (texto ou imagens), sendo especialmente indicado para servir como um periférico de armazenamento de informação para sistemas de bases de dados [**PHILIPS91**].

Na prática, uma boa parte da capacidade do disco é usada para guardar índices de pesquisa e programas que controlam o elevado volume de dados. Pode-se caracterizar o CD-ROM como um dispositivo robusto e eficiente para armazenamento não só de dados mas também de programas que permitem ao utilizador uma utilização mais versátil da informação.

O CD-ROM satisfaz as exigências de uma enciclopédia com algumas imagens, mas não é suficientemente poderoso e rápido para disponibilizar a variedade de dados requeridos por um sistema multimédia.

Photo CD

A plataforma PhotoCD foi desenvolvida pela Kodak em 1990, constituindo um meio de elevada qualidade para armazenamento de imagens.

Um disco PhotoCD pode armazenar aproximadamente **100 imagens**, com uma resolução de 18 milhões de pixels por imagem, que é 16 vezes superior à definição da televisão (NTSC). O processo de armazenamento das imagens utiliza técnicas de compressão que requerem em termos médios, aproximadamente 6 MB para cada imagem.

O sistema de gravação do PhotoCD processa normalmente um rolo fotográfico (35 mm), sendo as imagens transferidas para o CD (sistema de autoria PAS). Este sistema digitaliza a imagem do filme gerando um ficheiro que é passado a um computador (*IDM - Image Data Manager*) que se

encarrega dos ajustes necessários ao processo de reprodução fotográfica. Finalmente e depois de sofrerem o respectivo tratamento, as imagens são gravadas no disco que constitui o PhotoCD.

Sobre o disco PhotoCD podem-se realizar **quatro sessões de gravação**, permitindo ao utilizador adicionar rolos de 24 fotografias ao longo do tempo. Este disco pode ser reproduzido por leitores dedicados ou leitores CD-ROM XA e leitores CD-i.

CDTV

O CDTV (*Commodore Dynamic Total Vision*) a **primeira plataforma multimédia** a ser lançada no mercado de consumo, tendo sido desenvolvida pela Commodore em 1991 e destinando-se a sistemas baseados em computador, especialmente para jogos de vídeo.

O disco CDTV tem uma capacidade de armazenamento de dados em formato digital de **540 MB**, oferecendo uma combinação flexível de som, imagens, pequenas janelas de **vídeo** a 15 imagens por segundo (15 fps), animações gráficas, texto e dados. O leitor de CDTV oferece uma resolução de 640x512 (PAL) com o máximo de 4096 cores, sendo compatível com o CD-DA e disponibilizando dois canais stereo para o som.

O leitor CDTV resulta da fusão do computador Amiga com um leitor de CD-ROM, utilizando o sistema operativo da Commodore o Amiga Kickstart (é incompatível com as plataformas baseadas em DOS).

Esta plataforma acrescida de alguns acessórios (ex: rato) é também um sistema de apoio à educação e treino baseado em computador, destinado ao mercado doméstico (baixo custo).

CD-ROM XA

Esta plataforma é normalmente descrita como uma "ponte" (*bridge*) entre a plataforma CD-ROM e uma plataforma completamente multimédia (ex: CD-i). A designação XA (*eXtended Architecture*) veio colmatar as exigências específicas do texto, som e imagens, introduzindo técnicas de compressão de dados e um processo de gravação mais eficiente, que possibilita o aumento da capacidade de armazenamento e do processamento dos dados.

A extensão XA introduz a técnica de **entrelaçamento** (*interleaving*) de dados de diferentes tipos, isto é, em vez de gravar todo o texto ou todo o áudio ou todo o vídeo em três secções distintas (CD-ROM), é armazenada uma sequência com diferentes tipos de dados, de modo que estes sejam entregues os processadores apropriados, reduzindo os tempos de espera quando a informação muda de tipo. As aplicações multimédia interactivas exigem um fluxo contínuo de informação, libertando o processador para monitorar a acção do utilizador, pelo que esta nova especificação satisfaz melhor estes requisitos.

O CD-ROM XA pode suportar, para além de imagens e texto, sequências de vídeo, animações ou som (ao mesmo tempo). Pode armazenar **6320 imagens** com 256 cores de um espectro com 16,7 milhões de cores e animações (sucessão de imagens com ligeiras variações relativamente ao estado anterior) até 128 cores.

O CD-ROM XA não tem capacidade para processar vídeo real cobrindo todo o ecrã (*Full-motion Full-screen Video*), mas sim uma versão simplificada usando imagens mais pequenas e numa sequência mais lenta, tipicamente **15 imagens por segundo** (15 fps). Esta plataforma pode guardar 16 canais diferentes de áudio (mono), permitindo realizar aplicações em várias línguas ou juntar diferentes canais para obter uma melhor qualidade (stereo).

A estrutura de armazenamento dos ficheiros no CD-ROM XA está definida na norma ISO 9660 também designada por High-Sierra.

VideoCD

A plataforma VideoCD permite a reprodução de sequências lineares de vídeo, gravado sobre um disco compacto, tratando-se de um CD-ROM XA especialmente dedicado ao vídeo. O VideoCD utiliza a norma **MPEG-1** para compressão de vídeo, permitindo armazenar até **70 minutos** de vídeo com qualidade VHS [PHILIPS94]. O áudio, embora comprimido, mantém praticamente a mesma qualidade de um CD-DA.

Um disco de VideoCD pode ser lido em diferentes plataformas: leitores dedicados de VideoCD, computadores pessoais com leitor CD-ROM XA e placa descodificadora MPEG-1 ou em leitores de CD-i desde que possuam a extensão FMV (*Full-Motion Video*) e o disco VideoCD tenha uma aplicação (CD-i) que faça o controlo da sua apresentação.

O disco **CD-i DV** é semelhante ao VideoCD mas só pode ser lido pelo leitor de CD-i, dado que se trata de um disco CD-i com segmentos de vídeo digital (DV).

3DO

O sistema 3DO baseia-se num processador ARM6 RISC com cerca de 6 MIPS a 32 bits, num leitor CD-ROM de dupla velocidade e possui processadores dedicados para o tratamento da imagem e do som. É uma plataforma multimédia destinada a competir com CD-i, especialmente no mercado de jogos de vídeo e filmes interactivos. O leitor 3DO lançado pela Matsushita permite a leitura de discos CD-ROM, CD-DA e PhotoCD [COTTON93]. O mercado de aplicações interactivas baseadas em CDs ainda não se encontra dominado por uma empresa ou norma internacional.

2.4 A PLATAFORMA CD-I

A plataforma **CD-i** *Compact Disc Interactive* (Disco Compacto Interactivo) surgiu como um novo conceito de comunicação com pontencialidades interactivas e a possibilidade de combinar som, texto, imagem e vídeo.

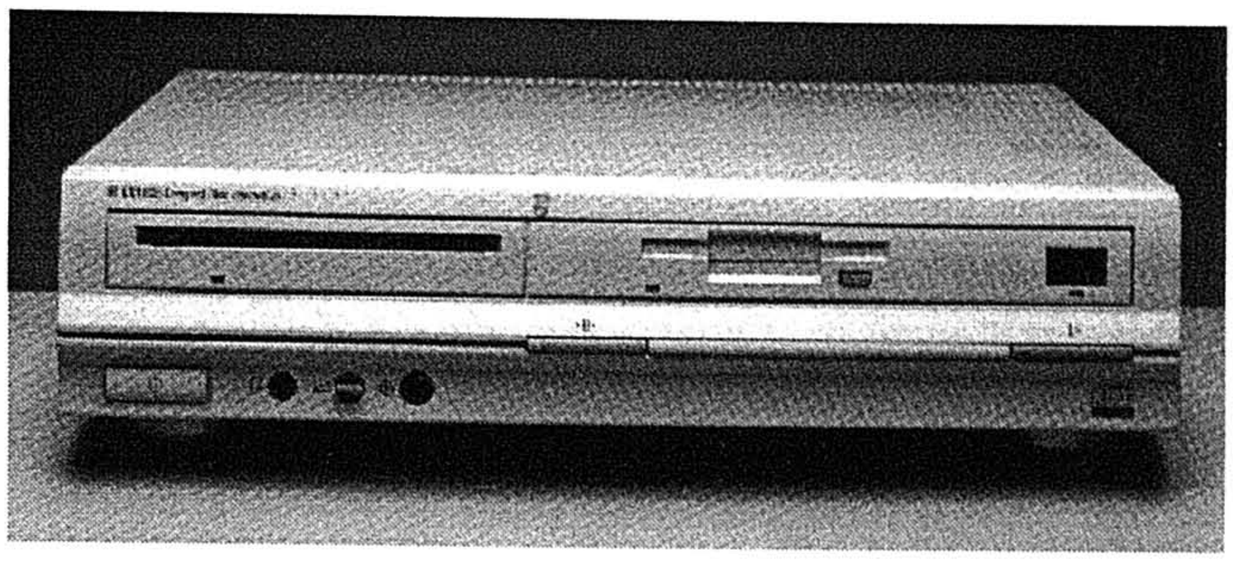


Figura 2.3 - Leitor de CD-i

Esta plataforma foi apresentada pela Philips (Eindhoven) e posteriormente desenvolvida em cooperação com a Sony e a Matsushita, tendo sido realizadas em Março de 1986 as primeiras tentativas de especificação do sistema CD-i. A especificação completa, o ***Green Book*** **[PHILIPS93],** foi publicada em Novembro de 1988 e os primeiros leitores de discos compactos interactivos foram postos à venda ao público em Outubro de 1991.

Encontram-se no mercado um vasto leque de discos CD-i, que se designam por **títulos**, cobrindo áreas diversas tais como; jogos, educação e quiosques multimédia.

Um disco CD-i pode armazenar até **650 MB** de informação que constitui o material multimédia. Por **título** entende-se o conjunto da informação armazenada no CD-i, ou seja, os dados e efeitos audiovisuais, juntamente com o programa que os apresenta interactivamente. A concepção de um título CD-i é realizada tendo por base um estudo do espaço em disco a atribuir a cada conjunto de efeitos audiovisuais. O disco CD-i pode conter quantidades variáveis de imagens, sons, texto e dados de acordo com o tipo de título desenvolvido.

A capacidade de **armazenamento** disponível para cada conjunto de efeitos é um elemento importantíssimo para a concepção da aplicação, porque permite chegar a uma solução de compromisso entre os vários efeitos que se pretendem armazenar. Assim, um CD-i pode conter até 250 000 páginas A4 de texto, mais de 6 000 imagens fotográficas coloridas, até 72 minutos de animação *full-motion* em *full-screen*, até 72 minutos de vídeo, mais de duas horas de som em alta fidelidade ou 19 horas de som mono para discurso falado, capacidade para 16 línguas diferentes com qualidade mono, ou dados num equivalente a 900 diskettes de baixa densidade.

As fases envolvidas no processo de **desenvolvimento de um título CD-i** têm custos e representam esforços que podem ser quantificados em função do esforço total despendido: a fase de **concepção** (10% a 20% do esforço total) engloba a concepção da interface, a definição do grau de interactividade e as características do título; a fase de **criação do conteúdo** (20% a 60% do esforço total); a fase de **teste** (20% a 70%) visa assegurar que o título funciona como pretendido nos diferentes sistemas de apresentação existentes no mercado (leitor CD-i , TV e Hi-Fi).

O CD-i foi concebido para se colocar na fatia de mercado dos electrodomésticos de alta fidelidade, isto é para proporcionar boa qualidade de imagem e de som, com facilidade de ligação

aos aparelhos receptores de TV e aparelhagem sonora (Hi-Fi), revolucionando os padrões do lazer e da difusão de informação.

Foi recentemente apresentado na 5ª Conferência Mundial de CD-i o **TeleCD-i**, constituído por um CD-i acoplado a um modem: o utilizador insere um disco para promoção de determinados produtos e pode imediatamente realizar a sua encomenda com uma simples selecção no ecrã da sua televisão.

Actualmente, já existem plataformas de **CD-i portáteis,** com ecrãs a cores de 10 cm, sugerindo fortemente que uma máquina multimédia portátil será a chave para o sucesso dos discos digitais interactivos.

Os aspectos fundamentais que distinguem o CD-i dos meios convencionais de comunicação assentam nas seguintes características: **natureza digital** (tecnologia óptica) e **operação em tempo-real,** proporcionando um **ambiente interactivo**.

O CD-i permite a ligação a televisores e monitores que obedeçam às normas NTSC (*National Television Standards Committee*), PAL (*Phase Alternate Line*) e SECAM (*Séquence Couleur à Mémoire*) e disponibiliza saídas de áudio em stereo totalmente compatíveis com os aparelhos Hi-Fi. Posteriormente foram desenvolvidos leitores **CD-i** de carácter **profissional** (de que é exemplo o CD-i 605 da Philips) que permitem a ligação extra a televisores e monitores que obedeçam às normas RGB Y/C e CVBS, e dispõem de duas portas entrada/saída, duas portas RS232, uma unidade de diskettes $3^{1/4}$ e uma placa de rede *Ethernet*.

2.4.1 IMAGENS E SOM EM CD-I

O CD-i oferece quatro métodos de codificação para **imagens**, sendo possível escolher vários níveis de qualidade, de acordo com a necessidade de um dado tipo de imagem:

- Até 6340 imagens fotográficas (ex: slides, imagens de vídeo paradas) combinando até 16,7 milhões de cores;
- Até 6340 imagens gráficas (ex: imagens geradas por computador) combinando até 256 cores, de uma escolha possível de 16,7 milhões de cores;
- Até 3170 gráficos complexos com uma definição de 32 768 cores simultâneas;
- Capacidade de realização de animações de imagens com no máximo 128 cores simultâneas.

Estes formatos suportam **efeitos especiais** que possibilitam a visualização das imagens ou a transição entre duas imagens.

Nos leitores de CD-i pode ser colocado (*upgrade*) um cartucho de expansão com capacidade **FMV - *Full Motion Video***, que permite a visualização de segmentos de vídeo em que as imagens são apresentadas à taxa de 25 fps. Trata-se de um sistema de vídeo digital que se prevê venha

revolucionar os sistemas tradicionais de vídeo baseados em cassetes, tal como aconteceu na música com o CD-DA [PHILIPS92a].

No que diz respeito ao **som**, o CD-i disponibiliza 4 níveis de qualidade, permitindo escolher entre dois factores antagónicos: qualidade e quantidade. Para se obter melhor qualidade sonora é necessário usar mais informação, vindo esta a ocupar espaço que não pode ser usado para outros fins [LIMA93].

Níveis de qualidade do som:

- **CD-DA** - até 72 minutos de som stereo, igual ao CD-DA (*Compact Disc - Digital Audio*);
- **Nível A** - até 144 minutos (2,4 horas) de som de alta qualidade em dois canais stereo, comparável a um disco de vinil;
- **Nível B** - até 288 minutos (4,8 horas) de som de média qualidade em quatro canais stereo, comparável a uma emissão de rádio em FM;
- **Nível C** - até 576 minutos (9,6 horas) de som de baixa qualidade em oito canais stereo, comparável a uma emissão de rádio em AM.

2.4.2 Software e Hardware do CD-i

O núcleo do CD-i é constituído pelo sistema operativo **CD-RTOS *Compact Disc - Real Time Operating System***. Trata-se de uma adaptação do sistema operativo OS-9 projectado para suportar a apresentação de produtos multimédia em tempo-real [MICROWARE91c].

A especificação completa do CD-i é conhecida por *Green Book* [PHILIPS93] e complementa a especificação dos CD-DA (*Red Book*), CD-ROM e CD-ROM XA (*Yellow Book*), VideoCD (*White Book*), o Photo CD e outros CDs graváveis (*Orange Book*).

Especificamente, o CD-i oferece três métodos de codificação de áudio (cada um com a possibilidade de ser mono ou stereo) e quatro formatos de codificação de imagens (alguns dos quais com vários níveis de qualidade diferentes). O CD-DA emprega a alta fidelidade da codificação PCM (*Pulse Code Modulation*) que ocupa toda a largura de banda, ou seja, ocupa todo o canal de dados, não deixando espaço para outra informação. O CD-i (e também o CD-ROM XA) usa um sistema modificado designado por ADPCM (*Adaptive Differential PCM*), que oferece três níveis de fidelidade (A, B e C). Estes níveis permitem diminuir a codificação do áudio (libertando espaço em disco) e também o esforço de processamento, à custa de uma redução da qualidade.

Em gráficos de alta qualidade são atribuídos oito bits a cada nível, ou seja, $2^8 = 256$ tonalidades diferentes para cada uma das três cores primárias RGB (*Red, Green* e *Blue*), o que perfaz 16 777 216 tonalidades. O CD-i emprega quatro métodos para a codificação de imagens: DYUV, RGB5:5:5, CLUT e RLE. Estes métodos podem, em níveis diferentes, empregar desde 7 a 16 777 216 cores numa única imagem [PHILIPS93].

2.5 O DISCO CD-I

O Disco Compacto (CD) é semelhante aos discos de vinil. A sua **estrutura** engloba:

- Uma área inicial, designada por ***Lead-in area*** - (pista 0) contém a tabela que descreve o conteúdo do CD;
- Uma área de programa, designada por ***Program area*** - contém todos os dados de apresentação numa ou em várias pistas;
- Uma área final, designada por ***Lead-out area*** - (pista $AA) que identifica o fim dos dados gravados no disco.

Sectores e Pistas do CD-i

A mais pequena unidade de dados que se pode gravar num CD, designa-se por ***Frame*** (no conceito CD-DA) ou **Sector** (no CD-i). Um **sector** do CD-i tem um comprimento igual a **2352 bytes**. Os dados que se podem armazenar num sector tem de ser do mesmo tipo (áudio, vídeo ou relacionados com o programa de controlo), mas pode-se definir um fluxo de dados de diferentes tipos usando uma sequência de vários sectores, assegurando um fluxo constante de informação de diferentes tipos, para as várias partes do sistema. O sistema CD-i processa os dados a uma taxa constante de **75 sectores por segundo**, ou seja, 13 ms por sector.

Toda a informação em cada aplicação é mantida numa única **pista** (*track*), do mesmo modo que as pistas de música são mantidas num LP de vinil. Um CD pode ter desde 1 até 99 pistas, que por sua vez podem ter um comprimento variável desde 300 sectores (4 segundos) até todo o disco. Cada pista pode conter qualquer combinação de dados (usando os formatos suportados pelo CD-i); texto, música, imagens paradas ou em movimento e programas de computador. Um CD de música clássica pode ter todo o concerto numa única pista, enquanto que um CD de música Pop oferece *n* pistas individuais de vários comprimentos. Um CD-i pode igualmente empregar algumas pistas de tamanho elevado, ou um conjunto de várias pistas mais pequenas.

A eficiência com que o sistema CD-i distribui os elementos multimédia pelos processadores específicos depende da organização desses dados no disco e da maneira como esses dados são lidos e processados. A eficiência é determinada pelos seguintes parâmetros:

- ***Seek Time***: tempo necessário para encontrar os dados no disco, deslocar a cabeça de leitura para a posição correcta (para o local onde estão os dados) e começar a transferência de dados;
- ***Transfer Rate***: taxa de transferência de dados entre o disco e o sistema, ou seja, é a taxa de transferência de informação à qual os dados são lidos do disco para processamento.

Os sistema CD-i e CD-ROM empregam controladores rápidos, que movimentam a cabeça de leitura radialmente relativamente ao disco, de um *pick-up point* até ao outro [**HOFFOS92**]. Uma vez posicionada a cabeça de leitura na pista correcta, esta espera a rotação do disco até ao ponto em surgem os dados desejados para iniciar a leitura efectiva.

Os *Seek Times* dependem da localização dos dados no disco, ou seja, encontrar um novo ficheiro demora mais do que encontrar dados dentro do mesmo ficheiro. A posição dos dados no disco (pistas centrais ou pistas na periferia do disco) é menos importante do que o caminho a percorrer entre duas posições de leitura consecutivas. É necessária uma qualquer forma de controlo de tráfego para manter um fluxo contínuo de informação desde o inicio da leitura, passando pelo processador apropriado e até ao respectivo dispositivo de saída. O fluxo contínuo de informação é assegurado pela forma com que os dados são colocados no disco, criando **ficheiros de tempo-real**, recorrendo a uma técnica de **entrelaçamento** (secção 3.3). O envio dos diferentes tipos de dados para as unidades de processamento é realizado pelo hardware, ou seja, pela arquitectura do sistema CD-i (secção 3.1).

A leitura começa mais cedo se for perto do centro do disco (onde a rotação é mais rápida - 500 rpm) do que na periferia do disco (200 rpm). Uma rotação típica demora cerca de 0.12 segundos no centro do disco e cerca de 0.28 segundos na periferia do disco. Dados experimentais revelaram que os *Seek Times* típicos variam entre 0.3 segundos (para dados no centro do disco) e 0.6 segundos (para dados na periferia do disco).

A figura 2.4 ilustra o ***layout*** dos ficheiros de tempo-real, evidenciando a organização das pistas de um CD-i, em espiral. Esta disposição das pistas permite a leitura com uma velocidade linear constante assegurando um maior número de sectores em cada disco, relativamente a uma disposição radial das pistas [**MOREIRA93**].

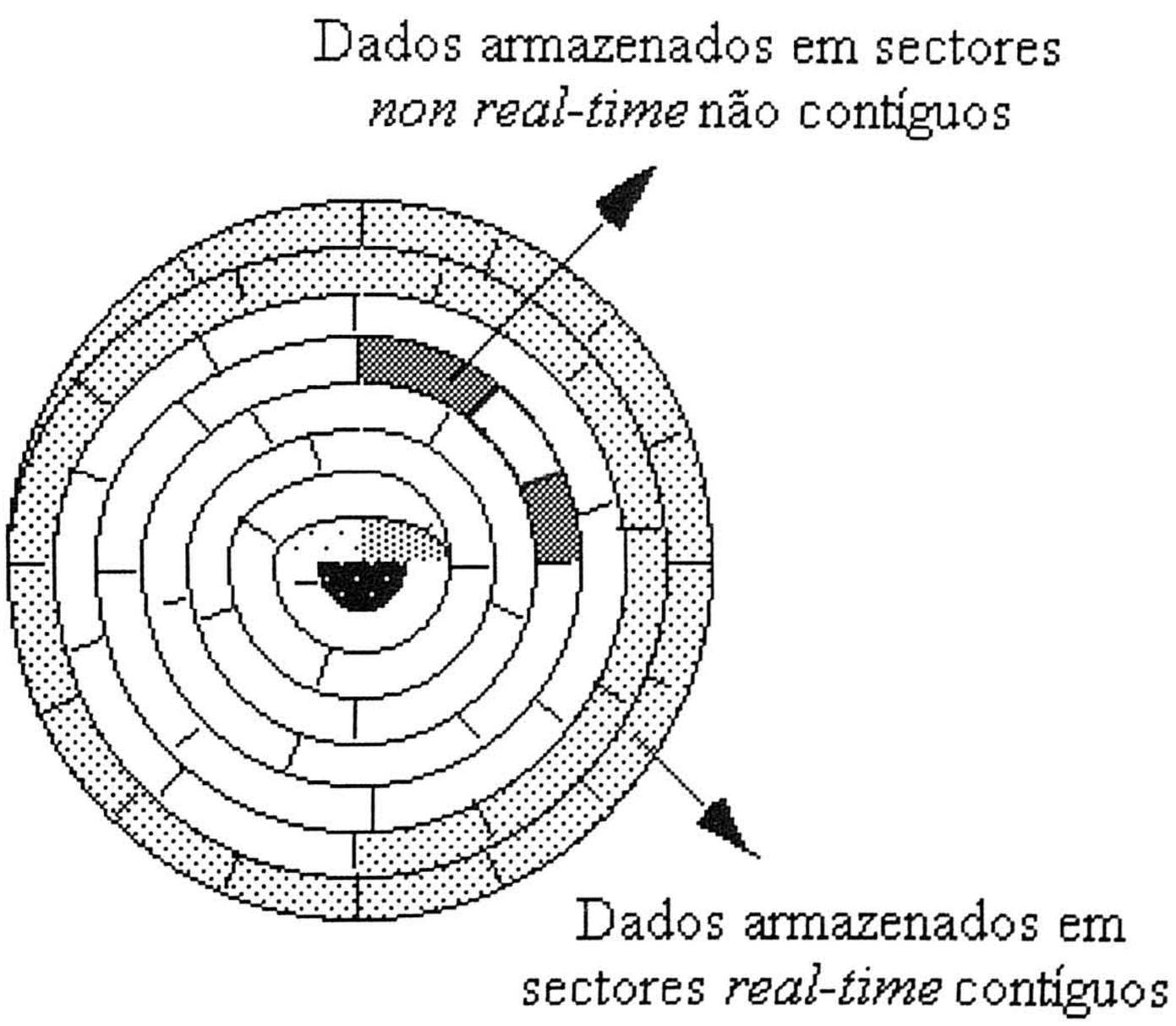


Figura 2.4 - Organização das pistas de um CD-i.

Os ficheiros armazenados no CD-i designam-se por **ficheiros de tempo-real** (*Real Time Files*) e são compostos por um ou mais *Real Time Records*. Um ***Real Time Record*** é uma sequência contígua de *Real Time Sectors* , terminado por um sector que tenha o bit *End Of Record* (EOR) a 1. Um sector de um disco CD-i contém dados áudio ADPCM, dados de imagens (*Green Image Data*) ou outros tipos de dados num formato acessível pelo hardware especializado do CD-i (DMA). O termo **tempo-real** (*real-time*) refere-se a um sistema hardware/software, que fornece ao programador a possibilidade de garantir que certos eventos (efeitos visuais, efeitos de áudio, eventos de programa - saltos, ...) ocorrerão num instante do tempo bem determinado. O processamento em tempo-real é necessário, por exemplo, para sincronizar efeitos de vídeo e efeitos áudio numa apresentação interactiva.

Um *Real Time Data Stream* é um fluxo constante de dados vindos do CD. No sistema CD-i, cada sector contém um número de canal que o leitor CD-i lê à medida que vai lendo o ficheiro. Os dados desejados são encaminhados para os processadores respectivos, enquanto que os dados que não são necessários, são ignorados (*dumped*).

Os dados são muitas vezes temporariamente armazenados em *buffers*, entre o disco e o processador, ou podem, alternativamente, ser encaminhados, ou seja, podem ser transferidos do disco para o processador à velocidade exacta a que são necessários, de modo a manter um fluxo contínuo e sem pausas.

Para os sistemas multimédia, a **sincronização** e o **entrelaçamento** (*interleaving*) são dois conceitos importantes (secção 3.3) para assegurar que todos os elementos díspares (som, imagem, texto e gráficos) de uma apresentação/cena são processados e apresentados na sequência correcta e no momento certo. Por exemplo, a sincronização dos lábios de uma pessoa a falar com o respectivo som é um desafio audiovisual significativo. Vários códigos (códigos de temporização) identificam os dados e sincronizam os elementos individuais numa apresentação multimédia. Para criar uma sequência (*stream*) de dados à taxa correcta, é normalmente necessário adicionar sectores vazios (***Blank Sectors*** - como por exemplo, *Silent-Audio*) para efectuar o ajuste, ao instante, do fluxo de dados. Ao adicionarmos sectores vazios (*Padding*) influenciamos o tamanho do ficheiro de dados, mas a tarefa de organizar eficientemente os dados, de modo a entrelaçar a informação de diferentes tipos, pode minimizar a redundância e diminuir o tempo de acesso aos dados, porque se reduz a movimentação da cabeça de leitura, já que os registos de diferentes tipos de informação se encontram colocados sequencialmente [PHILIPS 92e].

2.5.1 Modos e Formatos do CD

No sistema CD-ROM foram estabelecidas convenções para o armazenamento de dados em Compact Disc (CD), que o CD-i também emprega. Os dados em CD podem ser gravados em dois **modos** (*Modes*), um dos quais tem ainda dois **formatos** (*Forms*), fornecendo uma gama de

mecanismos para detecção de erros e códigos de correcção da informação de diferentes tipos. A escolha entre os dois modos e os dois formatos de gravação da informação depende do facto de querermos mais protecção ou um processamento mais rápido [**HOFFOS92**].

MODO 1 - Este modo contém um nível mais elevado de detecção de erros e códigos de correcção do que o modo2. Foi concebido para dados sensíveis a pequenos erros. As tecnologias CD-ROM, DVI e CDTV empregam o modo1.

MODO 2 - Este modo foi concebido para dados áudio e sinais vídeo que necessitam de ser entrelaçados, sector a sector, para se conseguir um fluxo de informação contínuo com áudio e vídeo, mas que não se deixe perturbar por outros sinais gerados pelo computador. Os sistemas CD-i e CD-ROM XA empregam o modo2.

O **modo2** contém dois formatos; **formato1** (*Form1*) que contém o mesmo nível de detecção de erros e códigos de correcção de dados que o modo1; mesmo os dados no **formato2** (*Form2*) são examinados durante o processamento, de modo a que erros, tais como um pixel ou uma linha mal colocada, possam ser corrigidos antes da informação chegar ao dispositivo de visualização. O sistema PhotoCD emprega sempre o modo2 e formato1. O CD-i emprega ambos os modos, garantindo assim uma elevada protecção com o **formato1** e uma grande rapidez com o **formato2**. Os dados relacionados com o programa (*Run Time Text*, índices, *HotSpots*, navegação, dados não áudio/vídeo ou código) são armazenados em **modo2, formato1**. Os dados no formato1 são tipicamente transferidos à taxa de 150 Kbps; enquanto que no formato2 são transferidos à taxa de 170 Kbps.

Todos os dados gravados em modo1 ou modo2, formato1 (com protecção extra contra erros) têm de passar pelo processador central do sistema. Dados em modo2, formato2 podem ser enviados directamente para os canais de saída apropriados.

A opção de modo1 ou modo2 (formato1 ou formato2) determina a rapidez e a facilidade com que os dados podem ser transferidos e entrelaçados. Por exemplo, o CD-ROM XA, em modo2, envia dados áudio, codificados em formato ADPCM, directamente para o processador áudio; ao passo que o CD-ROM em modo1 tem que enviar todos os dados através da unidade central de processamento. Tanto o sistema DVI como o CDTV, que empregam modo1, usam *buffers* para entrelaçar áudio e vídeo. A tecnologia DVI requer um processador mais poderoso, assistido por *chips* com microcódigo na placa principal do sistema. O CDTV emprega um processador menos poderoso e um conjunto de técnicas inteligentes para lidar com os dados entrelaçados. O CD-i e o CD-ROM XA empregam modo2, que fornece um bom entrelaçamento dos dados e permite a leitura com menos exigências para o processador central do que o modo1. A transição de dados áudio e vídeo em tempo-real (a essência multimédia) é mais fácil em modo2 [**MOREIRA93**].

2.5.2 DETECÇÃO E CORRECÇÃO DE ERROS

O sentido de uma palavra (semântica) é muitas vezes percebido do contexto onde ela se encontra, mas os dados numéricos ou a informação codificada são mais difíceis de interpretar. Muitas técnicas de **compressão** de dados introduzem deliberadamente informação redundante (ex: nos sinais de vídeo) para que seja possível recuperar os dados a partir de uma zona não afectada pelo erro. Quando ocorrem erros, o que acontece normalmente é perder-se não mais do que um campo de vídeo ou uma amostra de áudio. O CD-ROM e o CD-i empregam 3 níveis de codificação para detectar e corrigir dados:

- EDC (*Error Detection Code*)
- ECC (*Error Correction Code*)
- CIRC (*Cross-Interleaved Reed-Solomon Code*)

O conjunto dos 3 níveis de codificação usados no CD-i asseguram uma taxa média de 1 bit de erro em cada 10^{18} bits. Um disco CD-i poderia teoricamente ter um buraco de 2,5 cm de diâmetro e ainda assim ser lido correctamente, dado que existe informação redundante (noutro local) para ser utilizada pelos códigos de correcção de erros CIRC **[HOFFOS92]**.

2.5.3 CANAIS DE TRANSFERÊNCIA DE DADOS

O **entrelaçamento** dos dados e a **sincronização** da informação encontram-se facilitados pela existência de **32 canais** de dados (separados) no sistema CD-i. Dos 32 canais, 16 são usados para dados de qualquer tipo e os outros 16 canais são usados para quaisquer tipo de dados excepto áudio. Um canal de dados (*Data Channel*) é um caminho através do qual os dados do CD-i são encaminhados (*streamed*). Estes canais funcionam como os canais de um televisor, que pode receber muitos sinais difundidos e saltar rapidamente de um para outro, mas só pode visualizar um canal de cada vez. Há no entanto diferenças fundamentais: os televisores têm um par sincronizado de sinais áudio e vídeo, de cada vez (alguns receptores já permitem a recepção de som e imagem num único canal); isto é, a televisão está sempre a receber todos os canais difundidos mas apenas alguns são visualizados. O sistema CD-i tem que ser mais selectivo e oferecer apenas as opções que os utilizadores poderão querer em cada momento. Todos os sinais que passam através do sistema consomem espaço de armazenamento e tempo de processamento; daí que seja um desperdício disponibilizar mais do que é necessário em cada momento **[MOREIRA93]**.

O **modo2** suporta as várias utilizações dos múltiplos canais, mas tanto o sistema CD-i como o sistema CD-ROM XA incluem hardware que garante que os canais são sempre utilizados eficientemente, minimizando as exigências feitas ao processador principal. Por exemplo, num programa de treino de uma linguagem, quando é tomada uma opção, o CD-i emprega simultaneamente canais áudio para oferecer pistas de som complementares, de modo a que os

utilizadores possam comutar instantaneamente de uma para a outra. Outras vezes, com uma utilização eficiente de todos os canais de dados disponíveis, é possível garantir um fluxo continuo de informação díspar a ser apresentada.

Os 32 canais podem ser empregues em qualquer combinação, desde 0 até 32 (cerca de 4 294 967 variações possíveis).

A identificação do canal ou conjunto de canais a serem usados em cada momento é controlada pelo comando ***Channel Selection Mask***. Esta máscara pode por sua vez ser modificada de um registo (*record*) para outro, de modo a que o número das potenciais combinações de canais durante a apresentação seja elevado.

2.6 O FUTURO

Sem pretender fazer futurologia sobre o desenvolvimento dos sistemas multimédia, parece-nos que as grandes linhas de desenvolvimento dos mais importantes laboratórios nesta área serão:

A curto prazo

Os sistemas multimédia vão fazer parte do quotidiano das pessoas de um modo análogo ao que aconteceu com os computadores; o HyperCard já é distribuído juntamente com o computador Macintosh, para os PCs está a ser desenvolvido o Manhole, uma ferramenta que permite facilmente construir sistemas multimédia.

Os sistemas multimédia começam também a ser integrados conjuntamente com aplicações de **inteligência artificial**. Um destes sistemas foi construído por Scott M. Stevens na Universidade de Carnegie Mellon para o ensino de uma técnica de engenharia de software chamada **inspecção do código** (*code inspection*). Esta técnica baseia-se na discussão dos vários aspectos de uma aplicação durante uma reunião em que se atribui aos participantes um determinado papel (ex: moderador, investigador). Na prática revelou-se impossível ensinar esta técnica sem que previamente os participantes tenham participado numa reunião de atribuição de tarefas. A deslocação de peritos a uma reunião de treino para principiantes é cara dado que é necessário realizar mais do que uma reunião.

Este sistema realiza a **simulação** em computador de outros participantes usando inteligência artificial. Assim um principiante pode entrar em várias reuniões de treino e até participar múltiplas vezes na mesma reunião e experimentar diferentes papeis, conseguindo deste modo apreender o método da inspecção do código. A simulação permite o estudo de diferentes cenários, em que o utilizador se vê confrontado com as mais prováveis ocorrências.

A médio prazo

A vasta quantidade de sistemas multimédia que estarão disponíveis ao utilizador de computadores pessoais virá a colocar graves problemas de compatibilidade entre os diferentes sistemas. Actualmente é impossível fazer a leitura num PC de hipertexto desenvolvido sobre a plataforma Mac. Será necessário construir um **sistema multimédia abstracto** que permita a tradução de uma qualquer aplicação multimédia de modo que seja possível correr a mesma aplicação em diferentes plataformas.

A definição de um formato comum a todos os sistemas não resolve a totalidade do problema, dado que a natureza das aplicações multimédia envolve ligações dinâmicas entre diferentes documentos. Não será tão cedo que existirá um sistema onde se tenha acesso a toda a informação literária do mundo (Xanadu) dado que surgem enormes problemas quando um hiperdocumento estabelece uma ligação a outro documento externo. Alguns leitores terão acesso mas outro não, formando-se ilhas isoladas de informação sem um elevado grau de ligação a outros documentos. Esta situação é conhecida por cenário **ilhas de documentos** (*docuislands*).

É necessário resolver problemas de carácter prático, tal como definir um modo de referência normalizado de hiperdocumentos análogo à numeração ISBN dos livros impressos. Será necessário desenvolver canais de distribuição de hiperdocumentos com capacidade de os fazer chegar até junto do utilizador. Tipicamente, as bibliotecas deveriam ter disponíveis hiperdocumentos e permitir a consulta de livros electrónicos (*electronic books*).

III

O SISTEMA CD-I

Um dos aspectos mais significativos do sistema **CD-i** é que ele constitui uma norma mundial, desenvolvida pela Philips, Sony, Matsushita e Microware e, como tal, pode ser considerado como o primeiro meio autêntico de publicação electrónica internacional. Esta norma está compilada com precisão num documento intitulado, ***The Green Book*** [PHILIPS93], documento este que estabelece integralmente as especificações técnicas a que todos os leitores e **discos compactos interactivos** devem obedecer [PHILIPS93]. Desta forma fica assegurada a compatibilidade entre os leitores e os discos produzidos e comercializados em qualquer parte do mundo.

O leitor de CD-i interliga-se às aparelhagens de alta fidelidade, aos receptores de televisão comuns e opera-se por intermédio de um controle remoto manual. A compatibilidade entre o sistema CD-i e os meios audiovisuais existentes é total; o sistema CD-i é compatível com os sistemas de televisão PAL, NTSC e SECAM, bem como com os sistemas áudio de disco compacto.

Neste capítulo serão focados alguns dos aspectos técnicos e práticos considerados relevantes para a compreensão do sistema CD-i.

3.1 ARQUITECTURA DO LEITOR DE CD-I

O leitor de CD-i tem uma arquitectura modular, com blocos específicos que procedem à leitura do material contido no disco, identificam o tipo de informação lida e, de seguida, convertem-no em som, imagens ou texto [PHILIPS92a]. A informação pode ser apresentada imediatamente ou armazenada na memória do leitor até ser chamada pela aplicação ou pelo utilizador de uma forma interactiva. A arquitectura do leitor CD-i engloba os seguintes blocos:

- **Controlador CD-DA**; este bloco descodifica sinais provenientes de discos CD-DA e som de qualidade CD armazenado nos discos CD-i. Os sinais áudio podem ser ouvidos à medida que são lidos do disco.
- **Descodificador ADPCM**; descodifica sinais áudio contidos em discos CD-i.
- **Controlador de vídeo**; bloco que fornece a interface entre o descodificador de vídeo, os dois bancos de RAM e o barramento principal do sistema.
- **Descodificador de vídeo**; descodifica sinais de vídeo para a maior parte do material visual com excepção do FMV. Os sinais de vídeo são guardados na memória RAM do sistema (1 MB) e são sincronizados com o áudio pelo software (aplicação).
- **Memória RAM**; a memória do leitor utiliza-se para armazenar o software da aplicação (ou o título) e outras peças tais como ***soundmaps*** (pequenos blocos de som que são utilizados várias vezes no título) e ***sprites*** (pequenas imagens utilizadas frequentemente). Também se utiliza a memória para armazenar imagens, vídeo e gráficos. Esta memória encontra-se dividida em duas áreas de 500 KB intituladas **planos de imagens** (secção 3.5.2.1). Estas áreas são utilizadas para a manipulação de imagens.
- **Memória NVRAM**; o leitor de CD-i permite reter informação em memória através da NVRAM (*Non-Volatile Random Access Memory*) com 8 KB de memória permanente que podem ser utilizados para armazenamento de dados em aplicações específicas, como por exemplo os quadros de pontuação de um jogo de vídeo. Assim os utilizadores podem desligar a máquina quando quiserem sem perder essa informação e mais tarde recomeçar o jogo na posição em que se encontravam. A memória NVRAM é alimentada por uma pequena bateria para proteger os dados quando falha a alimentação do CD-i.
- **DMA**; o leitor de CD-i tem um modo de acesso directo à memória (DMA) que permite transferir informação directamente do disco para os *buffers* de memória e daí para os processadores e dispositivos de saída apropriados. O controlador DMA pode ainda produzir efeitos visuais e outros efeitos especiais, através da movimentação de dados na memória. Com o DMA, os dados podem ser movimentados no sistema sem *mono*polizar a unidade de processamento principal (MPU - *Main Processing Unit*), que pode por sua vez dedicar-se a tarefas mais exigentes.

- **Descodificador *Full-Motion Video***; o processamento necessário para o tratamento de material FMV é mais complexo do que o efectuado durante o tratamento de imagens paradas. Por isso, o leitor de CD-i possui um chip dedicado a estas funções e com 500 KB de memória adicional. Quando este subsistema não está a ser utilizado para tratar vídeo *full-motion*, pode ser utilizado para outros fins. As imagens FMV são apresentadas no plano de vídeo (secção 3.5.2.1).
- **Canais**; os dados são encaminhados do disco para o ecrã ou para as saídas de som através de um conjunto de canais (existem 32 canais no total). O fluxo global de dados é dividido em vários fluxos separados que são colocados nestes canais. Um som de boa qualidade requer uma maior quantidade de dados áudio do que um som de qualidade inferior. Por isso não é de admirar que necessite de um maior número de canais, podendo no máximo ocupar até 16 dos 32 canais disponíveis.
- **Microprocessador**; no leitor CDI605 da Philips é um Motorola 68020 que, tal como em qualquer sistema computadorizado, fornece toda a capacidade de processamento de dados do sistema.
- **Dispositivos apontadores XY** (*pointing devices*); como por exemplo um Mouse, uma Tracker-ball, um Joystick ou uma Lightpen, cujos movimentos são descritos em posições absolutas ou incrementais, ou pela velocidade do movimento. O sistema básico exige que este dispositivo tenha dois botões (*trigger buttons*) e que seja capaz de apontar qualquer pixel no ecrã com resolução normal.
- Um **relógio** e um **calendário** (com bateria); para manter um estado actualizado do tempo e da data (em segundos, minutos, horas, dias, meses e anos).
- Memória **ROM;** que contém o sistema operativo **CD-RTOS** (secção 3.2).

Tal como em muitos sistemas computadorizados, o sistema CD-i tem um **barramento** (*central system bus* ou *signal route*) que interliga todos os elementos descritos anteriormente, e que permite a circulação da informação entre esses elementos, utilizando os canais apropriados.

O *Green Book* especifica o hardware básico que todos os leitores CD-i devem ter. Com o crescimento do mercado, os produtores rivais procuram aumentar as suas vendas fazendo evoluir os leitores de CD-i, acrescentando-lhe novas funcionalidades de modo a haver uma distinção entre as várias marcas. A figura 3.1 mostra o fluxo de dados e os sinais de controlo no esquema básico do sistema CD-i.

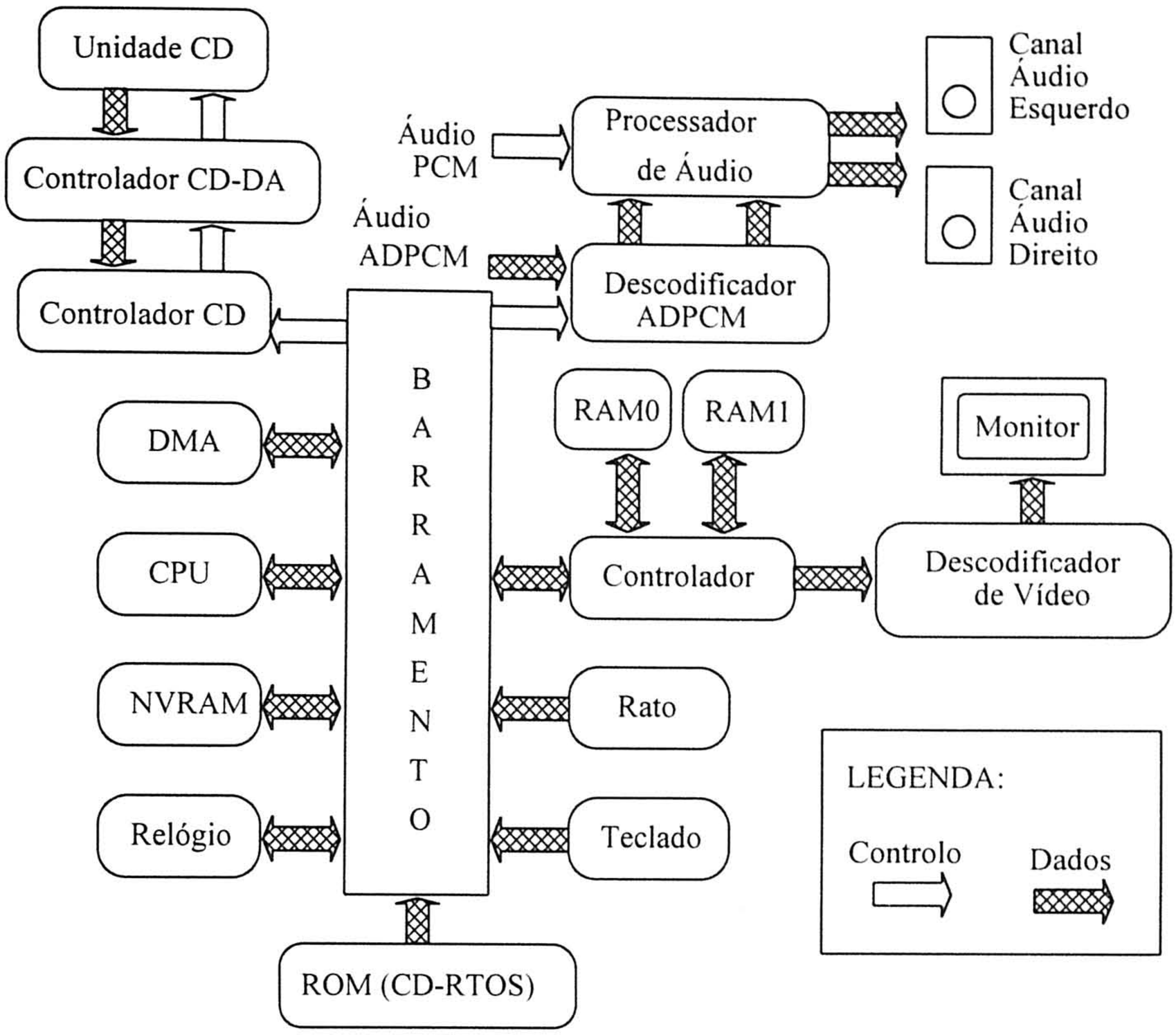


Figura 3.1 - Arquitectura do leitor CD-i

3.2 Sistema Operativo CD-RTOS

Todos os computadores são geridos por sistemas operativos, que não são mais do que conjuntos de programas que controlam todos os recursos de uma máquina computacional e fornecem a estrutura base para a construção das aplicações [TANENBAUM92]. Qualquer sistema computadorizado é um universo controlado pelo seu sistema operativo.

O sistema operativo **CD-RTOS** é restrito ao CD-i. Baseia-se no sistema operativo **OS-9** concebido originalmente para o microprocessador 6809 da Motorola e depois adaptado para o 68020. Estendendo o poder do OS-9, o CD-RTOS gere todos os dados dentro do sistema CD-i em **tempo-real**, ou seja, gere os acontecimentos à medida que eles acontecem. O CD-RTOS utiliza alguns artifícios (*shortcuts - drawmaps, soundmaps*), mas todos os dados são em última análise entregues à medida que são necessários, sob as directivas do utilizador.

Para coordenar o elevado número e a variedade de actividades necessárias em apresentações multimédia de tempo-real, o CD-RTOS tem de ser também um sistema **multitarefa** capaz de gerir concorrentemente vários processos. As características multitarefa do sistema CD-RTOS permitem-lhe controlar uma apresentação de dados dissimilares (áudio, vídeo) a partir do disco até às saídas finais (colunas de som, ecrã) [MICROWARE 91a].

Tecnicamente podemos salientar várias características do sistema operativo CD-RTOS:

- É um sistema operativo **multitarefa** modular. Permite uma estrutura multi-nível, em árvore, dos directórios em disco e suporta não só o acesso aleatório bem como a existência de ficheiros de tempo-real;
- Está dependente do **hardware**, ou seja, requer equipamento específico para poder fazer a apresentação de dados multimédia;
- Suporta o **processamento de entradas/saídas**, de modo a gerir as saídas de áudio e vídeo, as entradas de sinais vindos do disco e de equipamento opcional como o rato e o teclado;
- Suporta ainda **processamento aritmético**;
- Compreende 4 blocos principais: o ***kernel***, os **gestores** (memória, I/O, ...), os **controladores** e as **bibliotecas**.

3.2.1 *KERNEL*

Tal como outros sistemas operativos, o ***kernel*** (núcleo) do CD-RTOS representa o coração do sistema operativo: é a camada mais baixa de qualquer sistema operativo na qual todos os restantes processos se baseiam (ou sobre a qual desenvolvem a sua actuação). O ***kernel*** atribui os recursos de hardware aos vários processos e funções, que o sistema de apresentação e o programa requerem [**MICROWARE91b**]. O *kernel* aloca e desaloca dinamicamente os vários recursos de memória de modo a proporcionar o melhor uso possível do espaço disponível, à medida que o programa se desenrola.

O CD-RTOS controla ainda os aspectos multitarefa (tal como a comutação entre várias tarefas ou a suspensão de uma tarefa enquanto outra está em execução) e responde a interrupções (como por exemplo as resultantes de decisões ou de reacções dos utilizadores).

O ***kernel*** também responde às necessidades do sistema de apresentação realizando chamadas ao sistema e chamadas I/O, à medida que os dados passam do disco para os diferentes processadores e depois para os dispositivos de saída. A configuração do hardware é armazenada no **CSD**.

Configuration Status Descriptor **(CSD)**

O **CSD** fornece um *device status descriptor* para cada elemento discreto do sistema de apresentação. Cada *device status descriptor* define um tipo, um nome, um estado (activo ou inactivo) e parâmetros específicos para cada dispositivo de saída. Partindo desta descrição o *kernel* pode determinar as capacidades de cada dispositivo (desde o descodificador até qualquer acessório opcional).

O *kernel* necessita desta informação para determinar a configuração exacta do hardware do sistema de apresentação: tipo de monitor, destino das saídas áudio (TV e Hi-Fi) e potenciais entradas de dados (rato, teclado, ...). Alguma desta informação está armazenada em RAM não volátil (NVRAM) e outra parte está armazenada em RAM/ROM. Os utilizadores podem fornecer

alguma da informação para configurar os descritores de extensões ou dispositivos opcionais (joystick, teclados, ...).

Inicialização

O procedimento de arranque do sistema é similar ao de todos os sistemas computadorizados. O hardware é inicializado assim que se liga a alimentação, ou seja, são-lhe dadas as instruções iniciais e o *kernel* começa a sua actuação. É visualizado um ecrã pré-definido (***start-up screen***) com uma mensagem (*copyright*) do produtor de hardware. Se existir um disco na unidade de CDs, o sistema arranca a partir das instruções nele contidas, caso contrário o *kernel* compila o CSD para verificar a natureza e o estado de todas as partes do sistema de apresentação.

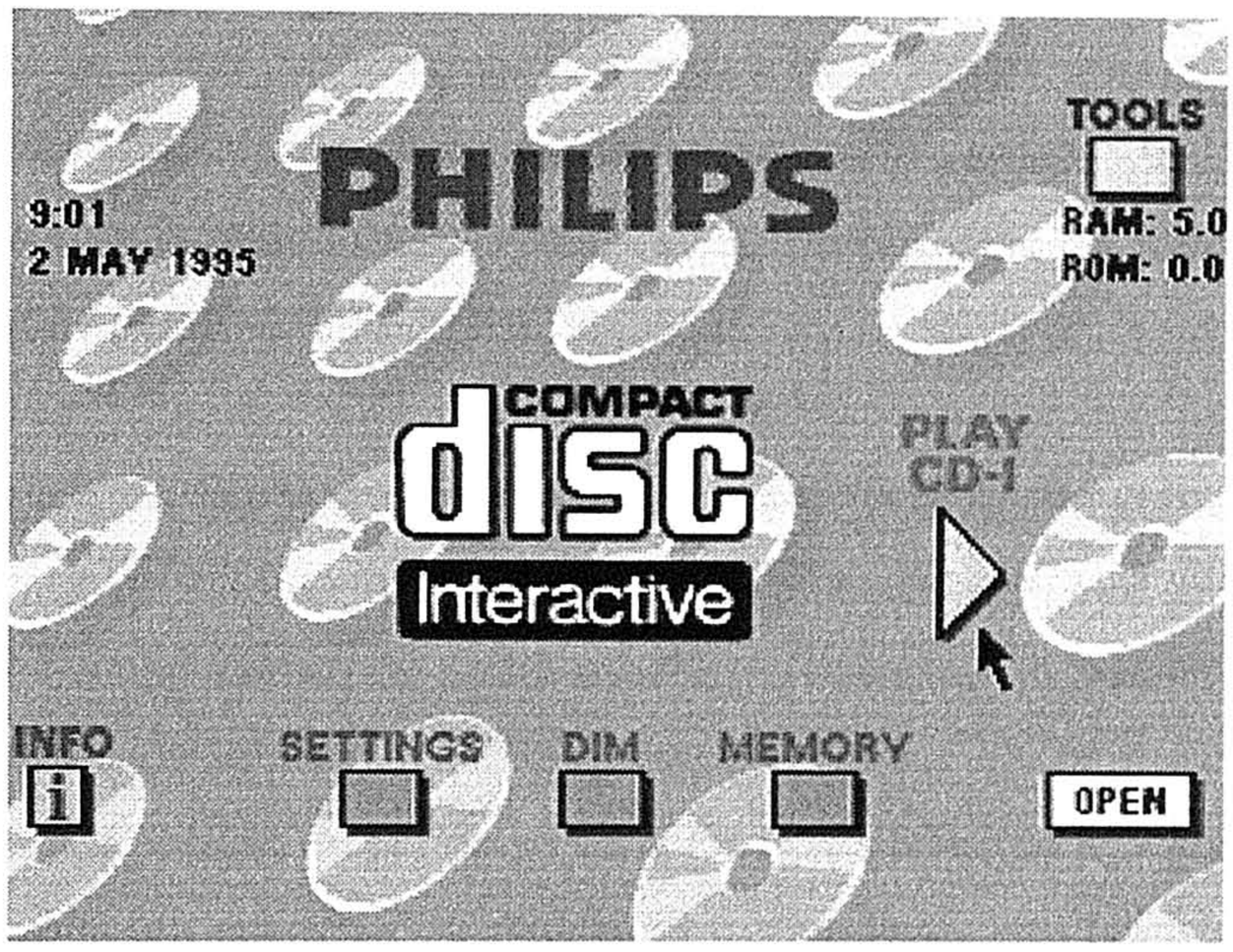


Figura 3.2 - *Start-up Screen*

3.2.2 GESTORES DE FICHEIROS

O sistema operativo CD-RTOS emprega 3 gestores centrais, para controlar as actividades principais do sistema:

- O **CDFM** (*Compact Disc File Manager*) controla os ficheiros do disco CD-i.
- O **NRFM** (*Non-Volatile RAM File Manager*) controla o banco de memória permanente do sistema.
- O **UCM** (*User Communication Manager*) controla os dispositivos audiovisuais e a interface com o utilizador [**MICROWARE 91b**].

O **CDFM** traduz comandos de alto nível do programa (no disco CD-i) em instruções para os vários controladores (dos dispositivos de saída) que encaminham o fluxo de informação para o sistema de apresentação. O CDFM controla o acesso aos vários ficheiros em disco, o que pode incluir a utilização de uma *password* (por exemplo, para garantir confidencialidade ou para aceder

a jogos). Inclui ainda o escalonamento necessário para coordenar os vários acessos ao disco, requeridos pela gestão multitarefa. Através do CDFM os dados podem não ser lidos pela ordem que são inicialmente requeridos, mas sim na ordem que permite a utilização mais eficiente dos recursos disponíveis (ex: a movimentação da cabeça de leitura ao longo do disco). Sendo os dados lidos à taxa de 75 sectores por segundo, o CDFM pode ajustar a ordem pela qual os acessos pedidos são processados, permitindo que o sistema trabalhe o mais eficientemente possível.

O CDFM tem comandos especiais para enviar dados CD-DA directamente para o descodificador ou para o controlador CD-DA e fornecer funções básicas comuns a muitos leitores CD-DA (*Pause*, *Eject*, ...).

Através da **NRFM** uma aplicação num disco CD-i pode criar os seus próprios ficheiros, sendo limitada pela memória existente no sistema. O sistema base requer apenas 8 KB de memória NVRAM; daí que, num directório sem subdirectoras, este espaço tenha que ser usado eficientemente. O gestor de ficheiros fornece uma versão simplificada do **RBFM** (*Random Block File Manage*) do sistema operativo OS-9 [**MICROWARE 91b**].

O **UCM** controla todos os dispositivos audiovisuais do sistema, assim como os dispositivos de entrada (rato, teclado, ...). O UCM controla o fluxo de sinais áudio ADPCM para o descodificador ADPCM (secção 3.6) e inclui efeitos de áudio especiais (atenuação e *panning*) que são criados dentro do próprio sistema de apresentação. As funções gráficas UCM do cursor controlam o tamanho, a forma, a cor e a posição do cursor no ecrã. Os dois planos de imagem (secção 3.5.2), as funções de texto e desenho em *run-time*, os *drawmaps* e *soundmaps* são também controlados pelo UCM.

O UCM aloca e desaloca memória para *drawmaps* e *soundmaps* através da chamada das funções *create* e *close*. O UCM move ainda os dados entre *drawmaps* separados para produzir efeitos tais como *screen updates*, *scrolling*, *panning* ou animação. As funções gráficas UCM de desenho fornecem capacidades de desenho básicas em *run-time* e funções de texto. As funções de desenho incluem linhas, rectângulos, polígonos, círculos, elipses, arcos e enchimentos (*fills*) com escolha de padrões (*patterns*), cores e largura de linhas. O UCM pode definir regiões de várias formas básicas (simétricas ou poligonais) para limitar a área de desenho dentro de um *drawmap*, ou para criar *mattes* para efeitos especiais. As formas básicas (*shapes*) podem ser combinadas para acomodar formas gráficas irregulares (ex: para delimitar uma imagem e depois cortá-la para se poder visualizar noutro *drawmap*). O UCM controla também o DCP (*Display Control Program*).

O **DCP** regula cada campo (*field scan*) do televisor ou monitor, e serve para gerir a visualização de imagens no ecrã ou para produzir **efeitos especiais**. Uma imagem no ecrã resulta da sobreposição de vários **planos de imagens**.

O DCP controla vários parâmetros, incluindo os métodos de codificação de imagem, transparência, a cor do plano de fundo e a ordem pela qual aparecem os planos (Plano A sobre o

Plano B, ou vice-versa). Define, para cada plano, as **LUTs** (*Lookup Table*), máscaras de cor e transparência, efeitos especiais (mosaicos e *mattes*), e o factor de contribuição de imagem (intensidade ou brilho relativo). O DCP fornece uma **FCT** (*Field Control Table*) e uma **LCT** (*Line Control Table*) com instruções para cada campo e para cada linha dentro de cada campo. Como o CD-i tem dois planos centrais de imagem (A e B), também há **DCPs** separados para cada um, ***path0*** e ***path1*** respectivamente, e as instruções que afectam os dois planos são sempre suportadas no *path0*.

A **FCT** é executada 50 vezes por segundo, num sistema PAL/SECAM, ou 60 vezes por segundo, no sistema NTSC. Pode conter até 1024 instruções para estabelecer os parâmetros de cada campo. As variáveis dos atributos podem também ser actualizadas (ex: determinar a ordem dos dois planos em qualquer momento).

A **LCT** é executada 625 vezes por segundo, num sistema PAL/SECAM, e 525 vezes por segundo, num sistema NTSC, em resolução normal. Fornece cerca de oito instruções discretas por cada linha e pode fornecer funções separadas para cada linha do ecrã. O DCP suporta até oito fronteiras *matte* em cada linha, embora uma ou mais se possam sobrepor.

Em conjunto estas duas tabelas colaboram na apresentação de *mattes*, fronteiras, *scrolling*, *panning*, *sun-screens*, *partial-screens updates* e outros efeitos onde instruções diferentes têm que ser enviadas para várias zonas do ecrã. Especificando individualmente o *display start address* para cada linha, por exemplo, o DCP pode conter uma parte da imagem do ecrã e pode fazer o *scroll* de outra parte. Definindo uma combinação de *mattes* opacas e transparentes, em dois planos de imagem, o DCP pode suster uma imagem parada numa *matte* no topo do plano e passar um vídeo (animação) a partir do segundo plano através das *mattes* transparentes (por exemplo para produzir a animação característica dos *cartoons*).

Outros Gestores de Ficheiros

O sistema CD-i tem ainda outros gestores de ficheiros, baseados em modelos do sistema operativo OS-9 e em sistemas mais evoluídos do que o sistema básico. Estes gestores incluem:

- Um **NFM** para controlar um conjunto de leitores CD-i e também, se necessário, um conjunto de sistemas OS-9, ligados em rede.
- Um **RBFM** para controlar ficheiros num dispositivo de armazenamento de massa, como por exemplo um disco duro, uma unidade de diskettes ou numa unidade na memória de acesso aleatório RAM (RAM *disk drives*). O **RBFM** (que faz a gestão dos ficheiros e outros meios de armazenamento - ex: disco duro) complementa o *compact disc file manager* (que faz a gestão dos ficheiros do disco CD).
- Um **SCFM** (*Sequential Character File Manager*) que controla dispositivos periféricos opcionais, tais como: impressora ou modem. O **SCFM** complementa o UCM que controla os dispositivos do sistema CD-i base.

Todos os sistemas de gestão de ficheiros comunicam com o equipamento do sistema de apresentação ou com a rede através de controladores (*device drivers*) intermédios.

3.2.3 CONTROLADORES (*DRIVERS*)

Todos os sistemas operativos empregam *drivers* para controlar os periféricos do sistema de apresentação (*delivery system*). Os controladores utilizam funções de baixo nível necessárias para lidarem directamente com vários dispositivos de entrada/saída.

O leitor CD-i pode por exemplo ser ligado a diferentes modelos de televisores e a vários modelos de teclados ou joysticks, na medida em que muitos produtores começam a fornecer produtos para o mercado CD-i. Os controladores eliminam as diferenças entre os vários produtos, normalizando a comunicação entre o sistema de apresentação, os vários gestores de ficheiros e os dados do disco.

Para identificar o equipamento do sistema de apresentação, o sistema operativo CD-RTOS emprega *device descriptors* que incluem o nome do respectivo *device driver* e do gestor de ficheiros, juntamente com informação essencial acerca do dispositivo. Esta informação fornece ao sistema operativo os endereços correctos para comunicar com os vários dispositivos de entrada e saída do sistema.

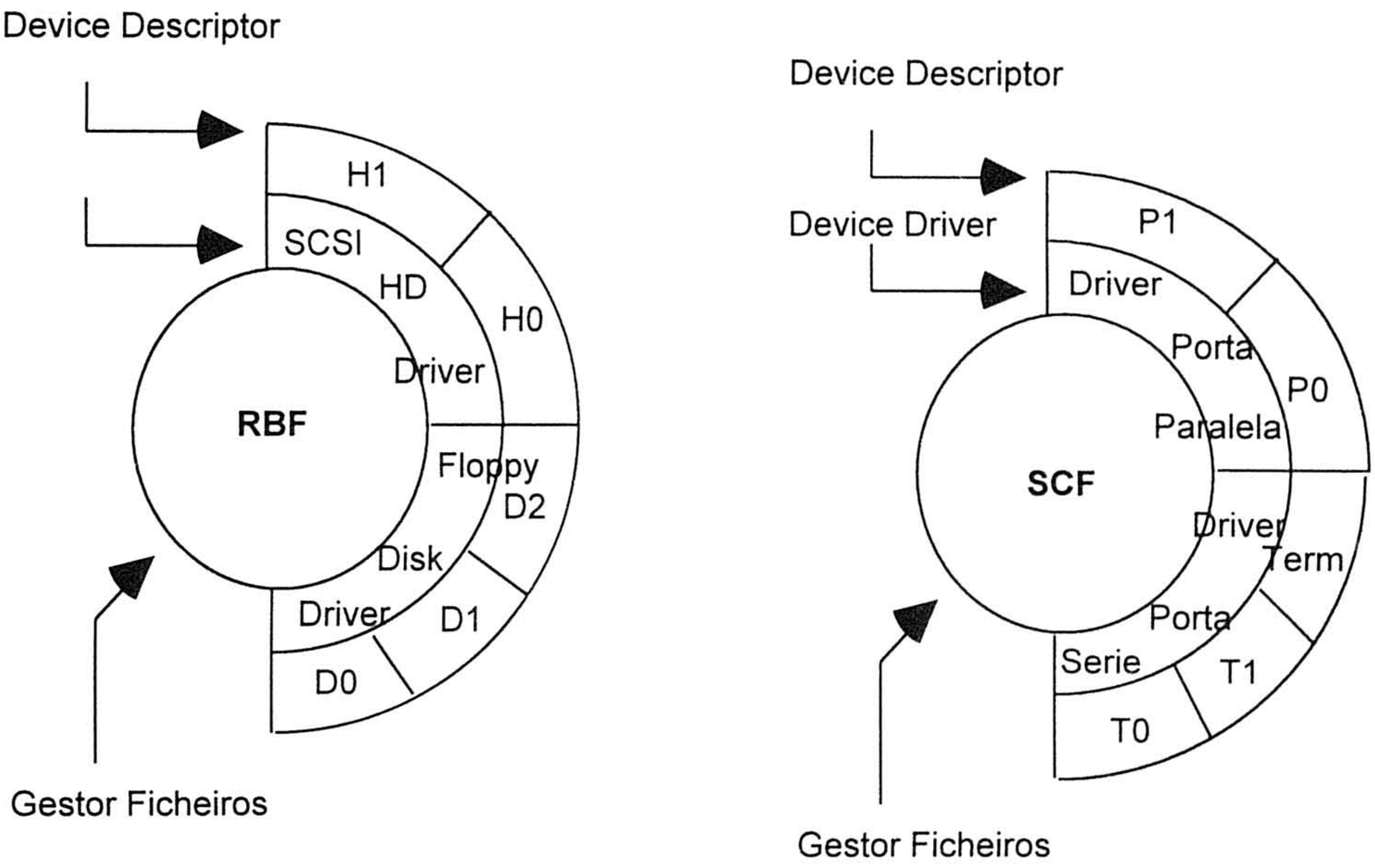


Figura 3.3 - Camadas (*devices*) do Sistema Operativo.

Sobre as camadas do sistema operativo são desenvolvidas **bibliotecas** que o programador pode incluir no seu código ao implementar uma aplicação, sem necessidade de um conhecimento exaustivo dos vários componentes do sistema operativo.

Bibliotecas

A maior parte dos sistemas operativos disponibiliza programas ou bibliotecas, que fornecem o código de funções básicas frequentemente executadas [MICROWARE90]. Estas bibliotecas incluem utilitários (que executam várias tarefas úteis, tais como funções de formatação de texto ou cópia de

ficheiros) e *assemblers* ou compiladores (que traduzem o código fonte escrito em *assembly* ou numa linguagem de alto nível para código binário) [MICROWARE91a].

O sistema operativo CD-RTOS tem as suas próprias rotinas, que incluem funções matemáticas necessárias para lidar com dados de vários tipos, funções de entrada/saída e sincronização. Os programadores de aplicações CD-i podem incluir bibliotecas ou funções já produzidas (*proprietary software*) no código da aplicação, para facilitar a programação. No apêndice A encontra-se uma breve descrição dos comandos do sistema operativo OS-9.

3.3 Entrelaçamento de Áudio e Vídeo

Os dados lidos do disco CD-i são controlados cuidadosamente, assegurando-se assim que os dados audiovisuais e outros tipos de dados chegam aos dispositivos de saída do sistema no momento exacto e na ordem exacta. Para conseguir isto, os dados têm que ser eficientemente organizados nos sectores do disco, ou seja, têm que ser **entrelaçados** de modo a serem facilmente lidos em tempo-real. O CD-i armazena os diferentes tipos de informação (áudio, vídeo, texto , ...) em sectores separados do disco. O modo como os diferentes sectores são organizados afecta a maneira como a informação é lida e apresentada aos utilizadores. O processo de colocação dos diferentes tipos de dados em sectores de disco vizinhos é designado por ***interleaving***, ou seja, **entrelaçamento** de fluxos de dados relacionando diferentes imagens vídeo e pistas de som, simultaneamente disponíveis ao utilizador (ver a figura seguinte).

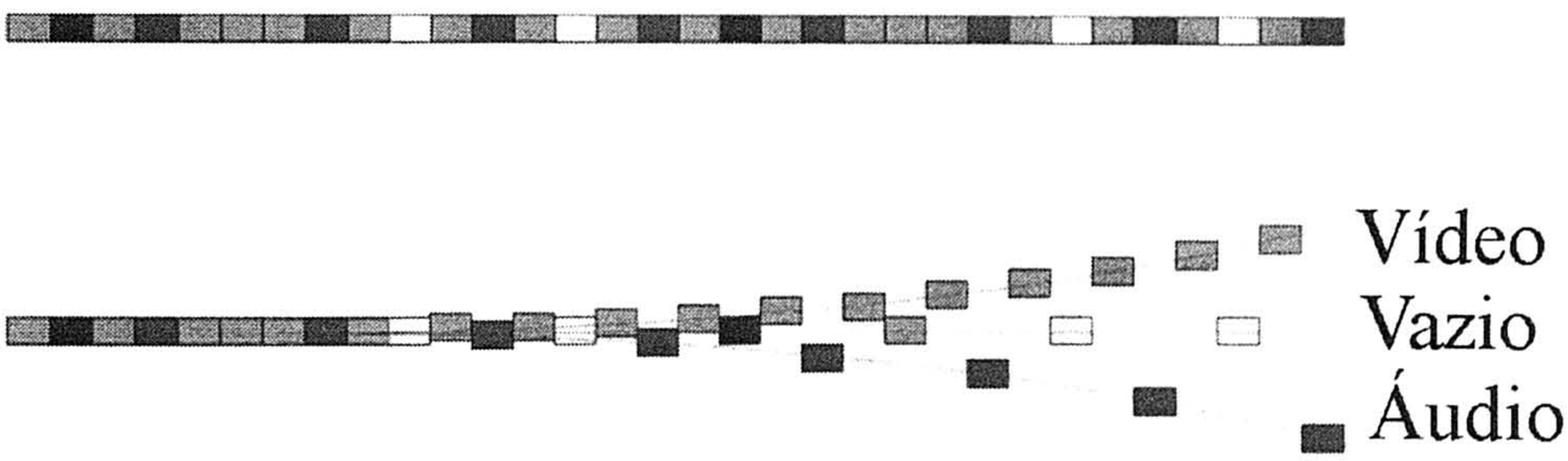


Figura 3.4 - Entrelaçamento de vídeo com áudio.

O disco CD-i é constituído por espirais lineares de sectores. Tal como uma trama num disco de vídeo, um sector digital é a menor unidade de informação disponível no disco (2048 bytes). Os **ficheiros de tempo-real** (*Realtime Files*) distribuem-se pelos sectores do disco segundo uma técnica designada por **entrelaçamento de sectores** (*sector interleaving*). Os bits de vídeo são cuidadosamente alternados com a quantidade de informação áudio necessária para assegurar a sincronização da imagem com o som. Este processo complexo de entrelaçamento de sectores é automatizado por aplicações de construção do disco.

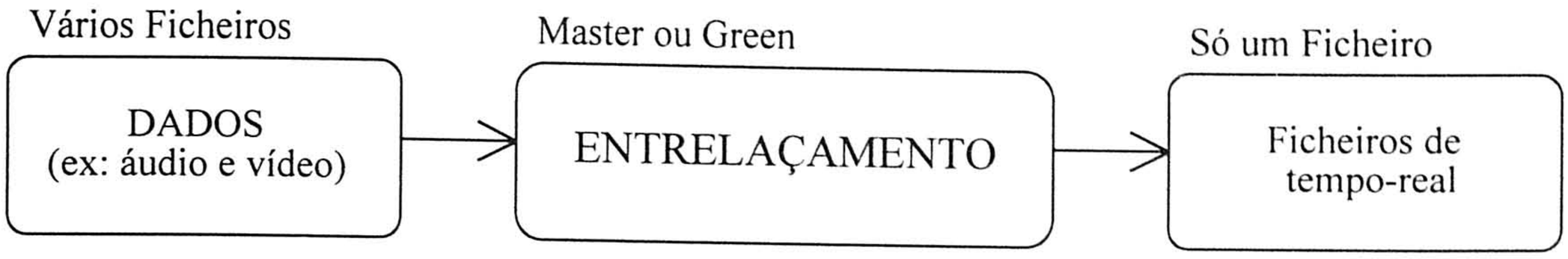


Figura 3.5 - Ficheiros de tempo-real

A utilização do entrelaçamento é particularmente útil para:

- Combinar diferentes tipos de áudio num único disco CD-i: por exemplo, um produtor pode querer por música e efeitos especiais de elevada qualidade numa pista e um comentário (de baixa qualidade) noutra pista;
- Correr várias apresentações áudio em paralelo numa única apresentação, como é o caso de uma apresentação com comentários em múltiplas línguas.

A **sincronização** é realizada a partir do áudio; daí que os sectores áudio tenham que ser postos sequencialmente e ordenadamente no disco. Por exemplo, para o nível C *mono*, os dados áudio de uma **pista de som** (*soundtrack*) têm que ser gravados de 16 em 16 sectores e em mais lado nenhum. Sempre que for necessário mais do que uma pista de som, os sectores para cada uma têm de ser gravados perto uns dos outros, mas mesmo assim têm que manter um padrão interno consistente. Assim, em cada bloco de 16 sectores, 1 pode ser dedicado à pista A e outro à pista B, mas cada sector em cada pista está exactamente à mesma distância do último e do próximo sectores na mesma pista. É este modelo rígido que permite que o sistema possa ler todos os diferentes tipos de dados e possa ordenar os diferentes sinais de áudio em pistas de som separadas e coerentes.

Depois de colocados os dados áudio nos sectores do disco, de acordo com o modelo especificado, os restantes sectores ficam livres para dados vídeo ou outros (como por exemplo, texto, *hotspots*, ...). Numa apresentação que combine imagens paradas com música ou narração, as imagens são tratadas como gráficos individuais e um editor sequencial envia-as para o ecrã. Este editor usa códigos de tempo (*time codes*) para assegurar que os gráficos aparecem na ordem correcta e permanecem no ecrã durante o tempo desejado. Os cálculos são relativamente lineares, já que as imagens não necessitam de uma grande porção da banda do canal.

Quando se visionam imagens em movimento é necessária uma equação mais complexa para calcular a quantidade de espaço disponível para a sequência de imagens, depois de já terem sido colocados os sectores áudio e depois dos métodos de codificação, taxa de *frames* (imagens por segundo) e tamanho das janelas terem sido especificados. Os dados de cada imagem de vídeo devem ocupar um número inteiro de sectores, já que o sector é a mais pequena unidade de dados processada pelo sistema CD-i, tendo portanto de ser totalmente preenchidos (e não parcialmente). A dados que necessitem de 3.65 sectores serão atribuídos 4 sectores.

O número de sectores disponíveis para cada *frame* pode ser calculado de acordo com a seguinte fórmula:

Ns = Trunc[(16-As)*75/16/fps]

em que:

- **Ns:** é o número de sectores disponíveis para cada *frame;*
- **Trunc:** é a função que remove a parte fraccionária;
- **As:** é o número de sectores de áudio, em cada grupo de 16;
- **fps:** representa o número de *frames* por segundo.

com os seguintes valores intermédios:

- O número de sectores disponíveis para dados de vídeo determina-se depois dos dados de áudio terem sido colocados nos sectores e depois de subtraído o número de sectores requeridos para áudio do grupo de 16 sectores, ficamos com o número de sectores disponíveis para vídeo (16-As);
- O número total de sectores vídeo que passam no sistema em cada segundo é calculado pelo produto do número de sectores vídeo por 75 (taxa de transferência de dados) e dividindo o resultado por 16 (número total de sectores em cada grupo).
- Dividindo o resultado anterior pelo número de *frames* por segundo (fps) escolhido, determinamos o número de sectores disponíveis para cada *frame* em cada segundo.
- O número final é arredondado porque os dados têm que ocupar um número inteiro de sectores.

Por exemplo, para um segmento de áudio nível C (que ocupa um sector em cada grupo de 16) e uma sequência de vídeo com imagens a uma taxa de 12.5 fps, temos:

Ns = Trunc[(16-1)*75/16/12.5]

Ns = Trunc[5.625]

Resultado: **Ns = 5 sectores por cada *frame***

Cada sector no formato2 (no qual é gravado o vídeo) contém 2324 bytes de dados. Temos então 5 sectores que dispõem de 11620 bytes de dados vídeo. A restrição principal no entrelaçamento de dados, reside no facto dos sectores de som terem de ser colocados em posições pré-definidas de modo a assegurar uma pista de som com uma cadência suave. A colocação do som nos sectores depende do formato utilizado; contudo a organização dos sectores pode em muitos casos ser tratada por programas e ferramentas de software, e os programadores só têm que se preocupar em não deixar os seus planos ultrapassar os limites de largura de banda disponível.

3.4 CONSTRUÇÃO DO DISCO CD-I

A construção de um disco CD-i ocorre, durante o processo de produção de um disco, depois de termos recolhido e codificado nos formatos respectivos, todas as fontes de imagem, som, textos e gráficos, isto é, depois da fase de programação e antes da produção física do disco CD (CD *master disc*). O objectivo principal inerente à construção do disco é o de reunir num único ficheiro a **imagem de um disco** (*disc image file*) agrupando um conjunto de **ficheiros fonte** (*source files*), de modo a que possam ser interpretados pelo emulador de CD-i, ou então copiados para uma *tape* e enviados para uma fábrica de produção de CDs (CD *mastering facility*).

A imagem de um disco CD é normalmente gravada num disco magnético, a que o emulador de CD-i tem acesso, e que o leitor de CD-i interpreta como se tratasse de um verdadeiro disco compacto (CD). Deste modo pode-se simular o comportamento do título final sem ser necessário gravar um disco CD. A construção de um disco processa-se em duas fases:

1ª Fase - Construção dos ficheiros de tempo-real

- **Formatar** as fontes de dados em sectores CD-i de acordo com as especificações do *Green Book* ou seja, temos que preencher os campos de sincronismo (*sync*), o *header*, *subheader* e os dados; calcular os campos de controlo de erros e aplicar técnicas de *scrambling*;
- **Ordenar** os sectores, provenientes de várias fontes de dados, para produzir os ficheiros CD-i.

2ª Fase - Construção do volume

- Calcular e adicionar a **estrutura** dos sectores do disco CD-i para dar um nome ao disco (*disc label*), às directorias e aos ficheiros com tabelas de *paths*;
- **Dividir** o volume do disco em pistas separadas por pausas ou intervalos (*gaps*).

A **construção do disco** (*disc building*) refere-se à geração de uma imagem do disco numa plataforma magnética em vez de ser numa plataforma óptica. O ficheiro imagem do disco é um ficheiro que reflecte o *layout* e o conteúdo de um disco óptico [OPTIMAGE92a]. Para a construção de imagens de discos existem duas **ferramentas** (*utilities*).

As duas ferramentas são:

- **Master** - Ferramenta para a construção total do disco.
- **Green** - Ferramenta utilizada apenas para a primeira fase (descrita em cima) de construção do disco.

Estas ferramentas são utilizadas para criar um imagem de um disco óptico (CDs) nos formatos **CD-i**, **CD-ROM**, **CD-ROM XA** ou **CD-DA**.

O **Master** e o **Green** estão directamente relacionados e partilham muitas funções:

- Ambos usam um ficheiro escrito numa linguagem dedicada (***script***). Os comandos utilizados nos scripts do Master descrevem um *álbum CD-i* completo, enquanto que os comandos dos scripts do Green são um subconjunto da linguagem Master e descrevem apenas a estrutura dos ficheiros de tempo-real.
- Ambos os utilitários executam o seu trabalho em duas fases principais: **compilação** e **geração**. A maior parte das verificações são feitas na fase de compilação. A fase de geração é executada apenas depois de uma compilação bem sucedida.
- Ambas as aplicações produzem opcionalmente um **relatório** (*map report*) que descreve a alocação de sectores nos ficheiros de tempo-real.
- Ambas as aplicações operam em ***dry run mode***, no qual apenas se considera o tamanho dos ficheiros de dados, sem criar os principais ficheiros de saída (a criação dos ficheiros de tempo-real é muito lenta). Este modo de operar é útil para detectar e corrigir erros na compilação e para gerar e verificar os mapas dos ficheiros de tempo-real.

O **Master** tem como entrada um script que descreve a imagem do disco e o respectivo conjunto de ficheiros de dados, como por exemplo imagens, som ADPCM, som CD-DA, programas executáveis, dados arbitrários e ficheiros de tempo-real pré-produzidos (*premastered*). O Master assume que todo o processo de codificação dos dados já foi realizado.

As saídas principais do Master resumem-se a uma **imagem completa do disco** para cada um dos volumes definidos no álbum CD-i.

Existem contudo saídas secundárias opcionais, como por exemplo:

- Uma **Tabela** que descreve as **pistas** do volume do CD-i, fornecendo informação necessária, ao emulador ou a outras facilidades de *mastering*, para gerar o código do subcanal associado;
- O ***Map report***, que descreve as relações entre os ficheiros fonte e os sectores alocados para cada ficheiro de tempo-real.

O algoritmo executado pelo **Master** pode ser decomposto nas seguintes fases:

1. O script é examinado (***parsed***) e colocado numa lista ligada com as descrições do volume. Cada descrição tem a forma de uma árvore de directórios e cada nó da árvore do volume descreve um ficheiro CD-i em termos dos ficheiros fonte pelos quais é constituído. As verificações léxicas, sintácticas e semânticas (incluindo a acessibilidade dos ficheiros fonte, largura de banda máxima, número máximo de pistas, tamanho mínimo das pistas e tamanho máximo do volume) são feitas nesta fase de análise do script;

2. Se a análise anterior foi completada sem erros, o processo de **geração de volumes** é chamado para cada volume do álbum;
3. O processo de geração de volumes é responsável por todo o volume de pistas e por toda a estrutura de ficheiros do disco. Dependendo da presença ou não de *red files* (ficheiros só com áudio CD-DA), esta fase é feita em dois ou três passos sobre a árvore de descrição do volume. O primeiro passo gera sequencialmente para uma única *green track* todos os *non-red files*, chamando um construtor especial de ficheiros para cada tipo de ficheiro. Os sectores para as directorias são reservados à frente dos ficheiros que elas contêm e não são analisados nesta fase. Existem duas políticas de análise da árvore, para o controlo global do posicionamento dos ficheiros: a primeira (usada por omissão) faz uma análise da árvore em largura (*breath first*); a segunda faz uma análise da árvore em profundidade (*depth first*).

O ficheiro com a imagem do disco é criado no directório onde se encontra o script e usa o mesmo nome com a extensão que indica o tipo de imagem do disco que foi criada. Existem diferentes extensões para os ficheiros (Tabela 3.1) de acordo com o tipo de imagem do disco a criar:

Tipos de *Disc Image*	Extensão do ficheiro
CD-i	.cdi
CD-DA	.red
CD-ROM XA	.xa

Tabela 3.1 - Extensões dos ficheiros gerados pelo Master

O **Green** implementa um subconjunto de funções do Master, para construir ficheiro de tempo-real. A sintaxe do script para esta aplicação é totalmente compatível com a sintaxe dos scripts do Master; um script para o Green pode ser incluído num script para o Master sem quaisquer modificações.

A saída do Green é um ficheiro de tempo-real que pode ser uma fonte de entrada para o Master. Existe contudo um compromisso entre o espaço de disco ocupado pelos ficheiros de tempo-real e o tempo de processamento desses mesmos ficheiros. O pré-processamento de ficheiros de tempo-real duplica a informação mas acelera o tempo de construção do disco, já que a inclusão de um ficheiro de tempo-real no disco envolve apenas uma cópia.

3.5 CODIFICAÇÃO DE IMAGENS NO SISTEMA CD-I

As imagens no sistema CD-i classificam-se em duas categorias principais:

- A **primeira** inclui uma vasta gama de material visual, desde sequências animadas a imagens naturais paradas, texto e gráficos;
- A **segunda** é *motion video*. As imagens naturais de vídeo em movimento possuem o seu subsistema próprio no leitor de CD-i e são codificadas de forma diferente de todos os outros tipos de materiais vídeo.

Quando se fala genericamente de **vídeo** no domínio do CD-i, pode-se estar a referir aos vários tipos de imagens:

- imagens paradas que resultam de uma digitalização, tais como *slides*;
- *frames* (imagens paradas) capturadas com uma câmara de vídeo;
- pequenas janelas de vídeo (*partial-motion video*), ocupando parte do ecrã;
- segmentos de vídeo a 25 imagens por segundo (*full-motion video*), ocupando todo o ecrã.

3.5.1 CARACTERÍSTICAS TÉCNICAS DAS IMAGENS EM CD-I

Uma das características particulares do CD-i é a variedade de opções que proporciona para a codificação de imagens paradas e em movimento. Um disco CD-i pode conter as seguintes quantidades de dados:

- até 6 340 imagens fotográficas (*slides*, transparências coloridas ou *frames* de vídeo paradas), combinando até 16 777 216 cores;
- até 6 340 gráficos com 32 768 cores em qualquer ecrã;
- animação, isto é, desenhos com movimento limitado e com 128 cores.

Estas imagens podem ser apresentadas numa área ocupando a totalidade do ecrã ou confinadas a janelas, com uma imagem parada maior preenchendo o resto do ecrã e podendo utilizar qualquer número de cores. Antes do aparecimento do vídeo a 25 fps (*Full-Motion)* em ecrã completo (*Full-Screen)*, o movimento das imagens era restringido por um certo número de limitações, mas podia ser apresentado numa área cobrindo 1/8 do ecrã com movimento parcial (*partial motion*). Todos estes factores, tamanho, cor, detalhe e movimento, influenciam decisivamente a quantidade final de dados visuais que podem ser armazenados num único disco [**RIBEIRO93**].

As imagens CD-i podem ser apresentadas no ecrã em 3 níveis diferentes de resolução. Estas especificações são normalmente expressas em termos de pixels, os elementos de imagem individuais que constituem a mais pequena unidade de informação visual nos sistemas de ecrã digital. O *aspect ratio* (proporção entre altura e largura) de um pixel adapta-se ao *aspect ratio* do ecrã no qual ele é apresentado; por isso a forma como o pixel é efectivamente visualizado varia

ligeiramente entre ecrãs NTSC e PAL/SECAM e os diferentes níveis de resolução. Isto contribui para uma ligeira distorção que é particularmente perceptível em formas geométricas como círculos e triângulos.

A **resolução normal** é apropriada para a qualidade de imagem de um receptor de televisão bem sintonizado. A **resolução dupla** tem o dobro de pixels por linha em relação à resolução normal, e o mesmo número de linhas por ecrã. A resolução dupla é equivalente à qualidade de imagem de um computador com placa gráfica VGA. A **alta resolução** tem o dobro de pixels na horizontal e na vertical relativamente à resolução normal, isto é, o dobro de pixels por linha e o dobro de linhas por ecrã. A alta resolução é equivalente à melhor qualidade de vídeo digital gerado num estúdio profissional.

No mesmo bloco de dados, a resolução normal utilizaria apenas 4 pixels para descrever aquela porção de imagem, a dupla resolução utilizaria 8 pixels e a alta resolução utilizaria 16. Cada pixel representa tipicamente um byte de dados. Por isso, quanto mais pixels existirem mais espaço e poder computacional é necessário para armazenar e processar a imagem. Como facilmente se depreende, dividindo a imagem em 16 unidades discretas de informação visual, a alta resolução produz linha mais suaves, maior detalhe e uma utilização de cores mais subtil do que os outros níveis de resolução.

A **resolução normal** é suficiente para a maior parte do material audiovisual contido num título CD-i. A **resolução dupla** proporciona imagens mais nítidas que são apropriadas para texto detalhado e gráficos. A **alta resolução** é particularmente útil para gráficos muito detalhados.

As resoluções normal e dupla **não** são entrelaçadas e estão, por isso, livres dos problemas associados com televisão entrelaçada e sinais de vídeo. Por outro lado, as imagens em alta resolução possuem uma quantidade muito maior de dados para processar, são entrelaçadas e sujeitas ao efeito de *flicker*, particularmente em linhas muito finas ou entre áreas de contraste elevado.

O CD-i é independente das normas nacionais de difusão de televisão; no entanto a maior parte dos receptores de televisão e monitores de vídeo são concebidos de acordo com pelo menos uma das normas aplicadas à composição de imagens no ecrã. Isto significa que o número real de pixels apresentados ao utilizador depende da norma utilizada no receptor de televisão ou monitor de vídeo [**RIBEIRO93**].

O leitor CD-i pode ser ligado a um qualquer receptor de televisão, mas os vários modelos e marcas têm ligeiras diferenças no modo de interpretação do sinal de vídeo. Isto levanta um problema grave, pois não está garantida a visualização total do ecrã. Caso o receptor de televisão corte a orla da imagem numa zona crítica (ex: texto, botão de interactividade, ...) o funcionamento normal do título CD-i fica comprometido [**PHILIPS93**]. Para contornar este problema é definida uma zona do ecrã em que se garante a sua visualização em todos os receptores de TV e que se designa por **zona de visualização** (***safe area***).

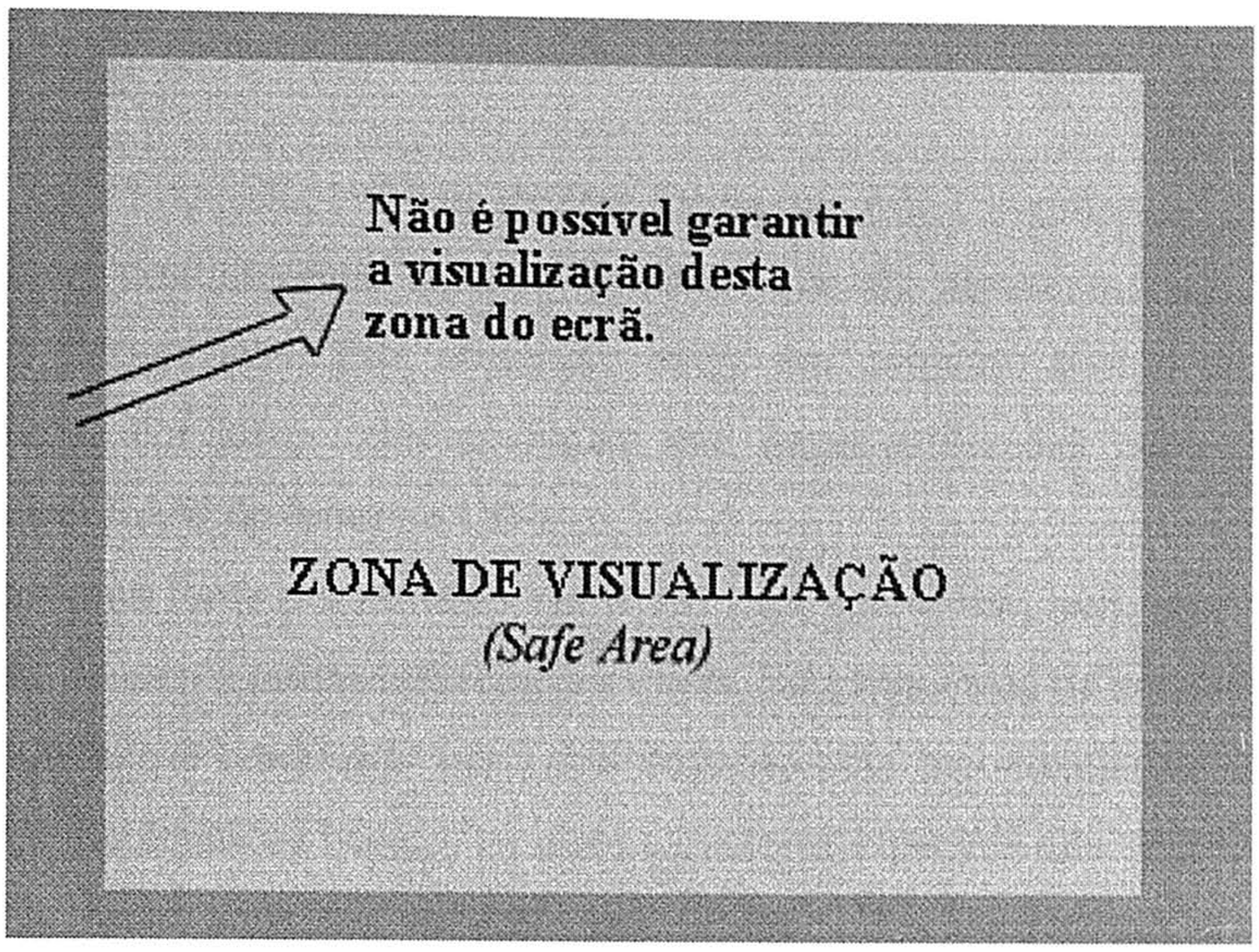


Figura 3.6 - Zona de Visualização

Resolução do ecrã (em pixels)

Normal: PAL - 384 x 280 (105 K) e 320 x 250 de ***safe area***

NTSC - 360 x 240 (84 K) e 320 x 210 de ***safe area***

Dupla: duplica o número de pixels por linha

PAL - 768 x 280 pixels

NTSC - 720 x 240 pixels

Alta: duplica o número de pixels por linha e coluna

PAL - 768 x 560 pixels

NTSC - 720 x 480 pixels

O sistema NTSC apresenta 525 linhas e o PAL/SECAM apresenta 625 linhas; na prática traduz-se numa diferença de 40 linhas para uma imagem CD-i em resolução normal. Para contornar este problema, a equipa de tratamento de imagens adiciona um bordo ao ecrã se usar a norma NTSC em monitores PAL ou simplesmente inicia a apresentação da imagem na vigésima linha do ecrã PAL, o que deixa 20 linhas em branco no fundo da imagem. Todo o material visual deve ser confinado à zona de visualização que é livre de qualquer distorção e estabelecida para ser visualizada em todos os ecrãs, seja qual for a sua qualidade.

A quantidade de dados necessária para armazenar uma imagem individual é calculada com base no tamanho do ecrã. Uma imagem que preencha o ecrã em resolução normal ocupa 92 160 bytes de dados em NTSC (384x240) e 107 520 bytes em PAL/SECAM (384x280), de que resulta em 90 KB por imagem em NTSC e 105 KB por imagem em PAL/SECAM. Quando a resolução é mais elevada é necessária uma maior quantidade de dados e o número de bytes necessários para armazenar a imagem aumenta proporcionalmente.

3.5.1.1 TEXTO

O sistema CD-i pode manusear texto da mesma forma que este é manuseado pelo computador, isto é, como caracteres contidos num ficheiro de dados que são apresentados no ecrã com uma fonte, cor e efeitos tipográficos. Esta utilização exige um conjunto básico de comandos que descrevem cada fonte. Por isso, todas as fontes que forem utilizadas pelo título devem ser carregadas na memória RAM do leitor de CD-i.

O sistema contém uma fonte por omissão em ROM para apresentar, num terminal que lhe seja acoplado, mensagens de erro e outras informações vitais. Esta fonte está de acordo com o conjunto de caracteres normalizado e conhecido por *Latin* definido pela recomendação ISO 8859-1. Esta fonte inclui símbolos matemáticos básicos e outros caracteres especiais que aparecem num teclado de computador. Esta fonte pode ser utilizada por títulos CD-i, no entanto o aspecto estético proporcionado por outras fontes ou conjuntos de caracteres pode ser preferido na concepção do texto.

São necessárias fontes especiais para linguagens que utilizam outros caracteres, ou aplicações que exigem símbolos especiais não contidos na fonte definida por omissão. O leitor de CD-i pode apresentar todos estes caracteres como fontes *bit-map* ou imagens, tal como num sistema de *display* de alta qualidade baseado em computador.

3.5.1.2 MÉTODOS DE CODIFICAÇÃO DE IMAGENS PARADAS

O CD-i proporciona um leque de técnicas [HOFFOS92] apropriadas para a redução da quantidade de dados armazenada e processada com cada imagem, permitindo optar pelos formatos de codificação apropriados às variadas exigências de fotografias, *slides* e *frames* de vídeo paradas, texto e gráficos de computador, animação e imagens obtidas a partir de trabalhos de arte e fontes de arquivo.

Os dados visuais sofrem um tratamento que se pode resumir nas seguintes fases:

- **Captura de imagens naturais e gráficos**; uma imagem natural pode ser uma fotografia, um *slide*, uma *frame* única de um filme. Este material deve ser obtido no formato RGB através de um *scanner* ou através de uma câmara de vídeo. No formato RGB, cada componente de cor é codificado com 8 bits e as imagens são capturadas com alta resolução;
- **Conversão da imagem RGB para os formatos CD-i**; pode ser utilizado por exemplo o Adobe Photoshop com os *"CD-i Plug-Ins"* que convertem directamente as imagens analógicas para o formato CD-i pretendido, mediante uma codificação digital da imagem [STERN93]. A maior parte destes formatos comprime a quantidade de dados necessária para guardar uma imagem ou uma sequência de imagens;
- **Codificação para o formato IFF**; todo o material visual é finalmente integrado na imagem final do disco para posteriormente ser emulado.

O CD-i oferece **4 técnicas ou formatos de codificação e compressão básicos** que se aplicam a imagens de vários tipos. Estas técnicas proporcionam diferentes níveis de qualidade, permitindo a escolha de soluções criativas e de baixo custo.

O cálculo do espaço de cor **RGB** divide cada pixel nos componentes Vermelho (**R**), Verde (**G**) e Azul (**B**) de cada cor. Os sistemas de imagem que empregam o espaço de cor **YUV** exploram a diferença entre brilho e cor dividindo o pixel entre **luminância (Y)** e **cor (U** e **V)**. A componente **Y** é na realidade a imagem a preto-e-branco, as componentes **U** e **V** adicionam o sinal de cor. Isto significa que a luminância e a crominância (cor) podem ser processadas separadamente: a tecnologia DVI, por exemplo, comprime mais o sinal de cor (ao qual o espectador é menos sensível) do que o sinal de luminância.

A manipulação da cor é umas das formas principais através das quais os 4 sistemas básicos de codificação utilizados no CD-i e as suas variações conseguem uma economia do espaço de armazenamento. Os 16 milhões de cores disponíveis no final do processo representam realmente as 16 777 216 combinações possíveis de 24 bits (2^{24}) ou 256 tonalidades em cada uma das *palettes* RGB. Na prática, a primeira geração de leitores CD-i apenas suportava 6 bits para cada nível RGB, o que representa 262 144 cores (2^{18}), ou seja, 64 tonalidades diferentes em cada nível RGB (2^{6}).

Pode-se utilizar a técnica de codificação mais apropriada para cada recurso visual, minimizando o espaço ocupado, dividindo a área total do ecrã em bandas horizontais denominadas sub-ecrãs, onde para cada banda se pode empregar uma técnica de compressão diferente.

Um procedimento para a aquisição de imagens e vídeo é converter este material a partir de outros ambientes para o ambiente CD-i. A conversão de dados de um formato de ficheiros para outro é uma característica muito comum nos ambientes de computação actuais. Como tal, existem produtos de software desenvolvidos pela Philips e pela *OptImage* que fornecem serviços semelhantes a todos aqueles que desenvolvem os seus projectos CD-i. Os respectivos pacotes de software encontram-se disponíveis para plataformas PC, Mac e SUN [**RIBEIRO93**].

No apêndice A descrevem-se os 4 formatos básicos de codificação a partir das características particulares que apresentam em relação aos seguintes aspectos, utilizando a letra correspondente à classificação a seguir especificada:

A. Modo como efectuam a **codificação** de imagem (dado que os vários tipos de imagens gráficas dentro do sistema CD-i são codificadas de forma diferente, mesmo entre os vários níveis de qualidade dentro do mesmo formato);

B. Modo como lidam com a **cor**;

C. Adequação aos 3 níveis de **resolução;**

D. Espaço de armazenamento que ocupam (de modo a proporcionar a escolha ponderada entre: mais imagens em disco ou menos imagens com melhor qualidade);

E. Material visual a que melhor se aplicam.

3.5.2 EFEITOS COM IMAGENS EM CD-I

A imagem é um elemento muito importante num sistema multimédia, principalmente se for possível realizar algumas operações sobre o seu modo de apresentação. Os efeitos com imagens permitem cortar a *mono*tonia de uma apresentação com imagens umas a seguir às outras (ex: *slideshow*). O sistema CD-i implementa os efeitos com imagens usando diferentes **planos de imagem**. Este conceito está relacionado com a ideia de uma pilha de *slides*, sendo no entanto a relação entre eles dinâmica. A transição entre um plano e o seguinte pode ser efectuada de várias formas: *cuts*, *wipes fades*, etc (secção 3.5.2.2). Adicionalmente, podem-se visualizar partes de três planos ao mesmo tempo utilizando *mattes* e transparências. Os planos podem ser utilizados para conter imagens paradas, imagens em movimento e botões de interactividade. A posição relativa dos planos de imagem pode ser alterada, isto é, a sua ordem na pilha de apresentação no ecrã, permitindo controlar a transição entre imagens.

3.5.2.1 TIPOS DE PLANOS

Uma das características específicas do CD-i que lhe conferem a versatilidade na apresentação de gráficos e imagens é a utilização de planos nos quais se apresentam o cursor, texto, gráficos e imagens vídeo contidas num título CD-i.

Existem **4 tipos de planos** de imagens destinados à manipulação das mesmas que estão colocados numa pilha (na perspectiva de visualização no ecrã):

- **Plano de vídeo -** É utilizado para vídeo *full motion* em *full screen*, para conter cores de fundo (escolhidas numa gama de 16 cores onde a cor por omissão é azul) ou para apresentar vídeo proveniente de fontes analógicas externas, isto é, não proveniente do disco CD-i ou da memória.
- **Planos A e B -** São os dois planos centrais de 8 bits destinados aos outros tipos de imagens. Estes planos podem ser codificados separadamente de modo a proporcionar duas camadas de gráficos ou texto e gráficos utilizando codificação DYUV, CLUT ou RLE. Estes planos podem também ser fundidos e codificados como uma única imagem RGB5:5:5, a qual necessita do dobro da quantidade de dados em relação aos outros métodos de codificação. Cada plano contém 512 KB de dados e fornecem várias técnicas de manipulação de imagens, incluindo efeitos de *partial-motion*.
- **Plano frontal** - É o plano do cursor. É um plano invisível através do qual se dá a movimentação do cursor que pode ter qualquer forma dentro de um bloco de pixels de dimensão 16x16. O cursor está restringido a uma de 16 cores possíveis.

Podem-se manipular estes planos de qualquer forma de modo a produzir os efeitos desejados. A escolha da ordem pela qual os planos são manipulados (A em frente de B ou B em frente de A) é ajustada pela parte do programa que controla o ecrã.

Os planos referidos podem ser utilizados de várias formas para criar um conjunto de efeitos que melhoram a apresentação e abrem um espaço muito vasto à criatividade:

- Sobreposição de imagens de forma a criar uma combinação de imagens e a mover um objecto para cima de outro;
- Criação de efeitos visuais tais como *wipe*, *dissolve* e *fade*;
- Realçar com *highlight* certas zonas de menus;
- Carregar e esconder rótulos (*labels*);
- Colocar uma janela contendo uma imagem DYUV sobre uma imagem CLUT, conseguindo uma mistura simultânea no ecrã de imagens DYUV e CLUT;
- Conceber imagens compostas com vários formatos.

É importante referir que cada linha de cada ecrã pode ser codificada individualmente de modo a suportar uma variedade de gráficos e efeitos simultaneamente. Este processo é conseguido por intermédio do DCP.

A versatilidade conferida ao sistema de visualização pela existência de planos de imagem é aumentada pela utilização de **sub-ecrãs** em cada plano. Estes sub-ecrãs são faixas horizontais que podem conter informação adicional à contida no resto do plano, podendo ser codificados separadamente. O número de sub-ecrãs, o seu tamanho e localização são parâmetros controlados inteiramente pelo programador. Por isso, podem existir tantos sub-ecrãs quantas as linhas do ecrã.

Um sub-ecrã pode ser utilizado para apresentar texto opcional tal como legendas. Os sub-ecrãs podem também ser utilizados para proporcionar *partial-screen updates* de forma eficiente nas sequências que exigem movimento e variedade de imagens. Os sub-ecrãs podem ainda ser utilizados para combinar técnicas de codificação diferentes, por exemplo para apresentar texto e gráficos em dupla resolução (CLUT4 ou RLE3) num ecrã em resolução normal. Esta possibilidade é útil para melhorar o aspecto de conjuntos de caracteres complexos ou gráficos que tiram partido da suavidade proporcionada pela dupla resolução.

A utilização de sub-ecrãs e *partial-screen updates* pode reduzir o tempo de processamento porque disponibiliza nova informação apenas quando esta é estritamente necessária e refrescando cada parte do ecrã à taxa apropriada.

3.5.2.2 Criação de Efeitos Especiais em CD-i

A perspectiva de controlar e produzir efeitos especiais numa *workstation* é uma das características do ambiente de vídeo digital. Na produção convencional de imagens e vídeo, os efeitos especiais são concebidos e gravados recorrendo a um estúdio profissional de edição gráfica. Na plataforma CD-i (e outras plataformas digitais), os efeitos especiais tornam-se tão simples de produzir sob o controlo do computador como qualquer outra funcionalidade.

O CD-i utiliza os seus planos de imagem para criar efeitos visuais no ecrã de apresentação. Assim, existe uma distinção técnica entre efeitos realizados num plano e efeitos que necessitam de ambos os planos.

Efeitos realizados num plano:

- ***Cuts*** - Este efeito coloca rapidamente uma imagem no ecrã. Pode-se dar a sensação de movimento *full-motion* se os *Cuts* entre *frames* paradas forem suficientemente rápidos, se possível a uma taxa entre 25 e 30 *frames* por segundo;
- ***Partial updates*** - Este efeito é análogo ao *Cuts* mas só envolve uma parte do ecrã, isto é, só se actualiza um sub-ecrã;
- ***Panning e Scrolling*** - Envolve o movimento de uma área do ecrã sobre uma imagem maior quer na direcção vertical quer na horizontal;
- ***Mosaic effects*** - Estes efeitos envolvem a manipulação directa de pixels da imagem de modo a fazer com que esta se desintegre em blocos ou de modo a aumentar o seu tamanho;
- ***Fade up*** **ou** ***Fade down*** - Aumenta ou diminui o brilho de uma imagem entre preto e intensidade máxima, percorrendo 64 níveis.

Efeitos que envolvem ambos os planos (planos A e B):

- ***Cuts*** - O DCP pode alterar rapidamente a ordem pela qual os planos aparecem, de forma a que um plano substitua o outro, preenchendo o ecrã com outra imagem;
- ***Fade*** - A imagem do plano B surge à medida que a imagem do plano A desaparece;
- ***Wipe*** - A imagem do plano (A ou B) que está na frente é progressivamente substituída pela imagem do outro plano, seguindo uma linha que atravessa o ecrã a partir de qualquer dos lados em direcção ao outro. Existem três espécies de *wipes*:
  1. ***curtain***: um *wipe* que abre a visualização da imagem no plano da frente, dando a ideia de um par de cortinas que se abrem, para revelar a imagem que está no plano de trás.
  2. ***blind***: um *wipe* no qual o plano da frente abre para revelar progressivamente a imagem do plano traseiro.
  3. ***square***: uma secção quadrada, que aumenta progressivamente de tamanho, aparece em qualquer lugar do plano da frente e mostra a imagem do plano de trás abrindo o quadrado simultaneamente em todas as direcções horizontais e verticais.
- **Transparências** - As áreas transparentes permitem que o plano de trás seja visualizado através do plano da frente, geralmente abrindo janelas no plano da frente, através das quais se pode ver uma imagem do plano de trás (que pode por exemplo estar em movimento). Existem dois tipos de transparência disponíveis:
  - *Chroma key*: Este é um método que permite fazer com que uma ou várias cores seleccionadas num plano se tornem transparentes. Pode ser utilizada para isolar

uma imagem no plano da frente de forma a que possa aparecer contra qualquer imagem que exista no plano traseiro.

- *Mattes*: Uma *matte* é uma área de qualquer forma na qual os pixels da imagem são tornados transparentes. As *mattes* utilizam-se para obter efeitos do tipo *dissolve* no qual a imagem do plano da frente se dissolve uniformemente na imagem do plano de trás. As *mattes* também são utilizadas para fazer com que uma janela do plano da frente mostre uma imagem do plano de trás.

3.5.3 SISTEMA FMV (*FULL-MOTION VIDEO*) EM CD-I

O *Full-Motion Video* é um sistema com capacidade de mostrar sequências vídeo com uma qualidade idêntica à da televisão (30 imagens por segundo) sendo definido como uma extensão do CD-i que possui capacidades para mostrar imagens naturais em movimento e em *full-screen* (ecrã completo). O áudio associado possui qualidade *Compact Disc*. Para mostrar sequências *full-motion* a partir de um disco CD-i é necessário comprimir a informação audiovisual para o *bitrate* CD-i. O sistema **FMV** *(Full-Motion Video)* aplica um método de compressão baseado na norma **MPEG (*Motion Picture Experts Group*)**.

O FMV introduz no CD-i a capacidade de mostrar 72 minutos de imagens naturais em movimento e em *full-screen*, com áudio de qualidade CD. Cada disco CD-i FMV pode ser lido em leitores de 50 ou 60 Hz. Um disco CD-i FMV foi concebido para alocar 1.2 Mbps para vídeo em *full-screen,* e 0.2 Mbps para áudio *stereo* de qualidade CD. Deste modo, o ***bitrate*** total do fluxo de dados proveniente do disco iguala 1.4 Mbps. Para comprimir a informação audiovisual para estes *bitrates* e para armazenar os dados (de áudio e vídeo) codificados obtidos após a compressão em discos CD-i é aplicada a norma MPEG. A norma MPEG é definida pela ISO/IEC como uma norma genérica, isto é independente das aplicações.

O descodificador FMV contém um banco de **memória** de pelo menos 4 MB para manipular a descodificação de vídeo MPEG e reordenação bem como para a conversão de frequências (50 Hz ou 60 Hz). Quando esta memória não é utilizada para a descodificação de vídeo *Full-Motion*, pode ser utilizada para aumentar a quantidade de memória disponível no sistema para as aplicações. Nesse caso, o descodificador funciona como um controlador de memória, de tal modo que o acesso à memória se torne transparente para a aplicação.

3.5.3.1 APLICAÇÃO DA NORMA MPEG AO FMV

A norma **MPEG** é uma definição genérica para a codificação audiovisual, isto é, é independente de qualquer aplicação e permite o desenvolvimento de muitas aplicações que necessitam de vídeo e áudio digital comprimido. Os meios de armazenamento visados pela norma MPEG são: CD, DAT e discos magnéticos. As tecnologias baseadas em MPEG irão ser utilizadas

numa grande variedade de canais de comunicação como RDIS, redes (LAN's) e mesmo em aplicações de difusão.

A norma MPEG especifica uma ***stream*** MPEG como um formato genérico de troca de informação. Para armazenar uma *stream* MPEG num dado meio, é necessária uma **camada específica do meio** (ver figura 3.7). No caso do FMV, o meio é um disco CD-i ou a memória e a camada específica do meio é o formato do disco.

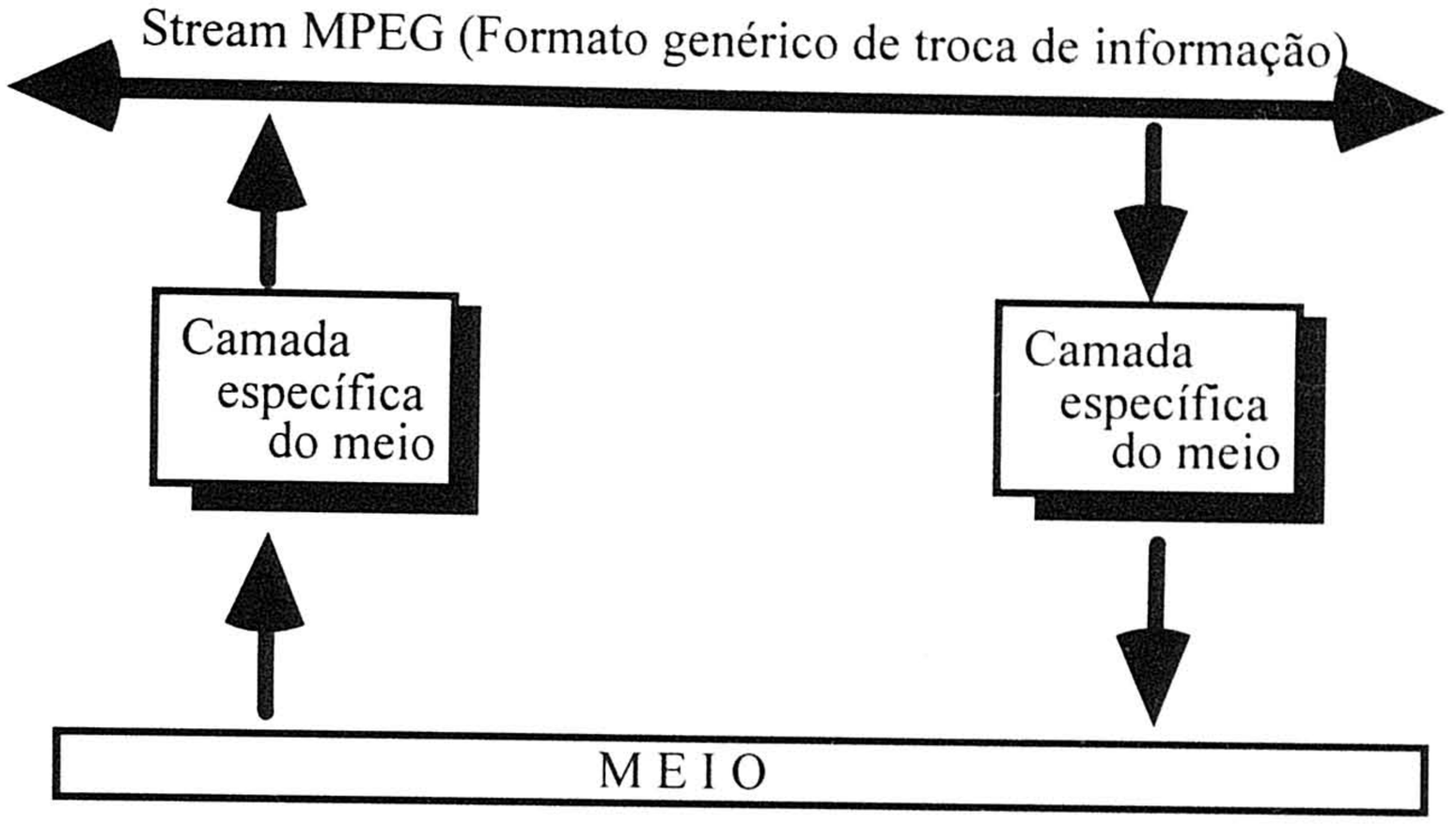


Figura 3.7 - Armazenamento de uma *stream* MPEG num dado meio

A norma MPEG constituída por três partes:

- Na **primeira parte** especifica-se a **camada de sistema** da norma, incluindo a estrutura multiplexada que permite combinar áudio MPEG, vídeo MPEG e informação de tempo necessária à sincronização da leitura;
- Na **segunda parte** especifica-se a **camada de codificação de vídeo**;
- Na **terceira parte** especifica-se a **camada de codificação de áudio**.

Na figura 3.8 apresenta-se um sistema de codificação MPEG genérico em que as fontes de áudio e vídeo são entradas para os codificadores de áudio e vídeo respectivamente.

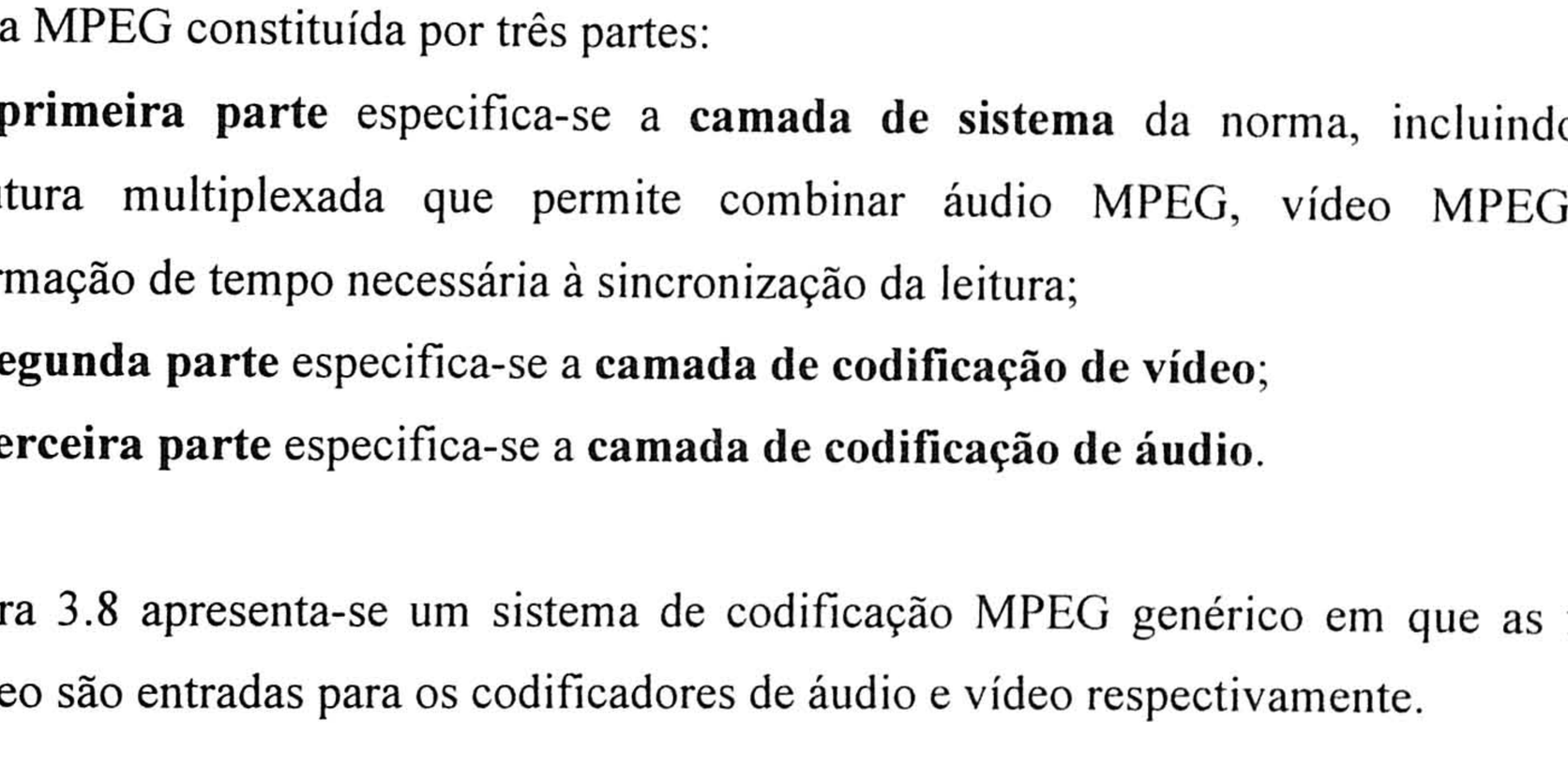


Figura 3.8 - Sistema genérico de codificação MPEG

As *streams* MPEG áudio e vídeo geradas pelos codificadores são multiplexadas numa única *stream* MPEG. Na mesma *stream* MPEG podem ser multiplexadas 16 *streams* de vídeo MPEG e 32 *streams* de áudio MPEG. Existe também um relógio do sistema (90 KHz) que é utilizado para sincronização.

Na norma MPEG são definidos alguns parâmetros tais como:

- a taxa de imagens (*picture rate*);
- tamanho das imagens;
- frequência de amostragem de áudio;
- a *bitrate* (taxa de transferência de dados MPEG).

Estes parâmetros possuem uma gama de variação muito vasta de modo a adaptá-los a uma grande variedade de aplicações. Para definir a gama de variação dos seus parâmetros, uma aplicação especifica o seu ***profile* MPEG**. Este *profile* também define as restrições específicas da aplicação na sintaxe e semântica MPEG. O FMV adoptou o ***profile*** conhecido por **CSPS** (*Constrained System Parameter Stream*), tal como está definido pela norma MPEG para a camada do sistema.

3.5.3.2 CODIFICAÇÃO FMV (*FULL-MOTION VIDEO*)

Quando o CD-i foi lançado não suportava vídeo *Full-Motion* em *Full-Screen*. No entanto estavam disponíveis duas técnicas que proporcionaram uma solução temporária:

- Refrescamento parcial do ecrã (*Partial-Screen Updates*);
- Janelas de vídeo (*video windows*).

A escolha do sistema que permitisse a apresentação de segmentos de vídeo digital esteve restringida às capacidades de processamento do sistema CD-i: o vídeo exige um número elevado de imagens de alta qualidade, de forma a simular o movimento com realismo fotográfico, mesmo à velocidade de 15 fps. Por outro lado, a taxa de transferência de dados do CD-i limita-se a 75 sectores por segundo (172 Kbps). Se a totalidade desta largura de banda fosse dedicada a imagens de vídeo digitais a uma taxa de 15 fps, cada imagem (*frame*) teria que estar contida num bloco de apenas 11.3 KB (170/15) - não deixando espaço para o áudio, texto ou dados relacionados com o programa de aplicação.

Como o vídeo é uma sequência de imagens, a codificação DYUV é a mais apropriada para imagens em movimento que simulam vídeo real. Uma imagem DYUV típica exige 90 KB em NTSC e 105 KB em PAL/SECAM. Assim, a uma taxa de transferência de 172 KB por segundo, só seria possível entregar 2 imagens por segundo, o que é claramente pouco para simular vídeo real e muito rápido para apresentações de *slides* (*slideshow*).

Dado que uma imagem *Full-Screen* não pode ser refrescada com a frequência necessária, a solução a adoptar é **reduzir o tamanho da janela** na qual o vídeo aparece. Se um bloco de 11.3 KB é o máximo permitido por *frame* de imagem a uma taxa de transferência de 172 Kbps, isto representa cerca de 12.5 % dos 90 KB exigidos por uma imagem DYUV no sistema NTSC e perto de 11 % dos 105 KB necessários no sistema PAL/SECAM. Por conseguinte, a imagem poderia preencher uma janela que ocupasse essas percentagens do ecrã.

Se a *frame rate* é reduzida para 12.5 fps, mantendo o mesmo volume de dados, o tamanho da janela pode ser aumentado para 13 % em PAL/SECAM ou para 15 % em NTSC. Em alternativa, a utilização de janelas mais pequenas deixa espaço no canal de dados para áudio, texto, gráficos e outros elementos do programa. Um algoritmo de compressão relativamente simples pode aumentar o tamanho da janela para 16 % em PAL/SECAM e 19 % em NTSC. Em alternativa, a interpolação vertical pode aumentar o tamanho da janela para 1/3 do ecrã total (*Full-Screen*). Estas restrições podem ser adaptadas de forma a criar dois efeitos:

- **Pequenas janelas de visualização de segmentos vídeo**, com imagens reduzidas proporcionalmente, para apresentar *partial-motion video* dentro de zonas com pelo menos 1/8 (13 %) do tamanho total do ecrã;
- **Vídeo em *Full-Screen* com refrescamento parcial do ecrã**, apresentado dentro de "janelas invisíveis" que se movem pelo ecrã de modo a refrescar as diferentes partes da imagem, uma de cada vez e segundo uma dada ordem.

O leitor de CD-i possui um *chip* especial para processar vídeo e uma memória de 0.5 MB para guardar e processar imagens em movimento, além de 1 MB de memória reservado para os outros tipos de vídeo. Como este tipo de imagens ocupam muito espaço em disco, existem métodos de compressão de *motion video* que poupam espaço em disco e da memória do leitor. Para esse efeito e como vimos, pode-se utilizar ***partial-screen video*** que faz o movimento numa janela que utiliza apenas uma fracção do ecrã, e pode utilizar 1 MB da memória do leitor que é normalmente utilizada para outros tipos de imagens, em vez de utilizar o *chip* de *motion video* e a respectiva memória. Pode-se também utilizar ***partial-motion video*** que utiliza uma velocidade de 20 fps e é adequada para demonstrações que correm a baixa velocidade. Esta técnica também não necessita do *chip* especial de *motion video* nem da respectiva memória especial para *motion video*, podendo ser utilizada a memória normal de 1 MB do leitor CD-i.

3.5.3.3 Áudio e Vídeo no Formato MPEG

Para codificar áudio, o FMV aplica as camadas I e II da parte de áudio da norma MPEG. O áudio pode ser codificado em ***stereo***, em ***joint stereo*** (isto é, *intensity stereo*), em **canais duais** ou em **canais simples**. No caso do ***joint stereo***, explora-se a redundância existente entre os canais *stereo* para melhorar a eficiência da codificação. A frequência de amostragem de áudio aplicada é

de 44.1 KHz. Para que a integração das funções de descodificação de áudio MPEG com as funções de descodificação de base seja simples, não se permitem outras frequências no FMV.

O sistema FMV suporta os **parâmetros restritos para vídeo** definidos na parte de vídeo da norma MPEG. No entanto, alguns parâmetros possuem uma gama de variação maior no FMV. Assim, é permitida uma taxa máxima para o vídeo de 5 Mbps, que excede o limite de 1.856 Mbps definido na norma MPEG para os parâmetros restritos de vídeo. Para além disso, e para as imagens paradas; entrelaçadas e codificadas, o tamanho máximo da imagem excede o limite máximo para os parâmetros restritos de vídeo. Finalmente, no FMV é permitida a codificação de vídeo com uma taxa variável.

Uma *stream* de vídeo MPEG pode resultar da concatenação de sequências de vídeo com parâmetros diferentes. Por exemplo, a frequência (taxa) de imagens ou o tamanho das imagens pode variar. O FMV não aplica restrições no *aspect ratio* dos pixels. As aplicações devem levar em conta que os descodificadores FMV não compensam a distorção do *aspect ratio*. Por isso, são recomendados três valores opcionais para utilização prática: um valor óptimo para monitores de 60 Hz, um valor óptimo para monitores de 50 Hz e um valor intermédio destinado a ambos os monitores.

3.5.3.4 Sincronização com a Leitura (*Playback*)

A sincronização entre áudio e vídeo MPEG é assegurada implicitamente pela norma MPEG. O descodificador regenera o relógio do sistema e apresenta imagens descodificadas e tramas de áudio com base nos *Time Stamps*. No FMV, os descodificadores de áudio e vídeo MPEG possuem o seu próprio relógio de sistema, mas ambos os relógios são fixados com o valor de 75 Hz (a taxa de entrega) e o seu ***offset*** mútuo permanece constante durante uma leitura. A sincronização entre áudio e vídeo é assegurada pela aplicação, que só avança quando a diferença entre os dois relógios do sistema seja igual ao *offset* mútuo.

A norma MPEG assume que a descodificação é realizada em tempo-real e não contempla atrasos de processamento mas, na prática, ocorrem atrasos de processamento nos descodificadores de áudio e vídeo MPEG.

A sincronização entre a leitura *Full-Motion* e leituras de áudio ADPCM é da inteira responsabilidade da aplicação. Para compensar os atrasos de processamento de dados de áudio e vídeo MPEG, as aplicações devem atrasar a leitura do áudio ADPCM. Para possibilitar a sincronização entre acontecimentos da aplicação e acontecimentos *Full-Motion*, o descodificador FMV informa a aplicação, a pedido desta, sobre acontecimentos de FMV. Por exemplo, pode informar a aplicação sobre o início da próxima imagem ou o início da próxima trama de áudio.

3.6 ÁUDIO NO SISTEMA CD-I

A qualidade máxima do áudio no CD-i é igual à normalizada para o CD-DA, mas estão disponíveis vários níveis de qualidade sonora, permitindo escolher o formato que melhor se adapta ao efeito sonoro que se pretende, controlando o espaço ocupado em disco e a largura de banda disponível. No CD-i existem basicamente dois mecanismos de reprodução áudio. O mais utilizado consiste na reprodução de um **ficheiro em tempo-real** que contém os sectores de áudio entrelaçados directamente no disco. O outro método utiliza ***soundmaps***, em que o áudio é guardado na memória e passado para o processador de áudio sob controlo do CPU. Durante as primeiras fases do projecto de títulos é necessário fazer uma análise cuidada dos requisitos do áudio, escolhendo qual o mecanismo que melhor se adapta para obter um determinado resultado, porque estas decisões têm consequências na sincronização, temporização e utilização da memória.

O sistema de áudio contém dois descodificadores distintos para leitura de dados a partir do disco, fazendo a conversão do áudio para a saída do leitor:

- O **descodificador CD-DA** permite a leitura de áudio *compact disc*;
- O **descodificador ADPCM** permite o tratamento do áudio como um sinal digital especificamente produzido para o CD-i. O material de áudio tem de ser convertido para o formato ADPCM, para ser aceite nas aplicações CD-i.

O sistema CD-i utiliza a técnica de compressão ADPCM para o som, permitindo reduzir o tamanho dos ficheiros de áudio relativamente ao ficheiros PCM (*Pulse Code Modulation*) característicos dos CDs de áudio (CD-DA). O CD-i tem dois descodificadores distintos para manter a compatibilidade com os CDs de áudio. A codificação ADPCM pode ser realizada com diferentes níveis de qualidade, permitindo uma escolha que satisfaça dois critérios contraditórios: utilizar uma boa (suficiente) qualidade de som sem aumentar o tamanho dos ficheiros. Ao som de boa qualidade **nível A**, correspondem os ficheiros maiores e ao som de menor qualidade **nível C** correspondem os ficheiros mais pequenos. Existe ainda um nível de qualidade média **nível B**.

Nível	**Amostragem**	**Canais**	**Utilização**	**Tempo**
PCM CD-DA	20 KHz	1 *stereo*	1/1	1.2 h
ADPCM Nível A	17 KHz	2 *stereo*	1/2	2.4 h
		4 *mono*	1/4	4.8 h
ADPCM Nível B	17 KHz	4 *stereo*	1/4	4.8 h
		8 *mono*	1/8	9.6 h
ADPCM Nível C	8.5 KHz	8 *stereo*	1/8	9.6 h
		16 *mono*	1/16	19.2 h

Tabela 3.2 - Qualidade do áudio em CD-i

O número de canais usados para o áudio influência a qualidade final do mesmo. Usando formatos de baixa qualidade para o áudio, pode-se misturar até 16 canais, permitindo fazer comentários paralelos em várias línguas. O som *stereo* utiliza aproximadamente o **dobro** dos canais do som *mono*. Quantos mais canais de som se utilizar, menos efeitos se pode usar ao mesmo tempo no disco.

O processo de digitalização de áudio está perfeitamente dominado: podem ser usados DAT (*Digital Audio Tape*), ou recorrer a sofisticados computadores e software para o controlo do processo de aquisição do som. O software Digidesign permite fazer uma emulação da gravação digital de áudio em várias pistas. Existe software (Microsounds da Microtechnology) que permite a edição e manipulação do áudio, de uma forma não destrutiva, criando efeitos sonoros do tipo *fade-in* ou *fade-out*.

A conversão de um sinal sonoro analógico para digital é tipicamente realizada usando a técnica **PCM**, em que a digitalização do som analógico é realizada em intervalos fixos de amostragem. Posteriormente é gerado um código para cada amostra, logo o espaço ocupado em disco é elevado. Esta técnica é usada no CD-DA. No CD-i é usado o ADPCM com uma frequência de amostragem de 44.1 KHz e com 16 bits para a amplitude de cada canal *stereo*. Cada amostra requer 32 bits (16 bits por cada canal).

Quando o áudio se encontra quer no formato analógico ou digital, é necessário utilizar conversores (disponíveis para o Macintosh e PC) para ficheiros PCM. A fase inicial consiste em guardar num disco duro de um computador os ficheiros áudio em PCM de todo o material necessário ao título. Os dispositivos de conversão para PCM são placas de aquisição de áudio com frequências de amostragem 44.1 KHz ou 48 KHz e com uma relação sinal-ruído (SNR) dentro do intervalo 85-90 dB com 16 bits ou 32 bits.

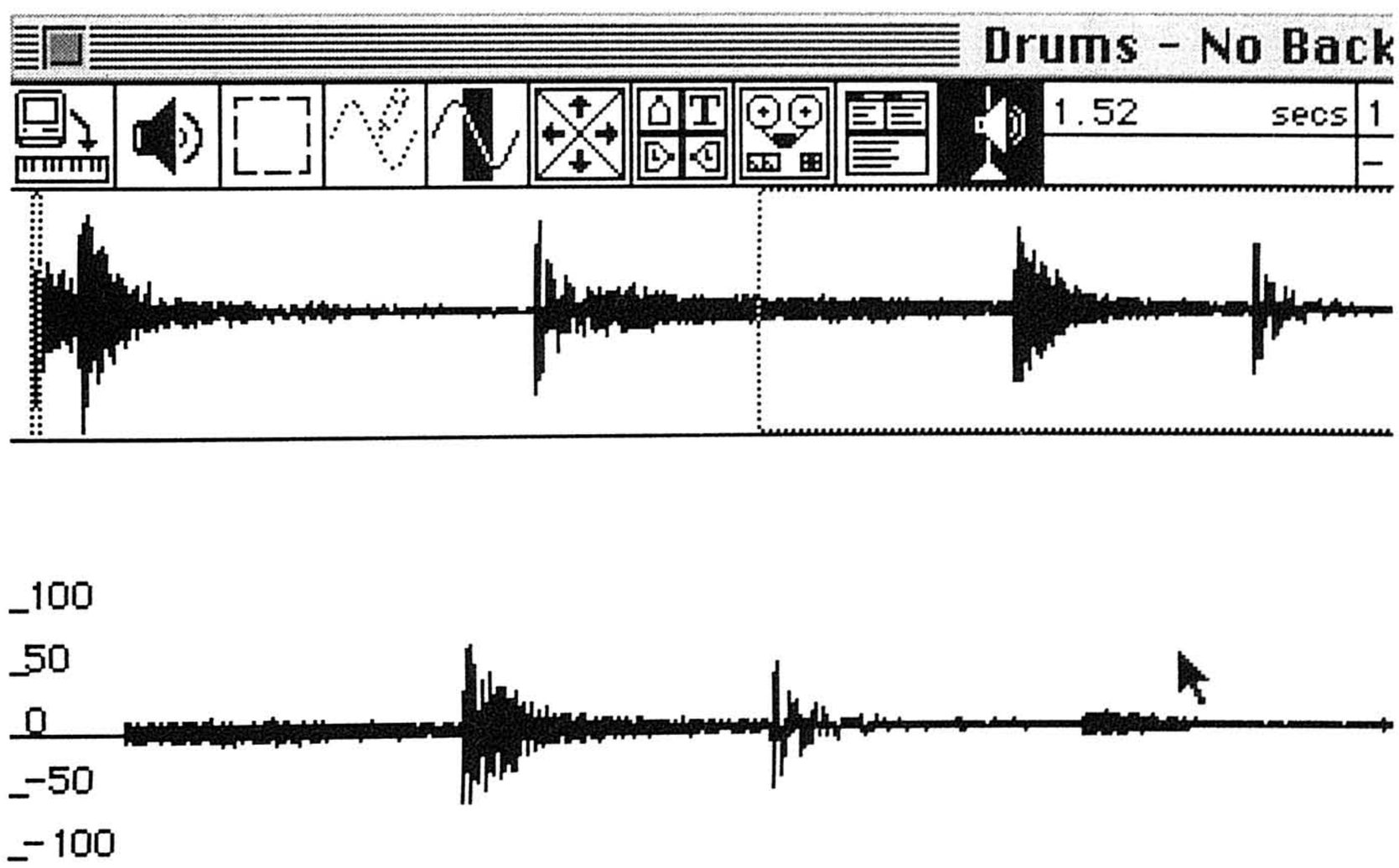


Figura 3.9 - Digitalização de áudio (*Digidesign*)

A codificação usando **ADPCM** pode ser decomposta em duas fases, correspondendo a primeira à digitalização e codificação PCM e a segunda à optimização do código gerado na primeira fase, usando algoritmos adaptativos às diferenças (*AD - Adaptive Delta*). O método ADPCM apenas codifica as diferenças (*Delta*) entre os valores das amostras em intervalos consecutivos e utiliza palavras de código mais pequenas (quando comparadas com o método PCM), realizando assim uma compressão eficiente para o som. Os métodos adaptativos usam algoritmos que incorporam memória sobre o passado do sinal, permitindo prever se a diferença para o próximo valor será grande ou pequena, e com esta informação optimizar o código [TAUB86].

A utilização de áudio ADPCM no sistema CD-i permite:

- Combinar diferentes normas para o áudio num único CD-i. Permitindo ter música clássica com grande qualidade e discursos (voz), numa outra pista, usando uma qualidade inferior;
- Correr várias peças de áudio em paralelo sobre uma apresentação em vídeo.
- Fazer a transição entre as diferentes pistas sonoras sem se notar os ruídos de transição.

A conversão do formato PCM para ADPCM é realizada usando ACUs (*Audio Conversion Utilities*) sobre as plataformas Macintosh ou PC. A frequência de amostragem depende do nível de qualidade escolhido.

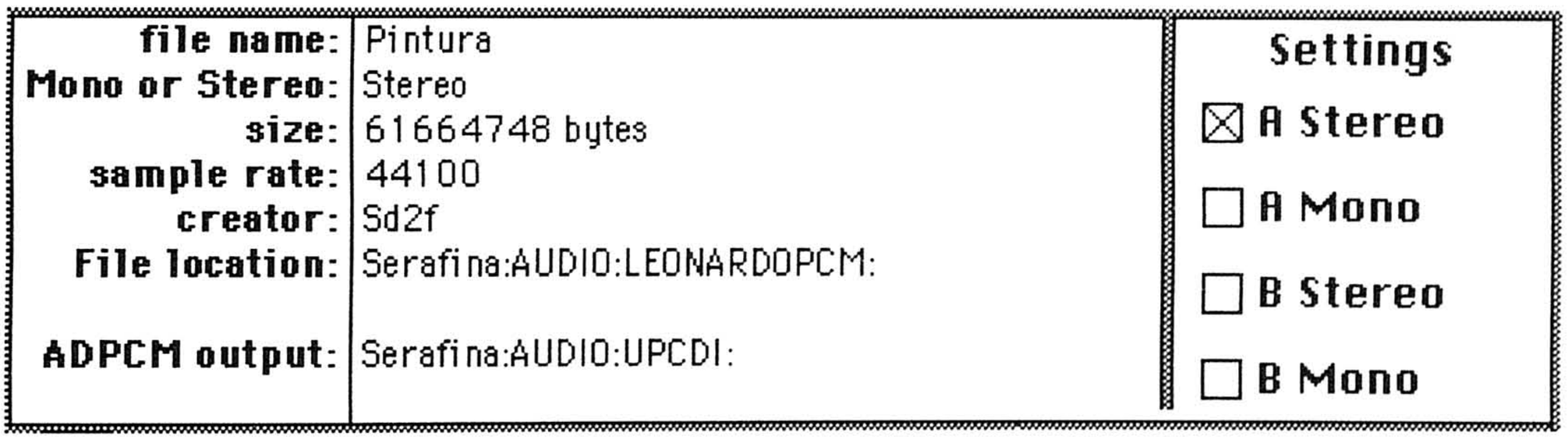


Figura 3.10 - Conversor de áudio PCM em ADPCM

O CD-i tem uma unidade de processamento de áudio que se encarrega de direccionar o áudio para os canais de saída esquerdo e direito. Esta unidade processa o áudio no formato PCM directamente a partir do disco ou áudio proveniente do descodificador ADPCM, sem consumir tempo de CPU.

A principal função do descodificador ADPCM é converter os dados de áudio dos sectores do disco CD-i para PCM com 16 bits em complemento para 2, convertendo assim uma palavra de 4 ou 8 bits numa palavra de 16 bits. O descodificador ADPCM elimina as flutuações do sinal recorrendo a filtros activos.

IV

PROGRAMAÇÃO NO SISTEMA CD-I

Em áreas como CBT (*Computer-Based Training*) e IV (*Interactive Video*) está disponível uma grande variedade de produtos de software para ajudar os programadores a trabalhar mais eficientemente. Permitem também que autores sem um treino especial criem programas interactivos. A programação de títulos CD-i está a começar a descobrir ferramentas que permitirão, pelo menos, facilitar as actividades mais básicas e até obviar a necessidade de uma programação formal. Em seguida descrevem-se algumas técnicas usadas para a programação no sistema CD-i.

4.1 PROGRAMAÇÃO DE APLICAÇÕES INTERACTIVAS

As apresentações em **multimédia interactiva**, que parecem fluir continuamente do início até ao fim, compreendem um número variado de módulos que podem ser ordenados em sequências muito diferentes entre si, resultando em apresentações coerentes. A chave para o desenvolvimento de uma aplicação interactiva é a programação modular e estruturada, que permite uma optimização da utilização do espaço de armazenamento no disco, bem como do tempo de processamento do leitor CD-i.

Um editor de jogos de computador, John Garcia da Novallogic, considera que o **código** de uma aplicação é apenas cerca de **0,3%** dos dados de um disco comum. Paradoxalmente, a concepção de um bom jogo de computador é conseguida à custa das mais avançadas metodologias da engenharia de software. As proporções podem ser diferentes noutras aplicações, mas a mensagem é a mesma: Qualquer que seja o conceito criativo, intelectual ou visual, uma apresentação multimédia interactiva de sucesso depende igualmente de uma boa programação. É essencial uma boa colaboração entre a equipa de concepção e os programadores para criar um programa interactivo que seja eficiente. O aspecto fulcral da interactividade de tempo-real é, que as acções dos utilizadores nunca podem ser totalmente previstas.

Sempre que possível, é vantajoso organizar os dados em grupos de pequenos ficheiros, que podem assim ser rapidamente recuperados e processados segundo várias combinações.

Podem ser efectuadas algumas pequenas **economias em tempo-real** (durante o decorrer da aplicação), por exemplo: usar a largura de banda disponível para carregar os ficheiros auxiliares, enquanto que os dados de tempo-real fluem para fora do disco, ou realizar o reposicionamento da cabeça de leitura, enquanto o sistema está à espera da resposta do utilizador. Também se podem utilizar estas pausas, enquanto se espera pela resposta do utilizador, para procurar e abrir ficheiros que provavelmente serão requisitados a seguir e até mesmo, pré-carregar dados que os utilizadores podem vir a necessitar em qualquer ponto de decisão.

No entanto, mesmo dentro de uma apresentação didáctica (na qual o estudante tem um número limitado de opções em cada ponto) **não é sempre prático pré-carregar todos os dados** para todas as potenciais opções. Os tempos de resposta podem ser retardados, particularmente se todos os *buffers* estiverem carregados à espera de uma resposta, e o utilizador escolher um caminho menos usual. Num disco que combine muitas facilidades o utilizador pode subitamente decidir interromper a sequência normal do título e, em vez disso, jogar um jogo. Dados sem qualquer relação intrínseca podem ser armazenados em qualquer parte do disco e então o tempo de resposta da aplicação depende principalmente de uma estrutura eficiente e de uma organização que permita carregar e processar novas informações rapidamente, onde quer que elas se encontrem.

O áudio e o vídeo podem não começar simultaneamente; a sobreposição do som de uma cena com a imagem de outra é uma técnica de televisão eficiente e familiar para a maioria dos observadores. No CD-i uma transição sem cortes de um segmento para outro pode ser efectuada

apresentando o som e as imagem à medida que cada um está pronto, de modo que o primeiro disponível cubra a demora na preparação do outro.

A **consistência** é, normalmente, mais importante do que a velocidade, permitindo assegurar que os utilizadores não são confundidos por diferenças erráticas nos **tempos de resposta**. Os utilizadores podem tolerar que demore alguns momentos mudar de uma sequência de imagens para uma outra, mas podem ficar surpreendidos ou fazer suposições inválidas se uma rotina do tipo pergunta/resposta demorar mais tempo sem razões óbvias.

Ocasionalmente pode valer a pena **duplicar dados** que são frequentemente utilizados, de modo a que as cópias dos ficheiros essenciais sejam rapidamente acedíveis de diferentes locais do disco CD-i.

Programas modulares e versáteis optimizam os recursos do sistema, carregando e processando apenas a informação requisitada pela aplicação um dado instante. À medida que a apresentação decorre os bancos de memória do leitor são constantemente renovados com os dados apropriados, dando origem à optimização da utilização dos recursos e, em particular, à realização de uma troca de informação entre o disco de CD-i e a audiência o mais rápida possível.

4.2 PROGRAMAÇÃO EM CD-RTOS

Os **módulos** dentro de um programa de CD-i compreendem quer o código quer constantes (valores que não mudam) de um programa convencional, que estão guardados no disco, e ainda módulos de dados criados dinamicamente para acomodar toda a informação variável durante a apresentação de um CD-i [MICROWARE91b].

O sistema operativo CD-RTOS tem uma arquitectura modular, quer para suportar as exigências das aplicações multimédia interactivas, quer para facilitar a sua actualização (*upgrade*), num mercado que muda rapidamente. O CD-RTOS tem um número de **características** especiais que são muito do agrado dos programadores:

- Pode organizar os **ficheiros** dentro duma estrutura hierárquica, sem limitações à profundidade da respectiva árvore. No entanto este facto pode dar origem a caminhos (*path*) muito longos, através das muitas camadas descendentes e pode tornar esta estrutura impraticável. Para minimizar este problema, o CD-RTOS providencia também dois directórios de recurso (*default*). Os ficheiros abertos para iniciar um processo ficam contidos num directório de execução, enquanto que os ficheiros que não são executáveis ficam contidos dentro de um directório de dados.
- Pode iniciar **processos** através do comando *chain* (que substitui o processo corrente por um novo) ou com o comando *fork* (que permite criar um novo processo deixando o último ainda a correr). O processo pai pode designar o número de processo, a prioridade do processo filho, os caminhos I/O que herda, o conteúdo dos parâmetros e a memória extra para a sua pilha (*stack area*).

- É um sistema operativo **multitarefa sem gestão de memória**, mas permitindo que os programas sejam executados em qualquer ponto da memória. Quando é iniciado um novo processo, a maior parte dos registos na unidade de processamento central são inicializados, incluindo o apontador da pilha e o apontador da memória livre, de modo que o sistema operativo possa simplesmente apontar os programas para as localizações apropriadas. Os programas devem ter uma localização independente das suas variáveis e das subrotinas, podendo ser localizados em qualquer posição da memória.
- O acesso a segmentos de memória livre pode ser obstruído por blocos ainda referenciados por outros processos, originando a **fragmentação da memória**. O melhor processo para contornar este problema é utilizar uma chamada ao sistema (*system call*) que permite alocar segmentos de memória em zonas que se encontrem livres.
- A **sincronização**, essencial às apresentações em tempo-real, é realizada por sinas de interrupção (*interrupts*), que actuam sobre o processo corrente. Em seguida é realizada uma acção de controlo pelo gestor de sinais, que envia o endereço da função que directamente responde ao pedido de interrupção. É possível mascarar os pedidos de interrupção, de modo que o processo seja insensível a determinados sinais, continuando a sua normal execução. Os sinais podem ser trocados entre processos ou enviados a partir de gestores de ficheiros.
- Utiliza uma **interface genérica para todos os dispositivos de entrada ou de saída**. Um programa deve endereçar um novo ficheiro ou dispositivo usando o seu nome para o abrir pela primeira vez. No entanto, isto estabelece um número que é directamente associado ao ficheiro ou dispositivo específico, podendo o programa usar este número como uma referência ao ficheiro ou dispositivo, sem necessitar de mais informação. A utilização destas referências permite efectuar atalhos significativos na programação, proporcionando um acesso mais rápido e directo.

4.2.1 *Compact Disc File Manager* (CDFM)

O maior desafio para o programador de CD-i é garantir que os dados podem ser acedidos e processados tão rapidamente quanto possível, satisfazendo os requisitos de entrelaçamento e sincronização. Isto representa uma **combinação optimizada do fluxo de dados de tempo-real e de outros dados**, envolvendo o gestor de ficheiros do CD, que, por sua vez, controla o fluxo de informação a partir do CD.

Os dados de tempo-real, particularmente o áudio, têm de ser colocados sob parâmetros rigorosos e etiquetados com atributos de tempo-real que os distinguem dos registos normais.

Um ficheiro pode conter dados de todos os tipos. Os dados genéricos podem ser armazenados nos espaços entre os ficheiros de tempo-real, para optimizar a capacidade de armazenamento e de processamento. Quanto a elementos audiovisuais e dados relacionados com a programação, estes são armazenados segundo uma determinada ordem, e é mais económico encher os sectores livres com dados que não são de tempo-real do que desperdiça-los com a inserção de zeros (*blank padding*), cujo único propósito é regular o fluxo dos dados de tempo-real.

O arranjo dos directórios e dos ficheiros é uma questão de preferência pessoal ou de prática local, mas deverá ser prático e consistente. Um modo de apresentação seria, por exemplo, colocar todos os dados para um segmento (lição, jogo, etc.) num só directório, com o áudio, o vídeo, os elementos relacionados com o programa e o código respectivo em subdirectórios separados.

O gestor de ficheiros do CD lê todos os dados do mesmo modo a não ser que haja algum requisito especial, através do comando ***Play*** do CDFM. Quando se executa o comando *Play*, os dados passam a fluir do disco para o sistema o mais rapidamente possível. O comando *Play* destingue os dados CD-DA e CD-i (áudio, vídeo e outros formatos relacionados com o programa) para os dirigir para canais apropriados, bancos de memória ou processadores. No entanto, não faz nada para controlar a velocidade de transferência de dados. Apenas o ordenar dos sectores do disco pode assegurar que os ficheiros de tempo-real são entregues uniformemente aos seus destinos de modo a apoiar uma apresentação sincronizada. Os tempos de acesso são mais pequenos no centro do disco; no entanto, a proximidade da colocação dos dados é mais importante que a sua posição real no disco.

O desafio do **entrelaçamento** e da **sincronização** depende da variedade dos dados usados na apresentação. Para ouvir apenas música necessitamos só de um comando para abrir o ficheiro de áudio apropriado e de outro para executar o comando *Play*, que identifica os dados como registos de tempo-real e os manda directamente para o processador de áudio, deixando o microprocessador do sistema CD-i livre para fazer outra coisa, desde que não interfira directamente com o disco ou com o processador de áudio. Por exemplo, para abrir um ficheiro de texto ou de gráficos armazenado na memória. No entanto, os registos de tempo-real combinam sectores entrelaçados de áudio, vídeo e de outros dados relacionados com o programa, tais como texto ou botões. Para realizar uma apresentação sincronizada destes dados tão diversos, os registos de tempo-real contêm também comandos ou *triggers* para sincronização, que constituem o sistema de controlo da entrega das instruções necessárias para uma apresentação eficiente.

Todos os **ficheiros de tempo-real** começam com um RTCA (*Real-Time Control Area*), que armazena os comandos que controlam a execução de registos de tempo-real, o carregar e descarregar dos dados de vídeo e áudio, a saída do áudio, e ainda o manuseamento de *drawmaps*. Estes comandos podem ser executados sequencialmente, em paralelo, ou nas combinações apropriadas para a apresentação em causa. Estes comandos são descodificados por um interpretador de registos de tempo-real RTRI (*Real-Time Record Interpreter*), que é um descodificador multitarefa capaz de manusear várias actividades em paralelo.

O **RTRI** não é notavelmente rápido, mas utiliza eficientemente os recursos do CD-i, carregando e descarregando muitos registos de tempo-real, pequenos e discretos, conjuntamente com as instruções que lhes dizem respeito.

Devido ao facto de os dados serem requisitados ao acaso num ambiente de tempo-real, as instruções do RTRI não seguem a sequência prevista por muitos programas de computador, mas responde a *triggers* que actuam como deixas para actores à espera (ponto). Cada instrução responde ao seu sinal de *trigger* e junta-se à fila de espera para execução. As instruções básicas de RTRI contêm as funções mais comuns e podem ser estendidas adicionando funções para aplicações específicas.

Os **ficheiros de tempo-real** são executados através de uma chamada ao sistema, PSC (*Play System Call*), que selecciona sectores individuais pelo seu número de ficheiro (*file number*), tipo de dados (áudio, vídeo ou outros dados relacionados com o programa) ou número de canal (*channel number* que pode variar entre 0 a 15 para áudio e 0 a 31 para outros dados).

A PSC contém o PCB (*Play Control Block*) que desfaz o entrelaçamento dos dados de tempo-real, separando os sectores de áudio, vídeo e de outros dados relacionados com a programação, e enviando-os para os processadores específicos ou para os bancos de memória apropriados.

O PCB atribui listas PCL (*Play Control Lists*) individuais a cada tipo de dados. Simplificando, cada entrada nestas listas descreve os campos de controlo e de estado, aponta para os *buffers* apropriados e para a próxima entrada da PCL. As **PCLs de áudio** proporcionam uma máscara de selecção de canais (qualquer grupo de canais de 0 a 15) que determina se o áudio deve ser mandado directamente para o processador ou para um *soundmap* num dos dois bancos de memória RAM. As **PCLs de vídeo** direccionam gráficos para o *drawmap* apropriado e as PCLs de dados enviam os dados relacionados com o programa para a zona apropriada da memória do sistema CD-i, para processamento.

Um programa pode controlar os comandos de execução (*Play*) através do PCB, mas a actividade real tem lugar no CDFM sob a forma de um serviço prestado pelo sistema operativo. Também neste caso, esta breve descrição parafraseia uma actividade muito mais complexa. Cada entrada na PCL inclui também um número de sinal que o CDFM pode enviar para o programa quando o comando *Play* termina, ou quando encontra um *trigger* no disco ou um *buffer* que está temporariamente cheio de mais para aceitar mais dados. Os dados podem assim fluir facilmente para os *buffers*, sendo este processo mais rápido do que a capacidade de processamento dos dados já disponíveis.

4.2.2 *USER COMMUNICATIONS MANAGER* (UCM)

O UCM faz a gestão dos sinais I/O. O UCM está relacionado com o periférico de entrada que controla a posição do cursor (normalmente um rato ou teclado) e com as saídas do som e da imagem.

O **cursor** é controlado usando um modo análogo ao que se encontra em ambientes computacionais. Num determinado instante, as coordenadas correntes do ecrã podem ser fornecidas através de uma função chamada *PT-Coord*, que o CD-TROS emprega sempre que se pretende determinar em que ponto se encontra o cursor. Um programa pode pedir ao UCM para determinar as coordenadas, a uma cadência apropriada, sempre que o cursor se movimenta. O sistema pode seguir um movimento rápido fazendo 20 amostras por segundo, registar menos dados se o movimento for mais lento ou não registar nenhum quanto o cursor está parado. O UCM pode controlar a forma do cursor, cor, localização, cintilação, e até mesmo a sua presença ou ausência em cada instante. O plano do cursor do CD-i pode suportar qualquer forma desde que possa estar contida num bloco de 16x16 pixels, numa só cor escolhida entre 16.

O **teclado** também funciona numa forma familiar, com os dados a passarem através de um *buffer* de entrada. Quando for solicitada uma entrada através do teclado, o programa pode emitir um pedido de leitura e esperar por dados ou, alternativamente, pedir ao UCM para enviar um sinal sempre que se actua no teclado. O programa também pode pedir ao UCM que determine quantos dados estão no *buffer*, quer para iniciar a leitura quer para esperar por mais entradas.

A forma como o CD-RTOS trata o áudio e o vídeo permite realizar a sua distinção relativamente ao OS-9 e a outros sistemas operativos familiares em ambientes de computador convencionais. As funções áudio providenciam um controlo de volume, funções de comando (FF, REW, PLAY, etc) e de distribuição da potência sonora pelas diferentes saídas de modo independente.

A maioria das funções de **áudio** do UCM dizem respeito aos ***soundmaps*** que normalmente são gerados através dos comandos *Play* do CDFM. O UCM cria uma estrutura de dados para os *soundmaps*, na qual se pode colocar os dados de áudio, e um identificador para lhe dar um nome. Os *soundmaps* podem ser misturados dentro do UCM e a sua saída direccionada para o processador de áudio.

O UCM contém uma grande variedade de funções que permitem gerir o **vídeo** sobre a forma de ***drawmaps***. Dentro de cada *drawmap*, o UCM define duas ou mais regiões discretas ou sobrepostas, que podem ser movimentadas dentro do *drawmap*, copiadas para trocar dados entre *drawmaps*, e até mesmo, apagadas em conjunto, realizando operações de *clipping*. Estas regiões podem ter uma definição que pode chegar até ao nível dos pontos (*pixels*) e linhas (por exemplo, para realizar economicamente uma actualização de uma parte da imagem, nomeadamente a parte que está relacionada com a informação mais importante da imagem). Funções de desenho em tempo-real, semelhantes às que se podem encontrar num pacote de software de desenho, também podem ser executadas dentro das zonas definidas pelos *drawmaps*.

O UCM oferece uma grande variedade de opções para escrever **texto** em *drawmaps* e codificar vários tipos de fontes de texto e caracteres especiais. Com o UCM pode-se escrever texto num *drawmap* partindo de dois acessos diferentes. Funcionando no modo de emulação de um terminal, o UCM trabalha como um terminal de computador simples com sua linha de comandos e atributos de caracteres. Alternativamente, com mais programação, as cadeias de caracteres (*strings*) de texto podem ser desenhadas ponto-a-ponto (*bit-mapped*), tal como qualquer outro objecto, permitindo especificar com precisão o ponto de partida e o tamanho de cada *string*. Consegue-se assim um controlo mais preciso do texto no ecrã do que com o emulador de terminal.

O UCM pode suportar **fontes de texto** em módulos de dados que têm de ser colocados na memória. Cada letra de cada fonte tem um só número de código, *bit map* e tamanho (embora todas as letras de qualquer fonte possam partilhar atributos, como o espaçamento, altura e o formato CD-i nos quais elas estão codificadas). Estes códigos podem ser interpretados de três maneira diferentes, e daí resulta o inevitável dilema entre a complexidade e a compreensão. Para conjuntos de caracteres complexos com possibilidade de escolha de tipos de estilo, os códigos de 16 bits podem conter até 65536 letras com 16 bits por letra. Para texto mais simples os códigos de 8 bits podem conter 256 letras. Para combinar códigos sucintos, para caracteres usados frequentemente, com opções para fontes ou letras especiais, o método de 7/15 bits permite a utilização de códigos de 0 a 127, a 8 bits por letra, e de 32768 a 65535, a 16 bits.

O UCM também pode vigiar o DCP. As únicas instruções que o UCM tem que suportar são aquelas que ligam as tabelas de controlo de campo e linha do DCP. A maior parte dos efeitos especiais são controlados indirectamente, manipulando os comandos do DCP. No entanto, as acções do DCP são sincronizadas apenas com instruções de um programa de controlo. O DCP controla a saída de gráficos para o processador de vídeo, linha a linha e campo a campo, mas as suas funções são aplicadas sob a direcção de um programa que especifica como os vários parâmetros devem ser manipulados para criar os efeitos desejados. Afortunadamente para o programador, estas instruções podem ser expressas sucintamente com a ajuda de bibliotecas de funções e de ferramentas como os sistemas de autoria.

4.3 Ambientes para Produção de Aplicações CD-I

Muitos dos pioneiros da programação em CD-i desenvolveram as suas próprias ferramentas de produção de aplicações CD-i, para facilitar a criação dos primeiros projectos experimentais. Empresas como a Philips, a OptImage e a Capitol Disk Interactive, tomaram também a iniciativa de desenvolver e comercializar uma gama de ferramentas dedicadas à produção de títulos para CD-i.

Na prática, muitos dos programas interactivos envolvem uma combinação de aplicações criadas com sistemas de autoria e bibliotecas de funções. Para a equipa de produção o ponto crítico é a escolha da combinação ideal.

Actualmente os produtores de aplicações interactivas têm ao seu dispor uma gama de opções que incluem:

- Linguagens de programação de **baixo nível**, tais como a linguagem *assembly* do processador 68020, muito próxima do **hardware** e que requer programadores especializados em CD-i;
- Linguagens de programação de **alto nível**, que utilizam palavras e comandos com um grau de abstracção elevado diminuindo a proximidade com o hardware, mas ainda necessitando de peritos em **software**;
- Ambientes completos baseados em **sistemas de autoria**, que oferecem *templates* e rotinas a partir das quais o **utilizador** pode moldar as suas apresentações;
- **Bibliotecas de funções**, que fornecem código pré-definido disponibilizando tarefas ou rotinas vulgares.
- **Motores de aplicações**, que providenciam uma estrutura que pode ser utilizada várias vezes durante uma apresentação, mudando apenas os dados iniciais e sem alterar a forma como são apresentados.

As primeiras aplicações para o sistema CD-i foram desenvolvidas principalmente através de programação usando linguagens de baixo-nível, mas actualmente existe um número variado de ferramentas que permitem uma automatização das actividades mais básicas [MICROWARE90].

Na maior parte das aplicações multimédia, as acções racionais dos utilizadores podem ser reduzidas (virtualmente) a um número finito de opções; o que permite aos programadores construir previamente um conjunto de módulos que podem ser utilizados em qualquer aplicação [SMITH90].

A construção de aplicações para CD-i deverá assentar sobre metodologias de reutilização de software. As rotinas pré-compiladas contornam a necessidade de cada equipa/projecto *reinventar a roda* para satisfazer as suas necessidades. Os ambientes de desenvolvimento de aplicações devem fornecer aos programadores uma boa interface gráfica e outros efeitos relacionados, tais como ícones, menus, caixas de diálogo e a possibilidade de recorrer à ajuda prestada por comandos do tipo *help*. Assim, o programador tem acesso, dentro do próprio software de desenvolvimento, a todos os memorandos de que pode necessitar. A tendência, em praticamente todas as áreas de aplicações de software, tem sido produzir programas com ferramentas que até os utilizadores casuais (não programadores) possam adaptar às suas necessidades.

O desafio para os programadores está em ser pioneiro, já que em cada campo só após a realização de alguns projectos é que se pode determinar qual é a gama normal de opções funcionais e criativas. Os primeiros trabalhos num novo ambiente tendem, necessariamente, a ser desenvolvidos a partir do nada ou então recorrendo a aplicações já desenvolvidas para outros fins. No entanto, assim que existam alguns modelos, os produtores de software rapidamente põem no mercado novas aplicações.

4.3.1 LINGUAGENS DE PROGRAMAÇÃO

Praticamente todas as primeiras aplicações de CD-i foram escritas numa única linguagem de programação: C. A **linguagem C** foi desenvolvida a partir de uma linguagem mais antiga, chamada BCPL, que por sua vez derivava da linguagem B e foi construída de forma a poder ser utilizada dentro do sistema operativo UNIX.

A linguagem C é popular porque combina a conveniência de uma linguagem de alto nível com a capacidade de escrever em níveis inferiores (por exemplo, para endereçar uma instrução específica que se encontra directamente ligada ao hardware do sistema). Isto é precisamente o contrário do que acontece no desenvolvimento de aplicações multimédia, onde a maior parte das instruções são escritas usando linguagens de alto nível. Contudo, são necessários alguns módulos de baixo nível para executar directamente as instruções para os dispositivos de entrada e saída .

As **linguagens 4GLs** constituem uma alternativa à linguagem C e a outras linguagens de aplicação geral (BASIC, Pascal, etc.). Estas linguagens tem uma sintaxe idêntica às linguagens de programação de alto nível mas vêem as suas funcionalidades melhoradas, providenciando ferramentas para indivíduos sem experiência na programação de computadores. Professores e especialistas, por exemplo, podem criar os seu próprios trabalhos interactivos com ferramentas que estão mais direccionadas para as suas necessidades do que para as necessidades do hardware. As 4GLs e outras ferramentas constituem sistemas de autoria de alto nível e utilizam frequentemente um ambiente gráfico com janelas, bem como oferecem a possibilidade de acesso a um *help*, menus e caixas de diálogo, fornecendo ao utilizador um conjunto de opções para cada tarefa que pretenda implementar. As linguagens de programação convencionais fornecem ao autor uma folha em branco e um conjunto de ferramentas de desenho, que o autor pode combinar de diferentes maneiras. As 4GLs oferecem ao utilizador um conjunto de desenhos base e várias ferramentas com as quais o utilizador pode construir a aplicação. Elas também tendem a utilizar uma combinação de palavras comuns, metáforas desenhadas com alusões aos objectos do mundo real, com gráficos e outros efeitos visuais, para melhorar a apresentação de um texto. Os utilizadores vêem apenas os comandos e as opções de que necessitam para definir o conteúdo e a estrutura da sua apresentação. A linguagem de programação realiza internamente toda a programação extra, necessária para a tradução das instruções do utilizador em comandos e chamadas ao sistema operativo.

Nas 4GLs, cada elemento discreto de uma apresentação é transformado num objecto com os seus próprios atributos, alguns dos quais partilhados com objectos semelhantes. Num ambiente multimédia, por exemplo, pode-se assim distinguir facilmente o áudio, o vídeo e os dados relacionados com o programa, e descrever cada um com os termos apropriados.

As 4GLs apareceram ao mesmo tempo que os sistemas periciais, o hipertexto e outros ambientes que deram um novo aspecto às comunicações entre os indivíduos e os computadores. Os fluxogramas de blocos descrevem adequadamente os programas de computados convencionais, com as suas ramificações hierárquicas, partindo de um único ponto, através de um certo número de caminhos e chegando a um número finito de conclusões. Os sistemas intuitivos ou analíticos, que

utilizam elementos de inteligência artificial, oferecem muitos caminhos para percorrer os diferentes módulos de informação e necessitam de uma estrutura menos parecida com uma árvore e mais com uma teia, formando um padrão complexo e permitindo muitas viagens e experiências diferentes. As duas inovações, uma linguagem acessível a um não programador e uma estrutura que aspira a ter a flexibilidade do pensamento humano, são complementares e normalmente aparecem em conjunto, permitindo que indivíduos especialistas noutras áreas apliquem o seus conhecimentos directamente num sistema interactivo.

4.3.2 Programação com Sistemas de Autoria

O termo **sistemas de autoria** (capítulo V) é ambíguo, podendo ser aplicado quer às ferramentas utilizadas por programadores de computadores, quer às ferramentas que ajudam especialistas sem formação específica na área das ciências da computação. Os primeiros escrevem o código de programação numa linguagem de computador usando comandos muito próximos da máquina e os segundos utilizam ferramentas de software que minimizam o esforço da programação recorrendo a interfaces gráficas amigáveis.

Os sistemas de autoria para aplicações multimédia interactivas podem ser implementados em qualquer nível de programação, a partir de código dificilmente reconhecível na notação binaria do próprio hardware ou usando produtos hipermédia que interagem com o autor em todos os estágios da produção, utilizando linguagem e conceitos correntes. Os sistemas de autoria tem caído em duas categorias principais:

- **Sistemas baseados em menus,** oferecem ao programador uma gama limitada de opções. Estes pacotes de software são relativamente fáceis de aprender e de utilizar, mas são um tanto ou quanto inflexíveis. Tal como todos os sistemas baseados em texto, também tendem a ter dificuldades no que respeita às funções matemáticas necessárias para realizar simulações e jogos, e não estão preparados para o desenvolvimento de produtos para entrega numa grande variedade de plataformas de sistemas.
- **Sistemas baseados em linguagens**, combinam muitas ou todas as características das linguagens de programação de alto nível com rotinas que facilitam as funções mais comuns, tais como a análise das resposta do utilizador e a ramificação interactiva. Com esta capacidade de funcionamento e versatilidade, estes sistemas levam normalmente algum tempo para aprender e utilizar eficazmente. Estes pacotes de software oferecem algumas facilidades matemáticas, mas as simulações e jogos complexos requerem geralmente programação adicional. O aparecimento de novos produtos para CD-i e outras plataformas multimédia podem seguir um padrão semelhante, isto é, quer software dedicado quer software que seja independente da plataforma final de execução.

Os sistemas de autoria **convencionais** baseiam-se nos princípios básicos da programação de computadores e requerem utilizadores com uma grande perícia. A procura para estes sistemas está a diminuir devido à concorrência de produtos com interfaces mais amigáveis, mas aqueles que ainda restam constituem um pequeno mas importante mercado para aplicações altamente elaboradas.

Os sistemas de autoria de **alto nível** controlam todos os aspectos da produção, apresentação e gestão de uma apresentação interactiva. Com o surgir de sistemas completamente digitais dever-se-ia controlar também a aquisição, criação e manipulação de gráficos, imagens de vídeo, e ainda o texto, o áudio e outros dados associados.

Um sistema de autoria para ser completo, deverá suportar todas as fases desde a criação no estúdio até à apresentação no sistema escolhido e providenciar uma capacidade de gestão, tal como relatórios do estado do sistema. Obviamente que o trabalho no estúdio é mais complexo que o trabalho do sistema de apresentação, mas o software deve ser o mesmo desde o início até ao fim.

A figura 4.1 mostra a sequências das várias etapas fundamentais de um sistema de autoria para CD-i, de modo que todo o processo de desenvolvimento do título seja realizado de uma forma integrada.

No capítulo V são apresentados exemplos de sistemas de autoria para a plataforma CD-i.

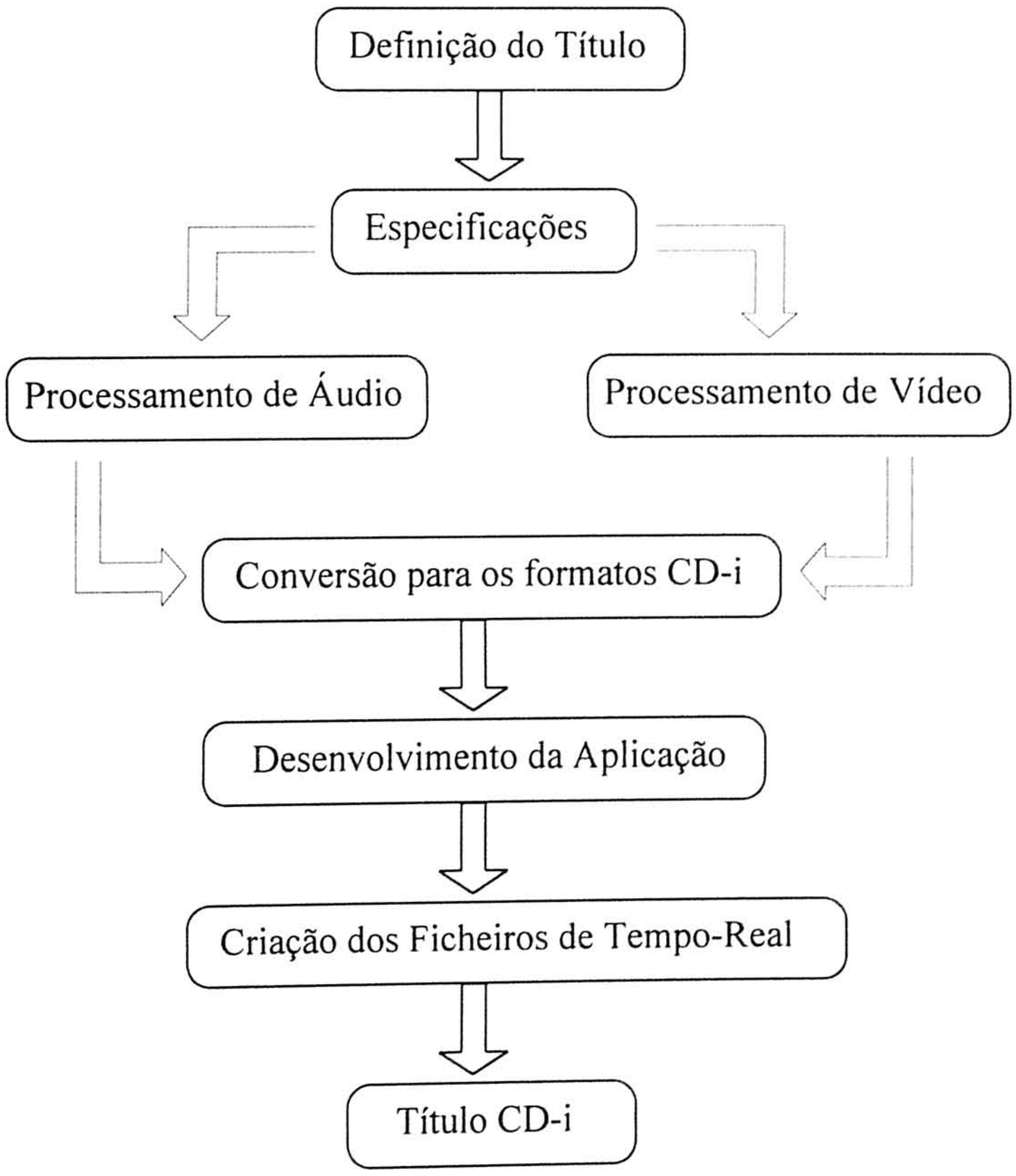


Figura 4.1 - Sistema de Autoria para CD-i

4.3.3 BIBLIOTECAS DE FUNÇÕES

As bibliotecas fornecem um conjunto de **funções pré-definidas** que ajudam os programadores a executar tarefas básicas sem ter que escrever instruções que implementem essas tarefas. As bibliotecas de CD-i providenciam tarefas básicas de que todos os programadores necessitam para controlar o fluxo de dados através do sistema, e efeitos especiais tais como botões ou efeitos de imagens. O sistema operativo CD-RTOS por seu lado também utiliza bibliotecas internas para implementar muitas das suas funções [MICROWARE90].

Para o desenvolvimento de um título sobre a Universidade do Porto, foram desenvolvidas bibliotecas de funções, usando Balboa, de modo a reduzir o tempo de construção da aplicação e optimizar os recursos invocados [RIBEIRO95].

Como virtualmente todas as aplicações de CD-i iniciais foram escritas em C, também as primeiras bibliotecas foram desenvolvidas para fornecer rotinas que os programadores em C pudessem optimizar com programação própria. A utilização de bibliotecas de funções para procedimentos normalizados, permitiu a pré-compilação do código associado às tarefas especiais, optimizando o processo de produção de um CD-i.

O ambiente de rotinas **Balboa** (secção 5.7), desenvolvido pela Philips e pela OptImage, foi o primeiro modelo de bibliotecas de CD-i. O Balboa é modular, com 16 módulos (gestores) independentes que providenciam uma vasta gama de funções, e pode ser instalado num ambiente multitarefa numa SUN, num Mac II ou num computador PC-compatível [OPTIMAGE94b].

4.3.4 MOTORES DE SOFTWARE

Os motores de software proporcionam uma funcionalidade semelhante às bibliotecas de funções, criando uma estrutura que pode ser usada várias vezes durante a mesma apresentação, sendo apenas alterados os dados que constituem as diferentes sequências. Uma apresentação do tipo "projecção de slides", em que as imagens são passadas segundo uma sequência e acompanhadas por uma narração ou música, emprega um conjunto de ferramentas básicas que serve para uma qualquer sequência de imagens e áudio. Um motor de software funciona como uma plataforma a partir da qual se podem construir diferentes modelos de aplicações.

Em Itália foi desenvolvido um motor de software para fazer a apresentação de cidades, usando o sistema de autoria HAL-90. A plataforma utilizada para fazer um título sobre a cidade de Veneza serviu também para fazer a apresentação de Lisboa, ou seja, usando o mesmo motor de software foram criados dois títulos CD-i.

4.4 O ESTÚDIO CD-I

Só faz sentido começar a construir um disco CD-i depois de todos os elementos audiovisuais e o código do programa estarem terminados e convenientemente testados. A fase final do processo de desenvolvimento de um disco CD-i consiste na emulação por hardware do comportamento final (real) da aplicação; se a emulação da aplicação for realizada durante a fase de desenvolvimento, permite ao programador navegar pela aplicação e verificar se está a funcionar de acordo com as especificações inicias [OPTIMAGE92b].

As estações de trabalho dedicadas ao desenvolvimento de títulos CD-i apresentam dois modos típicos de funcionamento:

- Execução de tarefas que permitem o desenvolvimento de um programa interactivo;
- Simulação de como o trabalho desenvolvido irá aparecer no ecrã e nas saídas áudio.

Uma **estação de trabalho** pode ser constituída por apenas um computador com acesso ao restante equipamento do estúdio CD-i, ou proporcionar todas as funcionalidades necessárias ao desenvolvimento, teste e construção de títulos CD-i, numa única estação de trabalho. Entre os requisitos básicos de uma estação de trabalho contam-se as possibilidades de:

- Criar texto e gráficos recorrendo a software de tratamento de texto, imagem e animações;
- Manipular o material audiovisual que se encontra digitalizado e capturar algum deste material recorrendo a placas de aquisição de vídeo, *scanners*, placas de aquisição de áudio e software de processamento de imagem;
- Desenvolvimento de protótipos da aplicação que permitam fazer uma análise da aplicação final e antecipar refinamentos ao conjunto da aplicação;
- Fornecer um conjunto de ferramentas de programação, desde a utilização de sistemas de autoria de alto nível até às linguagens de programação avançadas;
- Proporcionar a emulação e criação de imagens de disco CD-i, que permitam a compilação final do programa para passar a fase de teste e validação do produto final, bem como transferir a aplicação para um disco CD-i [OPTIMAGE92b].

Uma configuração mínima do **hardware** inclui:

- Um leitor de CD-i ligado a um terminal e a um emulador, com controladores para ambos os tipos de diskettes, disco duro e com expansão de memória para 5 MB;
- Um computador (PC ou Mac) com um disco interno de pelo menos 700 MB, mas se possível com 2 GB;
- Um WORM (*Write-Once Read-Many*) que permite a gravação de discos compactos (CD-i, VideoCD, CD-ROM, PhotoCD e CD-DA) [PHILIPS92b].

Especificamente, o material de captura **audiovisual** mínimo é:

- Um *scanner* a cores de elevada qualidade com 24 bits e uma câmara para captação de imagens reais;
- Um computador a 32 bits com monitor de 19'', para criar e retocar os gráficos que são capturados;
- Software de edição e manipulação de imagens digitalizadas, mais software para criação de gráficos e imagens virtuais;
- Software para aquisição, edição e codificação de sequências de áudio.

A instalação de uma rede com uma estrutura modular permite acompanhar as necessidades futuras de um estúdio, e a partilha de recursos e informação. Um estúdio típico deve combinar os seguintes sistemas de informação: uma SUN para a programação, um PC para tratamento de imagem, um Amiga para edição de vídeo e um Mac para desenvolvimento de um protótipo em *HyperCard* bem como para a edição do áudio, mais alguns computadores para ajudar em tarefas rotineiras tais como o processamento de texto, a administração financeira e a gestão de projectos.

V

SISTEMAS DE AUTORIA

Neste capítulo serão apresentadas ferramentas que permitem o desenvolvimento expedito de títulos multimédia sobre a plataforma CD-i. Estas ferramentas são constituídas por software e hardware e são normalmente designadas como **sistemas de autoria (*Authoring Systems*)**. Os sistemas de autoria mais recentes possuem interfaces gráficas amigáveis, baseadas em ambientes de janelas, que eliminam o trabalho fastidioso da programação de baixo nível. Os sistemas de autoria são limitados pelas funções elementares que os constituem, daí que os títulos produzidos não utilizem todas as potencialidades da plataforma CD-i.

5.1 O QUE É UM SISTEMA DE AUTORIA PARA CD-I

Os sistemas de autoria permitem o desenvolvimento de títulos em CD-i de uma forma expedita (secção 4.3.2). O principal objectivo destas ferramentas é libertar as mentes criadoras dos procedimentos técnicos que levam à produção final de um título. Uma das fases mais importantes no desenvolvimento de aplicações interactivas é a **prototipagem**, visto ser possível analisar se o produto que está a ser desenvolvido tem possibilidade de satisfazer os requisitos definidos pela equipa de projecto. Assim que um conceito está totalmente definido é necessário mostrar que se pode passar à sua implementação; para tal será desenvolvido um protótipo da aplicação, constituindo um projecto piloto capaz de ilustrar o comportamento do título, quer à equipa de produção, quer ao utilizador final. Estes programas piloto são muito utilizados para fazer a apresentação genérica do futuro título. Naturalmente, a maior parte das ferramentas de prototipagem estão limitadas ao ambiente por elas proporcionado não podendo emular todas as capacidades do CD-i.

A **plataforma do sistema de autoria** deverá ser partilhada em rede, permitindo a manipulação e tratamento dos elementos constituintes do título pelo mesmo sistema que se encarrega da produção do CD-i, facilitando a troca de ficheiros entre as equipas de produção e evitando a duplicação excessiva de ficheiros. Caso não seja possível instalar um sistema de autoria em rede, a manipulação e tratamento dos elementos constituintes do título pode ser realizada em plataformas separadas. O operador poderá usar um Macintosh para realizar o tratamento da imagem e um PC para o tratamento do áudio, dado que esses ficheiros serão depois transportados (por rede ou manualmente) para um sistema de autoria mais amigável capaz de criar as cenas finais do título.

Estão disponíveis várias ferramentas que permitem transportar os vários ficheiros dos diferentes sistemas de processamento para o sistema CD-i. O leitor CD-i 605 tem um interpretador MS-DOS que permite a leitura e a escrita de ficheiros em diskettes formatadas por este sistema operativo. Os utilizadores Macintosh necessitam do pacote de software *MacRead*, que permite apenas a leitura (não se pode escrever directamente em discos Mac) de discos formatados pelo sistema Macintosh. Por último a opção mais versátil está disponível para os utilizadores que dispõem de uma rede *Ethernet*, baseada num pacote de software chamado NFS (*Network File System*) e numa placa *Ethernet* que se coloca num dos dois *slots* do CD-i, facilitando a troca de ficheiros.

Dependendo das necessidades e habilidade da **equipa de programação**, poder-se-á optar por uma das seguintes plataformas: Sun SparcStation, PC, PS/2 ou Macintosh. Todas dispõem de pacotes de programação e emulação.

É muito importante assegurar uma ampla **capacidade de armazenamento**, factor muitas vezes esquecido pelos produtores de títulos multimédia e como um disco CD-i é capaz de armazenar até 750 MB, será boa norma estabelecer um valor que seja superior. Os requisitos mínimos em termos

de capacidade de armazenamento para um sistema de autoria de CD-i será na ordem de grandeza de 2 GB, assim distribuídos:

2 GB = 750 MB (Disco) + 750 MB (Dados do Disco) + 500 MB (Dados Intermédios)

O processo central do desenvolvimento de uma aplicação num sistema de autoria é o **sequenciamento**, que consiste em realizar a sincronização do áudio com os efeitos vídeo e as imagens. O sequenciamento deverá ser realizado num sistema de autoria baseado num leitor CD-i, para evitar conversões que podem alterar o comportamento final do título e usufruir das virtudes da interface gráfica do CD-i.

Em seguida apresenta-se uma lista das ferramentas necessárias para a construção e teste de um título CD-i:

- Uma biblioteca de gestores, usada para criar a estrutura principal do programa multimédia - *Runtime Development*;
- Um programa para criar uma imagem de um futuro disco CD-i - *Disc Building*;
- Um programa de emulação que permita testar e validar a imagem do disco através do leitor CD-i tal qual como se o disco já estivesse gravado - *Emulation*.

Para fazer a emulação das sequências multimédia na estação de programação, será necessária a instalação de uma ponte entre os sistemas de programação e o CD-i. Esta ligação de hardware é realizada pelo **Emulador** (*OptImage Emulator*) que interliga o CD-i a um computador PC, Mac ou UNIX através de uma interface SCSI. Este Emulador foi construído com um processador 68020, para permitir que a maior parte da manipulação dos dados associados à criação de uma imagem de um disco seja retirada da memória do computador, diminuindo o tempo da operação. O mesmo Emulador pode ser usado em diferentes sistemas, sendo para tal necessárias umas ligeiras alterações do software [OPTIMAGE92b].

5.2 ASLAN

O ASLAN é um sistema de autoria para automatizar muitos dos aspectos redundantes associados ao desenvolvimento de títulos CD-i [SENFTNER93]. Basicamente este sistema consiste em:

- Ferramentas de codificação de imagens paradas para o formato DYUV;
- Ferramentas de codificação de sequências animadas usando imagens RGB888 que permitem a criação de animações codificadas segundo o método RL7. Podem-se ainda seleccionar várias opções para configurar a actuação dos filtros sobre as imagens, ler scripts de protótipos, verificar a integridade dos elementos que

constituem o título, optimizar o entrelaçamento do disco, gerar scripts que permitem a construção final do disco CD-i;

- Compiladores de animações que permitem gerar animações a partir de um directório contendo apenas as imagens codificadas em RL7;
- Um compilador de módulos CDRTOS, permitindo a criação de módulos executáveis a partir de um determinado script;
- Scripts de controlo da construção da imagem do disco; trata-se de um conjunto de scripts em UNIX de controlo dos vários passos da construção da imagem de um disco CD-i;
- Uma aplicação CD-i que controla a apresentação multimédia dos elementos que se encontram no disco.

Inclui ainda um vasto número de programas utilitários desenvolvidos para simplificar os cálculos envolvidos na determinação do espaço ocupado pelos elementos do título. Por exemplo o programa "*frame_average*" aceita como parâmetro um conjunto de nomes de ficheiros (*imagem**, correspondente às imagens imagem*1*, imagem*2*, imagem*3*, ... , imagem*n*) de imagens RL7, dando como resultado um relatório explicitando quantas imagens por segundo (*frame rate*) podem ser arquivadas tendo em conta o tamanho das imagens RL7 e os respectivos níveis de áudio. Basicamente, um título ASLAN é construído por sequências e menus.

5.2.1 SEQUÊNCIAS

Uma sequência é uma apresentação linear de áudio/vídeo em que a interacção do utilizador está limitada a operações do tipo pausa, interrupção total da sequência ou permitir a sua reprodução até ao fim. A apresentação de elementos vídeo está limitada a imagens do tipo DYUV e a animações do tipo RL7, a uma qualquer taxa temporal de apresentação desde que não seja ultrapassada a largura de banda do sistema. Relativamente ao som pode-se incluir qualquer nível *stereo/mono* (A, B e C) CD-i. As sequências podem ter vários pontos de entrada (*entry points*) e vários modos de execução.

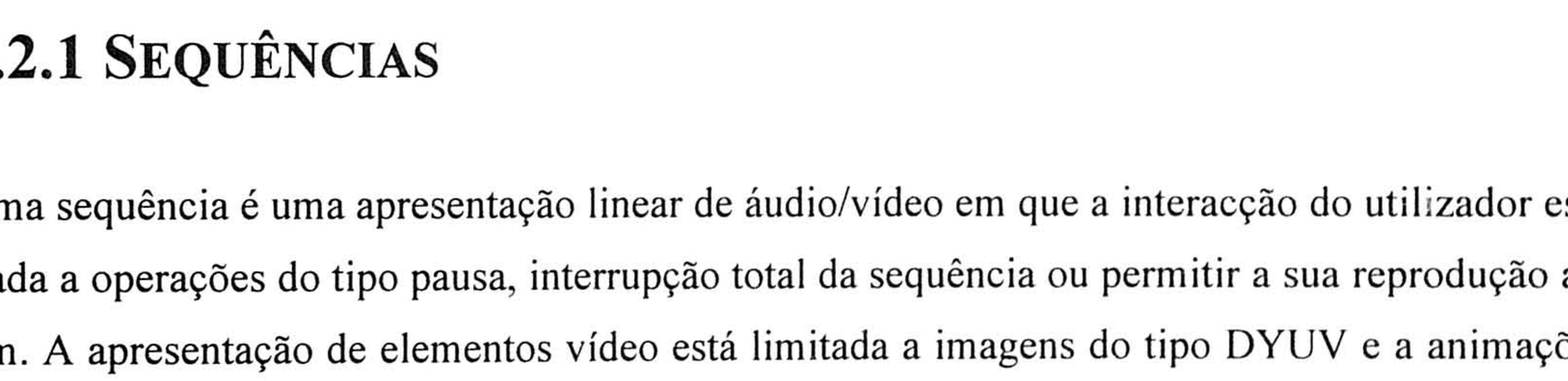


Figura 5.1 - Uma sequência ASLAN com 2 pontos de entrada

Entre duas imagens paradas é realizado um dos seguintes efeitos de transição:

- **cut** - transição rápida de uma imagem para outra;
- **dissolve** - fusão de uma imagem noutra, podendo controlar a duração da fusão;
- **wipe** - possibilidade de fazer entrar a nova imagem de cima para baixo, debaixo para cima, da esquerda para a direita ou da direita para a esquerda;
- **curtain** - fazer um efeito de abertura ou fecho de uma cortina (tipo teatro),

Pode ser apresentada uma **animação** entre duas imagens paradas, do tipo:

- **Normal** - animação RL7 sobre uma imagem DYUV em que uma das cores é transparente;
- **Animate** - animação RL7 por trás de uma imagem DYUV.

Pode ser definida uma interrupção que permita ao utilizador saltar para um outro ponto qualquer do título, tipicamente uma outra sequência ou menu. A sequência pode saltar automaticamente (*branch*) para qualquer ponto do título quando completa.

5.2.2 MENUS

Um menu é construído por uma colagem sobre uma imagem DYUV de uma imagem de ecrã completo em formato RL7 evidenciando determinadas zonas da imagem. Quando o utilizador movimenta o cursor sobre um determinado botão, é mostrada uma imagem completa (*full screen*) evidenciando a zona correspondente ao botão. Caso o utilizador seleccione um determinado botão pode-se despoletar uma determinada acção, por exemplo iniciar uma sequência, tocar um som ou saltar para outro menu. O botão é especificado como uma imagem RGB888. O pré-processador do ASLAN converte a imagem numa região comprimida (*compressed region*) para ser usada quando a aplicação estiver a correr. A região assim definida permite a realização de botões com formas irregulares. Um ou vários pixels ligados ou separados podem ser usados para definir um botão.

Os **botões** de imagens completas RL7 tem várias características com interesse especial:

- Possibilidade de simular sistemas mais complexos de controlo (ex: barra de controlo por arrastamento, visualização tipo processador de texto página-a-página e técnicas do ecrã azul, em que se revelam detalhes que estavam escondidos no botão);
- As imagens são apresentadas quase instantaneamente.;
- O utilizador não consegue bloquear a aplicação através do menu;
- Facilidade de usar versões temporárias dado que as imagens são em ecrã completo;
- Cada zona activa pode usufruir de 128 cores.

Normalmente a configuração da capacidade de armazenamento de botões é de 47 sectores, aproximadamente 106 KB e para uma configuração em que se usa o máximo de memória a capacidade de armazenamento é de 320 KB [SENFTNER93]. Caso esteja definida uma configuração que utilize toda a memória é desactivada a possibilidade de anular a última acção do utilizador.

Os **menus** incluem opções tais como:

- Caso o utilizador não exerça qualquer comando durante um determinado tempo, a aplicação começa a percorrer os diferentes menus e sequências de uma forma pré-definida;
- Tornar o cursor invisível ou visível;
- Coordenadas de posicionamento do cursor definindo a posição inicial;
- Possibilidade de reprodução das pistas sonoras aleatoriamente;
- Anular o último comando de controlo realizado pelo utilizador;
- Gerir a memória de modo a adaptá-la às características das imagens e dos respectivos efeitos;
- Possibilidade de definir no ecrã botões com funções idênticas os existentes no controlo remoto.

5.3 MEDIAMOGUL

Para realizar a **prototipagem** de títulos CD-i a Philips e a OptImage desenvolveram o sistema de autoria MediaMogul. Este sistema foi especialmente concebido para ser utilizado pelas equipas encarregues da concepção de títulos, com pouca ou sem experiência em programação. O software deste sistema de autoria proporciona um ambiente gráfico amigável, para a criação expedita de sequências audiovisuais e um simples mas eficiente motor de interactividade baseado num cursor. Já foram criados muitos títulos, desde o início até ao fim, usando unicamente este software. O hardware deste sistema de autoria consiste num leitor Philips CDI 605, teclado, disco duro e um monitor profissional RGB. O MediaMogul permite a realização de aplicações multimédia interactivas utilizando como base o sistema CD-i. É composto por cinco módulos: *File Director, Script Writer, Editor, Art Studio* e *CD-RTOS* [OPTIMAGE93].

O ***File Director*** realiza a gestão dos ficheiros usados pelo MediaMogul. Este módulo é chamado sempre que se pretende realizar operações de escrita ou leitura de ficheiros, permitindo navegar facilmente pela árvore de directórios. Podem-se ainda realizar sobre os ficheiros as seguintes operações: copiar, mover, apagar, atribuir um novo nome e ver o seu conteúdo.

O ***Script Writer*** permite a escrita de scripts que definem como as imagens de vídeo se combinam com um leque de efeitos especiais e com as diferentes sequências sonoras. Disponibiliza ainda comandos de controlo do programa que permitem a definição de variáveis ou a

inserção de subrotinas desenvolvidas pelo próprio programador. Trata-se do módulo principal do MediaMogul, oferecendo um conjunto de funções pré-definidas em menus, que permitem a implementação das aplicações CD-i.

O ***Editor*** é o módulo que se encarrega da interactividade de uma aplicação, permitindo a definição de botões que implementam a estrutura interactiva de menus. Cada menu pode ter até 64 botões (pontos de interactividade) e cada botão pode chamar qualquer *script* da aplicação.

O ***Art Studio*** permite a criação e manipulação de imagens, que posteriormente podem ser incluídas nos scripts. As imagens produzidas com este módulo tem uma resolução normal no formato CLUT8, que utiliza um máximo de 256 cores.

O ***CD-RTOS*** dá acesso ao sistema operativo do leitor CD-i, disponibilizando um interpretador de comandos (*shell*).

Figura 5.2 - Módulos do MediaMogul

5.3.1 CONFIGURAÇÃO DO SISTEMA MEDIAMOGUL

O MediaMogul pode ser configurado pelo utilizador através do ficheiro **Mogul.config**. Este ficheiro de configuração não é muito importante para o funcionamento do sistema (não é necessário que exista no sistema), mas permite ao utilizador poupar tempo quando constrói uma aplicação. Os comandos que se podem incluir neste ficheiro podem ser definidos através da linha de comandos CD-RTOS, mas sempre que o utilizador sair do MediaMogul esta informação perde-se, o que nem sempre é desejado, especialmente quando se definem os directórios iniciais de pesquisa para os seguintes módulos do MediaMogul: *File Director, Script Writer*, *Editor* e *Art Studio.*

No ficheiro Mogul.config pode definir-se quais os dispositivos (DEV) de armazenamento de dados que estão disponíveis para serem acedidos pelo *File Director*.

Exemplos:

```
DEV     cd      - Compact Disk (Unidade CD)
DEV     d0h     - Diskette de alta-densidade formatada em OS-9
DEV     pc0h    - Diskette de alta-densidade formatada em MS-DOS
DEV     h0      - Disco duro formatado em OS-9
DEV     mac     - Disco duro formatado em Mac
```

A palavra-chave **PRINTER** permite ligar uma qualquer impressora tipo *proprinter* (normalmente uma impressora de agulhas) na porta paralela do CD-i.

A palavra-chave **ANALYSIS** é usada para garantir que a aplicação cumpre as restrições necessárias para que funcione em qualquer tipo de leitor CD-i:

ANALYSIS **n**

A variável **n** pode tomar valores inteiros entre -1 e 255:

- Se **n** = -1, não verifica as restrições de tempo; usado sempre que utilizador não pretende construir um disco CD-i mas apenas correr a aplicação dentro do sistema MediaMogul;
- Se $0 \leq$ **n** < 75, os scripts que satisfaçam esta restrição podem ser usados com segurança em aplicações cujos ficheiros de tempo-real sejam normalizados e que não estejam compactados. Os ficheiros de tempo-real compactados servem para optimizar o espaço de armazenamento do disco CD-i;
- Se $75 \leq$ **n** < 256, os scripts que satisfaçam esta restrição podem ser usados com segurança em aplicações cujos ficheiros de tempo-real estejam compactados.

5.3.2 Módulos do MediaMogul

File Director

Este módulo serve de apoio aos restantes módulos do MediaMogul servindo de gestor de ficheiros, facilitando a pesquisa dos ficheiros no disco. O acesso aos ficheiros e aos directórios é realizado por dois *browsers*, sendo as funções básicas de um gestor de ficheiros realizadas através do manuseamento do rato. A estrutura de ficheiros é idêntica à de um sistema UNIX, existe um directório principal ou raíz onde se podem definir novos directórios ou depositar ficheiros. Apenas pretendemos salientar uma característica específica sobre o tipo de nome de um ficheiro no CD-RTOS: um ficheiro de um CD cujo o nome está escrito em maiúsculas é diferente de outro ficheiro com o mesmo nome escrito em minúsculas (*Audio1<>audio1*), mas se estiverem num outro dispositivo (ex: disco duro) não existe diferença entre eles (*Audio1=audio1*).

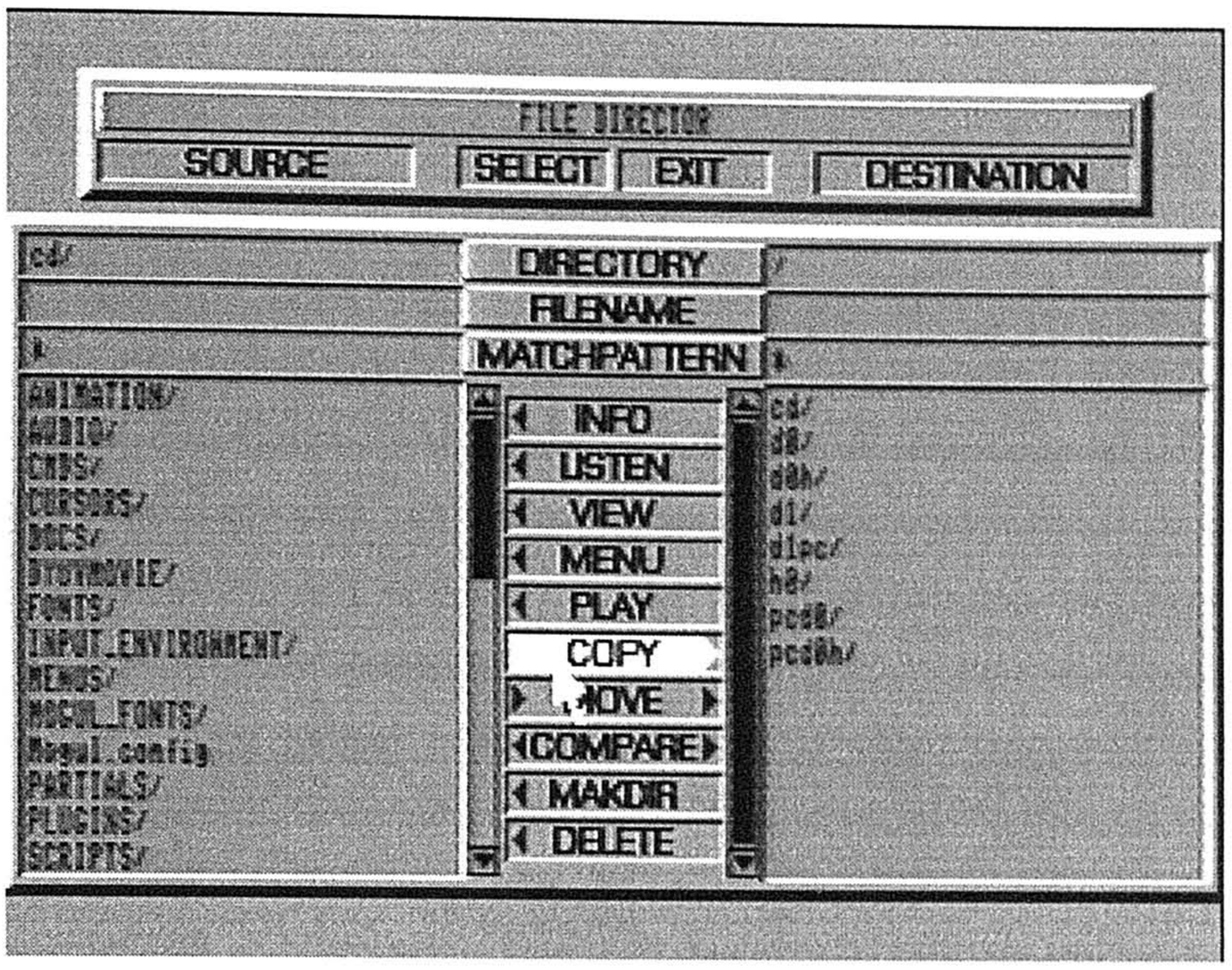


Figura 5.3 - Gestor de ficheiros do MediaMogul

Script Writer

A integração dos vários elementos que constituem uma aplicação multimédia é realizada por este módulo. Os ficheiros de vídeo, som, imagens e os menus são montados usando um script, de modo a construir uma aplicação coerente.

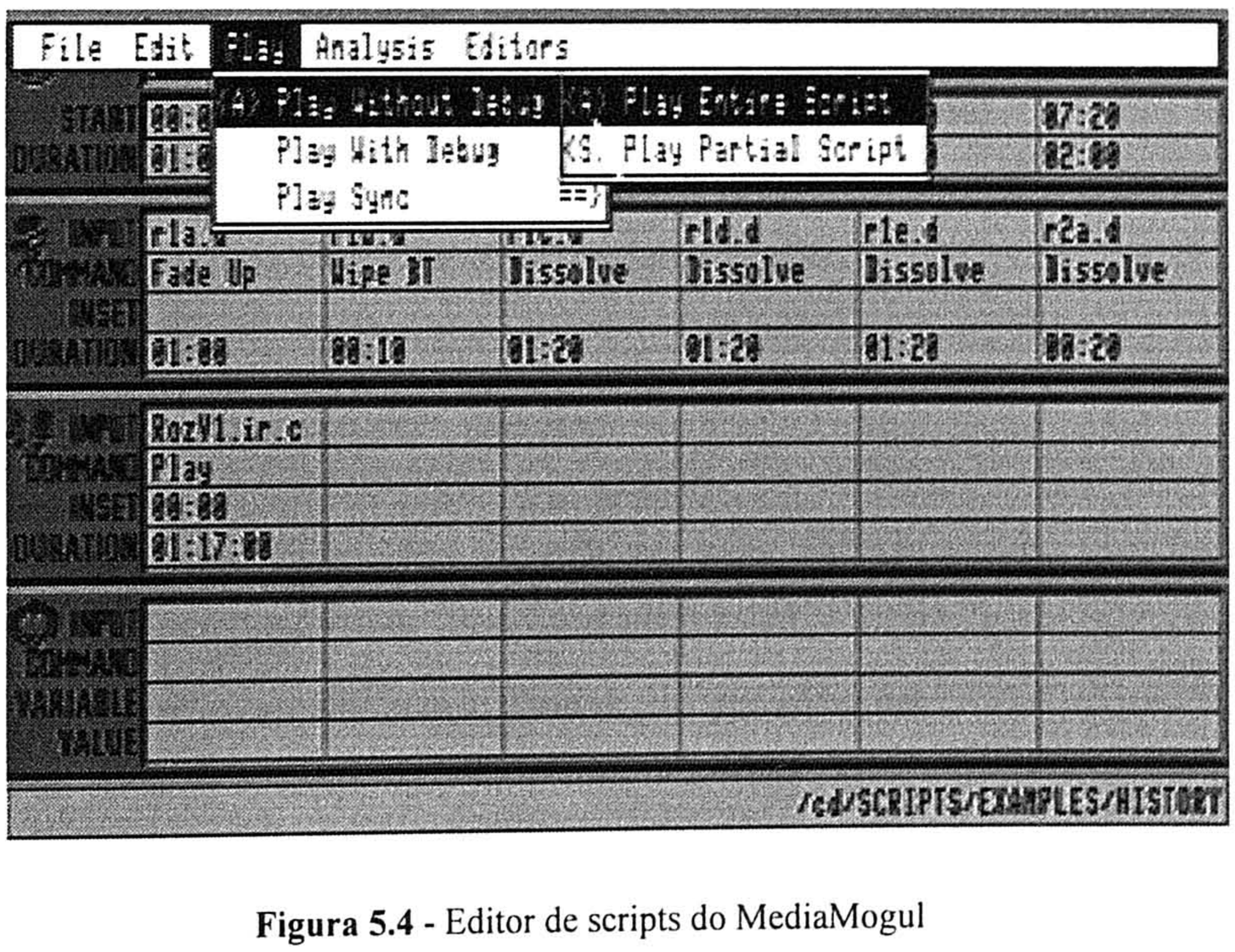


Figura 5.4 - Editor de scripts do MediaMogul

Para criar o script é necessário seleccionar quais os ficheiros de áudio e vídeo a serem reproduzidos, qual a ordem com que aparecem na reprodução, a duração e quais os efeitos a serem aplicados. Os ficheiros juntamente com os respectivos comandos e restrições temporais são inseridos em pequenas células de um quadro que tem associado a cada coluna um determinado intervalo de tempo. Para facilitar a escrita dos scripts, pode-se recorrer a um menu *pop-up* que

disponibiliza um conjunto de informações que podem ser transferidas para as células apropriadas com um simples *click* do rato. Por omissão são definidos valores inicias que posteriormente se podem alterar.

O *script writer* (figura 5.4) proporciona uma representação gráfica dos scripts criados pelo utilizador, associando um evolução contínua do **tempo** da esquerda para a direita. Cada uma das seis colunas que aparecem na figura 5.4 representam o instante de tempo inicial de cada período de tempo de um script. Tudo o que está na mesma **coluna** é chamado no instante de tempo indicado no topo da coluna. Cada coluna está dividida em quatro grupos distintos. O grupo de cima serve para introduzir o instante em que os comandos contidos na mesma coluna são executados. Os dois grupos seguintes servem para indicar quais os ficheiros de vídeo (imagem) e áudio que são chamados. O grupo final permite introduzir elementos que controlam a sequência da aplicação (ex: saltos, botões, ...). A construção de um script em MediaMogul consiste em preencher as células do quadro anterior, usando o rato e seleccionando a respectiva célula. Ao seleccionar a célula aparece uma janela pedindo a informação específica da célula, que o utilizador deve fornecer através do teclado. Algumas das janelas tem menus *pop-up*, devendo o utilizador seleccionar uma das várias opções possíveis ou navegar por uma lista de ficheiros.

A **duração** inclui o tempo de duração de um qualquer efeito vídeo de transição. Ao indicar este parâmetro é necessário garantir que o tempo de duração não passa para além do instante em que o próximo comando vídeo é executado, senão o efeito não fica completo.

Antes de simular a aplicação, correndo o script, pode-se fazer uma análise do script para verificar se o sistema é capaz de executar todos os comandos que lá foram colocados, garantindo que o script dá origem a uma aplicação. Se forem detectados problemas é enviada uma mensagem de erro que ajuda os utilizadores a identificarem qual a origem do erro. O script (sem erros) pode ser gravado num disco magnético para ser visto posteriormente.

Editor

O Editor é o módulo do MediaMogul que permite ao utilizador criar scripts interactivos, definindo zonas de interactividade sobre o ecrã, ou seja, definir botões que funcionam como menus dentro de um script. Cada menu pode conter 64 opções e cada botão associado pode fazer saltar a aplicação para um qualquer script. Depois de definido um botão, a sua selecção é feita através do *click* com o rato e o script que lhe estava associado ao botão passa a ser executado. Por sua vez, este script pode conter novos menus com botões que estejam associados a outros scripts. O limite para o número de saltos possíveis é o espaço de armazenamento no disco.

Como a interactividade dos títulos construídos com o MediaMogul é assegurada por este módulo é conveniente fazer um plano (representação em árvore) com todos os saltos possíveis, dada o pouca estruturação do processo de definição do grau de interactividade.

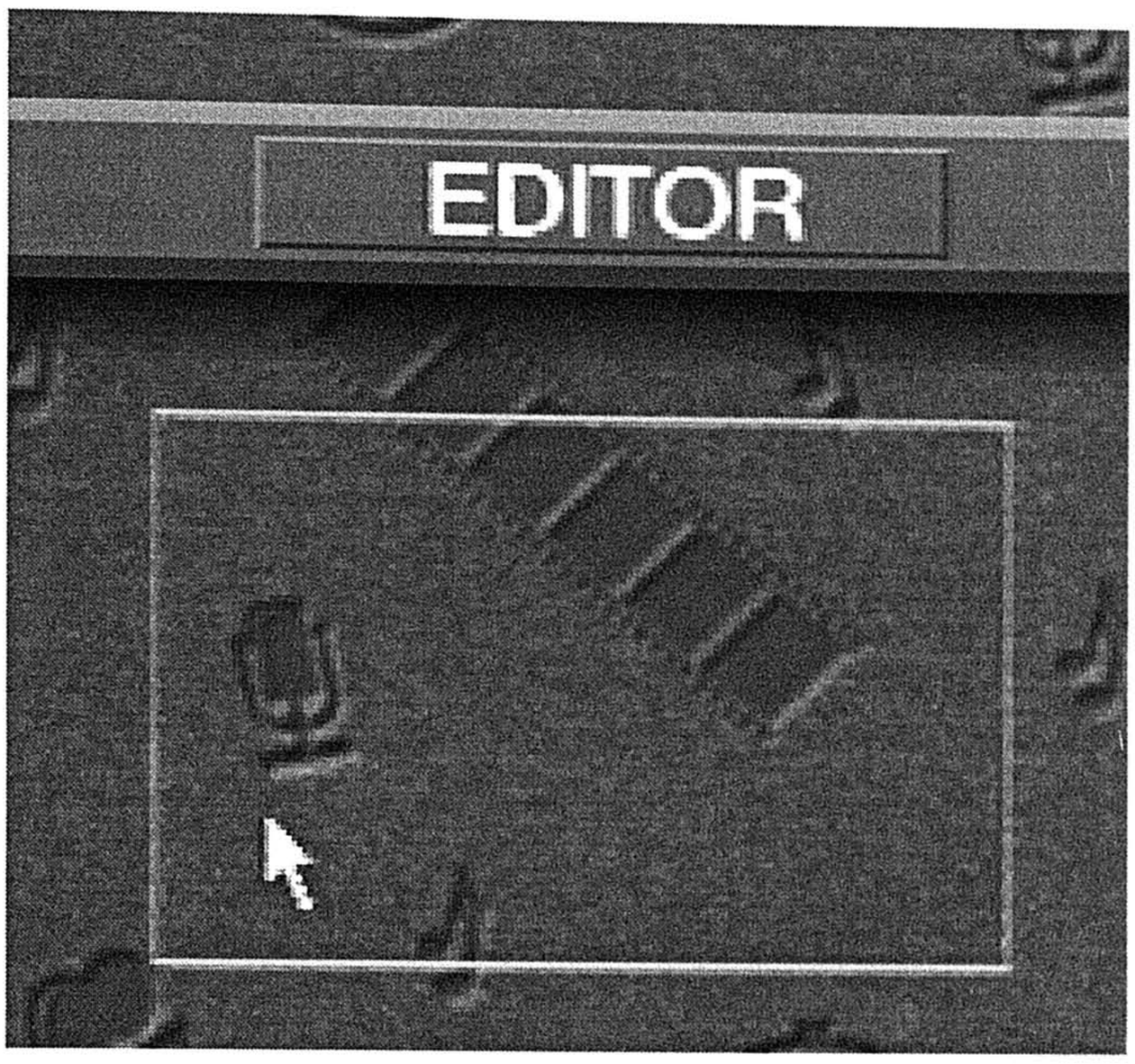


Figura 5.5 - Editor de botões do MediaMogul

O utilizador para definir um botão deve em primeiro lugar visualizar a imagem que vai estar associada ao botão e, em seguida e com a ajuda do cursor desenhar (sobre a imagem) o botão no sítio pretendido. Aparece no ecrã uma janela pedindo informação sobre o botão que acaba de ser desenhado, que o utilizador deve preencher de acordo com a estrutura definida para o título.

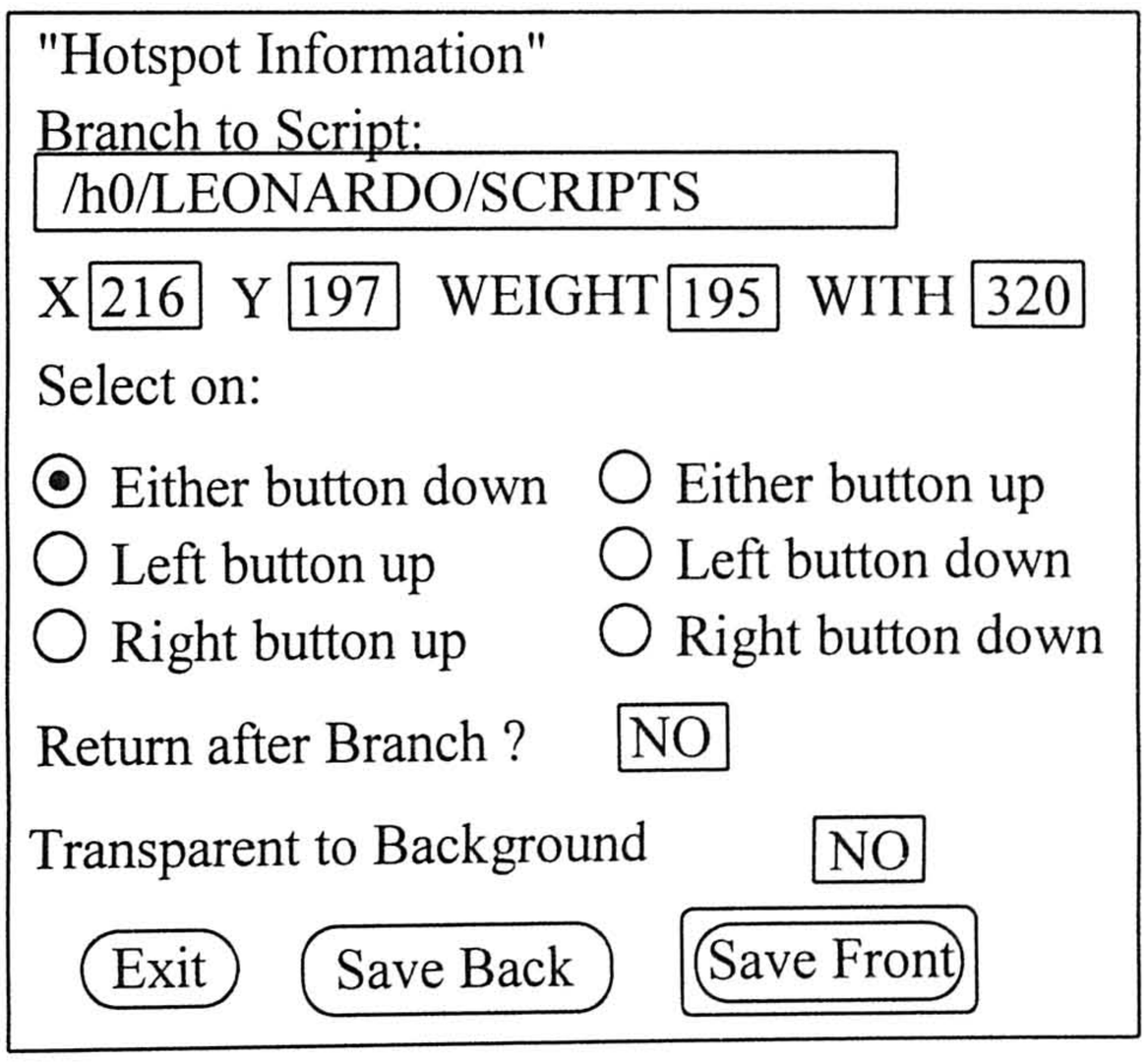


Figura 5.6 - Janela de informações dos botões

É sobre a janela da figura 5.6 que o utilizador define qual o script que vai estar associado ao botão, bem como as restantes propriedades do botão. Uma das propriedades mais importantes é a **transparência**, que permite ao utilizador definir que caso o cursor esteja sobre a zona do botão a

imagem que estava a ser visualizada (Plano A) será substituída por uma imagem que se encontrava em segundo plano (Plano B).

O editor do MediaMogul permite no máximo 128 cores, e por isso, quando se carregam imagens CLUT8 é necessário reduzir de 256 para 128 cores resultando uma degradação da imagem. Contudo a imagem não se degrada se for chamada a partir do script.

Art Studio

Este módulo do MediaMogul permite ao utilizador criar e editar imagens directamente sobre o leitor de CD-i, ou seja, é uma ferramenta de desenho para operar sobre imagens sem necessidade de recorrer a uma plataforma de processamento de imagens (estação gráfica). Todas as imagens produzidas por este programa são de resolução normal no formato CLUT8, com um máximo de 256 cores.

O rato é utilizado para seleccionar as opções e realizar as várias funções de desenho, sendo o teclado usado apenas para a colocação de texto. Os botões do rato têm funções diferentes: o botão esquerdo é o botão de selecção e o botão direito serve para anular a última acção tomada. As várias funções do módulo *Art Studio* encontram-se agrupadas em dois menus no fundo do ecrã. O menu da esquerda contém as funções que permitem escolher as acções que o utilizador pretende executar (pintar, desenhar um quadrado, gravar, ...), o menu da direita contém as configurações que permitem definir as opções de desenho (espessura do traço, cor, ...).

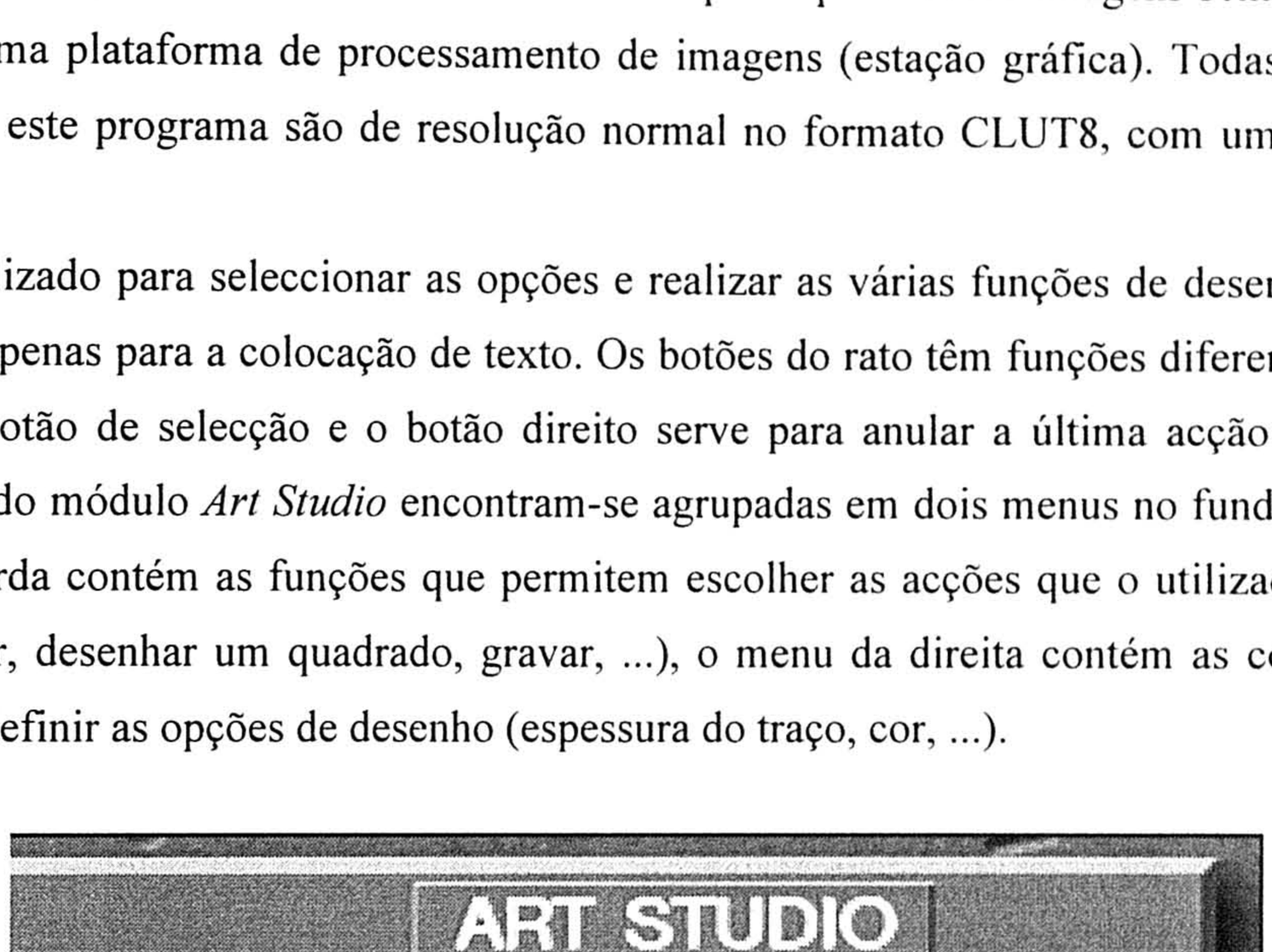


Figura 5.7 - Ferramenta de desenho do MediaMogul

O módulo ***CD-RTOS*** dá acesso ao sistema operativo do leitor CD-i, disponibilizando um interpretador de comandos (*shell*). A sua função principal é permitir executar programas CD-RTOS de apoio ao utilizador. Tipicamente serve para invocar ferramentas de reparação do disco,

executar o software de emulação, ver quais os processos que estão a correr, imprimir textos através da porta paralela, configurar o sistema para funcionar em rede, etc.

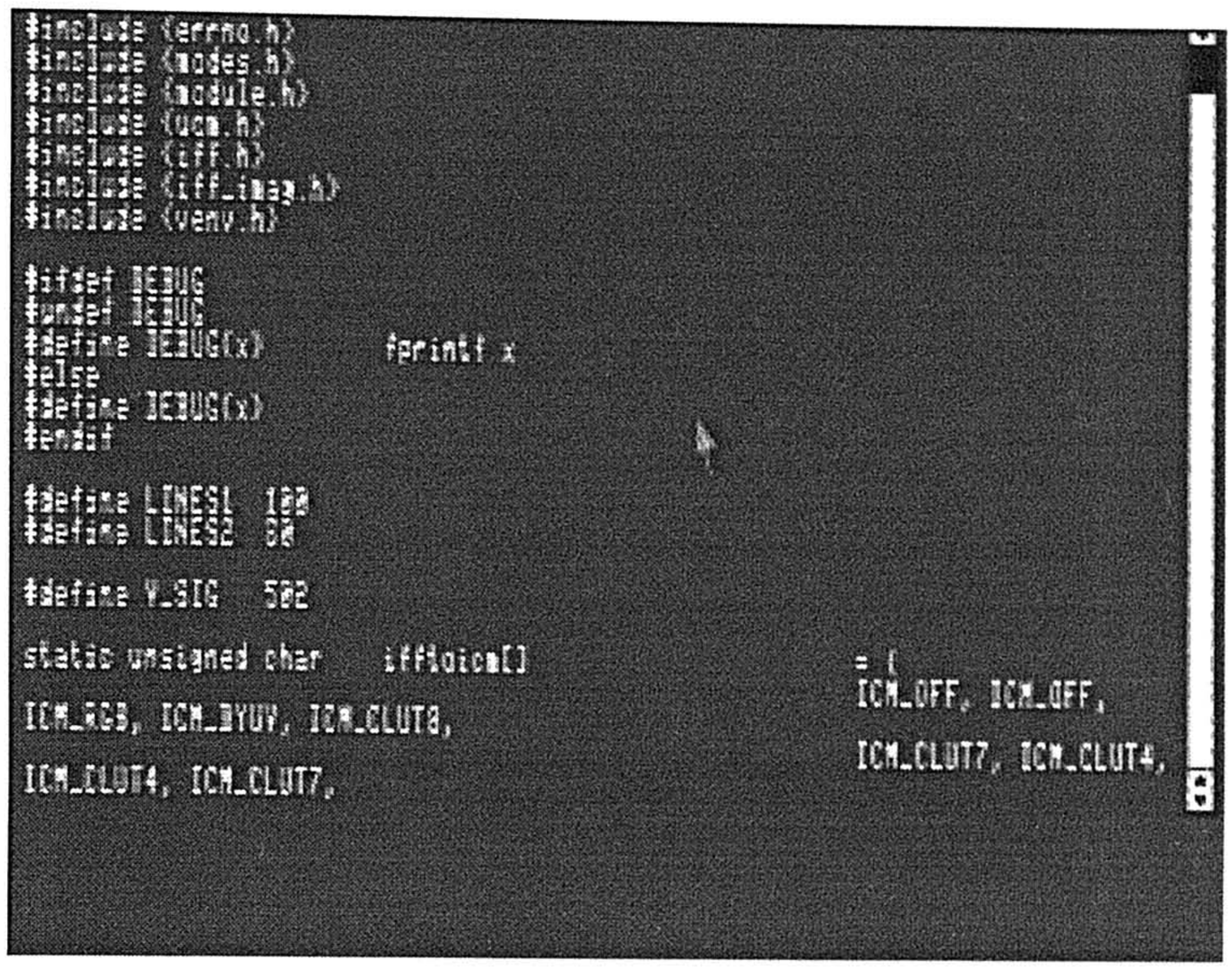


Figura 5.8 - *Shell* do CD-RTOS

5.3.3 Características e Limitações do MediaMogul

Devido às limitações do leitor CD-i, os efeitos de transição excepto o *Cut* não funcionam entre uma imagem CLUT8 e outra imagem qualquer CLUT ou RL. Transições entre imagens CLUT8 e DYUV não são afectadas. Os efeitos rápidos (*fast effects*) não funcionam para imagens CLUT8. Os comandos de desenho e texto (*Rect, Rectcolor Text* e *Textcolor*) só podem ser usados sobre imagens CLUT e o texto não pode conter o caracter ^.

Ao alterar a cor do texto é alterada a entrada 1 da CLUT da imagem. Os melhores resultados são obtidos quando o comando *Textcolor* é usado com imagens que não utilizam a entrada 1 da CLUT. Se estiver a usar fontes de caracteres que não as fornecidas com o MediaMogul, é necessário garantir que o nome do ficheiro das fontes e o nome do modulo OS-9 são idênticos **[ZEGERS95]**.

Uma imagem parcial tem de ser do mesmo tipo que a imagem principal, ou seja, uma imagem parcial do tipo CLUT só pode ser colocada sobre outra imagem CLUT e uma imagem parcial DYUV só pode ser colocada sobre outra imagem DYUV. O comando *Partial* pode ser usado para modificar a CLUT da imagem que está activa no momento, a partir de um ficheiro que tem apenas a tabela de cores (CLUT). O comando *Overlay_bg* só pode ser usado se a imagem do plano B for do tipo CLUT. Muito usado nos botões.

Se for definida uma transparência é possível mostrar a imagem que se encontra no plano B enquanto está a ser executado um ficheiro de animação, mas o comando *Transparency* não funciona se a imagem que lhe é anterior for uma imagem CLUT8.

5.3.4 FERRAMENTAS AUXILIARES PARA ANIMAÇÕES E JANELAS DE VÍDEO

O MediaMogul contém um conjunto de ferramentas auxiliares, para ajudar o utilizador a construir as suas aplicações CD-i. Estas ferramentas encontram-se acessíveis a partir do CD-RTOS.

Ferramentas para Animação

A ferramenta **animpack** cria ficheiros com animações que podem ser chamadas pelos comandos *Animate5, Animate10* e *Animate15* do MediaMogul, correspondentes às taxas de 5, 10 e 15 fps. Este programa lê uma lista de nomes de ficheiros com imagens RL7 ou RL3, que constituem a sequência das imagens da animação pretendida e de tamanho *full screen* (384x280). A CLUT usada na animação é a CLUT da primeira imagem da sequência, sendo possível definir como transparente a cor que se encontra na posição 0.

A seguinte linha de comando passa uma lista de nomes de ficheiros em formato RL7 (dir -u *.RL7) e define opções que se podem considerar como típicas para a criação de uma animação (*lima.anim*).

ex: CD-I> **animpack -b**=30000 **-c**=80 **-m**=25000 <dir -u *.RL7 >*lima.anim*

As opções que permitem controlar a realização da animação são:

-c = 80 define que esta animação tem um número máximo de 80 imagens;

-b = 30000 define que o tamanho máximo para os ficheiros RL são 30000 bytes;

-m = 25000 define um tamanho mínimo para os ficheiros RL como sendo 25000 bytes e sempre que existam ficheiros mais pequenos são acrescentados zeros. O objectivo desta restrição é assegurar uma taxa semelhante para a apresentação das várias imagens que compõem a animação.

A ferramenta **animinfo** permite enviar para o monitor o cabeçalho do ficheiro que contém a animação, neste caso o ficheiro é o *lima.anim*.

ex: CD-I> **animinfo** *lima.anim*

A ferramenta **animunpack** funciona ao contrário do animpack, ou seja, sendo-lhe fornecida uma animação o animunpack cria cada uma das imagens que compõem a respectiva animação.

ex: CD-I> **animunpak** *lima.anim*

Ferramenta para Janelas de Vídeo

A ferramenta **dyuvmoviepack** cria ficheiros com sequências de vídeo que podem ser chamadas pelos comandos *Dyuvmovie5, Dyuvmovie10* e *Dyuvmovie15* do MediaMogul, correspondentes às taxas de 5, 10 e 15 fps. Este programa lê uma lista de nomes de ficheiros correspondentes a imagens DYUV, que constituem o segmento de vídeo que se pretende construir. A largura das imagens tem de ser um múltiplo de 4 e a altura não pode exceder 140 pixels.

ex: CD-I> dir *.DYUV | **dyuvmoviepack** >*lima.dym*

As tabelas seguintes mostram a percentagem máxima do ecrã que pode ser usada para a janela de vídeo, tendo em conta o número de imagens por segundo (fps), o tipo de monitor e o nível de codificação do áudio. As dimensões apresentadas para as janelas de vídeo mantêm o mesmo *aspect ratio* de uma imagem que ocupa todo o ecrã. Só uma e uma só janela de vídeo pode ser mostrada no ecrã num dado instante.

Sem Áudio:

Nº de Imagens por Segundo (fps)	Monitores de 525 linhas	Monitores de 625 linhas
5 fps	39,8% 224x140	32,4% 188x140
10 fps	18,9% 164x104	16,2% 152x112
15 fps	12,6% 136x85	10,8% 124x92

Tabela 5.1 - Janelas de vídeo sem áudio

Áudio C-*mono*:

Nº de Imagens por Segundo (fps)	Monitores de 525 linhas	Monitores de 625 linhas
5 fps	35,2% 224x140	30,2% 188x140
10 fps	17,6% 160x100	15,1% 148x108
15 fps	11,7% 128x82	10,0% 120x88

Tabela 5.2 - Janelas de vídeo com áudio C-*mono*

Áudio C-*stereo* ou B-*mono*:

Nº de Imagens por Segundo (fps)	Monitores de 525 linhas	Monitores de 625 linhas
5 fps	32,6% 216x137	28,0% 188x140
10 fps	16,3% 152x96	14,0% 140x96
15 fps	10,8% 124x78	9,4% 116x85

Tabela 5.3 - Janelas de vídeo com áudio C-*stereo* ou B-*mono*

Áudio B-*stereo* ou A-*mono*:

Nº de Imagens por Segundo (fps)	Monitores de 525 linhas	Monitores de 625 linhas
5 fps	27,6% 200x126	23,8% 186x136
10 fps	13,8% 140x89	11,9% 132x96
15 fps	9,2% 116x72	7,9% 104x78

Tabela 5.4 - Janelas de vídeo com áudio B-*stereo* ou A-*mono*

Áudio A-*stereo*:

Nº de Imagens por Segundo (fps)	Monitores de 525 linhas	Monitores de 625 linhas
5 fps	19,0% 164x104	16,2% 152x112
10 fps	9,5% 116x73	8,1% 108x79
15 fps	6,3% 96x60	5,4% 88x65

Tabela 5.5 - Janelas de vídeo com áudio A-*stereo*

5.3.5 VÍDEO DIGITAL (DV) EM MEDIAMOGUL

O MediaMogul permite a inclusão de ficheiros com vídeo digital (*DV - Digital Video*) nas aplicações. O conjunto das rotinas que constituem as *DV Plug-Ins* permitem ainda o controlo dos dois planos de vídeo durante a apresentação de um segmento de vídeo digital, sendo possível ao utilizador activar ou desactivar cada um dos planos, independentemente do outro, e definir cores ou instruções *matte* para criar transparências entre os vários planos de imagem [OPTIMAGE93].

Usando transparências podem-se construir menus com botões de interactividade sem obscurecer o plano de vídeo digital, mantendo a funcionalidade dos botões; inclusivamente a capacidade de saltar de um script para outro. Podem ser definidos pelo autor um conjunto de botões especiais que permitem o controlo da apresentação dos segmentos de vídeo digital. A inclusão do vídeo digital no MediaMogul empobreceu as características de simulação e emulação deste sistema de autoria.

Emulação

Antes de usar o *Script-to-Disc* para criar uma imagem de disco com um segmento que contenha vídeo digital é necessário copiar para o disco magnético que contém o MediaMogul os ficheiros correspondentes aos segmentos de vídeo digital. A aplicação só pode ser visualizada depois de criada a imagem de disco, sendo esta restrição muito importante na definição do tempo que demora a criar um protótipo, dado que o desenvolvimento do protótipo é "feito no escuro" e os testes só podem ser realizados depois de criada uma imagem de disco [OPTIMAGE92b].

A rotina que se encarrega de iniciar o segmento de vídeo tem de ser chamada seleccionando a opção *Not Ready* (secção 6.1) dado que o material vídeo ainda não se encontra disponível.

Simulação

Caso se pretenda executar um ficheiro de vídeo digital directamente a partir do MediaMogul é necessário que o respectivo segmento de vídeo digital seja gravado num disco WORM, para ser acedido directamente pelo leitor de CD-i. Sem o ficheiro gravado no CD não é possível visualizar um segmento de vídeo digital directamente a partir do MediaMogul, mas continua a ser possível criar uma aplicação usando o respectivo ficheiro de vídeo digital e construir uma imagem de disco com o *Script-to-Disc* (secção5.3.6).

Criar uma aplicação com DV

Para criar uma aplicação usando as subrotinas de vídeo digital para o MediaMogul deve-se seguir o procedimento da figura 5.9.

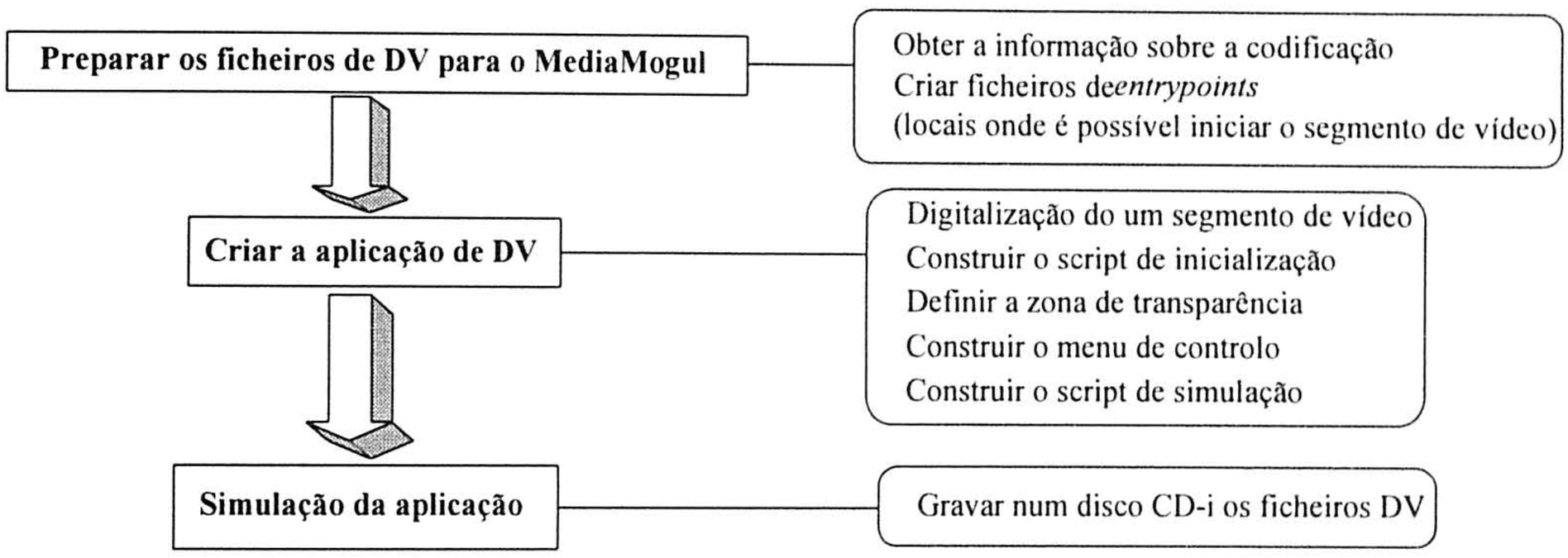


Figura 5.9 - Criar uma aplicação DV em MediaMogul

5.3.6 *SCRIPT-TO-DISK* (S2D)

O sistema de autoria MediaMogul necessita de software que permita converter a aplicação (conjunto de scripts) num título CD-i. A primeira fase é criar a imagem do disco CD-i de modo que a aplicação possa ser testada no emulador. Para esse efeito recorre-se ao ***Script-to-Disc*** (Script2Disc ou S2D). A **imagem do disco CD-i** é um ficheiro que contém toda a informação que será gravada no disco CD-i, mas que está armazenada num disco magnético de modo que seja prático realizar alterações ao seu conteúdo (não se pode apagar a informação num disco CD-i). A imagem do disco criada pelo S2D pode ser testada num emulador para efectuar as verificações finais antes da fase de impressão massiva dos CDs.

Para executar o S2D é necessário ter o seguinte equipamento; **software** MediaMogul, Script-to-Disc e um editor de texto (ex: *edt, memacs, vi)*; **hardware** disco rígido de 2 GB (SCSI-1), sistema capaz de correr o software S2D, um terminal VT100 ou qualquer programa que

realize uma emulação de terminal sobre uma máquina (ex: ***Red Ryder*** ou ***White Knight*** sobre um Macintosh, ***procomm*** ou ***telix*** sobre um PC ou ***tipcu*** sobre SUN), cabos de ligação, terminadores, teclado, rato, emulador e leitor de CD-i (para testar a imagem gerada pelo S2D), uma *tape* (com o software de controlo da unidade) para armazenar as imagens de disco (*BackUp*) e permitir o seu transporte. Opcionalmente pode-se ainda instalar um sistema *WORM*, que permite queimar um disco CD-i a partir de uma imagem de um disco já testada [**PHILIPS92b**].

O pacote de software ***Script-to-Disc*** inclui os seguintes programas:

- **cdbuild** - Lê scripts, menus e ficheiros de dados, criando os scripts que são usados directamente pelo *rtrb* e pelo *master*;
- **rtrb** - Cria um ficheiro de tempo-real (entrelaçado) para cada script;
- **master** - Agrupa os ficheiros de tempo-real e outros dados numa imagem de disco.

5.3.6.1 Espaço de Armazenamento em Disco

O disco magnético onde se está a desenvolver a aplicação com o MediaMogul deve ter espaço para armazenar o software S2D, os ficheiros de tempo-real e a imagem final do disco. Os elementos que constituem a aplicação estão gravados duas vezes no disco e por isso, a imagem do disco CD-i ocupe mais de metade do tamanho do disco duro, será necessário instalar e correr o software S2D num outro disco.

A imagem de um disco CD-i é substancialmente maior do que o espaço ocupado pelos elementos da aplicação em MediaMogul. Tipicamente uma aplicação realizada em MediaMogul com 40 MB (todos os ficheiros que constituem a aplicação) é transformada numa imagem do disco de 228 MB. Este aumento é devido ao entrelaçamento nos ficheiros de tempo-real.

O tamanho final da imagem do disco depende da natureza dos scripts (tempo, qualidade do som, utilização de ficheiros de tempo-real compactos ou normais). É necessário espaço para o software do MediaMogul mais os elementos da aplicação (som e vídeo), os scripts, mais o dobro do espaço da imagem final. Para uma aplicação de 44 MB em MediaMogul é necessário o seguinte espaço de disco:

44 M	**Aplicação em MediaMogul**
2 M	Software S2D
2*228 M	Executar o S2D
Total: 502 M	**Pode ser distribuído no máximo por 3 discos duros.**

A configuração que se aconselha para o desenvolvimento de aplicações em MediaMogul e utilização do S2D consiste numa instalação de um disco de 1 GB (MediaMogul + Todos os ficheiros associados a uma aplicação) e um disco de 2 GB (S2D e ficheiros intermédios gerados pelo S2D que constituem os passos necessários até à obtenção final dos ficheiros em tempo-real).

5.3.6.2 PLATAFORMAS DE DESENVOLVIMENTO

Existem duas plataformas sobre as quais se pode executar o S2D. A escolha de uma delas depende do hardware associado ao CD-i, da rapidez com que o software será executado e do número máximo de scripts que podem ser processados.

1º Plataforma - EMULADOR

Se o S2D for executado sobre o emulador, isso permite diminuir bastante o tempo de execução mas a sua utilização encontra-se limitada até cerca de 1400 scripts. O S2D demorou aproximadamente 3 horas a gerar a imagem do disco (228 MB) correspondente a uma aplicação com 40 MB em MediaMogul [OPTIMAGE92b]. O emulador pode estar isolado ou ligado a uma máquina (Mac, PC ou SUN) no caso de também estar ligado a um disco formatado em OS-9.

2ª Plataforma - CD-i (5 MB - RAM)

O S2D pode ser executado num leitor CD-i desde que tenha uma expansão de memória de pelo menos 5 MB (RAM), permitindo a construção de imagem de discos que utilizem até 4000 scripts e com uma velocidade de execução muito menor que o caso anterior. Por exemplo, um projecto demorou **30 minutos usando o S2D no emulador e 8 horas no CD-i**. É possível usar um leitor CD-i com 2 MB (RAM) mas a velocidade de execução ainda é menor e permite apenas criar imagens de disco de aplicações com 1400 scripts. Actualmente ainda não existe solução para as aplicações com mais de 4000 scripts. Contudo, já foi desenvolvido um protótipo para uma nova versão do S2D que permite ultrapassar esta limitação e aumentar cerca de 4 vezes a velocidade de execução.

Ligações do Hardware

O **emulador** para correr o S2D necessita de pelo menos um disco duro onde esteja armazenado o próprio S2D e onde serão gravados os ficheiros de tempo-real intermédios e a imagem do disco. A experiência pessoal do autor levou-o a concluir que o tamanho dos ficheiros intermédios (ficheiros de tempo-real) ocupam mais espaço do que aquele que é indicado no manual. O manual aconselha um disco cuja capacidade seja o dobro do tamanho da imagem do disco, mas constatou-se que isso não era suficiente e foi mesmo necessário instalar um disco com o quádruplo do tamanho da imagem do disco (2 GB), para não ter qualquer problema na execução do S2D. Só desta forma foi possível armazenar o MediaMogul, o S2D, todos os elementos da aplicação, os scripts, os ficheiros de tempo-real e a imagem do disco.

Para **gravar** um CD-i é necessário ligar ao emulador um sistema *WORM*. Este sistema permite uma e só uma gravação de um disco CD-i, tal como o seu nome indica (*Write Once Read Many*).

Por último é necessário um leitor CD-i 605, que ligado ao emulador, permite executar a imagem de disco criada pelo S2D.

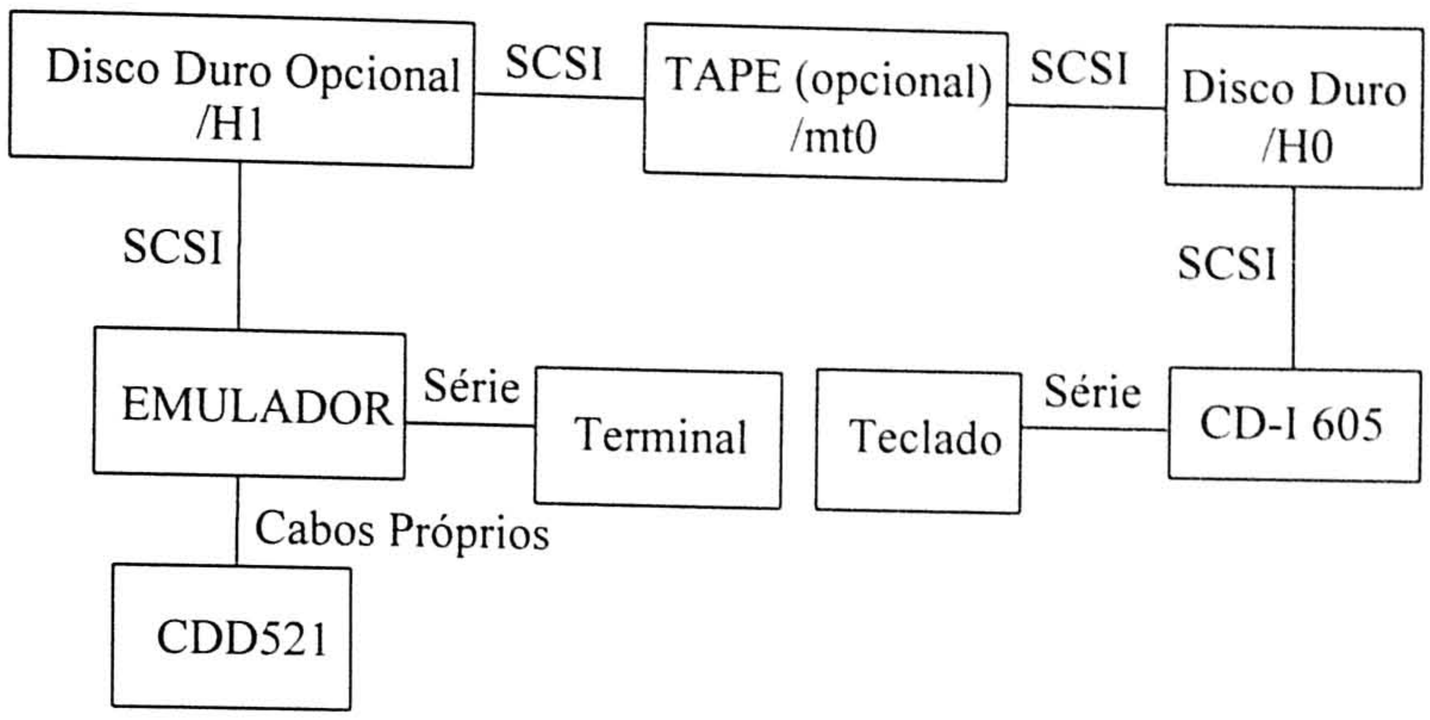


Figura 5.10 - Ligações do Emulador e do WORM

O emulador necessita de ter ligado um terminal ou uma máquina com software que permita a emulação de um terminal. A ligação série é estabelecida a 19200 *baud*, 8 bits de dados, sem paridade e um *stop bit*. Tipicamente será realizada uma emulação de um terminal VT100, usando os cabos apropriados [OPTIMAGE92b].

5.3.6.3 CONSTRUÇÃO DE UM DISCO CD-I

Depois de todos os elementos vídeo e áudio estarem adquiridos e convertidos para os formatos aceites pelo MediaMogul, num disco duro formatado pelo MediaMogul, são criados os scripts que constituem os vários segmentos da aplicação. Concluída a aplicação deve-se seguir a análise de cada script segundo as restrições necessárias à construção da imagem do disco. Este tipo de análise é realizada automaticamente pelo MediaMogul, sendo apenas necessário seleccionar a opção *Analysis*. O disco formatado pelo MediaMogul é directamente ligado ao emulador onde o S2D cria automaticamente a imagem correspondente ao título CD-i. Esta imagem do disco pode ser emulada para se verificar se a aplicação está isenta de erros; posteriormente será usada para servir de modelo para gravar o primeiro disco CD-i [VROONHOVEN92].

Para gerar a imagem do disco, usando o S2D, é necessário chamar o comando *cdbuild* passando-lhe como parâmetro o nome do script inicial da aplicação [OPTIMAGE92a]. O *cdbuild* começa por analisar todos os ficheiros com scripts e menus, para determinar os ficheiros de áudio e vídeo que serão necessários, verifica a sua presença e o seu tamanho, verifica se o *rtrb* pode gerar os respectivos ficheiros de tempo-real e caso não existam erros vai criar os *rtrb-scripts* necessários para o *rtrb*. Se nesta fase ocorrerem erros devem ser verificados os scripts respectivos usando o comando *analyse* do MediaMogul.

Assim que esta etapa esteja completada e isenta de erros, o *cdbuild* chama o *rtrb* que executa os *rtrb-scripts* gerados anteriormente e desta forma são criados os ficheiros de tempo-real que vão fazer parte da imagem do disco. O processo a partir deste ponto é completamente automático.

5.3.7 *MAC IMAGINATION PACK*

A OptImage anunciou recentemente o lançamento do pacote de software e hardware *Mac Imagination Pak*, tratando-se do primeiro sistema de autoria (hardware e software) para CD-i, funcionando totalmente sobre a plataforma Mac. Este pacote consiste numa placa *NuBus* e no software MediaMogul.

A placa *NuBus* emula as funções de um leitor CD-i, permitindo ao Mac reconhecer qualquer aplicação CD-i a partir da unidade de CD-ROM. Esta placa, usada em conjunto com o MediaMogul, permite a visualização e emulação das aplicações numa única plataforma, diminuindo consideravelmente o tempo de desenvolvimento de um título CD-i. Tradicionalmente, era necessário transferir o material (imagens, animações, música, ...) criado no Mac para um leitor CD-i antes de criar a aplicação com o MediaMogul. Com o *Mac Imagination Pak* é eliminado o tempo gasto na transferência destes ficheiros, possibilitando o acesso aos recursos disponibilizados pelo Mac, tais como partilha de ficheiros e acesso a uma rede.

Para instalar este pacote é necessário um Mac de 32 bits IIci ou superior, com o sistema 7. Está previsto o lançamento de uma versão para PC, mas só para Agosto de 1995.

5.4 HAL-90

Trata-se da primeira linguagem de autor de alto nível para aplicações CD-i. Este sistema de autoria é totalmente compatível com a norma do CD-i (*Green Book*), permitindo o desenvolvimento rápido de protótipos de aplicações CD-i. Este sistema encontra-se disponível para as plataformas SUN, Mac e PC-WINDOWS e surgiu da necessidade de libertar o utilizador do conhecimento do hardware CD-i, e do estudo do sistema operativo CD-RTOS.

O Hal-90 disponibiliza um conjunto de primitivas multimédia interactivas, chamadas **IM-Models (*Interactive Multimedia-Models*)**, que permitem a quem está a conceber uma aplicação implementar protótipos sem se preocupar com os detalhes de baixo-nível. O utilizador de uma aplicação CD-i interactua navegando pelas estruturas (árvores e redes) definidas com IM-Models de acordo com a complexidade necessária.

Este sistema de autoria ajuda nas várias etapas que levam à criação de um disco CD-i, através de um conjunto de ferramentas:

- ***User Interface*** - ambiente amigável que permite ao utilizador aceder às diferentes ferramentas do Hal-90, de modo a facilitar o desenvolvimento de aplicações CD-i;
- ***Compiler*** - esta ferramenta lê um ficheiro (tipo ASCII) produzido pelo utilizador e cria os módulos de objectos de acordo com o formato TLV (*Type Length Value*);

- ***Linker*** - permite agrupar os ficheiros dos módulos de objectos compilados que constituem a aplicação resolvendo todas as referências aos dados existentes no disco. Para esse efeito o *linker* utiliza um mapa do disco, gerado pelo programa *premaster*, que assegura que os dados armazenados no disco estão em conformidade com o *Green Book*.

O resultado final da escrita, compilação e "linkagem" constitui o código a ser executado no leitor do CD-i por um CPU virtual. Este **CPU virtual** é o programa que realiza a gestão dos recursos de um leitor de CD-i, por exemplo optimiza os tempos de pesquisa no disco à custa de uma leitura antecipada das imagens que são mais frequentemente utilizadas.

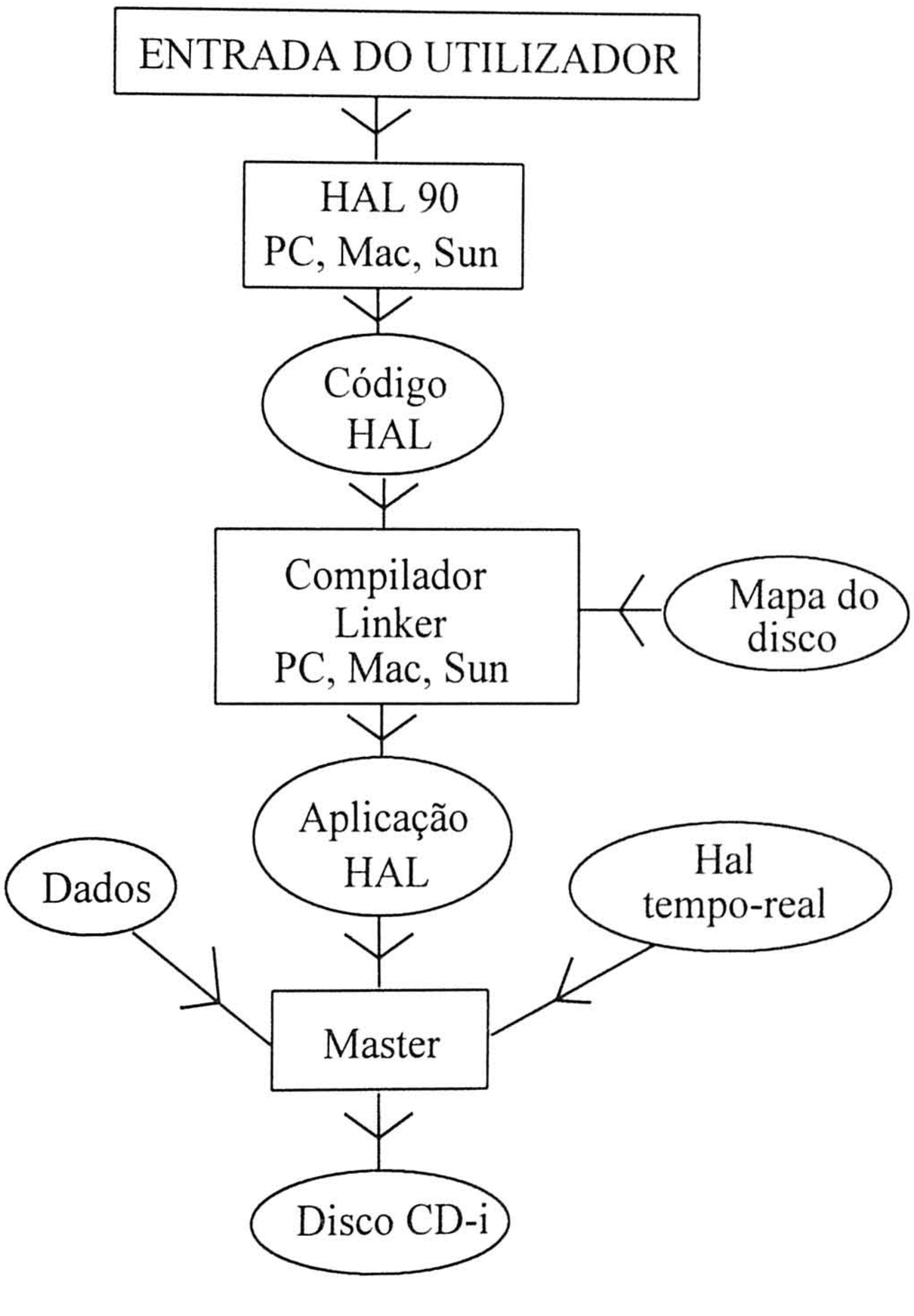


Figura 5.11 - Realização de um título em HAL-90

Este sistema de autoria foi utilizado para criar vários títulos CD-i, estando já disponíveis no mercado aplicação que fazem a apresentação das cidades de Veneza e de Lisboa.

A simplicidade do HAL-90 advém de uma limitação das capacidades do CD-i, restringindo-se às funcionalidades estritamente necessárias ao desenvolvimento de títulos CD-i. As primitivas do HAL-90 são destinadas ao desenvolvimento de títulos do tipo: apresentação de cidades, produtos e bases de informação.

As principais funcionalidades do HAL-90 são:

- Descrição da aplicação através de uma interface simples e eficiente;
- Compatibilidade total com o *Green Book*;
- Criação rápida de protótipos de aplicações;
- Acesso imediato à tecnologia CD-i;
- Redução dos custos e do tempo de desenvolvimento das aplicações;
- Descrição dos principais objectos envolvidos na criação de uma aplicação;
- Possibilidade de expansão recorrendo a rotinas desenvolvidas em linguagem C;
- O autor não necessita de ter conhecimentos sobre ciências da computação.

5.5 GRIFFIN TOOLS

Em Novembro de 1994 foi apresentado o sistema de autoria constituído pelas ferramentas *Griffin* (*GT-Griffin Tools*). Este sistema baseia-se na compilação de scripts de alto nível (usado o compilador de scripts **GTC**) para formatos que posteriormente são interpretados no leitor CD-i pelo ***GT runtime interpreter***.

Trata-se de um sistema de autoria com uma interface pouco amigável (baseada em comandos) mas com uma linguagem simples e fácil de dominar. Algumas das **vantagens** desta abordagem para a programação de aplicações CD-i:

- Programas mais pequenos relativamente aos equivalentes em C;
- Cada instrução é testada antes de ser executada, diminuindo a probabilidade de ocorrência de erros, mas à custa de perda de rapidez;
- A sintaxe é simples e clara para quem esteja familiarizado com o ambiente CD-i;
- Algumas das capacidades do CD-i, particularmente as que estão associadas a uma programação *multi-threading*, requerem uma programação muito elaborada usando C, mas, como estas ferramentas têm já desenvolvido um vasto leque de primitivas de programação, é minimizado o esforço que o programador tem de despender;
- A programação bem como a compilação podem ser realizadas directamente sobre o leitor de CD-i, não sendo necessário reconstruir a imagem de disco sempre que se realiza uma alteração.

O compilador deve ser invocado a partir da *shell* OS-9 (**gtc** *exemplo1.s* -d -e -o=exemplo1) e operar sobre o script (*exemplo1.s* - extensão **.s** não é obrigatória mas ajuda na legibilidade) sendo o resultado enviado para o ficheiro exemplo1. Caso não ocorram erros durante a fase de compilação é necessário interpretar o script compilado (exemplo1), ou seja, executar o script que previamente foi compilado (**gt** exemplo1) e a aplicação passa a ser executada no leitor de CD-i. As opções passadas para o **compilador GTC** são:

-o	=	<nome_do_ficheiro_de_destino
-d		Envia as mensagens para o terminal (Mac ligado à porta série)
-e		Envia um relatório sobre os erros de compilação para o terminal

5.5.1 PROGRAMAÇÃO CONCORRENTE

O exemplo seguinte permite mostrar com que facilidade é possível lançar processos concorrentes para serem executados nos leitores CD-i.

Este pequeno programa lança dois relógios (contadores) independentes. Os relógios despoletam a execução de duas subrotinas (*Ping e Pong*), em que cada uma delas imprime a respectiva mensagem. Como os relógios são inicializados com um intervalo de 1 segundo e disparam de dois em dois segundos, as mensagens associadas às rotinas aparecem alternadamente no terminal. Um terceiro relógio é inicializado após 10 segundos e só vai ficar activo uma única vez, tendo por função despoletar uma subrotina que apresenta a mensagem de aviso que o programa terminou e o programa chega ao fim.

Exemplo:

```
Start:
  every 2 seconds go to Ping
  wait 1 second
  every 2 seconds go to Pong
  after 10 seconds go to AllDone
  stop
Ping:
  message "Ping"
  stop
Pong:
  message "Pong"
  stop
AllDone:
  message "O Programa Terminou"

  exit
```

A instrução *stop* termina a execução do *thread* actual, mas o interpretador pode continuar a executar qualquer outro *thread* que ainda esteja activo. O programa termina a sua execução porque todos os contadores chegaram ao fim e todas as *threads* encontraram instruções stop.

5.5.2 EXEMPLO: VISUALIZAÇÃO DE IMAGENS

O exemplo que será apresentado de seguida, mostra com é fácil construir uma aplicação CD-i para mostrar uma imagem DYUV no ecrã.

No início do script é declarada uma variável do tipo *picture*, ficando definido um local para armazenamento da imagem que se pretende visualizar. Em seguida define-se o tipo de codificação da imagem e em que plano será colocada, usando a instrução *allocate*.

Em primeiro lugar o ICF (*Image Contribuition Factor*) da imagem é colocado no mínimo e activado o ecrã usando a instrução *show screen*. A execução das instruções anteriores levam o ecrã de azul para preto. A instrução *show PicA* coloca a imagem na primeira posição da lista (estrutura interna) de imagens associadas ao ecrã. A imagem é visualizada no ecrã mas sem conteúdo e sem brilho. Em seguida é lido o respectivo ficheiro de imagem (IFF) para a PicA. Finalmente foi

carregada uma imagem na estrutura respectiva, mas para que seja possível a sua visualização é necessário realizar um efeito de vídeo (*fade up*). Por último, após 5 segundos a imagem sofre o efeito *fade down* e o programa termina a sua execução.

Exemplo:

```
! Declarar a variável usada no script
  picture PicA
! Alocar as variáveis
Start:
  allocate PicA plane A coding DYUV
! Carregar uma imagem DYUV e fazer a sua apresentação no ecrã
  set icf of PicA to MIN
  show PicA
  show screen
  message "Carregar a imagem DYUV"
  load PicA from IFF "IMAGEMS/feup.dyu"
  fade up PicA then continue
  message "Espere 5 segundos"
  wait 5 seconds
Exit:
  message "Fim do programa"
  fade down PicA
  wait 2 seconds
  exit
```

5.6 BALBOA

O sistema de autoria Balboa (*Balboa Runtime Environment*) foi desenvolvido para ajudar **programadores avançados**. A programação de aplicações CD-i profissionais requeria programadores com uma vasta experiência, recorrendo-se habitualmente aos especialistas de software que estiveram envolvidos na especificação do sistema CD-i. Com vista a reduzir a complexidade da programação a Philips desenvolveu o Balboa, que se pode caracterizar por possuir um conjunto de bibliotecas de funções prontas para serem utilizadas [PIESING92]. A origem do Balboa está intimamente ligada a uma tentativa de optimização dos sistemas de autoria para CD-i. Trata-se do primeiro sistema de autoria da segunda geração, cobrindo todo o espectro da programação para CD-i: áudio, vídeo, interactividade com cursor e gestão de aplicações executada em tempo-real.

Os objectivos do Balboa eram:

- Proporcionar um interface entre a equipa de concepção e a equipa de programação, permitindo ao programador concentrar-se sobre a funcionalidade final da aplicação em vez de se preocupar com a complexidade do hardware do CD-i;
- Diminuir a dificuldade da concepção de um ambiente interactivo, onde as acções do utilizador são imprevisíveis, proporcionando um ambiente virtual de programação.

O Balboa é um ambiente de programação flexível e transparente, permitindo aos programadores mais avançados escreverem código directamente sobre o CD-RTOS e posteriormente combinar essas funções com as funções já existentes no Balboa. Desta forma é

facilitada a reutilização de software, de modo que o investimento no desenvolvimento de um título pode ser aproveitado para o próximo.

O Balboa funciona como uma camada superior do sistema operativo CD-RTOS aliando as potencialidades de um ambiente de programação por eventos (*event-driven*) a um elevado grau de abstracção das estruturas de dados [**RIBEIRO95**].

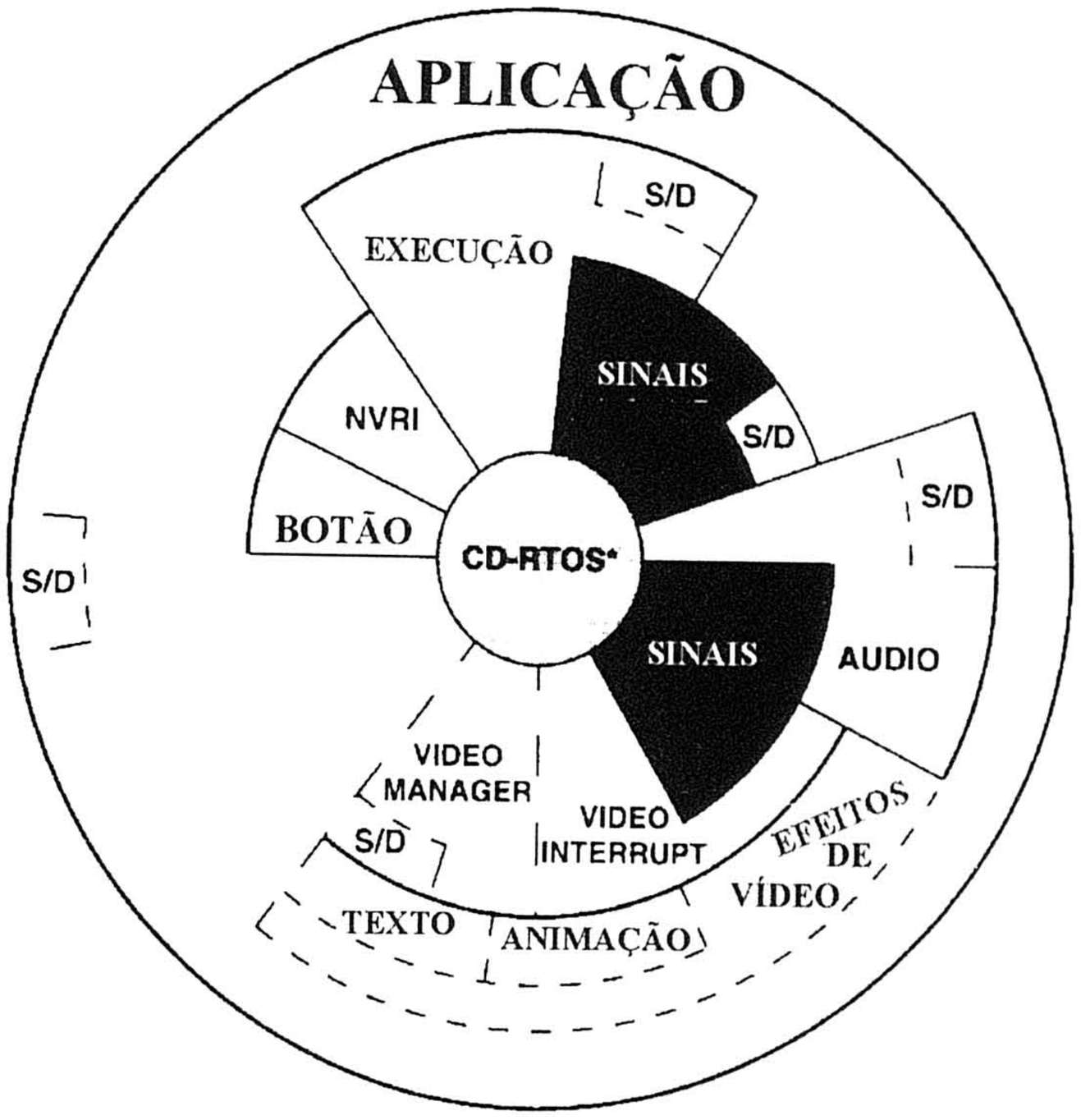


Figura 5.12 - Arquitectura do Balboa

A **arquitectura** do Balboa assenta no motor (*engine*) ***dispatch*** que se encarrega de activar as funções que implementam a aplicação, apoiando-se no motor ***signal*** para processar os sinais. Assim temos uma arquitectura de programação baseada em eventos; por exemplo, um sinal provocado pelo movimento do rato pode fazer activar uma função ou um sinal de alarme do hardware pode interromper a execução de uma função [**OPTIMAGE94b**].

As funções do Balboa estão agrupadas em 16 módulos, permitindo correr mais do que um programa lógico num dado instante, sendo esta a principal vantagem desta arquitectura modular. As tarefas implementadas pelos 16 módulos são:

- Tratamento de sinais CD-RTOS, contadores temporizados;
- Gestão do ecrã;
- Funções de interrupção para o vídeo;
- Tratamento de imagem;
- Efeitos de vídeo;
- Gestão do áudio;
- Manipulação de texto;

- Animações gráficas;
- Armazenamento de elementos multimédia em memória;
- Gestão do cursor;
- Gestão e configuração dos botões de interactividade;
- Gravação de dados inseridos pelo utilizador;
- Controlo da interactividade;
- Gestão da NVRAM;
- Relatório sobre o estado do sistema.

5.7 CORE

O sistema de autoria CORE (*CD-i Object-oriented Runtime Environment*) é um ambiente de desenvolvimento de aplicações CD-i **orientado aos objectos**, destinado a ajudar programadores avançados. O sistema CD-i realiza determinadas acções sobre certos objectos, então para se obterem aplicações mais flexíveis e reutilizáveis é melhor basear a estrutura do software nos objectos do que nas acções [PICOLET93].

A utilização de técnicas orientadas aos objectos no desenvolvimento de aplicações multimédia revela certas características importantes, das quais se destacam:

- Desenvolvimento mais rápido de código uniforme e reutilizável directamente a partir da análise dos modelos de dados orientados aos objectos;
- Uma solução geral e eficiente para a comunicação assíncrona de eventos;
- Um apoio integrado à correcção de erros (*debugging*) que inclui uma procura de erros em cada classe, por cada evento e por cada linha de controlo (*thread*);
- Um suporte para alguns dos mecanismos tradicionais orientados aos objectos implementados em C ANSI (e.g. abstracção, encapsulamento, herança e polimorfismo).

O CORE constitui uma nova arquitectura de desenvolvimento de aplicações multimédia em CD-i projectada com base em técnicas orientadas aos objectos. O domínio da hierarquia do CORE é semelhante ao de *Shlaer e Mellor* excepto no refinamento da camada de serviço [FAYAD93].

Os domínios específicos do CD-i que constituem as diferentes camadas são:

- **Aplicação**: normalmente é um processo OS-9 isolado - uma aplicação orientada para uma dada tarefa (e.g. simulação, base de dados, etc) ou um comando da *shell*;
- **Apresentação**: são cenas multimédia características (e.g. uma árvore de menus, um *slideshow*, uma animação, um *movie*);
- **Serviço de I/O**: existe um domínio por cada gestor de ficheiros do CD-RTOS;
- **Arquitectura do software**: um domínio que fornece o sistema genérico de serviços (e.g. gestão de memória, comunicação de eventos, gestão de estados, etc);

- **Implementação**: a implementação foi feita no sistema operativo CD-RTOS, em C ANSI e com o auxílio da biblioteca Balboa. Não é utilizada a linguagem C++ dado que é incompatível com o Balboa.

5.8 ESCOLHA DO SISTEMA DE AUTORIA

Seguidamente são apresentados um conjunto de critérios já estabelecidos como sendo os mais relevantes para a avaliação da próxima geração dos sistemas de autoria:

- Permitir uma interoperacionalidade com linguagens de programação genéricas (especialmente C) de modo a incorporar rotinas já desenvolvidas em projectos anteriores;
- Ter um conjunto de funções tais como os operadores matemáticos, primitivas para escolha entre diferentes respostas possíveis e um processo de entrada de dados de acordo com as exigências das aplicações finais;
- Integração das tarefas principais no desenvolvimento de uma aplicação (criação, edição e prototipagem);
- Suportar diferentes formatos para gráficos, captura de imagens e pacotes de animações;
- Possibilidade de importar dados proveniente de aplicações genéricas do tipo processadores de texto, bases de dados, folhas de cálculo;
- Possibilidade de exportar os dados nos formatos mais característicos;
- Qual a qualidade gráfica em termos de nº de cores disponíveis ?
- Quantos tipos de fontes de caracteres proporciona ou é capaz de suportar ?
- Verificar se não existe incompatibilidade com os sistemas que já usava ou que pretende vir a utilizar.

Independentemente da escolha final do sistema de autoria é essencial que a equipa de produção conheça o potencial e as limitações do sistema, bem como seja capaz de estimar o esforço que é necessário despender em cada uma das etapas necessárias à implementação da aplicação.

VI

SUBROTINAS PARA ESTENDER AS CAPACIDADES DO MEDIAMOGUL

As aplicações desenvolvidas usando sistemas de autoria encontram-se limitadas pelo conjunto de funções que se encontram disponíveis no sistema. Estas limitações podem ser ultrapassadas sempre que os sistemas de autoria permitam a programação numa linguagem mais genérica (independente do sistema CD-i) tal como a linguagem C, ou passar de uma linguagem de alto nível para uma de baixo nível. Este capítulo ilustra o trabalho de desenvolvimento de **subrotinas** em C, que podem ser chamadas pelo sistema de autoria MediaMogul em qualquer ponto da aplicação. As subrotinas são módulos CD-RTOS executáveis e pode-se estender as capacidades do MediaMogul através de subrotinas já existentes ou criando novas subrotinas com novas funcionalidades. É usual chamar-se a um conjunto de subrotinas *Plug-Ins*.

Uma subrotina é carregada em memória sempre que é chamada dentro de um script, sendo libertada a memória quando o script termina a sua execução ou salta para outro script. Contudo, é possível manter a subrotina sempre em memória activando o *sticky bit* (atributo dos ficheiros no sistema operativo CD-RTOS), o que pode ser muito útil na redução do tempo de transição de um script para outro.

6.1 COMO CHAMAR UMA SUBROTINA

Para chamar uma subrotina é necessário adicionar o comando *Call* ao script na coluna de controlo do programa. O comando *Call* tem associada a seguinte janela:

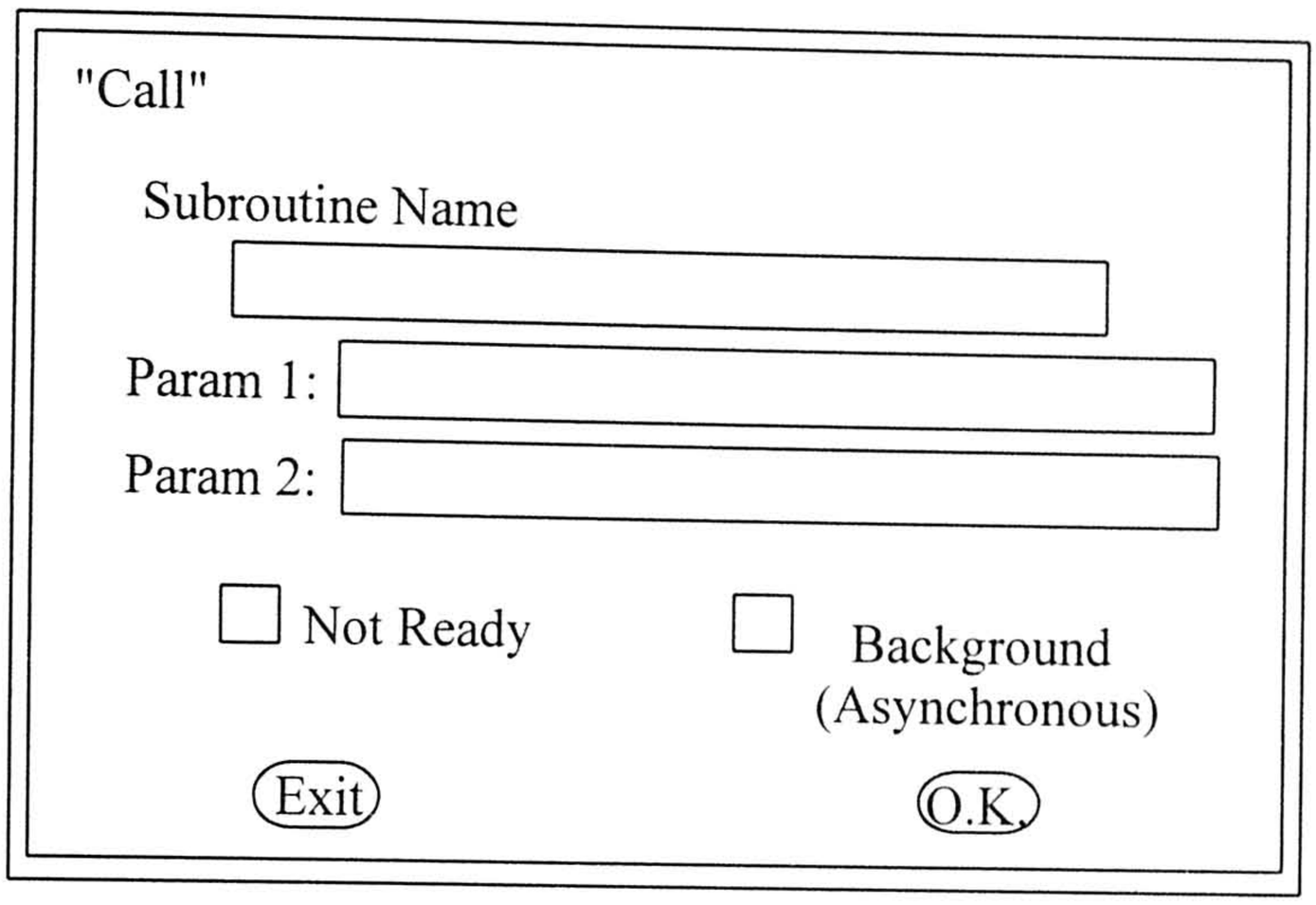


Figura 6.1 - Chamada de uma subrotina

Função de cada campo da janela de chamada de uma subrotina

Subroutine Name: indicar o nome da subrotina que se pretende usar. O nome da subrotina e o nome do módulo CD-RTOS devem ser iguais, para serem compatíveis com o comando *Call*. Pode-se usar o gestor de ficheiros do MediaMogul para realizar uma pesquisa interactiva da subrotina através da estrutura de ficheiros.

Param1: este é o primeiro parâmetro que é passado para a subrotina. Se o parâmetro tiver dois ou mais elementos deve ser colocada uma vírgula para os separar, não devendo ser usados espaços para servir de separadores.

Param2: este é o segundo parâmetro que é passado para a subrotina. Tem uma sintaxe igual ao *Param1*.

Not Ready: esta opção deve ser seleccionada sempre que uma subrotina for usada na imagem de disco actual mas não ser executada pelo MediaMogul (secção 5.3.5).

***Background*:** sempre que se pretenda chamar uma rotina de forma assíncrona, deve ser seleccionada esta opção. Isto significa que enquanto a subrotina está a ser executada, os comandos posteriores ao *Call* não ficam à espera que a subrotina termine a sua execução. As rotinas síncronas (*Foreground*) terminam a sua execução sempre que há um salto para outro script ou é executado um comando *skip*. As rotinas assíncronas continuam a sua execução.

Se um parâmetro contiver o caracter /, o S2D considera-o como um sendo um nome de um ficheiro. O ficheiro respectivo é incluído no disco, sendo passado como parâmetro para a rotina o nome do ficheiro, tendo em conta a sua localização no disco. A única excepção ocorre quando o caracter / é usado no nome de um periférico, tal como /p (impressora).

O MediaMogul tem um conjunto de subrotinas que permitem melhorar as aplicações que são realizadas com este sistema de autoria, disponibilizando um conjunto de funcionalidades de carácter profissional [OPTIMAGE93]. Para ilustrar estas funcionalidades apresenta-se um resumo da aplicação das subrotinas (o nome da respectiva subrotina aparece em *itálico*).

É possível definir a zona do ecrã em que o cursor se pode movimentar, limitando o utilizador a um espaço de interactividade menor e mais simplificado (*cdi_constrain_menu)*. O tamanho e/ou a localização da região em que se pode movimentar o cursor podem ser modificados (*cdi_set_constrain*). Podem-se definir botões irregulares (*irregular_menu*), ou seja botões que não têm uma forma rectangular e permitir activar menus sem ser visível qualquer apontador no ecrã (*no_pointer_menu*). Para preservar os écrãs que se encontram muito tempo ligados, sem que o utilizador exerça qualquer acção sobre a aplicação, é possível reduzir o brilho do ecrã (*cdi_dimmer*) após um determinado tempo. Pode-se colorir uma imagem durante a execução de uma aplicação, alterar individualmente os valores da tabela CLUT (*change_clut*) da imagem que está a ser visualizada. Pode-se fazer sair um ficheiro de texto no ecrã ou na impressora (*display_text*).

6.2 MEDIAMOGUL *PLUG-INS*

O MediaMogul tem um conjunto de subrotinas pré-compiladas que permitem executar tarefas que de outro modo não poderiam ser executadas pelo MediaMogul. Estas subrotinas são uma amostra do leque das possibilidades de desenvolvimento de novas funcionalidades para o sistema MediaMogul. As subrotinas apresentadas realizam tarefas muito específicas e básicas, em seguida apresenta-se o conjunto de subrotinas de carácter profissional (***Plug-Ins***) desenvolvidas pela OptImage e que se encontram disponíveis no mercado:

- ***Pop Up Menu Package***: permite criar uma barra com um menu de navegação que pode ser chamada através da pressão no botão do rato. Estas barras de menus podem ser transparentes, ou seja, mostrar a imagem do plano B quando o cursor entra na zona activa do botão. Ao activar os botões da barra de menus pode-se saltar para qualquer ponto da aplicação (para outro script);
- ***Run Length Packages***: este conjunto de subrotinas permite mostrar animações quando o cursor passa sobre a zona activa de um botão. As animações para serem usadas neste pacote têm de ser construídas com imagens RL7, usando a ferramenta ***anim_maker***. Usando estas subrotinas é possível fazer passar o texto no ecrã continuamente, tal como no final de um filme, onde se listam os nomes dos participantes. Permitem ainda mostrar imagens RL7 sobre qualquer outro tipo de imagem quando o cursor passa sobre a zona activa de um botão;

- ***QHY Package***: o MediaMogul não permite a visualização de imagens com dupla resolução (QHY), o que é um exemplo típico de uma limitação do sistema de autoria. Para colmatar esta lacuna a OptImage lançou o QHY Package que contém um conjunto de subrotinas que permitem a leitura de imagens com dupla resolução. As operações que se podem realizar com este tipo de imagens estão agrupadas em dois grupos: ***QHY Image Zoom*** para fazer uma ampliação a uma imagem QHY e ***QHY Highlighting Hotspots*** para usar imagens de dupla resolução nos botões de interactividade.

As subrotinas são desenvolvidas para realizar funções especiais que não fazem parte da estrutura base do sistema de autoria, destinando-se a aplicações de um determinado tipo (jogos, simulação de cor, etc). Os estúdios que desenvolvem aplicações CD-i começam a comercializar as subrotinas utilizadas nos títulos que produzem e que são suficientemente genéricas para serem reutilizadas:

- ***CD-i Test Engine:*** esta subrotina permite ao MediaMogul um título interactivo para treino de utilizadores, usando uma interface baseada em perguntas e respostas. A pontuação obtida pode ser armazenada em diskette (nos CD-i 60x) quando assim for desejado. Em conjunto com uma subrotina que implementa um teclado virtual, o utilizador pode escrever sobre respostas deixadas em branco;
- ***Base de Dados CD-i*:** é um motor de pesquisa de dados. Permite ao programador saltar para os scripts que satisfazem um critério de pesquisa fornecido pelo utilizador;
- ***Memory Game:*** trata-se de uma ferramenta que permite o desenvolvimento de aplicações que simulam a funcionalidade do jogo para crianças *"memory game"*;
- ***Paint Package:*** este utilitário permite incorporar um programa de pintura numa aplicação. Pode-se construir um título em que se altera o valor das cores na tabela de CLUT de uma dada imagem. Esta funcionalidade pode ser usada para colorir um objecto de uma dada imagem e depois saltar para uma sequência de imagens que mantêm as alterações na cor.

6.3 Gestão de Memória nas Subrotinas

O estudo para a alocação de memória das subrotinas tem por base um sistema CD-i com uma expansão de memória de 1 MB, que é o mínimo necessário para correr o sistema de autoria MediaMogul.

Como uma aplicação pode ser executada (interpretada) pelo sistema de autoria MediaMogul ou lida directamente de um disco CD-i (gerada pelo *Script-to-Disc*), as restrições para a utilização da memória são diferentes, pelo que se dividiu o estudo em dois casos:

1º CASO: aplicação a ser executada pelo sistema MediaMogul

A quantidade de memória que se encontra disponível para ser utilizada por uma subrotina (chamada pelo comando *Call* do sistema de autoria MediaMogul), que faz parte de uma aplicação que está a ser executada a partir do interpretador *Script Writer*, são 289 KB (mas o tamanho máximo de um segmento contíguo são 125 KB). Para uma aplicação que seja executada a partir do *File Director* a quantidade de memória disponível para uma subrotina são 511 KB (mas o tamanho máximo de um segmento contíguo são 248 KB).

2º CASO: aplicação a ser executada directamente de um CD-i

- Se as aplicações não utilizam ficheiros de tempo-real compactados que incluem áudio, temos disponíveis 299 KB de memória com segmentos contíguos em cada plano de vídeo com pelo menos 147 KB.
- Caso as aplicações utilizem ficheiros de tempo-real que incluem áudio do tipo Cmono, temos disponíveis 244 KB, com segmentos contíguos em cada plano de vídeo com pelo menos 120 KB.
- Para aplicações que utilizem ficheiros de tempo-real que incluem áudio do tipo Bmono ou Cstereo, temos disponíveis 190 KB, com segmentos contíguos em cada plano de vídeo com pelo menos 93 KB.
- Aplicações que utilizam ficheiros de tempo-real que incluem áudio do tipo Amono ou Bstereo deixam disponíveis 155 KB, com segmentos contíguos em cada plano de vídeo com pelo menos 75 KB.

Pode-se obter **memória adicional** para ser utilizada pelas subrotinas através do comando *FREE* (ou obtida de *buffers* de vídeo que não estejam a ser utilizados). O comando ***Free_video*** liberta um segmento adicional de memória em cada plano de 215 040 bytes. Depois de libertado este segmento adicional, não se pode aceder mais às funcionalidades de vídeo do MediaMogul até que seja executado o comando *Restore_video*.

O comando ***Free_audio*** liberta dois segmentos adicionais de memória caso seja utilizado um ficheiro de áudio de tempo-real compactado:

- Para cada Cmono 27 648 bytes
- Para cada Bmono ou Cstereo 55 296 bytes
- Para cada Bstereo ou Amono 82 944 bytes

Não se pode aceder ao som contido nos ficheiros de tempo-real de áudio, até que seja executado o comando *Restore_audio* [OPTIMAGE93].

6.4 LIMITAÇÕES DAS *PLUG-INS* DO MEDIAMOGUL

As *Plug-Ins* do MediaMogul têm de ser compatíveis com o ambiente que está definido para o MediaMogul e para o Script-to-Disc, o que na maior parte dos casos isto resulta em severas restrições no tipo de elementos que podem ser incluídos numa aplicação. As limitações das *Plug-Ins* são de vária ordem e dependem das funções que desempenham. Como um trabalho exaustivo do estudo destas limitações excederia o âmbito desta dissertação, são apresentadas algumas considerações sobre o funcionamento do pacote ***Run Length Packages*** apresentado na secção anterior.

O pacote de software *Run Length Plug-Ins* contém subrotinas para criar ligações entre botões, botões animados e deslocamento vertical dos botões. No caso específico dos botões animados que possibilitam ligar uma animação de ecrã completo a botões de um menu, cada animação tem início sempre que se entra num determinado botão e termina quando o cursor deixa de estar sobre o botão. Os elementos (ficheiros de animação com codificação do tipo *run-length*) podem ser convertidos com a ferramenta ***anim_maker***. Há no entanto uma forte restrição sobre estes elementos: o comprimento total dos ficheiros das animações de um determinado menu não pode exceder o tamanho máximo de uma imagem (aprox. 107 KB para imagens PAL). Trata-se de uma restrição típica das *Plug-Ins* do MediaMogul, pelo que a equipa que realiza o tratamento gráfico das imagens de uma aplicação deve ter muito cuidado para assegurar que as animações que produzem estão suficientemente comprimidas para não ultrapassar a memória disponível.

Caso se exceda esta restrição, o *anim_maker* retorna a seguinte mensagem de alarme:

```
Reading frame 15 this frame is 7696, and the animation buffer only had 852 bytes of room left.
Couldn't read in frame 15 of 749
```

6.5 UTILIZAÇÃO DE VARIÁVEIS EM SUBROTINAS

As variáveis do MediaMogul são globais, o que significa que as alterações realizadas dentro de um script permanecem no próximo. A maior parte das subrotinas fornecidas com o MediaMogul utilizam a variável Z para controlar a sua própria funcionalidade. Cada subrotina que utiliza a variável Z responde apenas a uma dada gama de valores e desta forma é possível activar uma opção de uma subrotina sem interferir com as outras subrotinas que se encontram activas. Por exemplo, se for colocada a variável Z com o valor 80 a subrotina *cdi_set_constrain* termina a sua execução mas se a subrotina *cdi_dimmer* estiver também activa ignora este valor da variável dado que foi construída para reagir apenas quando Z toma o valor 40. Sempre que se utilizem subrotinas num título, deve-se evitar a utilização da variável Z, que ficará reservada para uso interno das subrotinas.

As variáveis do MediaMogul são implementados como eventos CD-RTOS. É possível aceder a todas as variáveis do MediaMogul a partir de uma subrotina em C, através da utilização de chamadas ao sistema operativo CD-RTOS [MICROWARE91b]. Por exemplo, para adicionar o valor 100 à variável J é suficiente incluir o seguinte segmento de código dentro de um programa em C.

```
{
        int evento, valor;

        evento = _ev_link("J");                 /* Chamada ao CD-RTOS*/
        valor = _ev_read(evento);               /* Chamada ao CD-RTOS*/
        valor += 100;
        _ev_set(evento,valor,0);                /* Chamada ao CD-RTOS*/
        _ev_unlink(evento):                     /* Chamada ao CD-RTOS*/
}
```

O MediaMogul utiliza o ficheiro */nvr/variables* para gravar e ler os valores das variáveis. Neste ficheiro cada variável ocupa 4 bytes, correspondendo a um número inteiro com sinal de 32 bits. O MediaMogul permite definir apenas variáveis inteiras e o nome corresponde a uma letra do alfabeto; como o alfabeto internacional tem 26 caracteres, então o comprimento do ficheiro é de 4 x 26 = 104 bytes.

6.6 DESENVOLVIMENTO DE SUBROTINAS EM C

Para garantir a interoperacionalidade das duas linguagens (scripts do MediaMogul e subrotinas em C) o sistema MediaMogul tem definidos um conjunto de módulos e estruturas de dados que estabelecem a interface entre a linguagem C e as funções internas do sistema MediaMogul. A utilização da linguagem C permite aceder directamente a comandos do sistema operativo e a funções de baixo nível que controlam directamente o hardware. A ligação entre as linguagens é realizada à custa de dois ambientes: *Input Environment* e *Video Environment*. Estes ambientes definem um conjunto de estruturas e funções com capacidade de receber dados do exterior (subrotinas em C) para dentro das estruturas internas do MediaMogul.

As estruturas de dados que constituem estes ambientes estão comentadas com o objectivo de evidenciar a informação que pode ser processada pelas subrotinas, mas de um modo independente da aplicação a que se destinam.

6.6.1 *INPUT ENVIRONMENT*

Através de subrotinas é possível estender as capacidades de entrada de dados numa aplicação específica. O MediaMogul cria um módulo de dados associado a um ambiente de entrada (*input environment*) e associa um evento de entrada (*input event*) a cada subrotina criada pelo utilizador [OPTIMAGE93]. O evento de entrada permite que a subrotina se ligue e espere a ocorrência de um

determinado evento. O módulo de entrada de dados contém informação que é passada ao MediaMogul permitindo à subrotina a manipulação da entrada.

As **estruturas de dados** são apresentadas partindo das mais elementares até às mais complexas que são definidas à custa das mais simples, acrescidas de características complementares. Há um embebimento dos dados numa estrutura modular.

Estrutura de dados SHAPE:

Define a forma e a localização dos botões ou das respectivas regiões activas (zona do ecrã onde se encontra o botão).

Estrutura de dados CURSOR:

Contém informação sobre a definição do cursor, como por exemplo: coordenadas *x* e *y*, largura e altura, resolução, cursor cintilante, cor e forma do cursor. A forma do cursor pode ser alterada, mas o seu tamanho máximo encontra-se limitado a uma zona de 16 x 16 pixels.

Estrutura de dados ACTION:

Define qual a acção que deverá ser tomada quando o utilizador interactuar com o apontador do rato de uma dada forma. Pode-se associar mais do que uma acção a um botão ou região activa. É indicado qual o cursor que será apresentado (CURSOR) quando ocorre uma determinada acção. Para a aplicação saber qual será a acção a tomar, correspondente à escolha do utilizador, é passado um apontador para a respectiva função.

Estrutura de dados HOTSPOT:

Esta define quais as zonas activas de um dado ecrã, permitindo a implementação de botões. A maior parte da interactividade de uma aplicação é controlada por esta estrutura onde se indica a forma e a localização (SHAPE), uma matriz de acções associadas ao botão (ACTION), o nome do script para onde vai saltar e a possibilidade de voltar ao local anterior.

Estrutura de dados MENU:

Esta contém informação sobre os ficheiros que implementam os menus. A imagem associada ao menu não faz parte desta estrutura, dado que apenas se está a manipular o plano de botões do CD-i; para realizar a apresentação de um dado menu no ecrã será necessário chamar a função respectiva. Ao definir os diferentes botões pode ocorrer uma sobreposição das respectivas zonas activas. Nestes casos a região que permanece activa corresponde à que tem maior prioridade sobre as outras, ou seja, os botões que se encontram no início (com índices mais baixos) tem prioridade sobre os outros.

Nesta estrutura pode-se indicar quais os botões que estão activos (ACTION) e o respectivo número, a posição inicial do cursor e o respectivo cursor (CURSOR), o nome do script que será chamado caso não seja realizada qualquer acção, a possibilidade de voltar ao menu anterior e definir um *timeout* de espera pela decisão do utilizador.

Estrutura global de entrada de dados:

Esta contém um conjunto de parâmetros que permitem especificar o ambiente da entrada de dados, possibilitando associar diferentes cursores aos estados de funcionamento da aplicação (normal, zonas activas ou processamento). Esta estrutura tem um conjunto de variáveis para monitorar todos os meios de entrada de dados (cursor ou monitor sensível ao tacto). As variáveis indicam a posição do cursor, o menu que está activo e último botão seleccionado.

6.6.2 *VIDEO ENVIRONMENT*

O MediaMogul cria um módulo de dados contendo informação sobre o ambiente vídeo ao qual uma subrotina pode ter acesso, permitindo ao utilizador desenvolver as suas próprias subrotinas para estender as capacidades básicas do ambiente vídeo do MediaMogul [OPTIMAGE93].

Durante a instalação do MediaMogul no disco duro é criado o directório VIDEO_ENVIRONMENT que contém um exemplo com o código de uma subrotina e um ficheiro (**venv.h**) com definições de constantes associadas ao ambiente de vídeo. O ficheiro **venv.h** deve ser incluído em todas as subrotinas que afectam o ambiente vídeo.

O VIDEO_ENVIRONMENT tem dois estados: DISPLAY_CHANGED ou DISPLAY_NOT_CHANGED. Sempre que uma subrotina utilize as suas próprias LCTs e FCTs deve colocar o estado em DISPLAY_NOT_CHANGED antes de terminar a sua execução. Caso a subrotina utilize as LCTs e FCTs do ambiente vídeo do MediaMogul deve colocar o estado em DISPLAY_CHANGED antes de terminar a sua execução.

O **motor de vídeo** corresponde ao conjunto de software e hardware responsável pelos efeitos de vídeo que podem ser realizados pelo CD-i. O ambiente vídeo é definido por um conjunto de parâmetros que caracterizam as operações realizadas pelo motor de vídeo. O motor de vídeo tem associada uma zona da memória a cada plano de vídeo e para tal utiliza duas zonas de memória num total de quatro *buffers*, dois *buffers* por plano. Para aceder a estes *buffers* é necessário usar a matriz vmem. Os índices 0 e 2 correspondem aos *buffers* associados ao plano 0 e os índices 1 e 3 correspondem aos *buffers* associados ao plano 1.

Caso uma subrotina tenha necessidade de alterar as instruções numa LCT, devem-se seguir os seguintes procedimentos:

- As instruções sobre o método de codificação das imagens (ICM) devem ser colocadas no plano A e na coluna LCT_ICM, bem como só devem ser colocadas as instruções com os endereços em que começam os ecrãs (DADR) nas colunas LCT_DADR de ambos os planos. As outras colunas LCT podem ser usadas para outras instruções. Por exemplo, pode-se usar a coluna LCT_ICM do plano B para uma qualquer instrução. As instruções ICM não podem ser associadas ao plano B.
- Listar todas as instruções ICM e DADR por ordem crescente do número de linha na lista *lctbuf*. Podem ser inseridas instruções entre conjuntos sucessivos de instruções ICM e DADR, mas cada instrução ICM encontrada deve ter um número de linha maior do que a anterior. O mesmo se passa para as instruções DADR.

No apêndice B é apresentada a subrotina **videosc.c** que exemplifica a utilização deste ambiente. Esta rotina implementa um efeito de vídeo que realiza a deslocação de baixo para cima da imagem do plano B enquanto a imagem do plano A sai pelo topo do ecrã. Esta subrotina é chamada com um argumento onde é passado a duração do efeito.

6.6.3 COMPILAÇÃO DAS SUBROTINAS

Para tornar uma subrotina executável é necessário proceder à sua compilação criando um módulo executável pelo sistema operativo CD-RTOS. Este processo pode ser realizado sobre as distintas plataformas de programação, mas como este trabalho foi desenvolvido sobre uma plataforma Mac, passa-se a descrever os procedimento necessários para a compilação das subrotinas.

A compilação de uma subrotina é realizada automaticamente por software: o compilador interpreta as instruções que se encontram num ficheiro de texto segundo uma determinada linguagem de programação (neste caso C) e se não existirem erros nos vários módulos que constituem a subrotina, é criado um respectivo módulo executável.

O sistema operativo do Mac é diferente do CD-i, pelo que é necessário um C-*cross-compiler*, que é um compilador de C que é chamado dentro do Mac mas que produz à sua saída código para ser executado no sistema CD-i [**OPTIMAGE91**]. Os ficheiros gerados tem de ser transferidos para um disco OS-9 para serem executados pelo sistema CD-i, o que representa um inconveniente dado que se o compilador pudesse ser chamado pelo próprio sistema CD-i não era necessário a transferência de ficheiros e o resultado da compilação poderia ser imediatamente testado.

Contudo, é vantajosa a utilização de um *cross-compiler*, pois todo o trabalho de elaboração das subrotinas é executado no Mac, onde se dispõe de uma interface mais amigável e de um vasto conjunto de aplicações que ajudam na gestão de um projecto de software. Para ajudar no desenvolvimento das subrotinas foi utilizada a ferramenta MPW (*Macintosh Programming Workshop*).

Os comandos MPW necessários para uma compilação de uma subrotina (videosc.c) podem ser agrupadas numa *make_file* (videosc.make):

```
#  File:       videosc.make
#  Target:     videosc
#  Sources:    videosc.c
#  Created:    16 de Setembro de 1994
COptions     = {OS-9OPTS}
videosc ƒ videosc.make videosc.c
    xcc {Coptions} videosc.c
```

VII

DESENVOLVIMENTO DE UM PROTÓTIPO EM MEDIAMOGUL

O MediaMogul é um sistema de autoria que, pela facilidade de utilização, se pode considerar uma excelente ferramenta para a **prototipagem** de títulos CD-i. O objectivo principal do desenvolvimento de um protótipo é num curto espaço de tempo criar uma versão simplificada da aplicação final (título), sobre a qual se podem realizar testes de validação e testes de avaliação do grau de aceitação do título junto dos futuros utilizadores. Os testes de validação servem para verificar se o título que será desenvolvido pode satisfazer os requisitos definidos na fase de concepção.

Para ilustrar o processo de desenvolvimento de um protótipo usando o MediaMogul, foi implementada uma aplicação multimédia concebida pelo CMUP com o título de ***Leonardo***. Trata-se de uma aplicação interactiva sobre a vida de *Leonardo da Vinci*, podendo ser classificada como um exemplo típico de um discurso multimédia interactivo correspondente a uma apresentação.

Neste capítulo apenas se descrevem os aspectos relacionados com a implementação do protótipo; a concepção e o tratamento das imagens esteve a cargo dos restantes elementos do grupo de trabalho.

7.1 ESTRUTURA DO PROTÓTIPO

Conceptualmente, a aplicação foi dividida em **segmentos**, que se caracterizam por uma sucessão automática de imagens sincronizadas com o áudio, usando o diaporama (*slideshow*) como modelo do discurso multimédia.

A estrutura global do protótipo (figura 7.1) inicia-se com uma introdução, a que corresponde um segmento onde se faz a apresentação de *Leonardo da Vinci*. No fim da introdução o utilizador entra num menu onde pode seleccionar um de seis segmentos. Em cada um dos seis segmentos a informação é agrupada num contexto específico.

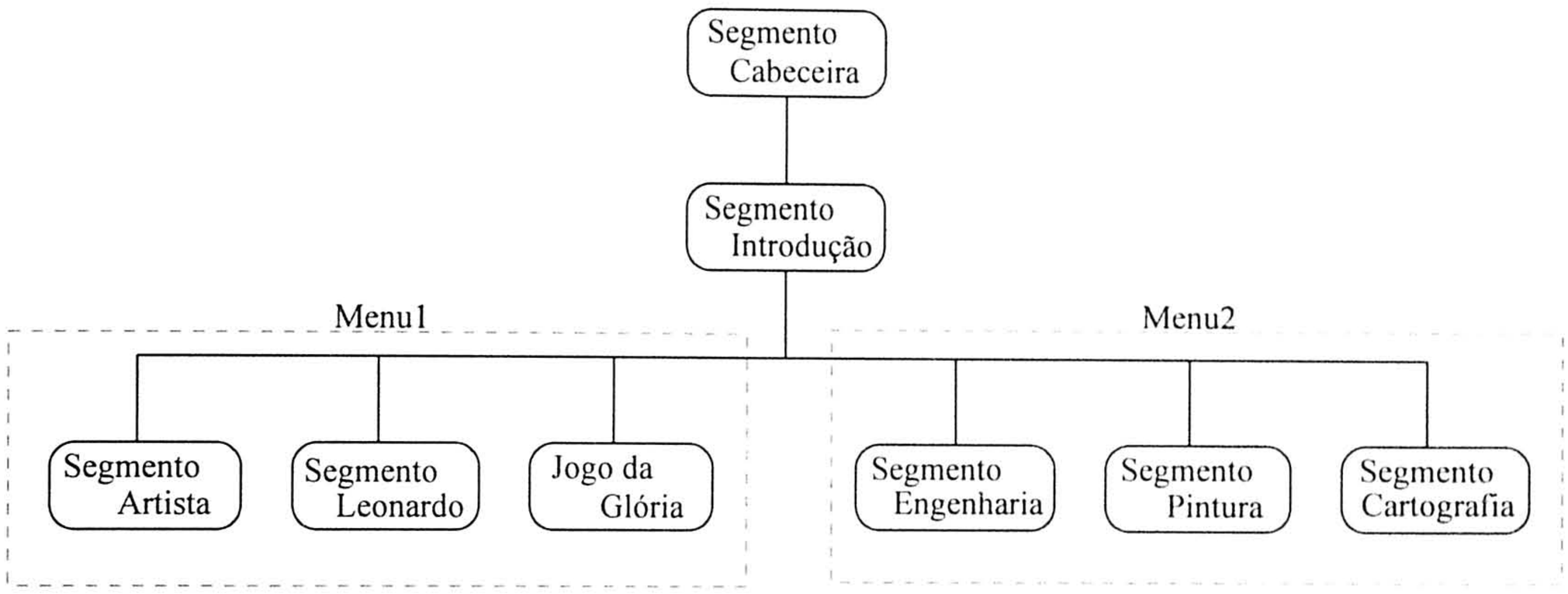


Figura 7.1 - Estrutura do protótipo Leonardo

Os segmentos

- Cabeceira - nomes dos autores do protótipo,
- Introdução - resumo sobre o conteúdo do protótipo,
- Artista - exploração da componente artística de *Leonardo da Vinci*,
- Leonardo - uma visão geral sobre a vida e obra de *Leonardo da Vinci*,
- Engenharia - a contribuição de *Leonardo da Vinci* para o avanço da engenharia,
- Pintura - análise de alguns quadros pintados por *Leonardo da Vinci*,
- Cartografia - o desenvolvimento da cartografia realizado por *Leonardo de Vinci*.

O Jogo

O estrutura deste módulo é baseada num jogo muito conhecido para as crianças, o Jogo da Glória. É apresentado um tabuleiro com casas numeradas que o jogador deve percorrer e, para tal deve lançar um dado, que se encontra sobre o canto inferior direito do ecrã, e avançar o número de casas que saiu (aleatoriamente). Então deve seleccionar o botão sobre o qual se encontra para ter acesso à informação contida na casa respectiva. É disponibilizado um pequeno texto informativo e uma imagem sobre a qual o jogador pode fazer uma ampliação.

Interactividade

Antes de iniciar a implementação de uma aplicação CD-i é importante definir o nível de interactividade que se pretende. Em seguida passa-se a apresentar as características pretendidas para a interactividade do protótipo, como foram definidas pela equipa de concepção.

Ao inserir o disco e ligar o leitor de CD-i é apresentada a cabeceira. Esta deverá ser vista uma única vez e não pode ser interrompida pelo utilizador. Posteriormente inicia-se o segmento com a introdução que termina num menu onde o utilizador pode seleccionar em qual das seis opções é que pretende entrar.

A cada uma das opções está associado um botão com uma imagem alusiva ao segmento, assim que o utilizador movimenta o cursor sobre o botão, este fica transparente, tornando visível a imagem do plano B. Caso o utilizador pretenda visualizar o segmento tem de seleccionar o respectivo botão.

Alguns dos segmentos têm uma duração considerável e, caso o utilizador não pretenda continuar a ver a sua apresentação ou tenha escolhido a opção incorrectamente, deve ser-lhe permitido interromper o segmento e automaticamente voltar ao menu anterior. Neste ponto o utilizador pode escolher novamente uma das seis opções possíveis.

Para simplificar a interface com o utilizador, as seis opções disponíveis para explorar a aplicação, estão distribuídas por dois menus, cada um com apenas três opções:

- O **menu1** disponibiliza o acesso aos segmentos: artista e leonardo, ao jogo (percurso) e ao menu2;
- O **menu2** permite voltar ao menu1 e aceder aos restantes segmentos: engenharia, pintura e cartografia. Desta forma foi possível usar botões de interactividade maiores, logo mais fáceis de localizar e seleccionar.

Se ao fim de 2 minutos o utilizador não actuar sobre os menus, é despoletado automaticamente o segmento *Leonardo*. No fim do segmento *Leonardo* volta a surgir o menu1 e o ciclo repete-se.

7.2 Desenvolvimento do Protótipo

O MediaMogul, sistema de autoria utilizado na implementação do protótipo, interpreta comandos que se podem armazenar num script. Seguidamente descrevem-se as fases de desenvolvimento do protótipo *Leonardo*, que servem também para sistematizar os procedimentos básicos para a realização de um protótipo genérico.

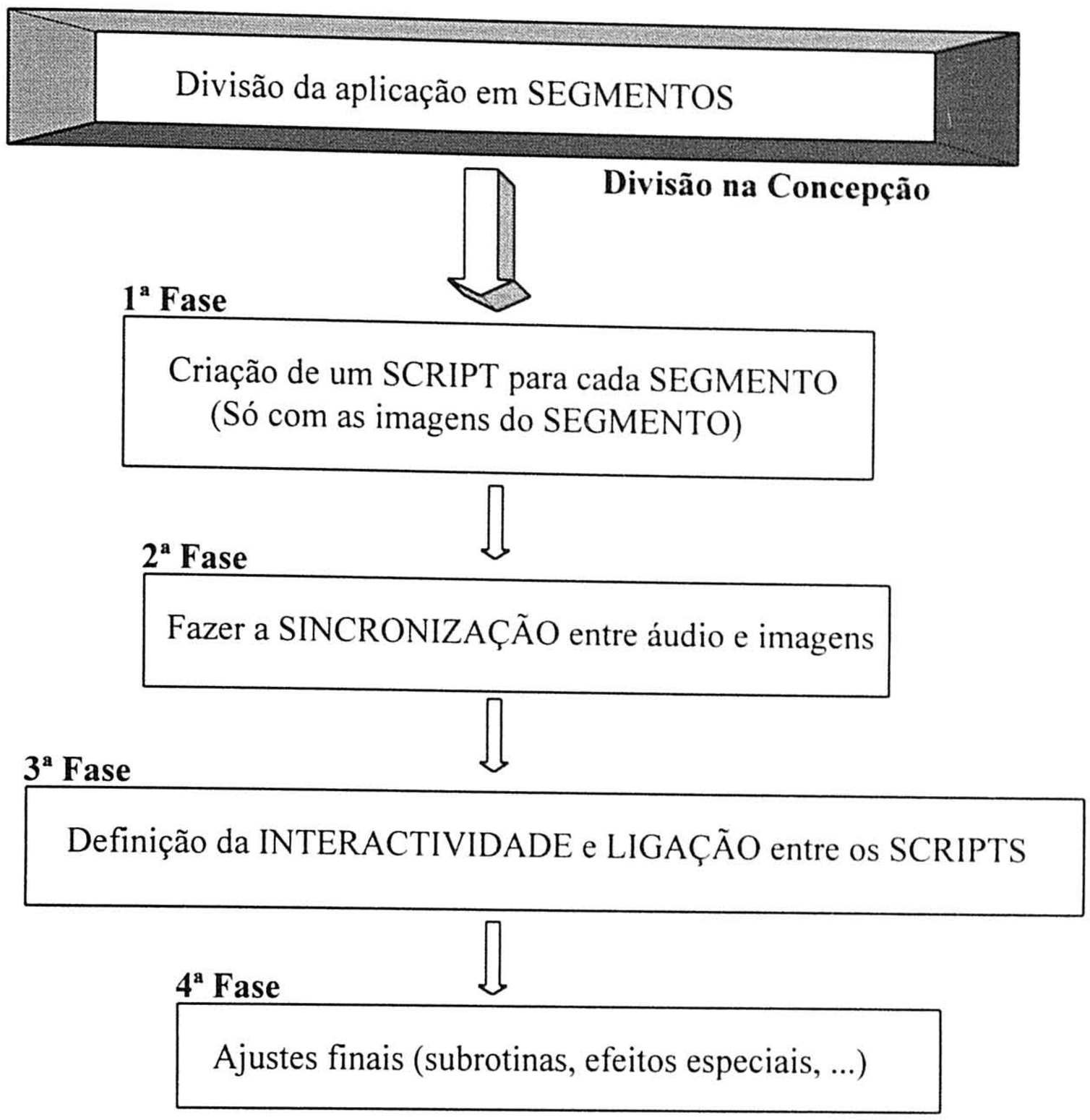


Figura 7.2 - Desenvolvimento de um protótipo em MediaMogul

A **primeira fase** da implementação consiste em criar **um script para cada segmento**. O MediaMogul só permite saltar de um segmento para o início do novo segmento; assim convém assegurar que esta restrição não entra em contradição com a especificação da aplicação. Depois de criado um segmento, não é possível aceder ao seu interior, apenas podendo saltar para o início do segmento. Logo a interactividade da aplicação pode ficar comprometida se a escolha dos segmentos não for criteriosa e não garantir que sempre que se deseja que o utilizador possa correr uma zona da aplicação tem de ser criado um novo script. Os scripts dos vários segmentos são construídos usando o editor de scripts do MediaMogul. No editor de scripts define-se a sequência de apresentação das imagens, controlando o tempo de exposição de cada uma e o efeito de vídeo que a faz surgir no ecrã. O efeito mais usado foi o *dissolve*, que realiza uma transição suave entre duas imagens: enquanto que a imagem anterior está a desaparecer começa a surgir a nova imagem. Neste ponto temos um conjunto de scripts isolados (sem qualquer ligação entre si), só com as imagens que fazem parte dos vários segmentos.

Na **segunda fase** adicionam-se os ficheiros de áudio que deveram acompanhar cada um dos segmentos, ou seja, para cada um dos scripts coloca-se o áudio respectivo. Esta fase é muito importante pois é aqui que se tem de ajustar os tempos de exposição das imagens de modo a assegurar a **sincronização** com o áudio. Uma sincronização deficiente entre imagem e som pode originar um discurso multimédia sem sentido.

Na **terceira fase** é definida a **interactividade** e as **ligações** entre os vários scripts que constituem o protótipo. Como foram estabelecidos dois menus de navegação pela aplicação, é necessário criar dois scripts onde são definidos os respectivos menus. Para criar um menu com um ou vários botões é necessário recorrer ao editor de botões do MediaMogul. Este editor permite visualizar a imagem do menu e sobre ela definir quais as zonas que vão ficar activas. Usando o rato, o autor desenha os rectângulos que definem os botões sobre a imagem de fundo, indicando as posições e os tamanhos dos botões de um modo interactivo. Depois de desenhado um botão, é apresentada uma janela com os comandos que se lhe podem associar, entre os quais se define qual é o script a ser chamado quando o utilizador premir o botão. Fazendo esta operação para cada uma das opções de navegação ficam estabelecidas as ligações dos menus aos vários scripts da aplicação. Como as opções estão dispersas por dois menus, foi incluído em cada um deles um botão que permite aceder ao outro menu, ou seja ao script onde está definido o outro menu. Para que as ligações fiquem completas é necessário que no fim de um segmento a aplicação retorne ao menu de origem do segmento, ou seja, é necessário acrescentar no final de cada um dos scripts uma instrução de salto incondicional para o script que contém o menu que dá acesso ao segmento em causa.

Na **quarta fase** fazem-se os **ajustes finais**, optimiza-se a temporização dos segmentos e acrescentam-se funcionalidades específicas recorrendo a subrotinas. Na secção seguinte é analisado com pormenor um dos segmentos ilustrativos do protótipo.

7.2.1 Implementação dos Segmentos

A funcionalidade requerida para os vários segmentos do protótipo é muito semelhante, justificando-se apenas a apresentação de um dos segmentos. O segmento escolhido foi a pintura, porque abrange os comandos mais significativos da aplicação.

```
SCRIPT PINTURA:
00:00       Menu          /h0/LEONARDO/MENUS/saltamenu2
00:20       Play          /h0/LEONARDO/AUDIO/Pintura.aas          00:00   00:00
01:10       Dissolve      /h0/LEONARDO/cdipintura/basecerto               04:00
01:10       Cursor_pos                                            707     280
09:10       Dissolve      /h0/LEONARDO/cdipintura/00ceia                  01:00
{...}
03:35:10    Dissolve      /h0/LEONARDO/cdipintura/00ceia                  04:00
03:45:10    Fade Down                                                     04:00
03:49:20    Fade Up       /h0/LEONARDO/cdipintura/mona1                   04:00
03:55:20    Menu_off
03:56:20    Skip_off
03:57:20    Pre_effect    /h0/LEONARDO/cdipintura/mona2
03:57:20    Call_bg       /h0/SUBROUTINES/videosc                 700
04:20:20    Dissolve      /h0/LEONARDO/cdipintura/mona11                  04:00
04:41:20    Skip_on
04:42:20    Menu          /h0/LEONARDO/MENUS/saltamenu2
04:43:20    Dissolve      /h0/LEONARDO/cdipintura/monacara1               04:00
{...}
05:40:20    Volume                                                220     04:00
05:45:20    Dissolve      /h0/LEONARDO/MENUS/menu2a                       02:00
05:45:20    Stop
05:47:20    Br_alwys_nr   /h0/LEONARDO/SCRIPTS/smenu2
```

Um script em MediaMogul é composto por 5 colunas:

1ª Coluna - Define-se o instante de tempo em que o comando da mesma linha é executado;

2ª Coluna - Contém o nome do comando a ser chamado;

3ª Coluna - O nome do objecto sobre o qual será realizado o respectivo comando;

4ª Coluna - Primeiro parâmetro passado ao comando (ex: tempo de espera);

5ª Coluna - Segundo parâmetro passado ao comando (ex: duração do efeito visual).

Na primeira linha é activado o **Menu** *saltamenu2*. A função deste menu é a definição de um botão correspondente a todo o ecrã, que se seleccionado pelo utilizador faz saltar a aplicação para o script que contém o menu de acesso a este segmento. Este menu é activado mas a aplicação não fica à espera de uma escolha do utilizador: caso o utilizador seleccione o botão o segmento é interrompido e aplicação volta ao menu anterior; caso o utilizador não faça qualquer acção o segmento continua a ser executado até ao final.

Na segunda linha é colocado o comando **Play** que dá início à leitura do ficheiro de áudio *Pintura.aas* (áudio ADPCM de nível A e em stereo). Este comando deve ter um esfasamento mínimo de tempo de 00:20 relativamente ao início de um script, restrição que é imposta para assegurar um bom desempenho do S2D, ou seja, para garantir que ao gerar a imagem do disco a aplicação mantém as funcionalidades observadas na simulação com o MediaMogul. Ambos os parâmetros são nulos o que significa que o áudio deve ser imediatamente iniciado e a sua reprodução só termina quando chegar ao fim.

Na terceira linha é realizado o efeito visual ***dissolve*** entre a imagem que estava no ecrã e a nova imagem (*basecerto*), com uma duração de 4 segundos (04:00).

Na quarta linha é colocado o cursor na posição definida pelas coordenadas x = 707 e y = 280, estas coordenadas são definidas usando a dimensão do ecrã em alta resolução.

De seguida há uma sequência de imagens até ao instante 3:45:10. Neste instante é inicializado o efeito ***Fade Down***, que consiste em diminuir o brilho do ecrã até se atingir um ecrã completamente preto.

Até este momento a realização do script não oferecia qualquer dificuldade de realização, porque se conseguiu satisfazer os requisitos da concepção recorrendo aos comandos básicos do MediaMogul. Na especificação do protótipo pretende-se visualizar a Mona Lisa (quadro pintado por *Leonardo da Vinci*), que dadas as suas dimensões não se ajusta ao formato do ecrã. Para resolver este problema e sem prejudicar a resolução da imagem, verificou-se que a imagem se adaptava bem a um formato com a largura normal do ecrã mas com o dobro da altura da imagem. Para tal a imagem foi cortada ao meio originando duas imagens (*mona1* e *mona2*) com as dimensões exactas do ecrã, permitindo a visualização de cada uma separadamente.

Figura 7.3 - Visualização de uma imagem 384x560

O efeito especial de transição entre estas duas imagens deveria ser um *scroll-up*, ou seja, à medida que a primeira imagem sai pelo topo do ecrã deverá entrar a segunda imagem pelo fundo, de modo que visualmente não exista um corte entre as partes que definem a imagem total. O ecrã funciona como uma janela que é deslocada sobre uma imagem com o dobro da altura do ecrã. A operação seguinte é mostrar a Mona Lisa e a solução adoptada para a resolução deste problema foi recorrer a uma subrotina especial.

A utilização da subrotina ***videosc*** (listagem no apêndice B) não é um passo trivial pelo que se descreve esta tarefa com mais pormenor. Depois de fazer surgir no ecrã a imagem *mona1* com um ***Fade Up*** e antes de chamar a subrotina, é necessário desactivar os menus que se encontram activos com ***Menu_off*** e não permitir que por pressão do rato o utilizador execute o próximo comando de salto incondicional (que se encontra no fim script) com o ***Skip_off***. Só assim é que está assegurado que o próximo comando não pode ser interrompido (não há interactividade) e então pode-se realizar o efeito de *scroll-up*. Para tal é necessário carregar a segunda parte da imagem da Mona Lisa (*mona2*) no plano B recorrendo ao comando ***Pre_effect*** e chamar a subrotina para realizar o *scroll-up*.

A subrotina *videosc* deve ser chamada em *Background*, ou seja, enquanto está a ser executada, o script continua a sua execução, mas como o comando seguinte só é chamado no instante 4:20:20 não se realiza nenhuma outra operação a não ser a própria subrotina. Caso a rotina não seja executada em *background* não há controlo sobre o comportamento final da aplicação depois de gerada a imagem do disco com o S2D. Depois de realizado este efeito especial é necessário voltar a repor o nível de interactividade anterior, ou seja, permitir a interrupção pelo utilizador com o comando ***Skip_on*** e activar novamente o menu *saltamenu2* com o comando ***Menu***.

Depois de resolvido este problema, continua a sucessão de imagens da aplicação até à última imagem da sequência e à medida que a última imagem desaparece diminui-se o volume do som, utilizando o comando ***Volume***. O comando ***Stop*** termina a reprodução sonora.

Por último o script deve saltar para um novo script correspondente ao próximo menu. Verificou-se que o salto incondicional ***Br_alwys_nr*** dá origem a um corte brusco na passagem de um script para outro. Para que o utilizador não sinta este efeito sobre a imagem, antes de se iniciar o salto carrega-se a imagem correspondente à primeira imagem que está definida no script para onde se vai saltar (*menu2a*), o que serve para enganar temporariamente o utilizador, fazendo-o pensar que o salto já se afectou. Na realidade só é realizado depois e como a imagem que é visualizada a seguir é igual aquela que já está no ecrã, não se nota qualquer perturbação na imagem.

7.2.2 IMPLEMENTAÇÃO DO JOGO

O mecanismo do jogo é bastante simples dado que o sistema de autoria MediaMogul não oferece primitivas para o desenvolvimento de jogos. Como a especificação conceptual é bastante simples, a única restrição importante é assegurar que o lançamento do dado é aleatório.

A estrutura do jogo está dividida em três módulos principais: o **tabuleiro**, o lançamento do **dado** e a selecção da **posição** em que se encontra.

O **tabuleiro** (figura 7.4) é uma imagem correspondente ao Jogo da Glória, que tem início na posição 1 e termina na posição 32, estando as várias posições agrupadas segundo uma espiral.

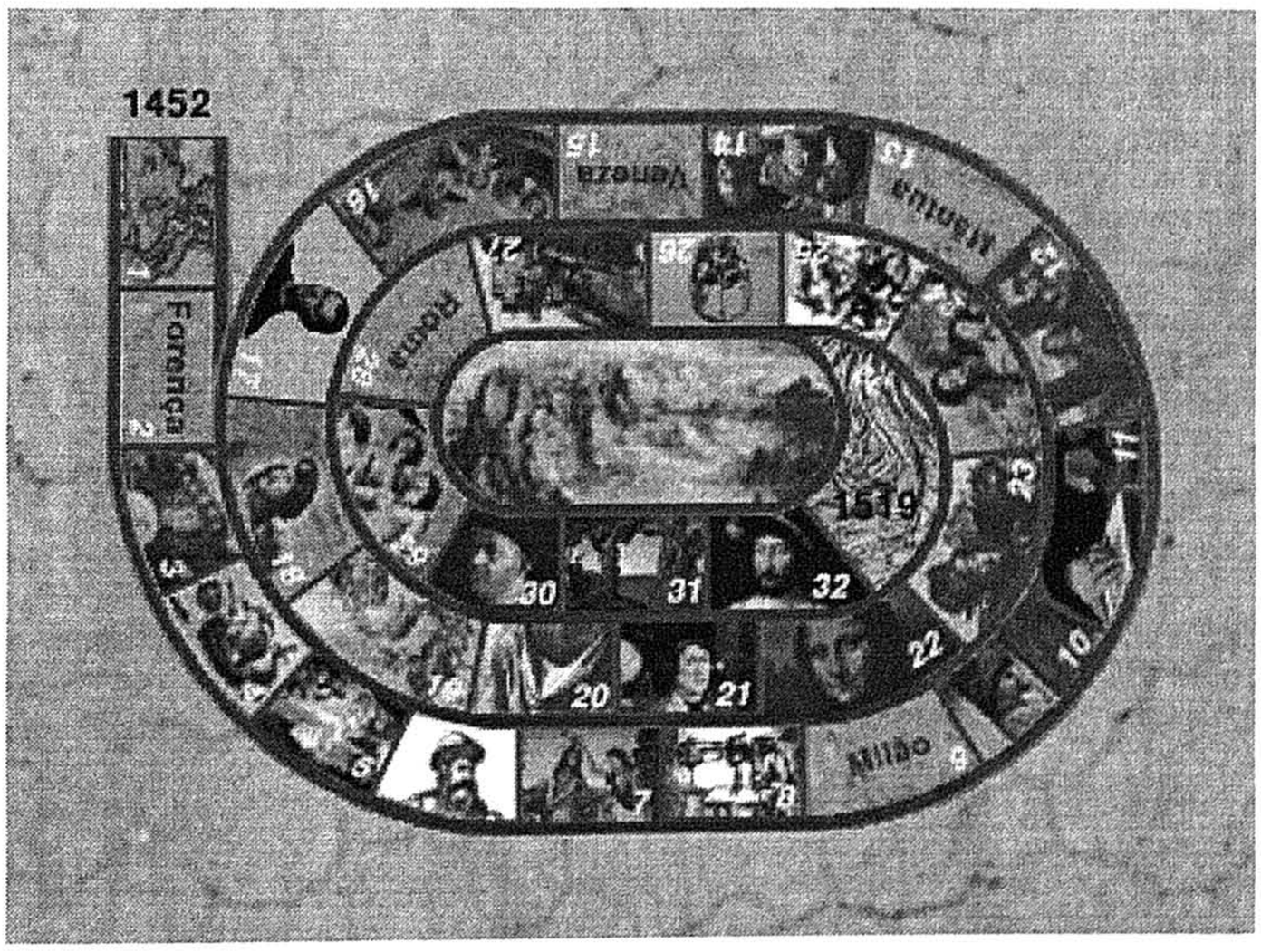


Figura 7.4 - Tabuleiro do Jogo da Glória

Tal como se vê na imagem anterior as posições têm formas irregulares e, como é necessário associar a cada posição o respectivo botão de entrada, só se pode implementar este jogo definindo botões irregulares. A possibilidade de realização de botões irregulares foi acrescentada na versão 2.0 do MediaMogul.

O lançamento do **dado** levantou dois problemas: gerar um número aleatório e mostrar o dado a rodar. Para gerar um **número aleatório** recorre-se a uma **subrotina** em C, cuja interface com o MediaMogul está definida na secção 6.6 e destina-se a aceder às variáveis do MediaMogul. No exemplo apresentado na secção 6.5 a subrotina adiciona 100 ao valor de uma variável, mas para realizar esta função especial, a subrotina foi modificada de modo a gerar um número aleatório e inteiro (usando as funções da linguagem C) compreendido entre 1 e 6, sendo depois esse valor copiado para uma variável interna do MediaMogul. Para se ver o **dado a rodar** é necessário construir uma **animação**. A animação é composta por uma sequência de imagens paradas com as várias posições que o dado pode tomar. Usando a ferramenta **animpack** (secção 5.3.4) agrupam-se as imagens num único ficheiro correspondente à animação final.

Ao seleccionar o botão da **posição** em que se encontra inicia-se a visualização do respectivo **segmento**. A construção destes segmentos é análoga ao segmento apresentado na secção 7.2.1.

7.3 Teste e Gravação do CD-i

Os testes aos protótipos em MediaMogul são fáceis de implementar. O utilizador deve carregar o editor de scripts com o script inicial da aplicação. Para se assegurar que a aplicação está de acordo com o *Green Book*, o MediaMogul disponibiliza a opção ANALYSE que se encarrega de fazer o teste a toda a aplicação, quando seleccionada sobre o script inicial.

Durante o desenvolvimento de uma aplicação é necessário testar o comportamento dos segmentos que se vão desenvolvendo, permitindo ajustar o seu funcionamento, detectar erros e descobrir se funcionam de acordo com o especificado. Como o MediaMogul opera directamente sobre o CD-i, isto simplifica o processo de simulação de cada script.

Depois do protótipo em MediaMogul estar concluído e antes de gravar o disco CD-i é necessário criar uma imagem do disco. Para criar a imagem do disco é necessário usar o S2D, em particular o comando *cdbuild*.

Depois de criada a imagem do disco é necessário proceder à sua emulação. Usando um **Emulador** pode-se executar a imagem do disco tal como se de um verdadeiro disco CD-i se tratasse, fazendo os testes finais à aplicação. Se durante a emulação não forem detectados quaisquer erros passa-se então à gravação do CD-i. Para fazer a gravação é necessário um gravador (WORM) de discos CD-i (*CDD - Philips 521*), onde se coloca um disco virgem para gravação. O software que é necessário usar para dar início a este processo é o *disctools*.

VIII

CONCLUSÕES

Nos últimos 5 anos notou-se um forte crescimento das aplicações de hipertexto, sistemas multimédia e sistemas de informação humana e geográfica.

As tecnologias **multimédia** envolvem uma grande quantidade de dados (texto, som, imagens e vídeo). É necessário ter meios para armazenamento dos dados e para o seu processamento. Os sistemas para satisfazerem estes requisitos têm de ser optimizados. Uma plataforma baseada em computador e disco magnético tem de se apoiar numa arquitectura complexa, recorrendo a processadores de grande velocidade e elevado poder de cálculo. Actualmente uma das formas de ultrapassar este problema é recorrer a plataformas baseadas em CD-ROM XA. Outra é usar o sistema CD-i.

O sistema **CD-i** é a melhor plataforma multimédia destinada ao mercado doméstico, porque para se ter acesso a produtos multimédia CD-i, não é necessário ter conhecimentos na área das ciências da computação. O CD-i está especialmente vocacionado para ser utilizado por um grupo de utilizadores (tipicamente o ambiente de uma sala de estar ou uma conferência), como é o caso do videogravador.

A inclusão do FMV (*Full-Motion Video*) no sistema CD-i, permitindo a visualização de VideoCD, veio acrescentar uma funcionalidade muito popular, originando um novo mercado de títulos que ajudam à implantação do CD-i. Os principais filmes realizados nos USA já estão comercializados em VideoCD. Existem várias plataformas semelhantes ao CD-i (3DO, SEGA, ...), mas espera-se que este se imponha sobre os restantes sistemas, principalmente porque assenta sobre uma norma internacional. Uma das desvantagens do VideoCD é a necessidade de se ter de trocar de disco a meio de um filme, porque normalmente um filme tem uma duração superior a 75 minutos, tempo máximo de armazenamento de um CD. Este problema deverá ser ultrapassado em breve com o lançamento do **HD-CD**, um CD de alta-densidade.

O HD-CD tem uma capacidade de armazenamento de 3,7 GB por disco, conseguida pela diminuição do tamanho dos furos de gravação do CD e pela colocação de mais pistas no disco, encurtando as distâncias que as separam. Na feira CES de *Las Vegas* desta Primavera, a Sony apresentou um leitor HD-CD com MPEG2, estando previsto o seu lançamento comercial no início de 1996. Este formato está a ser definido dentro de uma nova norma, o **livro dourado** (*Gold Book*), que integra todas as normas já existentes para CDs. Em desenvolvimento está o **CD de superdensidade com duplo lado** (duas camadas para armazenamento de informação), dispondo de uma capacidade de armazenamento total de 10 GB por disco.

A **programação** de aplicações para CD-i ainda não é trivial, sendo necessário recorrer a engenheiros de software com experiência na programação sobre a plataforma CD-i. Os sistemas de autoria têm aliviado o processo penoso de desenvolvimento de títulos CD-i, embora a facilidade de utilização crie limitações aos títulos. As ferramentas mais populares nos vários níveis de programação são: o MediaMogul, para a realização de protótipos, o HAL-90, para o desenvolvimento de títulos CD-i, o Balboa e o CORE, para o desenvolvimento de títulos cuja funcionalidade tem de explorar ao máximo as capacidade do sistema CD-i.

De momento colocam-se ainda bastantes problemas a quem pretende escolher um sistema multimédia que não seja fechado, porque ainda não existe compatibilidade entre os sistemas, dado que as investigações dos diferentes laboratórios desenvolveram diversos processos de fabricação de CDs que estão ainda muito longe da uniformização desejada. Contudo o **CD-R** (*Compact Disc Recordable*) será o sucessor natural do CD-i, dentro dos parâmetros que a investigação científica já definiu. A possibilidade dada ao utilizador de poder gravar no disco os resultados do seu próprio trabalho será mais uma parcela de valor a acrescentar à tecnologia do disco compacto, o novo papiro que permite reescrever em linguagem audio-scripto-visual todo o tipo de mensagens e que cobre campos tão diversificados como os da educação, formação profissional e indústrias do lazer.

8.1 CONTRIBUTO PRINCIPAL

Definidos os objectivos que nos propunhamos alcançar no início do trabalho, gostaríamos de destacar como contribuição principal:

- O levantamento de bibliografia sob o tema central, da qual resultou o estudo de diversas questões directa e indirectamente relacionadas com a utilização de sistemas de autoria para desenvolvimento de aplicações multimédia sob a plataforma CD-i;
- O desenvolvimento de um protótipo de uma aplicação CD-i sobre a vida e obra de *Leonardo da Vinci*.

No âmbito do grupo de trabalho do CMUP encontra-se em desenvolvimento um título CD-i sobre a Universidade do Porto. A criação do protótipo Leonardo permitiu adquirir a experiência e os conhecimentos relacionados com o desenvolvimento de aplicações interactivas sobre a plataforma CD-i.

8.2 RESULTADOS PRÁTICOS

O processamento de imagem em CD-i requer um operador com experiência, dado que a mesma imagem ao ser visionada no computador tem um aspecto diferente do que se obtém com um CD-i ligado a um monitor de TV. Esta ligeira degradação pode ser praticamente corrigida, aumentando o brilho da imagem em 20% e aumentando o contraste em 10%. Além disto a imagem na TV pode não aparecer na totalidade, sendo normalmente cortada as zonas periféricas, pelo que se aconselha a utilização do conceito apresentado na secção (3.5.1) sobre a zona de visualização.

O áudio não levanta este tipo de problemas: desde que a matriz inicial tenha uma boa qualidade e o processo de digitalização seja realizado sem erros, o resultado final é garantidamente satisfatório.

A conversão de imagem ou áudio para os formatos próprios do CD-i não apresentam degradações significativas.

O grau de interactividade que é possível imprimir a uma aplicação CD-i não é fácil de caracterizar dado que a sua génese está definida na especificação do título. Mas tipicamente podemos afirmar que o CD-i tem um sistema de interactividade baseado em interrupções, no qual o utilizador, através de dispositivos externos (monitor, comando por infravermelhos, ...), envia sinais ao sistema operativo que se encarrega de gerir os acontecimentos. As potencialidades da programação de aplicações interactivas sobre a plataforma CD-i são muito animadoras, tendo por limite prático a conjugação do poder criativo das equipas de concepção e de programação.

As primitivas do MediaMogul que se encarregam da visualização do texto são muito básicas e pouco flexíveis; não há qualquer controlo sobre a distância entre os caracteres vizinhos, todos os caracteres tem o mesmo tamanho (logo as letras mais estreitas ficam mais afastadas do que as letras mais largas), não é possível controlar o espaçamento entre linhas e não possui qualquer tipo de formatação (bold, itálico, centrar, marginar, ...).

Em jeito de resultado final apresenta-se uma classificação genérica das vários sistemas de autoria estudados, por ordem crescente das suas funcionalidades:

- ASLAN - Conjunto de ferramentas que ajudam na conversão do material original para os formatos próprios do CD-i;
- MediaMogul - Desenvolvimento de protótipos de títulos CD-i de um modo expedito, especialmente vocacionado para fazer apresentações de temas;
- Griffin Tools - Desenvolvimento de protótipos de títulos CD-i mas incorporando algumas primitivas de natureza interactiva e multitarefa;
- HAL-90 - Destinado ao desenvolvimento de títulos e conceptualmente bem estruturado;
- CORE - Desenvolvimento de títulos usando técnicas de programação orientada aos objectos e que se espera venha a ser o primeiro de uma série de novos sistemas de autoria;
- Balboa - Desenvolvimento de títulos profissionais, mas o seu elevado grau de funcionalidades é conseguido por uma proximidade grande do sistema operativo, logo pouco amigável.

8.3 LIMITAÇÕES

O protótipo Leonardo foi desenvolvido usando o sistema de autoria MediaMogul, logo as limitações do sistema de autoria vão-se fazer sentir no protótipo final. As capacidades da plataforma não foram totalmente exploradas. Esta limitação não é negativa para o projecto em questão, atendendo a que se considerou mais importante a realização de um protótipo num curto espaço de tempo.

Não foram abordadas as questões relacionadas com a ligação do leitor CDI605, através de uma rede local, aos postos de trabalho (tratamento de imagem, tratamento de som e programação), porque o CMUP não dispunha da estrutura física da rede *Ethernet*.

O protótipo não possui qualquer segmento de vídeo, pois foi decidido que não fazia sentido usar pequenas janelas de vídeo a 15 fps, cujo resultado final ficaria muito longe do desejado. Embora o MediaMogul 2.0 permita a utilização de FMV, este não foi incluído no protótipo porque o estúdio CD-i do CMUP ainda não possui ferramentas para tratamento de vídeo digital.

8.4 SUGESTÕES PARA DESENVOLVIMENTOS FUTUROS

Da análise das limitações apresentadas, resultaram algumas sugestões para trabalhos futuros. Em síntese indicam-se a seguir algumas propostas:

- Permitir ao utilizador um controlo mais apurado (parar, continuar, ...) do desenrolar de um segmento, usando menus *pop-up*, que surgem no ecrã por acção do utilizador (*click* com rato);
- Incluir segmentos de FMV no protótipo;
- Redefinir o Jogo da Glória de modo a disponibilizar marcadores da posição do jogador, movimentação automática dos marcadores e restringir a acção do utilizador apenas à posição em que se encontra;
- Disponibilizar um botão de mais informação que, num qualquer ponto do segmento, permitisse obter informação específica sobre o assunto que está a ser apresentado;
- Disponibilizar uma base de dados textual, em que o utilizador tenha acesso a informação escrita sobre *Leonardo da Vinci*;
- Fazer um mapa da aplicação, mostrando ao utilizador as várias opções que pode escolher para navegar pela aplicação e assinalar em que nível hierárquico se encontra;
- Evitar a interrupção do som durante a transição de um menu para outro;
- Criar novas fontes de caracteres;
- Melhorar a forma de aceder a um volume elevado de informação textual.

Bibliografia

[BUSH45]
Vannevar Bush
As We May Think
Atlantic Monthly, 176
Julho, 1945

[CAIN95]
Chris Cain
Plataformas Multimédia
Personal Computer World, Março de 1995

[COLLIN92]
Collin, Williamson, John and Henderson
Interactive Multimedia: Practice and Promise
Taylor and Francis, Inc., 1993

[CORREIA92]
Carlos Correia
Centro Nacional de Formação de Formadores
1992

[COTTON93]
Bob Cotton, Richard Olivier
The Cyberspace Lexicon
Phaidon, 1993

[COMPUSERVE94]
Mac Imagination Pack
File DISCOV.TXT
14 de Junho, 1994

[COMPUSERVE94]
Griffins Tools
Graham Trott
Compuserve 100115,1075

[C-TECH94]
C-TECH Verlags Gmb
Pablo Electrinic Publishing
CD-i Spezial Marketing-Kommunikation
1994

[DIGIDESIGN92]
Degidesign Inc.
Sound Designer II, Audio Editing Software
1992

[DUNCAN93]
Andrew Duncan
MPEG Data Structures
Philips Interactive Media of America
1993

[ENGELBART63]
D. Engelbart
A Conceptual Framework for the Augmentation of Man´s Intellect
Spartan Books, Vol I
Londres, 1963

[FAYAD93]
Mohamed Fayad; Louis Hawn; Mark Roberts; Jerry Klatt
Using the Shlaer-Mellor Object-Oriented Analysis Method
IEEE Software
March 1993

[GOLVIN92a]
Charles Golvin
Monitoring Audio Play From Memory
Technical Note #79
Setembro, 1992

[GOLVIN92b]
Charles Golvin
Using Subroutine Modules
Technical Note #80
Setembro, 1992

[HOFFOS92]
Hoffos Signe, Graham Sharpless,
Philip Smith e Nicholas Lewis
CD-I Designers Guide
McGraw-Hill, 1992
ISBN-0-07-707-580-3

[HUNT92]
Kevin Hunt
Real-Time Considerations in CD-i
Philips Interactive Media of America
1992

[LIMA93]
Rui Manuel Dias Ferreira de Lima
Áudio no Sistema CD-i
Faculdade de Engenharia da Universidade do Porto
1993

[LUIJT92a]
Alty van Luijt
Display Syncronization in CD-I
Technical Note #63, Julho de 1992

[LUIJT92b]
Alty van Luijt
Audio Considerations in CD-I
TSA Applications Notes TSA-002
23 de Julho, 1992

[LUNDY94]
Brad Lundy
MediaMogul Emulation v. Simulation
Compuserve Forum Multimédia Secção 17
Maio, 1994

[MEER92a]
Sander van der Meer
Media Mogul Script-to-Disc
TSA Application Notes TSA-005
24 de Julho, 1992

[MEER93b]
Jan van der Meer
The Full Motion System for CD-I
TSA Application Notes TSA-007
15 de Fevereiro, 1993

[MICROWARE91a]
Microware
OS-9 Technical Overview
Microware System Corporation, Março 1991

[MICROWARE91b]
Microware
OS-9 Using Professional
Microware System Corporation, Março 1991

[MICROWARE90]
Microware
CD-i System Libraries
Technical Reference Manual
Microware System Corporation, 1990

[MOREIRA93]
Rui Jorge da Silva Moreira
O Sistema CD-i
Faculdade de Engenharia da Universidade do Porto
1993

[MOREIRA95]
Rui Jorge da Silva Moreira
Interface entre MHEG e CD-i
Tese de Mestrado
Faculdade de Engenharia da Universidade do Porto
1995

[NIELSON86]
Jakob Nielson
Technical University of Denmark
Academic Press, Inc.
1986

[OPTIMAGE93]
OptImage Interactive Services Company, L.P.
MediaMogul 2.0
Novembro, 1993

[OPTIMAGE94a]
OptImage Interactive Services,
Balboa Run-Time Environment
Reference Guide, volume I e II
Philips, Abril 1994

[OPTIMAGE94b]
OptImage Interactive Services,
Balboa Run-Time Environment
Programmer's Guide, volume I e II
Philips, Abril 1994

[OPTIMAGE92a]
OptImage Interactive Services,
Disc Building - for Macintosh Computers
Philips, 1992

[OPTIMAGE92b] OptImage Interactive Services,
Emulation - for Macintosh Computers
Philips, 1992

[OPTIMAGE91] OptImage Interactive Services,
Macintosh C Cross Compiler
Philips, 1991

[PHILIPS91] Philips Consumer Electronics B.V.
System Description CD-ROM XA
Maio, 1991

[PHILIPS92a] Philips Interactive Media Systems
CD-i 605 - User Manual
1992

[PHILIPS92b] Philips Interactive Media Systems
CDD 521 - Operating Instructions
1992

[PHILIPS92c] Philips Interactive Media Systems
The CD-i Production Handbook
1992

[PHILIPS92d] Philips Interactive Media Systems
The CD-i Design Handbook
1992

[PHILIPS92e] Philips Interactive Media Systems
The Producer's Handbook of Multimedia
Authoring for CD-i
1992

[PHILIPS93] Philips Consumer Electronics B.V.
Compact Disc Interactive Full Functional Specification
Março, 1993

[PHILIPS94] Philips Consumer Electronics B.V.
Video Specification V2.0
Julho, 1994

[PIESING92] Jon Piesing e Andy Maxfield
A Beginner's Guide to Balboa
TSA Application Notes TSA-001
9 de Abril, 1992

[PICOLET 93] Randy Picolet,
CORE Technology:
Managing Concurrent Complexity
Philips Interactive Media
Abril de 1993

[RIBEIRO93] Nuno Jorge Gonçalves M. Ribeiro
O Tratamento de Vídeo no Sistema CD-i
Faculdade de Engenharia da Universidade do Porto
1993

[RIBEIRO95] Nuno Jorge Gonçalves M. Ribeiro
Biblioteca de Funções - UPToolkit
Tese de Mestrado
Faculdade de Engenharia da Universidade do Porto
1995

[SENFTNER93] Blake Senftner
ASLAN Overview
Philips Interactive Media of America
Abril, 1993

[SMITH90] Jerry D. Smith
Reusability and Software Construction: C and C++
John Wiley & Sons, Inc
ISBN 0-471-52411-5

[STERN93] Andrew Stern
Porting Adobe Premiere Slideshows to Media Mogul
The Interactive Engineer
Vol:1 Nº6 Outubro, 1993

[TANENBAUM92] Andrew S. Tanenbaum
Modern Operating Systems
Prentice-Hall International, 1992
ISBN 0-13-595752-4

[TAUB86] Herbert Taub
Principles of Comunication Systems
McGraw-Hill Series in Electrical Engineering, 1986
ISBN 0-07-100313-4

[TRIGG86] R. Trigg e M. Weiser
TEXTNET: A Network-based Approach to
Text Handling
ACM Transactions on Office Information Systems
Janeiro, 1986

[VROONHOVEN92] Joost van Vroonhoven
Emulator & Emulation
TSA Application Notes TSA-006
14 de Ourtubro, 1992

[WOODHEAD91] Nigel Woodhead
Hypertext and Hypermedia
Theory and Applications
ISBN 0-201-54442-3
Addison-Wesley, 1991

[ZEGERS95] Hein Zegers e Cees Rooijackers
Limitations to Media Mogul *Plug-Ins*
The Interactive Engineer
Fevereiro, 1995

APÊNDICE A

FORMATOS DE CODIFICAÇÃO DE IMAGENS

Formato DYUV

A. Codificação

A codificação DYUV atribui um valor ao sinal de luminância de cada pixel (Y) e de seguida define a cor (U e V) separadamente em relação ao seguinte: o Y compreende 6 partes de Verde (G) para 3 de Vermelho (R) e 1 de Azul (B). Assim, obtêm-se os seguintes valores:

Y= 0.6 G + 0.3 R + 0.1 B
U= R - Y
V= B - Y

O método DYUV opera em pares de pixels, atribuindo um valor Y a cada pixel individual e calculando o valor UV para cada par. Esta operação exige 2 Bytes de dados para cada par de pixels. Este método comprime tipicamente uma imagem fotográfica a uma proporção de 3:1 quando comparada com as técnicas RGB normalizadas de gráficos de computador. As imagens DYUV de 16 bits em resolução normal são codificadas como um conjunto de 4 pares de pixels, onde cada par de pixels compreende 4 bits, ou seja 2 Bytes de informação no total.

B. Cor

O formato DYUV, com um *display* de 24 bits, possibilita toda a gama de cores do CD-i (16 777 216), o que constitui mais do que um olho humano altamente treinado consegue distinguir, mas que é necessária para traduzir a subtileza das imagens fotográficas e os tons da pele humana.

C. Resolução

O método DYUV aplica-se a imagens com resolução normal, mas para alta resolução a codificação QHY (*Quantized High-Resolution*) pode ser aplicada para melhorar o sinal de luminância e trá-lo para o nível exigido pelo *display* de alta resolução.

D. Espaço de Armazenamento

A codificação DYUV ocupa 105 KB por imagem em PAL/SECAM e 90 KB por imagem em NTSC. Assim, um disco de 650 MB pode conter 6340 imagens fotográficas.

E. Material Visual

O método DYUV é mais apropriado para fotografias verdadeiras ou *frames* de vídeo paradas. Este tipo de imagens são detalhadas e profusamente coloridas, bem como sombreadas, exigindo uma reprodução de elevada qualidade. No entanto, os sinais U e V possuem uma resolução horizontal reduzida que não suporta alterações bruscas da tonalidade. A utilização de codificação delta com quantificação não linear significa que são necessários aproximadamente 3 pixels para mudar para a cor exactamente certa: quando o método DYUV é utilizado em actualização parcial do ecrã é muito importante restaurar as cores com o seu valor exacto.

F. Resumo das características

Tipo: DYUV
Descrição: Delta YUV
Resolução: Normal
Cores: 16 777 216
Planos: A ou B
Bits por pixel: 16 bits/2 pixels
Tamanho da imagem: PAL - 105 KB / NTSC - 90 KB
Utilidade: imagens naturais.

Este método de compressão de imagens naturais aproveita-se do facto de os pixels adjacentes serem frequentemente semelhantes. Por isso, em vez de guardar a informação referente a cada pixel, é guardada a informação das diferenças entre os pixels de cada linha. Guarda-se mais informação sobre o brilho (luminância) do que sobre a cor (crominância), o que poupa espaço em disco. Uma imagem DYUV necessita de 90 KB ou 45 sectores CD-i.

Formato RGB

A. Codificação

A codificação normalizada RGB proporciona cores com 8 bits (RGB 8:8:8), ou seja, 256 níveis distintos de cor para cada *palette* ou componente R, G e B do sinal de cor. Uma forma simples de comprimir o volume de dados contido neste registo sem perda significativa de qualidade é reduzir o número de bits com que se representa cada cor para 5 bits. Esta forma de codificação é conhecida por RGB 5:5:5 (5 bits para cada nível de cor). A taxa de compressão atingida é de 8:5. As imagens RGB 5:5:5 de 16 bits, que ocupam ambos os planos de imagem (A e B), são geradas em 2 bancos separados de 8 bits cada. O banco superior contém os bits superiores e o inferior contém os bits os inferiores. Este formato pode conter bits de transparência sem cor juntamente com os níveis de R, G e B. Os 5 bits mais significativos são atribuídos a cada nível de R, G e B e deixa os restantes 3 bits vazios em cada banco de 8 bits. Cinco bits para cada nível R, G e B deixa 1 bit para transparência no código final de 16 bits.

B. Cor

O formato RGB 5:5:5, permitindo uma escala de 32 valores para cada uma das cores primárias, proporciona um leque de 32768 cores, o que é suficiente para imagens que não exijam realismo fotográfico.

C. Resolução

O formato RGB 5:5:5 só se encontra disponível em resolução normal.

D. Espaço de Armazenamento

O formato RGB 5:5:5 ocupa 210 KB por imagem em PAL/SECAM e 180 KB em NTSC. Assim, o mesmo disco pode conter 3170 gráficos complexos de computador.

E. Material Visual

O formato RGB 5:5:5 proporciona *displays* de gráficos e fotografias com cores vivas sem as exigências do formato DYUV ou as limitações do formato CLUT. No entanto, como a taxa de compressão é relativamente baixa, o formato CLUT pode ser preferido nos casos em que a quantidade (em vez da qualidade) seja o facto crítico.

F. Resumo das características

Tipo: RGB888
Descrição: formato não reconhecido pelo CD-i, formato para gráficos de computador
Resolução: Normal (384 x 280 pixels)
Cores: 16 777 216
Planos: definição desnecessária
Bits por pixel: 24 bits/pixel
Tamanho da imagem: 600 KB
Utilidade: usado como fonte de todas as imagens a utilizar: captura digital de vídeo e gráficos.

Tipo: RGB555
Descrição: cada nível de cor (R, G e B) é codificada com 5 bits
Resolução: Normal
Cores: 32 768
Planos: A e B
Bits por pixel: 16 bits/pixel (5 bits p/ cada cor e 1 bit de transparência)
Tamanho da imagem: PAL - 210 KB / NTSC - 180 KB
Utilidade: imagens detalhadas.

Não existe compressão, logo é guardada a informação de cor de cada pixel. Uma imagem RGB necessita de 200 KB ou 100 sectores CD-i.

Formato CLUT

A. Codificação

A codificação CLUT (*Colour Lookup Table*) emprega uma economia de espaço muito simples através da criação de tabelas individuais (*palettes*) de 256 cores simultâneas a partir da gama final de 16 777 216 cores do CD-i. O método associa um endereço discreto da tabela a cada cor. Desta forma, e em tempo-real, o sistema só necessita de identificar a tabela correcta e o endereço da cor nessa mesma tabela, simplificando a procura de uma dada cor visto que passa da procura de uma cor entre 16 777 216 para a procura de uma cor entre 256 cores. Este procedimento atinge uma taxa de compressão de 3:1, análoga ao formato DYUV, mas com qualidade visual diferente. Em efeito, o formato CLUT oferece 3 níveis:

a) CLUT de 8 bits - 256 cores em simultâneo,
b) CLUT de 7 bits - 128 cores em simultâneo,
c) CLUT de 4 bits - 16 cores em simultâneo.

As imagens CLUT são geradas com 1 bit por pixel em resolução normal e 1 bit por cada par de pixels em resolução dupla. O formato CLUT8 utiliza 1 Byte (8 bits), o formato CLUT7 deixa 1 bit desse Byte livre e o formato CLUT4 em dupla resolução utiliza 4 bits por cada pixel.

B. Cor

O formato CLUT8 oferece uma *palette* de 256 cores em simultâneo a partir da gama final CD-i. O formato CLUT7 oferece 128 cores numa *palette* e o formato CLUT4 oferece 16 cores em dupla resolução.

C. Resolução

Os formatos CLUT8 e CLUT7 estão disponíveis em resolução normal e o formato CLUT4 em dupla resolução.

D. Espaço de Armazenamento

A codificação CLUT8 ocupa 105KB por imagem em PAL/SECAM e 90 KB por imagem em NTSC. Assim, um disco de 650 MB pode conter 6340 gráficos de computador, imagens convertidas a partir de obras de arte convencionais, tabelas ou outro material ilustrativo.

E. Material Visual

A codificação CLUT é a técnica mais versátil e económica para o armazenamento de gráficos e texto. As *palettes* de cor são limitadas, mas as imagens permanecem vivas e o formato CLUT4 em dupla resolução é útil para texto, em particular para conjuntos de caracteres complexos ou fontes pequenas que exijam grande detalhe.

F. Resumo das características

Tipo: CLUT8
Descrição: *LookUpTable*(LUT) com endereços de cores de 8 bits
Resolução: Normal
Cores: 256 (2^8 cores ou 2^8 endereços possíveis)
Planos: A ou B
Bits por pixel**:** 8 bits por pixel
Tamanho da imagem: PAL - 105 KB / NTSC - 90 KB
Utilidade: gráficos, imagens geradas em computador.

Tipo: CLUT7
Descrição: *LookUpTable*(LUT) com endereços de cores de 7 bits
Resolução: Normal
Cores: 128 (2^7 cores ou 2^7 endereços possíveis)
Planos: A ou B
Bits por pixel**:** 7 bits por pixel
Utilidade: gráficos, imagens geradas por computador.

Tipo: CLUT4
Descrição: *LookUpTable*(LUT) com endereços de cores de 4 bits
Resolução: Dupla
Cores: 16 (2^4 cores ou 2^4 endereços possíveis)
Planos: A ou B
Bits por pixel**:** 8 bits/ 2 pixels
Utilidade: gráficos ou texto detalhado.

As imagens gráficas geradas no computador podem ser comprimidas utilizando CLUT´s (*Colour Look Up Tables*). Estas tabelas têm comprimento variável e contém uma *palette* de cores. Cada cor possui uma posição bem determinada na tabela. Assim, as cores de uma imagem são referenciadas utilizando o respectivo endereço na tabela em vez de guardar o seu valor absoluto RGB de cor. O CD-i suporta 3 níveis de imagens CLUT: CLUT8, CLUT7, e CLUT4. O número indica o número de bits de dados utilizado para guardar o endereço da CLUT da cor para cada pixel. Imagens CLUT7 e CLUT8 necessitam de 90 KB ou 45 sectores CD-i.

Formato RLE

A. Codificação

O método RLE (*Run-Length Encoding*) utiliza técnicas de análise da cor apresentada em cada linha horizontal de pixels duma imagem. Este método codifica a cor do primeiro pixel na linha e de seguida grava o comprimento da linha (em pixels) para o qual a cor inicial não se altera (corrida -*run*-), isto é, até que apareça outra cor nessa linha. Para imagens que contêm grandes blocos (com muitas corridas) da mesma cor, este método proporciona uma poupança de espaço de armazenamento considerável. As imagens RLE devem registar em primeiro lugar o endereço da cor seleccionada na

tabela relevante e, de seguida, registar o comprimento da corrida. Em resolução normal, o formato RLE7 emprega 7 bits para descrever o endereço CLUT de 1 único pixel e divide este número de bits pelo endereço e pelo comprimento da corrida de pixels na linha. Em dupla resolução, o RLE3 define cada pixel num único par com um endereço de 3 bits distinto, e inclui ambos os endereços e o seu comprimento de corrida comum na descrição de uma corrida de pares de pixels.

B. Cor

O formato RLE7 utiliza uma *palette* de 128 cores em simultâneo enquanto que o RLE3, que codifica pares de pixels em dupla resolução, oferece 8 cores apropriadas para texto e imagens.

C. Resolução

O formato RLE7 está disponível em resolução normal e o formato RLE3 em dupla resolução.

D. Espaço de Armazenamento

O formato RLE reduz a quantidade de todos os dados necessários para registar uma tirada de cor única com um código de 2 Bytes. Assim, se uma linha de 255 pixels contém apenas 1 cor, são suficientes 2 Bytes de dados para definir aquela cor e especificar o comprimento da corrida para essa linha, o que de outra forma ocuparia 180 Bytes mesmo no formato CLUT4. Quanto maior for a redundância contida na imagem, maior será a poupança de espaço de armazenamento em RLE, podendo-se atingir taxas de compressão de 10:1.

E. Material Visual

A codificação RLE é o método preferido para animações, dado que preenche cada linha de saída com um único bloco de cor.

F. Resumo das características

Tipo: RLE7
Descrição: *Run-Length variation of* CLUT7
Resolução: Normal
Cores: 128 (2^7 cores ou 2^7 endereços possíveis)
Planos: A ou B
Bits por pixel**:** 7 bits por pixel
Utilidade: animações.

Tipo: RLE3
Descrição: *Run-Length variation of* CLUT4
Resolução: Dupla
Cores: 7
Planos: A ou B
Bits por pixel**:** 8 bits / 2 pixels
Utilidade: animações.

Sequências animadas ou desenhos animados são comprimidos utilizando a técnica *Run Length Encoding*, o que reduz ainda mais o tamanho das imagens CLUT. É guardado o valor da cor de um pixel particular e depois diz-se quantos pixels seguintes possuem essa mesma cor. Uma imagem RLE necessita de 10 KB ou 5 sectores CD-i.

APÊNDICE B

EXEMPLO DE UMA SUBROTINA: VIDEOSC

Antes de chamar esta subrotina deve ser usado o comando *Pre_effect* para carregar a imagem e preparar o controlador de vídeo para a visualização da imagem. O comando *Pre_effect* actualiza as FCT's e LCT's de modo que a imagem actual seja colocada no plano A e a nova imagem no plano B. No entanto, o comando *Pre_effect* coloca o ambiente vídeo em determinadas condições (associadas à imagem) no fim do próximo efeito de vídeo. Por exemplo, *fgplane* é associado ao plano da nova imagem, mesmo sabendo que esta operação só se realiza após a subrotina do efeito vídeo associar a respectiva imagem ao plano.

A subrotina *videosc* obtém o endereço da imagem do plano B a partir da matriz *lctbuf* que se encontra definida no ambiente vídeo e em cada passo da sua execução escreve esta instrução numa linha superior e apaga a linha anterior. Ao mesmo tempo activa uma *matte* transparente no plano A na mesma linha onde se encontra o endereço da instrução do plano B, permitindo a visualização da imagem.

Para deslocar a imagem do plano A para fora do topo do ecrã a subrotina *videosc* obtém o endereço do plano A a partir da matriz *lctbuf* e incrementa-o do número de linhas deslocadas até a um máximo de 384.

Caso a imagem tenha um tamanho inferior ao ecrã, o MediaMogul centra verticalmente a imagem e a subrotina *videosc* desloca essa imagem começando pelo fundo da imagem em vez de começar pelo fundo do ecrã.

```
#include <stdio.h>
#include <errno.h>
#include <modes.h>
#include <module.h>
#include <signal.h>
#include <ucm.h>
#include <venv.h>

#ifdef DEBUG
#undef DEBUG
#define DEBUG(x)  fprintf x
#else
#define DEBUG(x)
#endif

#define bgplane                 (fgplane ^ 1)

char                    *prog;
int                     vevent, *lctid, *fctid;
short                   fgplane;
short                   *vsig_plane, *vsig_line;
short                   done = 0;
```

```
int                     step = 0;

void
sighand(sig)
        register int            sig;
{
        switch (sig)
        {
        case SIGQUIT:
                done = 1;
                break;
        default:
                break;
        }
}

videosc(vpath, bgaddrinstr, bgaddrline, bgendline, fgaddr, fgaddrline, fgendline, duration)

        register int            vpath;
        int                     bgaddrinstr;
        short                   bgaddrline;
        short                   bgendline;
        char                    *fgaddr;
        short                   fgaddrline;
        short                   fgendline;
        int                     duration;
{
        register int            nop = cp_nop();
        register short          line0, line1, oldline0, oldline1;
        int                     tr[2];
        short                   startline;
        register int            temp;

        /* Colocar o plano A transparente */
        tr[fgplane] = TR_MAT0_T;
        tr[bgplane] = TR_OFF;
        dc_wrfi(vpath, fctid[0], FCT_TCI, cp_tci(MIX_OFF, tr[0], tr[1]));

        /* Iniciar o deslocamento pelo lado inferior da maior das duas imagens */
        if (bgendline > fgendline) {
                startline = bgendline;
        }
        else {
                startline = fgendline;
        }

        /* O MediaMogul aloca as hi res lcts e caso esteja activo o interlace todas as instruções lcts tem
        de ser escritas quer nas linhas pares quer nas linhas ímpares */
        oldline0 = startline & ~1;
        oldline1 = startline | 1;
        /* Definir o endereço da instrução para a imagem de fundo (Plano B)
        no fim da zona de deslocamento da imagem */
        dc_wrli(vpath, lctid[bgplane], oldline0, LCT_DADR, bgaddrinstr);
        dc_wrli(vpath, lctid[bgplane], oldline1, LCT_DADR, bgaddrinstr);
        /* Definir uma matte transparente no plano A correspondente ao endereço da instrução no plano B */
        dc_wrli(vpath, lctid[fgplane], 0, LCT_MAT1, cp_matte(1, MO_END, MF_MF0, ICF_MAX, 0));
        dc_wrli(vpath, lctid[fgplane], 1, LCT_MAT1, cp_matte(1, MO_END, MF_MF0, ICF_MAX, 0));
        dc_wrli(vpath, lctid[fgplane], oldline0, LCT_MAT0, cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));
        dc_wrli(vpath, lctid[fgplane], oldline1, LCT_MAT0, cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));

        /* A linha a partir da qual foi gerado o sinal é deslocada de modo que o scroll seja suave */

        /* remover a instrução signal da posição velha */
        dc_wrli(vpath, lctid[*vsig_plane], (*vsig_line) & ~1, LCT_SIG, cp_nop());
        dc_wrli(vpath, lctid[*vsig_plane], (*vsig_line) | 1, LCT_SIG, cp_nop());

        /* Colocar a instrução signal na nova posição */
        dc_wrli(vpath, lctid[*vsig_plane], oldline0, LCT_SIG, cp_sig());
        dc_wrli(vpath, lctid[*vsig_plane], oldline1, LCT_SIG, cp_sig());

        *vsig_line = oldline1;
```

```
for (step = 0; step < duration; ++step) {
        if (done) {
                step = duration;
                break;
        }

        line0 = (startline - (startline * step / duration)) & ~1;
        line1 = line0 | 1;

        if (line0 <= bgaddrline) {
                /*
                Caso linha da imagem que está a ser deslocada ultrapasse a linha correspondente ao
                topo da imagem do plano B, então a imagem do segundo plano não necessita de ser
                deslocada. Mas a imagem do plano principal pode ainda ter de ser deslocada para cima,
                então este ciclo é interrompido (break) sendo o respectivo tratamento realizado fora
                deste ciclo.
                */
                break;
}

/* espera pelo sinal de vídeo */
while (_ev_wait(vevent, 1, 1) != 1) ;
/*if (_ev_wait(vevent, 1, 1) == -1) {
        DEBUG((stderr, "%s: _ev_wait() failed\n", prog));
}*/


if (line0 != oldline0) {
        /* Colocar a instrução signal na nova posição */
        dc_wrli(vpath, lctid[*vsig_plane], line0, LCT_SIG, cp_sig());
        dc_wrli(vpath, lctid[*vsig_plane], line1, LCT_SIG, cp_sig());

        /* remover a instrução signal da posição velha */
        dc_wrli(vpath, lctid[*vsig_plane], oldline0, LCT_SIG, cp_nop());
        dc_wrli(vpath, lctid[*vsig_plane], oldline1, LCT_SIG, cp_nop());

        /* Reter a posição onde se encontra a instrução signal */
        *vsig_line = line1;

        /* Endereço da nova linha para o plano B */
        dc_wrli(vpath, lctid[bgplane], line0, LCT_DADR, bgaddrinstr);
        dc_wrli(vpath, lctid[bgplane], line1, LCT_DADR, bgaddrinstr);

        /* Remover o endereço da linha velha */
        dc_wrli(vpath, lctid[bgplane], oldline0, LCT_DADR, nop);
        dc_wrli(vpath, lctid[bgplane], oldline1, LCT_DADR, nop);

        /* Colocar transparente a nova linha */
        dc_wrli(vpath, lctid[fgplane], line0, LCT_MAT0,
                        cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));
        dc_wrli(vpath, lctid[fgplane], line1, LCT_MAT0,
                        cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));

                /* Actualizar o endereço da imagem do plano A */
                if ((line0 > fgaddrline) && (line0 < fgendline)) {
                        dc_wrli(vpath, lctid[fgplane], fgaddrline & ~1, LCT_DADR,
                                (temp = cp_dadr((int)(fgaddr + (((fgendline - line0)>>1) * 384)))));
                        dc_wrli(vpath, lctid[fgplane], fgaddrline | 1, LCT_DADR, temp);
                        }

                /* Remover a transparência da linha velha */
                dc_wrli(vpath, lctid[fgplane], oldline0, LCT_MAT0, nop);
                dc_wrli(vpath, lctid[fgplane], oldline1, LCT_MAT0, nop);

                oldline0 = line0;
                oldline1 = line1;
        }
}
```

```
if (step < duration) {
        /* O deslocamento da imagem do plano B está concluído, mas pode
        ainda ser necessário deslocar o plano A*/

        /* Esperar pelo sinal de vídeo */
        while (_ev_wait(vevent, 1, 1) != 1) ;
        /*if (_ev_wait(vevent, 1, 1) == -1) {
                DEBUG((stderr, "%s: falhou o _ev_wait()\n", prog));
        }*/

        /* Colocar a instrução na nova posição */
        dc_wrli(vpath, lctid[*vsig_plane], line0, LCT_SIG, cp_sig());
        dc_wrli(vpath, lctid[*vsig_plane], line1, LCT_SIG, cp_sig());

        /* Remover a instrução da posição velha */
        dc_wrli(vpath, lctid[*vsig_plane], oldline0, LCT_SIG, cp_nop());
        dc_wrli(vpath, lctid[*vsig_plane], oldline1, LCT_SIG, cp_nop());

        /* Reter a posição onde se encontra a instrução */
        *vsig_line = line1;

        /* Actualizar o endereço para a imagem do plano A */
        if ((line0 > fgaddrline) && (line0 < fgendline)) {
                dc_wrli(vpath, lctid[fgplane], fgaddrline & ~1, LCT_DADR,
                        (temp = cp_dadr((int)(fgaddr + (((fgendline - line0)>>1) * 384)))));
                dc_wrli(vpath, lctid[fgplane], fgaddrline | 1, LCT_DADR, temp);
                }

        /* Colocar o endereço da imagem do plano B na última linha */
        dc_wrli(vpath, lctid[bgplane], bgaddrline & ~1, LCT_DADR, bgaddrinstr);
        dc_wrli(vpath, lctid[bgplane], bgaddrline | 1, LCT_DADR, bgaddrinstr);

        /* Definir a nova linha como transparente */
        dc_wrli(vpath, lctid[fgplane], line0, LCT_MAT0, cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));
        dc_wrli(vpath, lctid[fgplane], line1, LCT_MAT0, cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));

        /* Remover o endereço da linha velha */
        dc_wrli(vpath, lctid[bgplane], oldline0, LCT_DADR, nop);
        dc_wrli(vpath, lctid[bgplane], oldline1, LCT_DADR, nop);

        /* Remover a transparência da linha velha */
        dc_wrli(vpath, lctid[fgplane], oldline0, LCT_MAT0, nop);
        dc_wrli(vpath, lctid[fgplane], oldline1, LCT_MAT0, nop);

        oldline0 = line0;
        oldline1 = line1;
        ++step;

        while (step < duration) {
                line0 = (startline - (startline * step / duration)) & ~1;
                line1 = line0 | 1;

                /* Esperar pelo sinal de vídeo */
                while (_ev_wait(vevent, 1, 1) != 1) ;
                /*if (_ev_wait(vevent, 1, 1) == -1) {
                        DEBUG((stderr, "%s: falhou 0  _ev_wait()\n", prog));
                }*/


                if (line0 != oldline0) {
                        /* Colocar a instrução na nova posição */
                        dc_wrli(vpath, lctid[*vsig_plane], line0, LCT_SIG,cp_sig());
                        dc_wrli(vpath, lctid[*vsig_plane], line1, LCT_SIG,cp_sig());

                        /* Remover a instrução da posição velha */
                        dc_wrli(vpath, lctid[*vsig_plane], oldline0, LCT_SIG, cp_nop());
                        dc_wrli(vpath, lctid[*vsig_plane], oldline1, LCT_SIG, cp_nop());

                        /* Reter a posição da instrução */
                        *vsig_line = line1;
```

```
/* Actualizar o endereço para o plano A */
if ((line0 > fgaddrline) && (line0 < fgendline)) {
    dc_wrli(vpath, lctid[fgplane], fgaddrline & ~1, LCT_DADR,
            (temp = cp_dadr((int)(fgaddr + (((fgendline - line0)>>1) * 384)))));
    dc_wrli(vpath, lctid[fgplane], fgaddrline | 1, LCT_DADR, temp);
}

/* Definir a nova linha como transparente */
dc_wrli(vpath, lctid[fgplane], line0, LCT_MAT0,
                cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));
dc_wrli(vpath, lctid[fgplane], line1, LCT_MAT0,
                cp_matte(0, MO_SET, MF_MF0, ICF_MAX, 0));

/* Remover a transparência da linha velha */
dc_wrli(vpath, lctid[fgplane], oldline0, LCT_MAT0, nop);
dc_wrli(vpath, lctid[fgplane], oldline1, LCT_MAT0, nop);
oldline0 = line0;
oldline1 = line1;
                }
                ++step;
        }
}

/* Esperar pelo sinal de vídeo */
while (_ev_wait(vevent, 1, 1) != 1) ;
/*if (_ev_wait(vevent, 1, 1) == -1) {
        DEBUG((stderr, "%s:falhou 0 _ev_wait()\n", prog));
}*/

/* Puxar para cima */
        dc_wrli(vpath, lctid[bgplane], bgaddrline & ~1, LCT_DADR, bgaddrinstr);
        dc_wrli(vpath, lctid[bgplane], bgaddrline | 1, LCT_DADR, bgaddrinstr);
        if ((oldline0) != (bgaddrline & ~1)) {
                dc_wrli(vpath, lctid[bgplane], oldline0, LCT_DADR, nop);
                dc_wrli(vpath, lctid[bgplane], oldline1, LCT_DADR, nop);
        }

        /* Trocar a ordem dos planos para o plano B */
        dc_wrfi(vpath, fctid[0], FCT_PO, cp_po((int)bgplane));
        dc_wrli(vpath, lctid[fgplane], oldline0, LCT_MAT0, nop);
        dc_wrli(vpath, lctid[fgplane], oldline1, LCT_MAT0, nop);
        dc_wrli(vpath, lctid[fgplane], 0, LCT_MAT1, nop);
        dc_wrli(vpath, lctid[fgplane], 1, LCT_MAT1, nop);
}

main(argc, argv)
        int                     argc;
        char                    **argv;
{
        int                     alarm_id;
        int                     duration;
        int                     vpath;
        char                    *venvmod;
        VE_ENV                  *venv;
        LCT_ELEMENT             **lctbuf;
        LCT_ELEMENT             *le;
        short                   bgaddrline;
        short                   bgendline;
        int                     bgaddrinstr;
        short                   fgaddrline;
        short                   fgendline;
        int                     fgaddr;

        prog = argv[0];
        if (argc != 2) {
                fprintf(stderr, "Tempo: %s duração_em_30_partes_do_segundo \n", prog);
                exit(0);
        }
```

```
intercept(sighand);
duration = atoi(argv[1]);
vpath = init_video();

/* link para o evento do sinal de vídeo */
if ((vevent = _ev_link("line_event")) == -1) {
        DEBUG((stderr, "%s: não foi possível criar o link para o sinal de vídeo\n", prog));
        exit();
}

/* link para o módulo de dados do ambiente de vídeo*/
if ((venvmod = (char *)modlink(VENV_NAME, 0)) == (char *)(-1)) {
    DEBUG((stderr, "%s: não foi possível criar o link para o módulo de dados do ambiente vídeo\n", prog));
    exit();
}

venv = (VE_ENV *)(venvmod + ((mod_exec *)venvmod)->_mexec);
lctid = venv->lctid;
fctid = venv->fctid;
lctbuf = venv->lctbuf;
/*
O comando Pre_Effect coloca na plano A do ambiente de vídeo a nova imagem.
Este comando actualiza o plano B, pelo que, antes desta subrotina terminar deve ser chamada a instrução
cp_po() para colocar a imagem no plano A. Mas dada a duração da subrotina o plano A é o oposto ao
definido no video environment, pelo que são necessárias as seguintes instruções */
fgplane = (*(venv->fgplane)) ^ 1;
vsig_plane = venv->vsig_plane;
vsig_line = venv->vsig_line;

/* Obter a linha e a instrução do endereço da imagem do plano B */
for (le = lctbuf[bgplane]; le->line_num != -1; ++le) {
        if (le->colm_num == LCT_DADR) {
                bgaddrline = le->line_num;
                bgaddrinstr = le->instruction;
                break;
        }
}
if (le->line_num == -1) {
        /* Não foi possível obter a linha e o endereço da imagem do plano B */
        DEBUG((stderr, "%s: Não foi possível obter a linha e o endereço da imagem do plano B \n", prog));
        exit(0);
}

/* Obter a última linha da imagem do plano B */
bgendline = *((venv->icm_off_line) + bgplane);

/* Obter a linha e o endereço da imagem do plano A */
for (le = lctbuf[fgplane]; le->line_num != -1; ++le) {
        if (le->colm_num == LCT_DADR) {
                fgaddrline = le->line_num;
                fgaddr = (le->instruction) & 0x00ffffff;
                break;
        }
}
if (le->line_num == -1) {
        /* Não foi possível obter a linha e o endereço da imagem do plano A */
        DEBUG((stderr, "%s: Não foi possível obter a linha e o endereço da imagem do plano B \n", prog));
        exit(0);
}

/* Obter a última linha da imagem do plano A*/

fgendline = *((venv->icm_off_line) + fgplane);
videosc(vpath, bgaddrinstr, bgaddrline, bgendline, fgaddr, fgaddrline,
                fgendline, duration);
_ev_unlink(vevent);
munlink(venvmod);
}
```

FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO
BIBLIOTE
0000002142