Edmundo B. Torreão • Marcos V. Villas

VAX

UMA INTRODUÇÃO

Editora Campus

TÍTULOS DE INTERESSE CORRELATO

ARQUITETURA DE COMPUTADORES E HARDWARE

MICROPROCESSADORES/MICROCOMPUTADORES
Vol. 1 — Arquitetura — *A.J. Khambata*
Vol. 2 — Software e Sistemas — *A. J. Khambata*

APLICAÇÕES DE MICROPROCESSADORES — *J.A. Kuecken*

INTRODUÇÃO À ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES — *H. Lorin*

CIRCUITOS INTEGRADOS CMOS — *R. Melen e H. Garland*

APLICAÇÃO DE MICROPROCESSADORES NA INDÚSTRIA — *NCC*

MICRO-MINICOMPUTADORES BRASILEIROS — *E.P.L. Passos*

CIRCUITOS SINCLAIR — *D. Santos Lima*

MANUTENÇÃO DE MICROS — *C. Costa*

ELETRÔNICA DIGITAL — *E.M. Oliveira*

Conheça toda a linha de Informática da Editora CAMPUS, com títulos nas áreas de: Introdução à Informática; Computação para Crianças; BASIC; COBOL; Outras Linguagens de Alto Nível; Microprocessadores e Linguagem de Máquina; Arquitetura de Computadores e Hardware; Apple; PC; TK85 e TK90X; TRS; Computação em Ambiente Empresarial; Programas e Aplicativos; Processamento de Dados; Teoria e Organização de Dados; Banco de Dados; Programação e Análise Estruturada de Sistemas; Sistemas Operacionais e Compiladores; Inteligência Artificial e Robótica; Interesse Especial; Videocassete e Videogames.

E, ainda:

DICIONÁRIO ENCICLOPÉDICO DE INFORMÁTICA — *A.H. Fragomeni*
Extenso e abrangente, reúne mais de 33.000 entradas em inglês e português pertencentes aos mais diversos campos da Informática e áreas correlatas.

Solicite nosso catálogo completo.

Procure nossas publicações nas boas livrarias ou comunique-se diretamente com:

EDITORA CAMPUS LTDA.
Livros Científicos e Técnicos
Qualidade internacional a serviço do autor e do leitor nacional.
Rua Barão de Itapagipe 55 Rio Comprido
Telefone (021) 284-8443 Telex (00038) 021-32606
20261 Rio de Janeiro RJ Brasil
Endereço Telegráfico: CAMPUSRIO.

VAX

UMA INTRODUÇÃO

VAX

UMA INTRODUÇÃO

VAX

UMA INTRODUÇÃO

Edmundo B. Torreão • **Marcos V. Villas**

Professor do Departamento de Informática PUC/RJ
Analista de Suporte de Sistemas

Professor do Departamento de Informática PUC/RJ
Analista de Sistemas

Editora Campus Ltda.

Rio de Janeiro

Capa
Otavio Studart

Projeto Gráfico, Composição e Revisão
Editora Campus Ltda.
Qualidade internacional a serviço do autor e do leitor nacional.
Rua Barão de Itapagipe 55 Rio Comprido
Tel.: (021) 284 8443 Telex (00038) 021-32606
20261 Rio de Janeiro RJ Brasil
Endereço Telegráfico: CAMPUSRIO

ISBN 85-7001-403-1

Ficha Catalográfica
CIP-Brasil. Catalogação-na-fonte.
Sindicato Nacional dos Editores de Livros, RJ.

Torreão, Edmundo Bastos
T642v VAX: uma introdução / Edmundo Bastos Torreão, Marcos Vianna Villas. — Rio de Janeiro: Campus, 1987.

ISBN 85-7001-403-1

1. VAX. 2. Computadores — Sistemas. I. Villas,Marcos Vianna. II. Título.

CDD — 001.61
86-1039 CDU — 681.3.32

À Marisy e
a meus pais

(Edmundo)

À Emília e
a meus pais.

(Marcos)

APRESENTAÇÃO

Este livro apresenta o sistema VAX para gerentes de processamento de dados, programadores de aplicações, estudantes de Informática e usuários não especializados de sistemas de computação.

O texto não tem como objetivo servir de manual de utilização de um sistema VAX (apesar de apresentar inúmeros comandos). Para esse fim devem ser utilizados os manuais da DEC (Digital Equipment Corporation) - fabricante dos computadores VAX no exterior - ou da Elebra Computadores, fabricante no Brasil.

O texto apresenta as principais características dos recursos de "software" e "hardware" do equipamento, os conceitos da tecnologia VAX, os comandos disponíveis, a operação, e a gerência do sistema.

Essa apresentação é feita de forma didática, os conceitos são apresentados de forma bastante simplificada.

Ambos os autores deste livro são graduados no curso de Tecnologia em Processamento de Dados da PUC-RJ. Atualmente, são professores do Departamento de Informática dessa Universidade, possuindo larga experiência em aplicações e equipamentos VAX.

Edmundo Bastos Torreão atua como analista de sistemas na área de controle de processos industriais, tendo anteriormente trabalhado em suporte de sistemas VAX/VMS.

Marcos Vianna Villas, também analista de sistemas, tem como área de atuação o desenvolvimento de aplicações administrativas com processamento centralizado e distribuído.

Os autores gostariam de ressaltar que não existe nenhuma relação de senioridade entre eles. A ordem de apresentação dos seus nomes é meramente alfabética.

SUMÁRIO

1. INTRODUÇÃO 11

2. UTILIZAÇÃO DO SISTEMA 16
   2.1. O básico da utilização 17
   2.2. Gerência do sistema 38

3. PROCESSADORES E PERIFÉRICOS VAX 45
   3.1. Processador — visão geral 45
   3.2. Registradores 48
   3.3. Modos de endereçamento 49
   3.4. Periféricos 51
      3.4.1 Terminais 51
      3.4.2 Impressoras 52
      3.4.3 Unidades de discos 53
      3.4.4 Unidades de fita 53

4. SISTEMA OPERACIONAL VAX/VMS 54
   4.1. Módulos do sistema 54
   4.2. Escalonamento de processos 57
   4.3. Sistema de memória virtual 58
   4.4. Comunicação entre processos 60
   4.5. Segurança do sistema 60

5. LINGUAGENS 63
   5.1. VAX-11 FORTRAN 66
   5.2. VAX-11 COBOL 67
   5.3. VAX-11 BASIC 68
   5.4. VAX-11 C 68
   5.5. VAX-11 PASCAL 69

6. DESENVOLVIMENTO DE PROGRAMAS 70
   6.1. Editor de textos 70
   6.2. Editor de ligações 74
   6.3. Biblioteca de rotinas "Run-time" 77
   6.4. Depurador simbólico interativo 78
   6.5. Utilitário de biblioteca 82
   6.6. Procedimentos 84
      6.6.1 Símbolos 85
      6.6.2 Expressões 86
      6.6.3 Comandos de controle 87
      6.6.4 Funções 89
      6.6.5 Comandos para manipular arquivos 90

6.7. Utilitário de diferenças 92
6.8. Formatador de textos 94
6.9. Utilitário de mensagens 97

7. ARQUITETURA DE INFORMAÇÃO 98
7.1. CDD (Common Data Dictionary) 101
7.2. DATATRIEVE 103
7.3. DECGRAPH 104
7.4. DECSLIDE 105
7.5. VTX 106
7.6. FMS (Forms Management System) 107
7.7. TDMS (Terminal Data Management System) 107
7.8. DBMS (Database Management System) 108
7.9. RDB (Relational Database) 109
7.10. ACMS (Application Control and Management System) 110
7.11. RMS (Record Management System) 110

8. COMUNICAÇÕES 111
8.1. DECNET 113
8.2. INTERNET 114
8.3. PACKETNET 115
8.4. ETHERNET 116
8.5. Emulação de terminal 116

9. CLUSTER 117
9.1. Hardware 119
9.2. Software 123

1. INTRODUÇÃO

Em termos bem gerais, o sistema VAX pode ser caracterizado como um sistema de computação multiprogramável de alta performance. Isto significa que vários programas podem ser executados concomitantemente numa mesma unidade central de processamento - UCP.

O sistema possui diferentes modos de operação. Um modo de operação é caracterizado pela maneira através do qual um usuário relaciona-se com o computador. Ele pode interagir através de um processamento "batch" (todos os comandos são grupados e submetidos para execução) ou, interativamente, num terminal.

Além do seu uso mais conhecido em aplicações comerciais, o sistema VAX também pode ser utilizado na área de controle de processos, devido a suas características de "hardware" e "software".

Num ambiente industrial, por exemplo, o sistema estará interagindo, diretamente, com outros componentes eletrônicos responsáveis pela comunicação com os equipamentos da fábrica. Diz-se comumente que, nesta situação, o sistema está efetuando controle de um processo industrial.

Dentre as aplicações mais conhecidas estão o desenvolvimento de programas - através de uma série de recursos existentes para esse fim - as aplicações comerciais/administrativas (folha de pagamento, contabilidade, etc) e aplicações científicas (aplicações em Engenharia e Estatística, entre outras ciências exatas).

Os componentes do sistema podem ser descritos, sumariamente, da seguinte forma:

* processador
* sistema operacional
* linguagens
* recursos de desenvolvimento
* recursos de gerência de informação
* recursos de comunicação de dados

O processador do sistema VAX faz parte de uma família, a família VAX. Nesta família estão o VAX 11/730, o VAX 11/750 (comercializado pela empresa brasileira Elebra Computadores, no equipamento MX-850), o VAX 11/780 e o VAX 8600.

O conceito de desenvolvimento de uma família de UCPs resulta em uma compatibilidade de "software" entre esses processadores. O mesmo "software" (programas) pode ser utilizado em todos os processadores.

A família VAX foi precedida pela série PDP-11 que apresenta características semelhantes. Tem-se o seguinte histórico desta série de UCPs:

* 1970 - PDP 11/20
* 1972 - PDP 11/40 e PDP 11/45
* 1973 - PDP 11/05 e PDP 11/10
* 1974 - PDP 11/70
* 1975 - PDP 11/04 e PDP 11/03
* 1976 - PDP 11/34 e PDP 11/55

* 1977 - VAX 11/780 e PDP 11/60
* 1979 - PDP 11/23 e PDP 11/24
* 1980 - VAX 11/750
* 1985 - VAX 8600

Os membros da família VAX diferem basicamente em termos de performance.

O endereçamento de memória é feito com 32 bits. Isto permite que sejam acessadas 2^32 posições de memória, mesmo que isso não ocorra na realidade.

O sistema possui 16 registradores de uso geral, cada um possuindo 32 bits. Estes registradores são utilizados pelo processador para operações lógicas e aritméticas, e para controle de execução de um programa.

O conjunto de instruções de "assembler" (linguagem de baixo nível) desse processador é extremamente poderoso. Permite fazer operações aritméticas em ponto flutuante e com campos em formato decimal-compactado, além da aritmética inteira binária. É possível a manipulação de cadeias ("strings") de caracteres e de bits.

Existem nove modos de endereçamento. Cada modo de endereçamento descreve uma maneira através da qual o processador acessa uma informação na memória ou nos registradores. Esta informação pode, também, estar contida na própria instrução.

As seguintes operações podem ser realizadas através de uma única instrução "assembler", dando uma amostra da sua capacidade do processador:

* somar, comparar o valor somado com uma determinada constante e desviar, segundo alguma condição especificada, para uma outra parte do programa;

* efetuar qualquer operação aritmética com duas variáveis e atribuir o resultado a uma terceira;

* editar (formatar) uma variável (por exemplo, para impressão de um relatório)

O processador também possui uma memória "cache" (memória com velocidade de acesso superior à memória principal), que contém as informações mais freqüentemente utilizadas.

A memória possui dispositivos capazes de detectar erros que porventura ocorram, aumentando assim a confiabilidade do sistema.

O processador executa "prefetch" de instruções de máquina. Isto significa que, enquanto a UCP está processando uma instrução, a próxima já está sendo acessada na memória.

O sistema operacional gerencia memória virtual. Isto significa que, neste tipo de equipamento, os programas podem ocupar mais espaço que a memória física disponível. O conceito de memória virtual libera o programador das preocupações quanto ao tamanho de seus programas.

O sistema é multiusuário. Várias pessoas podem utilizá-lo ao mesmo tempo.

A interação com o sistema é feita de uma forma amigável, através de comandos compostos de verbos em inglês. Esta linguagem de comandos pode ser adequada com novos comandos, inclusive em português, criados pelo próprio usuário.

Um usuário especializado em computação pode desenvolver programas em várias linguagens (Assembler, Fortran, Cobol, C e Pascal, entre outras) através de recursos de desenvolvimento com editores de texto, depuradores de programas, compiladores, gerentes de bibliotecas de programas e editores de ligação de programas.

Um usuário não especializado pode acessar o sistema através de "pacotes" de "software" como, por exemplo, gerência de banco de dados e geração de relatórios.

Existem vários recursos para interligação de computadores VAX com computadores de outras tecnologias, possibilitando um maior número de recursos para os usuários.

A integridade dos dados é garantida tanto durante a execução de um programa quanto no armazenamento permanente. Cada programa, mesmo que executando concorrente a outros programas de outros usuários na mesma memória física, só pode acessar seus próprios dados.

Um usuário pode proteger seus arquivos em disco ou em fita magnética, de forma seletiva ou geral, contra acesso indevido.

2. UTILIZAÇÃO DO SISTEMA

Os usuários de um sistema VAX podem ser divididos em grandes grupos, cada um com usos e características distintas. Estes são, em linhas gerais:

* programadores de aplicação
* programadores de sistemas
* gerentes de sistemas
* usuários não especializados em computação
* operadores do sistema

O programador de aplicação se utiliza do VAX para desenvolvimento de sistemas de processamento de dados, através das linguagens de programação disponíveis e outros produtos.

O programador de sistemas desenvolve "software" complementar ao já existente no sistema operacional, expandindo suas funções ou adaptando-o às peculiaridades de uma determinada empresa.

O gerente de sistemas organiza o uso do sistema, visando proporcionar a todos os usuários disponibilidade de recursos e confiabilidade das informações armazenadas.

Os operadores do sistema encarregam-se de funções como procedimentos de inicialização e montagem de fitas e discos, entre outras.

Os usuários não especializados utilizam-se do sistema para executarem suas aplicações. Sua interface com o sistema é adaptada para suas necessidades.

2.1 O básico da utilização

Todos os usuários dispõem do interpretador de comandos DCL - Digital Command Language, dos serviços de gerência de arquivos e registros (RMS - Record Management Services) e dos serviços básicos do sistema (Systems Services).

Através da DCL é possível não só chamar os utilitários para o desenvolvimento, como também a encadear de forma controlada a execução de programas.

Todos os comandos da DCL podem ser abreviados. Isto significa que de uma palavra de cinco ou mais caracteres, só três ou quatro precisam ser digitados.

Na maioria dos comandos são necessários parâmetros que informem sobre quais "objetos" o comando vai atuar. Esses parâmetros podem ser obrigatórios ou opcionais. Os parâmetros obrigatórios, se não forem digitados, serão requisitados. Dessa forma um usuário, que não conheça todo o repertório de comandos, pode ir "tateando" e esperar a requisição de dados por parte do sistema operacional.

Além dos parâmetros, os comandos podem receber qualificadores que mudam ou acrescentam suas características.

Os últimos vinte comandos digitados são armazenados e podem ser editados e reexecutados, facilitando a execução de tarefas repetitivas.

Cabe, aqui, apresentarmos alguns conceitos antes de continuarmos.

* Organização hierárquica de arquivos

Os arquivos são armazenados em discos magnéticos segundo uma estrutura hierárquica. Nesse tipo de estrutura, os arquivos estão dispostos como se estivessem em uma árvore. Nesta árvore existem os diretórios ("galhos"). Os arquivos, nesta analogia, poderiam ser associados às "folhas".

Existem os diretórios de nível mais "alto" que possuem subdiretórios, que por sua vez podem possuir outros subdiretórios e assim por diante. Dessa forma, os arquivos podem ser separados e agrupados de maneira que aqueles que possuírem informações relacionadas estarão agrupados num mesmo diretório.

Este tipo de catalogação facilita a organização dos arquivos, sobretudo quando a quantidade se torna elevada.

Surge, então, a idéia de estar "posicionado" dentro dessa "árvore".

Para que um usuário possa visualizar apenas os arquivos necessários para um determinado tipo de trabalho, facilitando sua concentração, ele se "posiciona" num disco e, dentro do disco, num determinado diretório - o disco e diretório "defaults".

Eles indicam, para o sistema, em qual unidade de disco e em qual diretório devem ser procurados os arquivos de um usuário, quando este não fizer nenhuma especificação explícita de disco e diretório.

Isso facilita o trabalho do usuário, reduzindo sua digitação.

* Nomes lógicos

O conceito de nomes lógicos também facilita o acesso a arquivos.

Nomes lógicos são termos definidos pelos usuários que passam a ter uma significação especial para o sistema.

Diz-se que o usuário "associa" um nome lógico a uma especificação de arquivo e que quando for utilizado, o sistema efetuará sua "tradução".

Nomes lógicos podem estar associados, também, a unidades periféricas (disco, fita, etc) e diretórios. Isso permite que o acesso a componentes físicos do sistema possa ser feito de modo lógico. A disposição física dos periféricos, diretórios e arquivos pode ser alterada, sempre que isto acarretar uma melhora na performance geral do sistema. Entretanto, programas que fizerem acesso a periféricos, diretórios e arquivos através de nomes lógicos não precisam ser alterados internamente, quando ocorrer uma alteração nessa disposição. Para que o acesso continue compatível, basta que seja feita uma nova associação dos nomes lógicos relacionados.

* Processo

A literatura VAX usa o termo "processo" para definir uma "entidade" que possui determinados privilégios, cotas de recursos do sistema e prioridades, que executa programas e que possui uma identificação para o sistema. Um processo pode ser uma sessão de terminal

entre um "login" e um "logout" (entrada/saída do sistema). Pode ser, também, uma tarefa executada num processamento em "batch".

Um processo é contabilizado, ou seja, serão guardadas informações sobre seu uso de UCP e tempo conectado ao sistema, entre outros dados.

Um processo pode estar executando alguma aplicação ou aguardando algum evento como a liberação da UCP por parte de outro processo, liberação de algum arquivo, disponibilidade de algum periférico ou dados de um terminal.

* Proteção de arquivos

Como já foi dito, os usuários podem proteger seus arquivos ou qualquer outro recurso contra o acesso indevido por parte de outros usuários.

Vamos dar as explicações, falando a respeito de arquivos. Os mesmos conceitos, entretanto, podem ser utilizados na proteção de discos e fitas magnéticos e regiões compartilhadas de memória, entre outros recursos.

Considerando-se a propriedade de um arquivo, existem quatro grupos de usuários:

- proprietário (OWNER) - aquele que criou o arquivo
- grupo (GROUP) - aqueles que trabalham no mesmo departamento ou mesmo projeto que o proprietário

- gerente de sistema (SYSTEM) - usuário responsável pela manutenção e otimização do sistema
- "mundo" (WORLD) - todos os usuários não categorizados acima

O proprietário do arquivo pode permitir diferentes tipos de acesso para cada um desses grupos de usuários. Como tipo de acesso, entende-se a capacidade de ler, gravar, deletar e executar um determinado arquivo. É claro que a execução de um arquivo só é possível se este for um arquivo contendo um programa executável.

A proteção de um arquivo é a forma como estão dispostos os tipos de acesso de cada um desses quatro grupos de usuário. Quando um usuário cria um novo arquivo, se nada for informado explicitamente, será usado para esse arquivo a proteção "default" (padrão que pode ser alterado pelo usuário).

* Filas de impressão e de "batch"

A impressão de arquivos numa mesma impressora é organizada em fila. Os usuários que desejarem ter seus arquivos impressos nessa impressora, devem colocá-los na fila associada. Com isso, os usuários podem continuar a trabalhar enquanto o sistema se encarrega de imprimí-los. Enquanto existirem arquivos para serem impressos nas filas, o sistema irá retirando-os e imprimindo-os.

O conceito de filas também é utilizado para execução de processos em "batch". Um processo, para ser executado em "batch", pode ser submetido através de

cartões perfurados ou de um terminal interativo. A vantagem básica do uso de processos em "batch" é a não alocação de um terminal do computador para a execução de uma tarefa que possui entrada de dados bem conhecida, isto é, uma tarefa que não necessita de uma interface conversacional com o usuário.

Os dados, que necessitar, estarão no próprio arquivo submetido ou em outro arquivo (em disco, fita magnética ou cartão perfurado).

O usuário pode ser avisado ao término da tarefa, ou ter a descrição dos resultados de cada comando executado armazenada em um arquivo.

Um arquivo submetido numa fila de "batch" (também chamado de procedimento de comandos) contém comandos que serão executados quando esta tarefa for uma das "n" primeiras tarefas nesta fila. O valor de "n" indica o número de tarefas que podem executar simultaneamente nesta fila. Esta idéia pode ser visualizada da seguinte forma: imagine uma fila única num banco para todos os guichês. Sempre que houver um guichê disponível, um novo usuário da única fila será chamado para ser atendido. O valor de "n" pode ser associado ao número de guichês disponíveis.

Esse valor é pré-fixado pelo gerente de sistemas.

* Procedimentos de comandos

Vimos, até aqui, procedimentos de comandos serem submetidos em filas de processamento em "batch". Este tipo de arquivo também pode ser utilizado numa

sessão interativa. Quando o usuário não deseja submeter seu procedimento em "batch", ou porque todas as filas estão sobrecarregadas, ou porque seu procedimento tem como função o encadeamento de programas interativos, ele pode fazê-lo executando esse procedimento interativamente.

Isso mostra que a linguagem de comandos, DCL, é idêntica para os dois modos de operação (interativo ou "batch").

Os procedimentos serão apresentados com mais detalhes no capítulo sobre Desenvolvimento de Programas.

* Alocação de periféricos

Para o sistema, existe uma diferença entre a alocação física e a alocação lógica de um periférico.

A partir do instante da alocação física nenhum outro usuário poderá se utilizar desta unidade, salvo se o volume tiver sido montado como público. No momento em que a unidade for desalocada fisicamente, outro usuário poderá utilizá-la.

Vamos supor que um usuário deseje utilizar uma unidade de fita e vai montar e desmontar várias fitas magnéticas para realização de seu serviço. Se ele fizer isso utilizando-se dos comandos de alocação física, um outro usuário poderá tomar-lhe o acesso exclusivo da unidade entre uma desmontagem e a próxima montagem. Para que isto não tenha a possibilidade de ocorrer, existem comandos de alocação lógica. Estes comandos alocam a unidade para o usuário, independente de existir algum volume montado.

* Organização interna dos arquivos

O sistema VAX permite as seguintes organizações de arquivos:

- Seqüencial

  registros são gravados seqüencialmente

- Relativa

  registros são gravados em posicões especificadas por uma chave numérica que determina a ordem relativa desse novo registro no arquivo

- Indexada

  registros são ordenados dentro de um arquivo através de uma ou mais chaves que são campos desses registros

O acesso aos registros dentro desses três tipos de organização pode ser feito de forma seqüencial ou randômica, isto é, um registro após o outro ou um registro específico.

Nas organizações seqüencial e relativa, o acesso randômico é feito através da especificação da "célula" dentro do arquivo, que deve ser lida.

O acesso randômico, dentro da estrutura indexada, é feito através de uma chave de acesso.

Os registros podem ter tamanho fixo ou variável. A performance de acesso é sempre melhor quando se trata de registros de tamanho fixo.

* Identificação dos arquivos

A especificação de um arquivo é feita através de três campos: nome propriamente dito, tipo e versão. É necessária, para identificação do arquivo pelo sistema, a especificação do disco e do diretório onde está "localizado".

O nome é utilizado como identificação básica do arquivo. O tipo como uma subidentificação, e a versão como grau de atualidade das informações.

A necessidade de uma subidentificação pode ser exemplificada no desenvolvimento de programas.

Existe um programa-fonte escrito numa determinada linguagem de programação. Os arquivos de programa-fonte escritos na linguagem COBOL possuem tipo "COB", em FORTRAN, "FOR" e em PASCAL, "PAS".

Os arquivos de programas-objeto (resultado de compilações) possuem tipo "OBJ" e os programas-executáveis, tipo "EXE".

Os campos de nome e tipo são alfanuméricos e a versão é um campo numérico.

Quando um novo arquivo é criado, caso já exista algum outro arquivo com mesmo nome e tipo, o sistema não deletará esse arquivo já existente. Para diferenciá-los, o sistema atribuirá ao novo arquivo uma versão superior à do já existente.

Esse tipo de procedimento, em geral, acarreta um excessivo acúmulo de versões

de um mesmo arquivo, sobretudo dos arquivos de textos e de dados. O sistema oferece comandos que facilitam a manutenção das versões de um arquivo mais recente.

* Cotas de utilização de espaço em disco

O responsável pelo uso do computador (gerente de sistemas) pode especificar a capacidade máxima de espaço em disco que um usuário pode utilizar. Dessa forma, fica assegurado que a área do disco não fique sendo utilizada de maneira indevida. A determinação das cotas dependerá de uma estimativa da real necessidade de espaço em disco de cada usuário ou grupos de usuários.

* Símbolos da linguagem de comandos

A DCL possui recursos que permitem o desenvolvimento de programas simples com estruturas lógicas de condição e desvio, atribuições e comparações.

Tal qual qualquer linguagem de programação, permite ao usuário o uso de variáveis. Estas variáveis podem assumir valores numéricos ou cadeias de caracteres. O usuário pode efetuar atribuições, comparações e operações aritméticas.

Serão apresentados mais detalhes no capítulo sobre Desenvolvimento de Programas.

O repertório básico de comandos pode ser apresentado da seguinte forma:

* Informações e controle geral da sessão

- ASSIGN

Permite associar um nome lógico a uma unidade de disco, diretório ou nome de arquivo (nome físico). Um outro comando para o mesmo fim é o DEFINE, que possui, entretanto, uma sintaxe distinta. A desassociação é feita através do comando DEASSIGN. Os nomes lógicos podem ser consultados através do comando SHOW TRANSLATION.

Exemplo:

```
$ ASSIGN ARQS1.DAT ARQUIVO
$ SHOW TRANSLATION ARQUIVO
  ARQUIVO = "ARQS1.DAT"
$ ASSIGN ARQS2.DAT ARQUIVO
  Logical name assignment replaced
$ SHOW TRANSLATION ARQUIVO
  ARQUIVO = "ARQS2.DAT"
```

- HELP

Através deste comando o usuário pode solicitar ao sistema informações sobre o seu uso. Isso significa que dúvidas podem ser esclarecidas durante uma sessão no computador. O utilitário HELP fornece informações de forma hierárquica (em "árvore"), ou seja, à medida que são necessários mais detalhes, o usuário pode ir solicitando.

Num primeiro nível, são apresentadas as funções mais gerais. Dependendo da dúvida, o usuário requisita informações sobre um determinado tópico que, por sua vez, apresenta subtópicos.

- MAIL

Através deste comando, os usuários podem se comunicar através do sistema de forma não interativa. O conceito é em todo análogo a uma troca de correspondência, neste tipo de comunicação.

As mensagens são enviadas e o usuário "destinatário" recebe uma informação de que existe "correspondência". Quando lhe for conveniente, ele irá "ler a correspondência".

- PHONE

Dois ou mais usuários, que desejem comunicar-se interativamente através do sistema, podem fazê-lo através deste utilitário.

A tela dos terminais se dividirá num determinado número de partes, dependendo do número de usuários que estiverem se comunicando entre si, e tudo que for digitado por um usuário aparecerá nas telas dos outros usuários.

- REQUEST

Através deste comando, o usuário pode mandar uma mensagem para o operador do sistema solicitando, por exemplo, a montagem de fitas ou discos magnéticos.

- SET

Permite alterar, entre outras características, disco e diretório "default", proteção de arquivos,

atributos de tarefas que se encontrem em filas de impressão ou de "batch".

- SHOW

Através deste comando e dependendo dos parâmetros, o usuário pode conhecer:

. dia e hora

. disco e diretório "default"

. status de periféricos (Estão ou não alocados a algum usuário? Qual a quantidade de erros observada durante sua utilização?)

. associação de nomes lógicos

. características de periféricos (Qual sua capacidade de armazenamento? Quanto está sendo utilizado?)

. status de processos em execução (Estão esperando algum evento para prosseguir sua execução? Qual o tempo de UCP já consumido?)

. proteção "default" de arquivos

. status das filas de impressão e de "batch" (Qual o número de tarefas aguardando nas filas? Quantas tarefas podem executar simultaneamente numa determinada fila de "batch"?)

. cotas de utilização de discos

. valores de símbolos da linguagem de comandos (Quais os valores que foram atribuídos?)

Obs: As perguntas acima são apenas exemplos, existindo outras informações que podem ser solicitadas.

* Controle específico de tarefas em "batch"

- DELETE/ENTRY

Através deste comando, um usuário pode retirar de uma fila de "batch" ou de impressão uma tarefa que já esteja executando ou que ainda esteja por executar. Este comando, em geral, torna-se necessário quando o usuário "descobre" que nem todos os dados estão corretos no arquivo submetido.

- SUBMIT

Este comando submete um arquivo numa determinada fila de "batch".

- Exemplo:

```
$ SUBMIT X.COM
    Job 53 entered on queue SYS$BATCH
$ DELETE/ENTRY=53 SYS$BATCH
```

* Controle de uso de periféricos e volumes (discos e fitas magnéticas, por exemplo)

- ALLOCATE

Aloca, logicamente, uma unidade. A desmontagem é feita através do comando DEALLOCATE.

- MOUNT

Requisita ao operador a montagem de volumes. Uma fita ou disco magnético para serem utilizados, precisam ser

montados. A desalocação é feita através do comando DISMOUNT.

* Manipulação de arquivos

- APPEND

  Copia o conteúdo de um ou mais arquivos para o final (após o último registro) de um outro arquivo.

- COPY

  Copia o conteúdo de um ou mais arquivos (dependendo de como for feita a especificação) para outros arquivos.

- CREATE

  Este comando cria arquivos ou diretórios com características específicas (se forem colocadas como parâmetros) ou com características "default" (assumindo-se a série de parâmetros pré-determinados pelo sistema).

- DELETE

  Permite deletar um ou mais arquivos dependendo de como for feita a especificação (parcial ou total).

- DIFFERENCES

  Este utilitário compara o conteúdo de dois arquivos exibindo as diferenças no formato especificado pelo usuário. Ele permite visualizar diferenças "sutis" entre arquivos em quase tudo semelhantes.

Este utilitário será visto, em detalhes, mais adiante no capítulo sobre Desenvolvimento de Programas.

- DIRECTORY

Esse comando dá ao usuário informações sobre os arquivos catalogados num determinado diretório. Ele informa, entre outras características, a data da criação, número de bytes ocupados em disco, proprietário dos arquivos e proteção. Este comando pode atuar sobre um arquivo específico, um conjunto de arquivos ou sobre o diretório por completo. A determinação de um conjunto de arquivos pode ser feita através de uma especificação apenas parcial dos nomes dos arquivos desejados. Essa especificação parcial se utiliza de caracteres especiais (por exemplo, o asterisco) que expressam mais que uma letra ou dígitos determinados.

- DUMP

Através deste comando o usuário pode visualizar o conteúdo dos arquivos em mais de uma representação: ASCII ou hexadecimal, por exemplo. Se for desejado, pode inclusive exibir os campos de controle pertinentes à organização do arquivo (campos gerenciados pelo Sistema Operacional).

- EDIT

Este comando permite editar arquivos seqüenciais. Em geral, esses arquivos contém textos (programas-fonte, manuais e cartas, por exemplo) ou outro tipo de dados, como por exemplo, entrada para algum programa.

O sistema possui editores de linha e de tela.

Os editores de linha têm como unidade de alteração um registro (linha). Para fazer alguma alteração o usuário deve localizar-se na linha a ser modificada e, então, efetuar a operação. O usuário também pode comandar a execução de alterações num grupo de linhas.

Nos editores de tela, as alterações são feitas nas informações exibidas na própria tela do terminal, facilitando o trabalho do usuário.

A descrição dos editores será mais detalhada quando tratarmos de desenvolvimento de programas.

- OPEN | WRITE | READ | CLOSE

Esses comandos são utilizados para acesso a arquivos, a nível da linguagem de comandos (DCL). Eles servem, respectivamente, para abrir arquivos, gravar e ler registros, e fechar arquivos. Isso significa que pequenas aplicações que necessitem fazer acesso a arquivos (seqüenciais, relativos ou indexados) podem ser desenvolvidas através da própria DCL.

Estes comandos serão vistos, com mais detalhes, no capítulo sobre Desenvolvimento de Programas.

- PRINT

Através deste comando, o usuário comanda a impressão de seus arquivos colocando-os nas filas de impressão. O usuário pode comandar a impressão

após uma determinada hora. Pode também determinar a deleção do arquivo após a impressão.

- PURGE

Este comando deleta as versões anteriores de um determinado arquivo, mantendo somente as mais atuais. O número de versões a serem mantidas pode ser especificado através de um qualificador.

- RENAME

Este comando efetua a mudança de especificação (nome, tipo e versão) de um arquivo ou diretório.

- SORT

Este utilitário classifica arquivos através de uma ou mais chaves de classificação especificadas pelo usuário. A classificação pode ser feita em ordem ascendente ou descendente. Uma chave pode ser numérica ou alfanumérica. Após a classificação é gerado um novo arquivo.

- TYPE

Este comando permite ao usuário visualizar o conteúdo dos seus arquivos de dados, sem que seja necessária sua edição. Os arquivos são exibidos no terminal.

- Exemplo:

```
$ CREATE A.DAT
A1
A2
A3
^Z   (teclas CTRL Z pressionadas
      simultaneamente)
$ CREATE B.DAT
B1
B2
B3
^Z
$ DIR

Directory DRA1:[EXEMPLOS]

A.DAT;1
B.DAT;1

Total of 2 files

$ APPEND A.DAT B.DAT
$ TYPE B.DAT
B1
B2
B3
A1
A2
A3
$ DELETE A.DAT;1
$ SORT B.DAT C.DAT
$ DIR

Directory DRA1:[EXEMPLOS]

B.DAT;1
C.DAT;1

Total of 2 files
```

```
$ TYPE C.DAT
A1
A2
A3
B1
B2
B3
$ PRINT C.DAT
  Job 45 entered on queue SYS$PRINT
$ COPY C.DAT B.DAT
$ DIR

Directory DRA1:[EXEMPLOS]

B.DAT;2
B.DAT;1
C.DAT;1

Total of 2 files

$ PURGE B.DAT
$ DIR

Directory DRA1:[EXEMPLOS]

B.DAT;2
C.DAT;1

Total of 2 files

$ TYPE B.DAT
A1
A2
A3
B1
B2
B3
```

* Controle de execução e desenvolvimento de programas

- LIBRARY

  Este comando chama o utilitário de gerência de bibliotecas. Através dele, o usuário pode atualizar ou acessar bibliotecas de programas-fonte, programas-objeto ou textos.

  Este comando será visto, com mais detalhes, no capítulo sobre Desenvolvimento de Programas.

- LINK

  O usuário chama o ligador do sistema para gerar um programa executável (também chamado na literatura VAX de imagem) a partir de programas-objeto (resultado de compilações).

  Este comando será visto, com mais detalhes, no capítulo sobre Desenvolvimento de Programas.

- RUN

  O usuário pode, dessa forma, comandar a execução de um programa executável.

- SYNCRONIZE

  O usuário pode escrever um procedimento de comandos que, num determinado ponto, dependa para sua continuidade do término de uma tarefa sendo executada em "batch".

Através deste comando, ele pode suspender a sua execução até que esta outra tarefa em "batch" chegue a seu fim.

- WAIT

Este comando possibilita o atraso de um procedimento de comandos por um determinado período de tempo.

2.2 Gerência do Sistema

O gerente de sistemas autoriza usuários, planeja como será feita a proteção dos dados, fornece privilégios, controla a utilização de recursos, analisa relatórios de contabilização, instala atualizações do sistema e executa diagnósticos do equipamento.

Autorizar um usuário significa criar uma conta através da qual o usuário identifica-se para o sistema. Ela é necessária para iniciar uma sessão de terminal ou uma tarefa em "batch". A identificação da conta é feita através de um campo alfanumérico. Em geral, os usuários necessitam provar ao sistema que não estão usando uma conta indevidamente. Isso é feito quando o sistema exige a entrada de uma palavra reservada ("password") quando do "login" (entrada no sistema). A princípio, o único que deve ter conhecimento dessa "password" é o proprietário da conta.

Além de criar a conta, o gerente de sistema, para autorizar um usuário, necessita fornecer disco e diretório "default" para essa conta. Essa especificação determina em qual diretório dentro do disco o usuário estará "posicionado" após a entrada no sistema.

Também é necessária a especificação de privilégios da conta. Entre esses privilégios podem ser citados:

* criar e acessar "mailboxes" (ver capítulo sobre Sistema Operacional)
* inserir nomes lógicos em tabelas
* obter acesso exclusivo a volumes
* requisitar a montagem de volumes
* bloquear memória de processo contra paginação
* executar comandos específicos do operador
* suprimir a contabilização

A determinação dos privilégios necessários depende das aplicações que serão executadas por um usuário.

Além da identificação da conta, os usuários possuem um código, o UIC (User Identification Code, ou código de identificação do usuário). Ele visa possibilitar a criação do conceito de grupos de usuários. Este código é composto da identificação do grupo e do usuário dentro deste grupo.

Cada usuário está limitado a cotas de utilização dos recursos do sistema especificadas pelo gerente. Entre esses recursos estão:

* tempo gasto de UCP
* número de páginas de um processo em memória real

* número de páginas no arquivo de paginação (onde estão armazenadas as páginas de um processo quando estão em disco magnético)

* utilização de espaço em disco

A especificação de cotas visa impedir que usuários utilizem-se do sistema em detrimento de outros usuários.

O sistema permite a contabilização dos recursos utilizados por um determinado processo. Essa contabilização é feita visando uma posterior cobrança ou uma avaliação da distribuição de utilização dos recursos entre os usuários.

São contabilizados, entre outros recursos:

* tempo gasto de UCP

* tempo de conexão ao sistema

* número de volumes montados

* número de páginas impressas

* número de "page-faults" (referência a partes de um programa que, devido ao gerenciamento de memória virtual, se encontram em disco magnético e não em memória real)

* número de requisições de entrada e saída a periféricos lentos (requisições "bufferizadas")

* número de requisições de entrada e saída a periféricos rápidos (requisições diretas)

Cabe ao gerente de sistemas avaliar quais os recursos que serão utilizados para a "cobrança" ou avaliações citadas acima.

O gerente também deve avaliar, periodicamente, a performance do sistema, verificando se alguma medida pode ser tomada visando sua otimização. Algumas das características que devem ser analisadas são as seguintes:

* número de usuários interativos e executando tarefas em "batch"
* estados dos processos (aguardando a UCP, aguardando operação de entrada e saída, utilizando-se da UCP, por exemplo)
* atividade de "page-fault"
* atividade de rede (se computadores estiverem interligados)
* tempo de resposta (após solicitação de operação de entrada e saída)

O sistema já possui utilitários para monitoração dessas características, inclusive para tratamento dessas informações e emissão de relatórios úteis para a avaliação.

Com base nos resultados do estudo, o gerente de sistemas deve propor, se for o caso:

* a expansão dos recursos computacionais (expansão de memória),
* aumento do número de unidades e controladores de discos e fitas,
* uso de um processador da família VAX com maior capacidade de processamento ou

* mudança nas características de utilização do sistema (por exemplo, a restrição do número de usuários interativos, aumento das facilidades para utilização do processamento em "batch").

2.3 Operação do sistema

A operação do sistema VAX é bastante simples. A determinação da quantidade de operadores necessários para uma instalação depende, sobretudo, do número de usuários e do tipo de serviço executado. Quanto maior o número de impressoras, unidades de fitas e discos, maior será o número de operadores necessários.

As funções básicas para a operação do sistema são:

* ligar e desligar o sistema ("startup" e "shutdown")
* controle de tarefas (troca de prioridades, suspensão ou deleção de tarefas)
* alocação de periféricos
* serviço de montagem e desmontagem de volumes
* cópias de segurança do sistema
* controle de filas de impressão e de "batch"
* impressão periódica do registro de ocorrências/erros armazenado pelo sistema

O controle das filas de impressão deve ser efetuado quando ocorrerem situações anormais. O operador poderá redirecionar uma fila. Isso significa que se ocorrer algum problema numa determinada impressora, todos os arquivos presentes na fila associada a este periférico poderão ser redirecionados para outra impressora do sistema, caso exista.

Se ocorrer algum problema durante a impressão de um arquivo que inutilize algumas de suas páginas, sua reimpressão poderá ser comandada a partir da última página inutilizada. O operador pode mandar imprimir qualquer arquivo:

* a partir de qualquer página,

* um determinado número de páginas anteriores à última página impressa ou

* após ser encontrado, desde o início do arquivo, um conjunto de caracteres um determinado número de vezes.

Se durante a impressão de um arquivo, com um grande número de páginas, surgir a necessidade de impressão de um outro arquivo, o operador poderá suspender a impressão do arquivo "longo" na próxima "quebra" de página, imprimir o arquivo "prioritário" e reiniciar a impressão do primeiro arquivo na página que ocorreu a suspensão.

As filas de processamento de tarefas em "batch" devem sofrer algum controle, por parte do operador, quando o sistema estiver sobrecarregado ou subutilizado.

Na sobrecarga, o operador poderá restringir o número de novos usuários (interativos/"batch") a entrar no sistema, talvez até suspendendo a execução de alguma tarefa menos prioritária.

Na subutilização, o operador poderá aumentar o número de tarefas executadas concomitantemente numa mesma fila ou aumentar a prioridade de tarefas normalmente executadas com baixa prioridade.

Esses procedimentos devem ser definidos pelo gerente de sistemas para que não acarretem problemas entre os usuários.

O sistema oferece recursos para segurança contra falhas de força elétrica. A recuperação e reinicialização ocorre sem intervenção do operador. Uma bateria, que garante sua integridade por um período de dez minutos, pode ser adicionada na configuração, para maior confiabilidade dos dados em memória.

3. PROCESSADORES E PERIFÉRICOS VAX

3.1 Processador - Visão Geral

Basicamente, a arquitetura dos processadores VAX possibilita, a nível do "hardware", a gerência do espaço de endereçamento virtual. Possui, também, um conjunto de instruções (mais de 200) capaz de executar em modo nativo (família VAX) e em modo de compatibilidade (família PDP-11).

Os processadores VAX 11/780 e VAX 11/750 se utilizam de 32 bits para endereçamento virtual e de 24 bits para endereçamento real. O "hardware" permite o uso de até oito megabytes de memória real.

Vamos nos deter um pouco mais no VAX 11/750, o MX-850 da Elebra Computadores.

O ciclo de UCP é de 400 ns. Sua capacidade de processamento é de 720 mil instruções por segundo. O acesso à memória para leitura é de 800 ns para cada quatro bytes e de 640 ns para gravação.

À esta UCP pode ser adicionado um processador de ponto flutuante, que acelera as operações aritméticas, sobretudo, em aplicações científicas.

Sua taxa de transferência de entrada/saída é de cinco megabytes. Essa comunicação com os periféricos é feita através de "barras de dados" (buses). Existem dois tipos de "buses" nos equipamentos VAX. O primeiro é voltado para

dispositivos de baixa velocidade, chamado de UNIBUS. O segundo, chamado de MASSBUS, é voltado para dispositivos de alta velocidade.

As mesmas instruções que acessam a memória principal, acessam as "barras de dados", facilitando a programação da interface com o "hardware".

A memória principal pode ter de um a oito megabytes. Os incrementos são de um megabyte.

Os processadores são capazes de manipular vários tipos de dados. Com aritmética binária inteira, manipula dados de oito a 128 bits. Com aritmética real de ponto flutuante manipula dados de 32 a 128 bits, ou seja, o intervalo de valores pode chegar de 10^-4932 a 10^4932 com uma precisão de até 33 dígitos. Manipula cadeias de decimais compactados de até 16 bytes (31 dígitos), cadeias de caracteres de até 65535 bytes, campos de bits de até 32 bits e possui instruções para manipulação de elementos de dados estruturados em listas.

A descrição detalhada dos tipos de dados pode ser vista na tabela a seguir.

O processador efetua suas funções baseado em informações de um "contexto". Esse contexto inclui a definição de um espaço de endereçamento no qual é executado um programa e o estado de processamento.

O processador e o sistema operacional gerenciam o ambiente multiprogramável dividindo o espaço virtual de cada processo em duas metades. Uma das metades é dependente do processo (contexto), contendo informações (dados e código) de um usuário. A outra metade é independente do contexto, o significado de cada endereço é o mesmo para todos os processos e contém informações referentes ao sistema operacional.

Isso não significa que o sistema operacional ocupe área de memória real em todos os processos. Só existe uma cópia do sistema operacional em memória real. Ocorre um mapeamento do sistema no espaço de endereçamento virtual de todos os processos.

TIPO	TAMANHO	INTERVALO	
INTEIRO		COM SINAL	SEM SINAL
BYTE	8 BITS	-128/127	0/255
WORD	16 BITS	-32768/32767	0/65535
LONGWORD	32 BITS	-2^31/2^31	0/2^32-1
QUADWORD	64 BITS	-2^63/2^63	0/2^64-1
OCTAWORD	128 BITS	-2^127/2^127	0/2^128-1
PONTO FLUTUANTE			DÍGITOS
F_FLOATING	32 BITS	10^-37/10^-38	7
D_FLOATING	64 BITS	10^-37/10^-38	16
G_FLOATING	64 BITS	10^-308/10^308	15
H_FLOATING	128 BITS	10^-4932/10^4932	33
DECIMAIS COMPACTADOS	0 A 16 BYTES		31
CADEIA DE CARACTERES	0 A 65535 BYTES		
CAMPO DE BITS	0 A 32 BITS		
CADEIA NUMÉRICA	0 A 31 BYTES	-10^31-1/10^31-1	
LISTA	PELO MENOS 2 LONGWORDS POR NÓ		

Obs: O símbolo ^ indica uma operação aritmética de exponenciação.

Mapeamento no espaço de endereçamento virtual significa estabelecer uma referência entre o conteúdo da memória real e as referências de endereços virtuais de cada processo.

Os programas de usuário e o sistema endereçam a primeira metade, chamada de espaço do processo. Apenas o sistema operacional referencia a segunda metade, chamada de espaço do sistema.

Esse tipo de arquitetura é uma garantia de proteção dos dados específicos do sistema. Além disso, os usuários não podem referenciar endereços virtuais que não estejam dentro do seu próprio espaço de endereçamento. Isso garante a proteção de memória entre os usuários.

Uma importante característica da arquitetura VAX para a multiprogramação é a maneira com que transfere o controle da UCP de um processo para outro (chamada de mudança de contexto). O processador possui instruções especiais em "assembler" para tal fim. Em operação normal, somente ao sistema operacional é dado privilégios para efetuar tal mudança. Quando ela ocorre, a UCP passa a reconhecer outro fluxo de código e dados. Em outras palavras, outro espaço de endereçamento.

3.2 Registradores

Como já foi dito, o processador possui 16 registradores de 32 bits. Quatro desses registradores são usados para funções especiais. Eles são:

* "program counter" (apontador de instruções)

* "stack pointer" (apontador de pilha)

* "argument pointer" (apontador de argumentos)

* "frame pointer" (apontador de "molduras" de chamadas a rotinas)

O "program counter" aponta sempre para a próxima instrução a ser executada. A arquitetura VAX possui um modo de endereçamento de memória baseado no "program counter" que permite ao usuário escrever código independente da posição de memória que o programa estiver posicionado. A esse tipo de endereçamento dá-se o nome de PIC (Position Independent Code, ou código independente de posição). Esse é o código normalmente gerado por todos os compiladores.

O "stack pointer" é um registrador projetado para manipular estruturas de pilhas (estrutura dinâmica de dados).

O "argument pointer" é usado para passar a uma sub-rotina o endereço de uma lista de parâmetros sobre os quais deverá atuar.

O "frame pointer" é usado para rastrear instruções encadeadas de chamadas de sub-rotinas ("CALL"). Através desse encadeamento, é possível para o sistema, quando do término anormal de um programa, reconhecer a seqüência de chamadas de sub-rotinas antes da ocorrência do erro. Este tipo de procedimento facilita a depuração de programas.

3.3 Modos de endereçamento

Cada modo de endereçamento indica uma maneira através da qual o processador interpreta a referência de endereço de dados de uma instrução "assembler".

O processador possui os seguintes modos de endereçamento:

* register mode
* register-deferred mode
* auto-decrement mode
* auto-increment mode
* auto-increment deferred mode
* displacement mode
* literal mode

No "register mode", o registrador utilizado na instrução contém o dado.

No "register-deferred mode", o registrador contém o endereço do dado (endereçamento direto).

No "auto-decrement mode", o conteúdo do registrador é decrementado do tamanho do dado (depende do tipo de dado acessado pela instrução) e, então, é usado como endereço do dado.

O "auto-increment mode" é semelhante ao "auto-decrement mode". Entretanto o conteúdo do registrador é incrementado.

No "auto-increment deferred mode", o conteúdo do registrador é usado como endereço de memória que contém o endereço do dado. Após o acesso, o conteúdo do registrador é incrementado.

No "displacement mode", o conteúdo do registrador é usado como uma base de endereçamento de memória. Uma constante, embutida na instrução, é somada ao registrador e, então, usada como endereçamento do operando.

No "displacement-deferred mode", o conteúdo do registrador é usado como base de uma tabela de endereços. Uma constante (embutida na instrução) é somada ao registrador, e a soma é o endereço de memória, cujo conteúdo é o endereço do dado.

Com exceção do "register mode", todos os modos de endereçamento podem ser modificados por um índice.

No "literal mode", um dado binário está embutido na própria instrução.

Todos esses modos de endereçamento possibilitam um acesso mais rápido a estruturas de dados mais complexas.

3.4 Periféricos

Serão apresentados os periféricos (terminais, unidades de disco, fitas magnéticas e impressoras) comercializados pela Elebra Computadores no Brasil.

3.4.1 Terminais

* EC1100

  É um terminal assíncrono alfanumérico no qual é possível a reconfiguração, por "software", de parâmetros de funcionamento, tais como: brilho, vídeo reverso e velocidade de transmissão. A comunicação assíncrona é feita no padrão RS-232 em full-duplex com velocidade de transmissão de 75 a 19200 bps.

* VT240 e VT241

Esses terminais, além das características alfanuméricas, oferecem recursos gráficos com resolução de 800 por 240 pontos. Também é reconfigurável por "software". A comunicação com o computador é feita de modo semelhante a do EC1100.

3.4.2 Impressoras

* EC8035

É uma impressora matricial projetada para impressão de textos e gráficos com alta resolução. Sua velocidade de impressão, em modo alfanumérico, é de 180 caracteres por segundo. Sua densidade, em modo gráfico, pode chegar a 21600 pontos por polegada quadrada. Ela pode ser conectada através de interface serial ou paralela.

* EC8040

Também é uma impressora matricial para impressão de textos. Possui, entretanto, velocidade de 400 caracteres por segundo.

* EC8030 e EC8060

São modelos de velocidade de 300 e 600 linhas por minuto, respectivamente. São conectadas através de interfaces paralelas, padrão Dataproducts ou Centronics.

* EC8120

É um modelo com velocidade de 1200 linhas por minuto. É conectada diretamente à barra de dados UNIBUS.

3.4.3 Unidades de discos

* EC9340 e EC9500

São unidades de discos seladas, de tecnologia Winchester, com capacidade de armazenamento de 256 e 436 megabytes, respectivamente. Possuem uma "panela" de sete discos rígidos não removíveis, de face dupla.

* RA81

É uma unidade de disco selada, de tecnologia Winchester, com capacidade de 456 megabytes numa "panela" de quatro discos rígidos, não removíveis, de face dupla.

3.4.4 Unidades de fita

* EC9180

Permite a leitura e gravação de dados a 1600 e 6250 bits por polegada. Um rolo de 2400 pés possui, quando formatado pelo sistema, capacidade de 37 e 123 megabytes, respectivamente.

4. SISTEMA OPERACIONAL VAX/VMS

4.1 Módulos do Sistema

O sistema operacional é composto de vários módulos. Vamos, resumidamente, conhecê-los.

O interpretador de comandos recebe as informações dos usuários interativos e das tarefas em "batch", e comanda os procedimentos necessários.

As rotinas de gerência de memória executam o mapeamento de memória real e virtual, carregam programas executáveis e efetuam a paginação de memória.

Existe uma série de rotinas que são compartilhadas em tempo de execução, isto é, elas estão permanentemente em memória e uma única cópia é utilizada por vários programas. São chamadas de rotinas da "Run-Time Library", ou seja, rotinas da biblioteca de tempo de execução. Este tipo de procedimento diminui a área de código de cada usuário.

O "swapping" (retirada/entrada total de páginas de um processo em memória real) também é efetuado por rotinas do sistema operacional.

Os serviços de gerência de arquivos e registros efetuam para o usuário o acesso aos arquivos do sistema.

Todos os compiladores geram chamadas para esses serviços, permitindo que arquivos sejam manipulados por aplicações escritas em diferentes linguagens.

Existem as rotinas de processamento de entrada/saída a nível de periféricos e as de tratamento de interrupções. Essas rotinas fazem a ligação entre o "hardware" e o "software".

Para que seja possível a execução de programas desenvolvidos para ambientes PDP-11, existem as rotinas de execução em modo de compatibilidade, que "traduzem" as chamadas feitas para o sistema operacional dos computadores PDP-11 (Sistema RSX-11) para as chamadas do VAX.

O sistema operacional também efetua a chamada de rotinas para tratamento de exceções tanto do "hardware" (por exemplo, erro nos dispositivos periféricos) como no "software" (por exemplo, divisão por zero). Esse tratamento pode ser feito tanto de uma forma padrão (conforme usualmente definido pelo sistema), quanto de uma forma especificada pelo usuário.

Essa interação Sistema Operacional/Programa de Usuário não existe só para o tratamento de exceções. A concepção do "software" do sistema permite que os usuários organizem seus próprios discos (utilização específica das trilhas e setores dos discos) e definam seus próprios protocolos com terminais específicos. Dessa forma, o usuário adapta o Sistema Operacional, no que ele tem de genérico, para suas aplicações visando, em geral, uma melhor performance.

Os serviços do sistema (Systems Services) são as funções básicas para requisição e alocação de recursos como o processamento de entrada e saída e a comunicação entre processos. As outras funções disponíveis (como por exemplo, o acesso aos arquivos) são derivadas dessas funções básicas.

Podemos citar os seguintes serviços do sistema:

* estabelecimento de canal de entrada/saída
* montagem de volumes
* obtenção de informações dos dispositivos periféricos
* alocação de dispositivos
* requisição de entrada/saída
* execução de saídas formatadas
* criação e comunicação através de "mailboxes"
* envio de mensagens para outros usuários
* criação e tradução de nomes lógicos
* bloqueio de recursos
* criação de novos processos
* obtenção de informação de processos
* hibernação e suspensão de processos
* reativação de processos
* alteração de prioridades e privilégios
* obtenção de data e hora
* criação de espaço virtual
* criação e acesso a seções globais (espaço de endereçamento compartilhado)
* proteção de páginas em memória

Existem módulos que controlam a alocação de recursos, efetuam a comunicação com o operador do sistema, e os que fazem o registro de erros e ocorrências, tais como, montagem e desmontagem de periféricos ocorridas durante a operação normal do sistema.

Existe um outro módulo do sistema responsável pelo escalonamento de processos.

4.2 Escalonamento de processos

Escalonamento de um processo significa passar o controle da UCP para este processo.

O processo que ganha o controle da UCP é aquele que possui a prioridade mais elevada num dado instante, e não está esperando nenhum evento do sistema.

Os processos, em geral, aumentam sua prioridade quando uma entrada/saída num terminal é efetuada ou quando um recurso, pelo qual estava esperando, torna-se disponível.

Os processos, usualmente, diminuem de prioridade após terem sido escalonados.

Como foi visto, os processos possuem prioridade dinâmica e se utilizarão da UCP por um período mínimo de tempo, ou até que uma requisição de operação de entrada/saída seja efetuada.

O "swapper" (módulo responsável pelo "swapping") garante ao escalonador de processos que aquele processo de maior prioridade possuirá um número mínimo de páginas em memória real, quando do seu escalonamento.

O sistema trata, diferentemente, processos interativos/"batch" de processos de tempo real.

Processos de tempo real são aqueles que necessitam de um tempo de resposta do sistema muito menor que os outros tipos de processos. Diz-se que são processos críticos em tempo.

Esse tipo de processo, em geral, só é escalonado em instalações VAX que sejam utilizadas na área de controle de processos.

Esses processos nunca perdem o controle da UCP, a não ser que entrem em estado de espera. Suas prioridades são superiores as dos processos do sistema operacional. Eles não possuem prioridade dinâmica, e sim, estáticas. Com isso, o sistema garante um pronto atendimento para esse tipo de aplicação.

4.3 Sistema de Memória Virtual

VMS significa "Virtual Memory System" (Sistema de Memória Virtual). Neste tipo de sistema, um programa de um usuário não necessita estar totalmente na memória real para sua execução. Um programa é dividido em partes (páginas), cada uma com 512 bytes e somente as partes que estiverem sendo referenciadas são mantidas na memória. As outras são armazenadas em disco magnético.

O sistema gerencia, por processo, o conjunto de páginas residentes em memória ou em disco.

O conjunto de páginas residentes em memória pode ir aumentando ou diminuindo, dependendo da disponibilidade de memória real e do número de processos executando simultaneamente.

Este número será reduzido até um mínimo, quando a carga do sistema estiver elevada. O sistema tenta reduzir o número de páginas residentes de cada processo, para tentar alocar mais processos concorrentes na memória real.

Quando um processo obtiver o ganho da UCP, for o de maior prioridade, e se todos os outros processos já estiverem com seu número de páginas em memória reduzidos ao máximo, será, então, necessária a retirada total das páginas de um desses processos para dar lugar ao processo prioritário.

Quando um processo faz referência à uma página ausente da memória real (armazenada em disco magnético) diz-se que ocorreu um "page-fault". O sistema se encarregará de acessar essa página em disco, tornando-a disponível, na memória real, para o processador.

À função de gerenciar páginas em memória/disco, transferindo-as de um meio para outro, dá-se o nome de paginação. À função de retirar totalmente um processo de memória e depois retorná-lo, dá-se o nome de "swapping".

O gerenciamento de memória (paginação e "swapping") é totalmente transparente para o usuário. Os programas podem ser escritos e testados, não sendo necessárias preocupações especiais quanto ao espaço de memória utilizado durante esta fase do desenvolvimento.

Na grande maioria das aplicações não é preciso nenhum tipo de otimização devido à capacidade computacional do equipamento. Entretanto, em aplicações especiais, isso pode vir a ser necessário e a redução da paginação/swapping (a partir de uma melhor estruturação dos dados e do código) pode otimizar a performance dos programas.

4.4 Comunicação entre processos

Nas aplicações mais usuais, os usuários estão preocupados que programas de outros usuários não interfiram com os seus. Em aplicações mais sofisticadas, entretanto, vários programas podem interagir de forma coordenada.

Esta interação pode ser feita de várias formas no Sistema VAX:

* através do compartilhamento de arquivos (onde dois ou mais usuários acessam um mesmo arquivo, ao mesmo tempo, de maneira ordenada),

* através de "mailboxes" ("caixas de correspondência" através das quais os usuários passam mensagens) ou

* através do compartilhamento do espaço de endereçamento de memória (dois ou mais usuários passam a usar a mesma área de memória conseguindo dessa forma a troca de informações).

4.5 Segurança do Sistema

A segurança do sistema, e especialmente dos dados, é uma das principais preocupações numa instalação com inúmeros usuários. É necessário que todos os usuários tenham garantido a confidencialidade de seus dados e que, ao mesmo tempo, exista um grau de interação de dados entre os usuários da comunidade, dentro de certas regras predeterminadas.

A proteção de dados, utilizando-se do conceito de grupo/usuário (UIC), é um método interessante para restrição de acesso a usuários que não pertençam ao mesmo grupo que o proprietário.

Para conseguir-se uma especificação de acesso de dados mais precisa, podem ser utilizados identificadores e listas de controle de acesso a arquivos.

O gerente de sistemas passa a atribuir a um usuário, além de uma identificação (conta) e um código (UIC), uma série de identificadores que lhe forem apropriados.

Um identificador pode representar um único usuário, uma comunidade de interesse, ou um tipo de trabalho ou projeto.

Um usuário pode então, quando for informar ao sistema a permissão de acesso de seus arquivos de dados, ler a base de dados contendo os identificadores existentes ("Rights Database" ou base de dados de direitos) e, a partir desse ponto, especificar sua lista de controle de acesso. Essa lista conterá identificadores e tipos de acesso permitidos para cada um dos identificadores.

Os usuários, que desejarem acessar um arquivo, deverão possuir um identificador presente na lista, e ter associado a esse identificador o tipo de acesso desejado.

Falamos até agora de segurança de dados. O sistema oferece, também, recursos para proteger-se contra o uso indevido de terminais. Isso é feito através de características que podem ser atribuídas a um terminal. Além da palavra reservada associada a identificação da conta, pode existir uma palavra reservada para o terminal, restringindo seu uso aos usuários que conhecerem esta senha.

Para prevenir o uso indevido de uma conta, o sistema pode caracterizar um usuário como intruso se, após inúmeras tentativas de efetuar um "login" num determinado terminal, não conseguir especificar a palavra-chave associada a identificação da conta informada. Este terminal pode ficar, então, bloqueado durante um certo período de tempo.

Para que o usuário tome conhecimento da utilização de sua conta, o sistema sempre fornece, quando da conexão, a informação do último "login" realizado (interativo e "batch") e das tentativas infrutíferas de "logins".

5. LINGUAGENS

Existem diversos compiladores e interpretadores desenvolvidos especialmente para o VAX/VMS.

Estes produtos podem ser divididos em dois grandes grupos:

* os que foram desenvolvidos pela DIGITAL e

* os que foram desenvolvidos por outras empresas.

A diferença entre estes dois grupos pode ser resumida nos seguintes itens:

* A DIGITAL conhece a fundo todos os seus produtos de software existentes (e planejados). Logo pode aumentar a compatibilidade dos seus compiladores entre si, e com outros produtos.

* Existe um "padrão de chamada de rotinas", que não são sempre seguidos por compiladores desenvolvidos por outras empresas.

A grande variedade de aplicações do VAX explica a quantidade de compiladores existentes. Existem compiladores para linguagens científicas, comerciais, para confecção de software básico, sistemas de tempo real e inteligência artificial.

Os principais compiladores desenvolvidos pela DIGITAL, atualmente, são os seguintes:

* VAX-11 FORTRAN
  (processamento científico)

* VAX-11 COBOL
  (processamento comercial)

* VAX-11 BASIC
  (instrução, uso genérico)

* VAX-11 C
  (software básico)

* VAX-11 PASCAL
  (instrução, uso genérico)

* VAX-11 PL/I
  (processamento científico e comercial)

* VAX-11 ADA
  (uso genérico, software de tempo real)

* VAX-11 RPG II
  (processamento comercial, geração de relatórios)

* VAX-11 BLISS-32
  (sistemas operacionais)

* VAX-11 CORAL-66
  (sistemas de tempo real)

* VAX-11 MACRO
  (software básico)

Outros compiladores que podemos citar são os seguintes:

* LISP
  (inteligência artificial)

* PROLOG
  (inteligência artificial)

* MODULA
  (sistemas operacionais, uso genérico)

* MUMPS
  (processamento comercial)

* APL
  (uso genérico, processamento científico)

Apresentaremos algumas características importantes das implementações em VAX das seguintes linguagens:

* VAX-11 FORTRAN

* VAX-11 COBOL

* VAX-11 BASIC

* VAX-11 C

* VAX-11 PASCAL

5.1 VAX-11 FORTRAN

- Implementação do FORTRAN (FORmula TRANslation) ANSI X3.9-1978 (FORTRAN 77), com extensões;

- Os nomes de variáveis podem ter até 31 caracteres, tornando mais confortável o uso de variáveis cujo nome indica a sua função;

- Permite o uso de constantes hexadecimais e octais;

- Existem rotinas para tratamento de informações a nível de bit;

- Comandos estruturados como "IF ... THEN ... ELSE" e "DO WHILE ... END DO";

- Variáveis e operadores lógicos, como por exemplo AND, OR, NOT, XOR e EQU;

- Acesso, via rotinas do VMS, a arquivos seqüenciais indexados;

- Comando "INCLUDE", que incorpora o texto de outro arquivo em tempo de compilação;

- Quando é usada uma variável REAL, como índice de uma variável subscrita, o valor correspondente é transformado em inteiro;

- Além dos comentários que são indicados pelo caracter "C" na coluna 7, é possível a inserção de comentários, ao final da linha de comando, delimitados pelo caracter "!".

5.2 VAX-11 COBOL

- Implementação do COBOL (COmmon Business Oriented Language) ANSI X3.23-1974;

- Possui comandos que permitem o acesso a banco de dados modelo CODASYL, da própria DIGITAL (VAX-11 DBMS);

- Possui comando estruturado para seleção, que implementa uma tabela verdade ("EVALUATE");

- Permite o uso de delimitadores explícitos, para indicar final de comando, como por exemplo "IF .. END-IF", "READ ... END-READ", etc;

- Possui comando "PERFORM" implícito, ou seja, sem que haja necessidade de definir o parágrafo a ser executado ("PERFORM ... END-PERFORM");

- Implementa REPORT WRITER (comandos específicos para geração de relatórios);

- Os comandos que fazem acesso a arquivos possuem extensões para controle de acesso concorrente. Por exemplo, um registro pode ser bloqueado no comando de leitura, de tal forma que outro programa não possa acessá-lo;

- Os comandos de acesso a vídeo ("ACCEPT" e "DISPLAY") podem ter endereçamento de cursor e outras características como por exemplo, "DISPLAY" com campo reverso e sinal sonoro, e "ACCEPT" com um valor "default" associado;

5.3 VAX-11 BASIC

- O BASIC (Beginners All-purpose Simbolic Instruction Code) do VAX pode ser executado de dois modos: interpretado e compilado;

- Os nomes de variáveis podem ter até 30 caracteres, tornando mais confortável o uso de variáveis cujo nome indica a sua função;

- O tamanho máximo do valor de uma variável alfanumérica é 64 K bytes (65.535);

- Permite que vários comandos sejam colocados em uma linha;

- Possui comando estruturado "IF ... THEN ... ELSE", que pode se estender por várias linhas.

5.4 VAX-11 C

- Implementação do C descrito em "The C Programming Language" (Kernighan & Ritchie, 1978) [1];

- Como o C está intimamente ligado ao sistema operacional UNIX, que não é o sistema operacional do VAX, existem algumas chamadas ao sistema que não foram implementadas.

[1] "C - Linguagem de Programação", publicado em língua portuguesa pela Editora Campus, 1986

5.5 VAX-11 PASCAL

- Implementação do PASCAL descrito em "PASCAL User Manual and Report" (Jensen & Wirth, 1974) [2];

- Permite a definição de módulo, que é um conjunto de rotinas associadas a uma única definição de dados;

- Extensão "OTHERWISE" para o comando "CASE";

- Extensões %DESCR, %STDESCR e %IMMED para passagem de parâmetros por descritor, por descritor de uma cadeia de caracteres e por valor imediato, respectivamente. Um descritor é um conjunto de informações que descreve as características de uma variável (ou parâmetro), tais como tipo e tamanho;

- Comando "INCLUDE", que incorpora o texto de outro arquivo em tempo de compilação;

- Operador aritmético de exponenciação;

- Permite o uso de constantes hexadecimais e octais;

- Permite a leitura e a gravação de tipos escalares (enumerados) definidos pelo usuário. Um tipo escalar é aquele que determina um conjunto de possíveis valores de uma variável;

- Acesso via comandos a arquivos com organização relativa e seqüencial indexada.

[2] "Pascal - Manual do Usuário" (título provisório) a ser publicado em língua portuguesa pela Editora Campus.

6. DESENVOLVIMENTO DE PROGRAMAS

Existem diversos recursos para desenvolvimento de programas no sistema operacional VMS. Estes recursos formam um "ambiente de desenvolvimento".

Os recursos existentes vão desde utilitários com objetivos específicos até bibliotecas de rotinas. Entre os utilitários podemos citar um poderoso editor de textos e um depurador simbólico.

A seguir, serão apresentados os recursos e, em seguida, será mostrado como esses recursos podem ser relacionados.

6.1 Editor de Textos

Existem dois editores de texto para o VAX, porém só é necessário aprender (e compreender) um deles.

Nas primeiras versões do VMS, o "editor principal" era o SOS, que é um editor orientado à edição de linhas. Este editor ainda está presente na versão atual do VMS, embora não seja mais o principal.

O editor principal agora chama-se EDT, que é extremamente interativo e que permite tanto a edição de linhas quanto a edição de tela cheia ("full screen").

Uma característica importante dos dois editores de texto é que podem ser utilizados tanto em modo "batch" quanto em modo interativo. Não é necessário aprender comandos de vários editores diferentes, que só podem ser usados em um dos principais modos de processamento ("batch"/interativo).

O editor de textos EDT é capaz de editar arquivos cujo conteúdo são caracteres, como programas-fonte e textos para documentação.

O comando para ativação do EDT é o seguinte:

$ EDIT <arquivo>

Após a digitação do comando acima, será aberta uma sessão de edição em <arquivo>. Se este ainda não existir, o EDT avisará ao usuário.

Aberta a sessão, existem dois modos de uso do editor: modo de linha e modo de tela cheia. Em sessões no modo de operação "batch", apenas o modo de linha é permitido.

Em termos de comandos disponíveis existem aqueles que só podem ser utilizados em modo de linha, aqueles que só podem ser utilizados em modo de tela cheia e aqueles que podem ser utilizados em ambos os modos.

No modo de linha, a cada linha está associado um número. Existem comandos para inserir, deletar, copiar e mover um conjunto de linhas. Este conjunto é identificado por sua linha inicial e por sua linha final.

No modo de tela cheia, as linhas não são numeradas. Existe movimentação do cursor nas quatro direções (para cima, para baixo, para esquerda e para direita) e teclas especiais que permitem:

* ir para o início/final do arquivo,

* ir para o início/final da linha,

* buscar uma cadeia de caracteres e

* deletar um caracter, uma palavra ou uma linha.

São muitos os comandos disponíveis. Existe um comando que mostra quais são estes comandos. Este comando está presente de duas formas: em modo de edição de linha ele se chama HELP e em modo de edição de tela cheia ele está associado à tecla <F2>.

Um conceito importante existente no EDT é o de "buffers" (áreas de trabalho). Quando um usuário abre uma sessão de edição, existem dois "buffers": MAIN e PASTE. Em geral um usuário trabalha no "buffer" MAIN, pois é para lá que um arquivo a ser editado é inicialmente copiado e ao final da sessão de edição é de lá que são lidos os dados para gravação do novo arquivo (editado).

O "buffer" PASTE é usado principalmente para deleção, cópia e movimentação de um conjunto de linhas quando se está em modo de edição de tela cheia. O procedimento é o seguinte:

* o usuário move o cursor até o primeiro caracter do conjunto de linhas,

* aciona a tecla especial <SELECT>,

* move o cursor para o último caracter do conjunto de linhas e

* aciona a tecla especial <CUT>.

O conjunto de linhas selecionado é copiado para o "buffer" PASTE e retirado do "buffer" MAIN. O retorno do conjunto de linhas para o "buffer" MAIN é feito no ponto onde se encontra o cursor, após o acionamento da tecla especial <PASTE>.

Esta característica do EDT de trabalhar com "buffers" permite a inclusão de um arquivo em um "buffer" (e posterior inserção no "buffer" MAIN, se for o caso), gravação de partes de um arquivo, e edição de vários arquivos simultaneamente (um em cada "buffer").

Outra característica do EDT está relacionada à segurança da sessão de edição. Um qualificador do editor (que pode ser desativado) indica que todas as operações executadas no arquivo que está sendo editado, além de produzirem o efeito desejado, serão registradas em um arquivo. Este arquivo é um registro de tudo que foi feito durante a sessão de edição. O seu nome é o mesmo nome do arquivo editado, sendo que o seu tipo é JOU ("Journal"). O uso principal deste arquivo JOU é na recuperação da edição, em caso de término anormal (por exemplo, queda do sistema por falta de energia elétrica).

A última característica a ser apresentada trata da parametrização do editor EDT. Por parametrização entendemos a definição do estado inicial da sessão da edição. Desta parametrização fazem parte definições de:

* uma palavra e uma frase (através da identificação dos delimitadores),

* estado inicial de edição (linha ou tela) e

* funções associadas a teclas (definição de teclas especiais).

A parametrização pode ser executada automaticamente toda vez que o editor é utilizado.

6.2 Editor de Ligações

O objetivo principal do editor de ligações do VMS é combinar um conjunto de módulos-objeto (resultados de compilação) para formar uma imagem executável. Uma imagem é, na literatura VAX, um programa.

Alguns sistemas operacionais usam uma combinação de compilador e carregador ("loader") para formar uma imagem executável.

A grande vantagem de se combinar diversos módulos-objeto em uma imagem é que os módulos-objeto podem ser resultado de compilações de linguagens diferentes, ou seja, uma imagem executável pode ser criada a partir de programas e rotinas codificadas em diferentes linguagens.

Esta combinação só funciona sem maiores problemas porque existe um "padrão de chamada" de rotinas no VMS.

O editor de ligações aceita vários tipos de entrada e pode produzir vários tipos de saída.

Vários arquivos de entrada podem ser especificados porém, apenas um arquivo de saída é gerado.

O formato do comando é o seguinte:

```
$ LINK <módulo-objeto> [,<arquivo> [,<arquivo>]..
```

O tipo do <módulo-objeto> em geral é OBJ, e a saída sempre será do tipo EXE. O <arquivo> pode ser um outro módulo-objeto ou biblioteca de módulos-objeto.

As entradas possíveis são as seguintes:

* Módulo-Objeto

  Resultado de uma compilação. Tem que existir pelo menos um.

* Imagem Compartilhável

  Resultado de uma execução anterior do editor de ligações, porem não executável. É usada para economia de espaço em disco. Deve ser ligada a uma imagem executável para poder ser utilizada.

* Tabela de Símbolos

  Tabela com definição de símbolos associados a valores, que é resultado do editor de ligações, porém não executável.

* Biblioteca

  Biblioteca de módulos-objeto ou de imagens compartilháveis.

* Opções

  Parâmetros para o editor de ligações, em geral usados para operações complexas.

A saída será de um dos seguintes tipos:

* Imagem Executável

  Saída mais comum, que pode ser executada pelo comando "$ RUN".

* Imagem Compartilhável

  Não pode ser executada. Precisa ser agregada a um módulo-objeto em uma outra fase de ligação.

* Imagem de Sistema

  Imagem executável que não é executada sob o controle do VMS (por exemplo, um sistema operacional).

* Tabela de Símbolos

  Conjunto de símbolos.

Exemplos:

```
$ PASCAL PROG              ! Compila PROG.PAS
$ LINK PROG                ! Liga PROG.OBJ
$ RUN PROG                 ! Executa PROG.EXE
```

Este é o procedimento mais comum, para a preparação de um programa executável.

```
$ COBOL MENU               ! Compila MENU.COB
$ PASCAL ROT1              ! Compila ROT1.PAS
$ FORTRAN ROT2             ! Compila ROT2.PAS
$ LINK MENU,ROT1,ROT2      ! Liga os .OBJ
$ RUN MENU                 ! Executa MENU.EXE
```

Neste exemplo o programa MENU, codificado em COBOL chama duas rotinas: ROT1, codificada em PASCAL e ROT2, codificada em FORTRAN. É na fase de ligação que todos os módulos-objeto (OBJ) são ligados, para formar um único programa executável. O programa principal (MENU neste exemplo) tem que ser o primeiro da lista de arquivos, no comando LINK.

```
$ COBOL MENU               ! Compila MENU.COB
$ LINK MENU,ROTS.OLB       ! Liga .OBJ usando ROTS
$ RUN MENU                 ! Executa MENU.EXE
```

Aqui um programa principal é ligado a uma biblioteca de módulos-objeto. Neste exemplo, os módulos ROT1 e ROT2 (já compilados) que estão na biblioteca ROTS serão ligados ao programa principal (caso o programa MENU só use estas duas rotinas "externas").

6.3 Biblioteca de Rotinas "Run-Time"

Existe no VMS dois conjuntos de rotinas genéricas, disponíveis ao usuário. O primeiro conjunto trata de "serviços do sistema". O segundo conjunto é conhecido por rotinas "Run-Time" (rotinas que são chamadas em tempo de execução de um programa).

A grande vantagem de rotinas "Run-Time" é que elas podem ser alteradas, sem que haja necessidade de re-compilar ou re-ligar programas. Outra característica interessante é que esta biblioteca é compartilhável, ou seja, só precisa existir uma cópia para uso de todos os usuários.

Estas rotinas "Run-Time" tratam dos seguintes temas:

* Alocação de recursos
* Tratamento de condições
* Uso geral
* Funções matemáticas
* Suporte a todas as linguagens
* Suporte a uma linguagem específica
* Tratamento de cadeia de caracteres

Estas rotinas podem ser usadas por programas codificados em qualquer linguagem. Isto é possível porque todos os compiladores DIGITAL, quando se trata de chamada de rotinas, usam um "padrão de chamada". Este padrão diz respeito a

* seqüência de ações necessárias para salvar o contexto (registradores) do programa chamador,
* identificação da forma de passagem de parâmetros para a rotina,
* ativação da rotina e
* restauração do contexto do programa.

Daí ser possível fazer um programa em PASCAL, que chama rotinas em FORTRAN, BASIC e COBOL, e que usa rotinas "Run-Time".

6.4 Depurador Simbólico Interativo

O depurador simbólico interativo do VMS (DEBUGGER) tem por objetivo auxiliar um usuário na depuração de programas. Ele pode ser usado em programas codificados em qualquer linguagem.

Algumas características importantes são:

* capacidade de verificar e alterar o conteúdo de posições de memória
* referenciar posições de memória via símbolos (nome de variáveis, nome de parágrafos)
* apresentar os dados de várias formas (ASCII, hexadecimal)
* ser totalmente interativo.

O uso do depurador é feito da seguinte forma:

* compila-se o programa com o qualificador /DEBUG

* liga-se o programa-objeto e outros arquivos também com o qualificador /DEBUG

* executa-se o programa

Assim que o programa começa a ser executado, o depurador entra em ação, permitindo o controle do programa.

Os principais comandos do depurador são os seguintes:

* SET BREAK

  Permite determinar pontos de parada ("break points") da execução do programa. Estes pontos podem ser indicados pelo número da linha no programa-fonte (indicado pelo compilador) ou por uma posição de memória indicada simbolicamente, como nome de parágrafo (COBOL). Vários pontos podem estar ativos simultaneamente.

* GO

  Executa o programa. Se existirem pontos de parada, o programa será executado até o próximo ponto senão, será executado até o final.

* STEP

  Executa o programa passo a passo. A cada comando STEP o depurador executa um comando do programa.

O comando pode ser tanto a nível da linguagem na qual o programa foi codificado, quanto a nível de linguagem de máquina.

* SET STEP SOURCE

Permite acesso ao programa-fonte, durante a depuração. O programa-fonte pode ser mostrado, segundo um intervalo de linhas. Se o usuário estiver usando o comando STEP, o depurador mostra, a cada passo, a linha do programa-fonte que foi executada.

* SET TRACE

Mostra a seqüência de comandos que está sendo executada. Os comandos a serem acompanhados devem ser indicados, da mesma forma do comando SET BREAK. Vários comandos podem ser especificados.

* SET WATCH

Permite o acompanhamento das mudanças do conteúdo de uma variável. Este comando mostra o conteúdo anterior e o atual a cada modificação da variável. Diversas variáveis podem ser acompanhadas simultaneamente.

* EXAMINE

Mostra o conteúdo de uma posição de memória. O conteúdo pode ser apresentado de várias formas. A variável, que estiver sendo examinada, será especificada com a mesma sintaxe da linguagem na qual o programa-fonte foi codificado.

* DEPOSIT

Muda o conteúdo de uma posição de memória. A posição de memória pode ser referenciada por um símbolo (nome da variável).

* EVALUATE

Avalia uma expressão. A expressão aritmética é digitada pelo usuário e o depurador mostra o seu resultado.

* CALL

Permite a chamada de sub-rotinas.

* SHOW

Mostra, de forma geral ou seletiva, todos os atributos ativos. Pode-se verificar quais são os pontos de parada, que variáveis estão sendo acompanhadas, e outros atributos.

* CANCEL

Cancela, de forma geral ou seletiva, atributos. Pontos de parada, por exemplo, podem ser cancelados a qualquer momento.

* @<arquivo>

Indica que comandos para o depurador devem ser lidos de <arquivo>. Muito útil quando trata-se de uma depuração complexa.

* IF / THEN / ELSE

Permite a execução condicional de comandos do depurador.

* WHILE

Permite a execução repetitiva de um comando do depurador, até que uma condição seja satisfeita.

6.5 Utilitário de Biblioteca

Existe no VMS um utilitário voltado para a criação e manutenção de bibliotecas. Estas bibliotecas são arquivos indexados que contêm código-objeto ou texto.

Existem 4 tipos de bibliotecas:

* Objeto

Contém rotinas que já foram compiladas, e que deverão ser agregadas a um módulo-objeto para formar uma imagem. É uma das entradas possíveis para o editor de ligações.

* Macro

Contém definições de rotinas e símbolos em MACRO "assembler", e é basicamente usada em programação "assembler".

* Texto

Contém textos de qualquer espécie. Pode ser usada para agrupar e controlar a manutenção de documentação. Também pode ser usada para armazenar trechos de um programa-fonte, como por exemplo definições de arquivo em COBOL (existe uma extensão ao comando COPY para acessar este texto: COPY <arquivo> IN <biblioteca>).

* Help

  Contém textos de ajuda. Existe um comando do VMS que é usado para consulta a bibliotecas deste tipo: o comando HELP. Existe uma biblioteca deste tipo que é do próprio VMS, onde estão descritos os comandos e outros assuntos. Este tipo de biblioteca pode ser usado para prover ajuda interativa a usuários de sistemas on-line. Em geral, os utilitários do VMS, que possuem algum tipo de ajuda, usam este tipo de biblioteca.

De um modo geral, uma biblioteca pode ser acessada pelo comando LIBRARY ou através de rotinas.

As rotinas de manipulação de bibliotecas podem ser chamadas de qualquer linguagem, e são as seguintes:

* Inicializar biblioteca

* Abrir biblioteca

* Procurar uma chave

* Colocar uma nova chave

* Recuperar todas as chaves

* Apagar uma chave e o texto associado

* Ler registro de texto

* Gravar registros de texto

O comando LIBRARY permite

* criação de biblioteca

* inserção de novos itens

* extração (cópia) de itens

* retirada (deleção) de itens

* consulta aos itens existentes

Uma das características interessantes em termos de auditoria é que a uma biblioteca podem ser associados vários registros de atualização (o número máximo é definido pelo usuário). Estes registros comportam-se de uma forma circular, ou seja, se o número de registros está no máximo, o próximo registro será incluído e o registro mais antigo será retirado.

A cada atualização da biblioteca fica registrado o nome do usuário, data, hora, a operação executada (inserção, deleção ou substituição) e a identificação dos itens envolvidos. Isto permite com que seja feita a verificação das operações executadas em uma biblioteca de rotinas, por exemplo.

6.6 Procedimentos

Os comandos para o VMS podem ser agrupados em arquivos, chamados procedimentos.

A execução dos comandos contidos em um procedimento é acionada da seguinte forma:

```
$ @<procedimento>
```

Se no nome do procedimento não for indicado o seu tipo, o VMS assume que ele é do tipo COM ("Command Procedure").

A execução de um procedimento pode ser comandada interativamente pelo usuário (nível DCL)

ou a partir de um procedimento em execução (como uma sub-rotina).

O VMS permite que sejam passados parâmetros para um procedimento. O número máximo de parâmetros permitidos atualmente é oito. Eles são identificados no procedimento como P1, P2, P3, P4, P5, P6, P7 e P8.

Existem vários recursos que tornam a DCL uma verdadeira linguagem de programação. Entre os recursos existentes, podemos citar:

* Símbolos
* Expressões
* Comandos de controle
* Funções
* Comandos para manipular arquivos

6.6.1 Símbolos

Um símbolo é uma variável, no sentido usado em linguagens de programação. A um símbolo pode ser atribuído um valor, que passa a ser o conteúdo do símbolo. Um símbolo pode conter valores numéricos (apenas inteiros) ou alfanuméricos.

Quanto a disponibilidade de uso, ele pode ser de dois tipos: local ou global. Um símbolo local é aquele que só existe (só pode ser usado) no procedimento no qual ele foi definido. Um símbolo global existe em qualquer procedimento, independente de onde ele foi definido.

Quando um símbolo é criado, fica determinado qual o seu tipo (numérico ou alfanumérico), e qual a sua disponibilidade (local ou global).

Um símbolo é criado por uma atribuição de valor. O sinal usado na atribuição é que determina as características do símbolo. Os sinais são os seguintes:

	Numérico	Alfanumérico
Local	=	:=
Global	==	:==

Um símbolo também pode ser criado por um comando, que mostra um texto e pergunta (no terminal) o valor a ser associado a um símbolo. O formato deste comando é

$ INQUIRE <símbolo> <texto>

Outra característica importante de um símbolo é que, quando ele estiver no início de uma linha de comandos, ele será substituído pelo seu conteúdo. Desta forma podem ser criados novos comandos.

Esta substituição, em uma linha de comando, do símbolo pelo seu conteúdo, também pode ser comandada, bastando para isso colocar o símbolo entre aspas simples.

6.6.2 Expressões

São três os tipos de expressões: aritmética, relacional (comparação) e lógica.

Na expressão aritmética os valores de dois símbolos (ou constantes) podem ser somados (+), subtraídos (-), multiplicados (*) ou divididos (/). Os símbolos (ou constantes) têm que ser numéricos.

Na expressão relacional o objetivo é comparar dois valores. Existe um conjunto de operadores para valores numéricos e outro para valores alfanuméricos:

Operação	Operador Numérico	Operador Alfanumérico
igualdade	.EQ.	.EQS.
diferença	.NE.	.NES.
maior	.GT.	.GTS.
maior ou igual	.GE.	.GES.
menor	.LT.	.LTS.
menor ou igual	.LE.	.LES.

Na expressão lógica, duas ou mais expressões relacionais, são tratadas, através dos operadores lógicos ".NOT.", ".AND." e ".OR.".

6.6.3 Comandos de controle

Existem comandos específicos para o controle da execução dos comandos de um procedimento. Os principais são os seguintes:

* IF <expressão> THEN <comando>

  Permite a execução condicional de um comando VMS. Na expressão podem ser usadas expressões relacionais e/ou lógicas.

* GOTO / <rótulo>

  Usado para desviar o fluxo de execução do procedimento. Em geral está associado a um comando IF. O rótulo pode ser formado por letras e dígitos e é identificado pelo caracter "dois pontos", que o sucede. Este comando é muito usado em

conjunto com o comando IF (desvio condicional).

* ON <exceção> <comando>

Usado para tratamento de exceções em um procedimento. Existem classes de exceções pré-definidas, como por exemplo erro na execução de um programa. Quando uma exceção ocorre, o processamento é interrompido e o comando indicado é executado. Em geral este comando é o comando de desvio (GOTO).

* EXIT

Termina o procedimento. Se o procedimento que possui este comando foi chamado por um outro procedimento ("chamador"), volta o controle ao procedimento "chamador". Senão, volta para o nível DCL.

* STOP

Termina a execução do procedimento, passando o controle para o VMS. Mesmo que o comando seja executado em um procedimento chamado por outro (sub-rotina), o controle passa para o VMS (nível DCL).

6.6.4 Funções

Existe uma grande variedade de funções disponíveis, a nível de DCL. Estas funções são conhecidas como "funções léxicas" e permitem:

* obter informações de um processo, como nome do usuário, modo de execução (interativo ou "batch"), privilégios e UIC;
* obter informações do sistema operacional;
* obter informações de arquivos como nome, data, proteção, organização e tamanho;
* obter informações de periféricos;
* traduzir nomes lógicos;
* executar operações sobre símbolos alfanuméricos, como procura de uma cadeia de caracteres, tamanho do símbolo, extração de parte do valor e formatação;
* transformar e/ou determinar o tipo de um símbolo.

Todas as funções léxicas tem "F$" como prefixo do seu nome e fazem uso de parâmetros para indicar o que deve ser executado. O resultado da função sempre estará no local do comando onde ela foi acionada.

6.6.5 Comandos para manipular arquivos

Arquivos do VMS, de qualquer dos tipos de organização (seqüencial, relativa ou indexada), podem ser acessados através de comandos. Os principais comandos são os seguintes:

* OPEN

  Torna disponível um arquivo. O modo de uso (leitura ou gravação, ou ambos) é indicado através dos qualificadores /READ e /WRITE. Este comando possui dois parâmetros, que são o nome pelo qual o arquivo será referenciado no procedimento e a identificação do arquivo para o sistema VMS.

* CLOSE

  Libera um arquivo, tendo como parâmetro o nome interno.

* READ

  Lê um registro de um arquivo e atribui o seu conteúdo a um símbolo. Neste comando pode ser indicada a chave de acesso (se o arquivo for indexado), e para qual rótulo deve ser desviado o processamento quando o arquivo terminar.

* WRITE

  Grava dados, a partir de um símbolo ou do resultado de expressão, em um arquivo. Neste comando pode ser indicado que um registro lido pelo comando READ será atualizado (regravação). Na regravação é necessário que o novo conteúdo seja do mesmo tamanho do conteúdo anterior.

Exemplo de procedimento:

```
$ LOOP:
$  INQUIRE Cad "Cadastro"
$  IF Cad .EQS. "" THEN GOTO LOOP
$  Aux := 'F$SEARCH(Cad)'
$  IF Aux .NES. "" THEN GOTO OK_Cad
$  WRITE SYS$OUTPUT "Cadastro nao existe"
$  GOTO LOOP
$
$ OK_Cad:
$  Usuario := 'F$PROCESS()'
$  IF Usuario .EQS. "VILLAS" THEN GOTO OK_Usuario
$  WRITE SYS$OUTPUT "Usuario nao autorizado"
$  EXIT
$
$ OK_Usuario:
$  OPEN/READ/WRITE Xpto 'Cad'
$  Cnt = 0
$
$ LEITURA:
$  READ/END_OF_FILE=FIM Xpto Reg
$  Cnt = Cnt + 1
$  Reg = Reg - 1
$  WRITE/UPDATE Xpto Reg
$  GOTO LEITURA
$
$ FIM:
$  CLOSE Xpto
$  WRITE SYS$OUTPUT "Registros atualizados : ",Cnt
$  IF Cnt .GT. 5 THEN-
      WRITE SYS$OUTPUT "Sao mais de 5"
$  EXIT
```

6.7 Utilitário de Diferenças

Este utilitário compara dois arquivos e mostra as diferenças encontradas entre eles.

Este utilitário está voltado para a comparação entre textos. As diferenças são apresentadas em grupos, e o que é mostrado é o próprio segmento de texto que não está presente nos dois arquivos.

Ao final da relação de diferenças é apresentada uma estatística mostrando quantos registros são diferentes e em quantos grupos.

Este utilitário é muito usado para gerar um arquivo com o registro de alterações efetuadas em arquivos que contenham texto, como por exemplo, programas-fonte e documentação.

Exemplos:

```
$
$ TYPE A.TXT
Este e um exemplo
do utilitario DIFFERENCES.
Veja como ele mostra
as diferencas entre estes
dois textos (A.TXT e B.TXT).
Esta linha so esta no arquivo A.
Esta tambem so esta no arquivo A.
$
$ TYPE B.TXT
Este e um exemplo do utilitario DIFFERENCES.
Veja como ele mostra
as diferencas entre estes
dois textos (A.TXT e B.TXT).
Esta linha so esta no arquivo B.
Esta tambem so esta no arquivo B.
```

```
$ type c.dif
$ DIFFERENCES A.TXT B.TXT
************
File DRA1:[VILLAS]A.TXT;1
  1    Este e um exemplo
  2    do utilitario DIFFERENCES.
  3    Veja como ele mostra
******
File DRA1:[VILLAS]B.TXT;1
  1    Este e um exemplo do utilitario DIFFERENCES.
  2    Veja como ele mostra
************
************
File DRA1:[VILLAS]A.TXT;1
  6    Esta linha so esta no arquivo A.
  7    Esta tambem so esta no arquivo A.
******
File DRA1:[VILLAS]B.TXT;1
  5    Esta linha so esta no arquivo B.
  6    Esta tambem so esta no arquivo B.
************

Number of difference sections found: 2
Number of difference records found: 4

DIFFERENCES /IGNORE=()/MERGED=1
     DRA1:[VILLAS]A.TXT;1-
     DRA1:[VILLAS]B.TXT;1
$
```

6.8 Formatador de Textos

O formatador de textos do VMS tem por objetivo produzir um texto formatado, a partir de um texto escrito pelo usuário.

Este formatador é acionado pelo comando

```
$ RUNOFF <arquivo>
```

Se não for especificado o tipo do arquivo, o VMS assume que o seu tipo é RNO. Será produzido um arquivo com o mesmo nome do arquivo de entrada, porém com o tipo MEM.

O texto a ser formatado deve possuir comandos para o formatador. Estes comandos não serão copiados para o texto formatado. Eles indicam parâmetros e ações a serem tomadas durante a formatação.

Os comandos do formatador permitem

* definir o tamanho de uma página
* definir título e subtítulo
* definir o uso ou não de paginação (no título) e data de formatação (no subtítulo)
* centralizar uma linha de texto
* definir se o texto será formatado em espaço simples ou espaço duplo
* comandar o salto de linha
* comandar o salto de página
* realçar trechos do texto, através de ênfase (negrito) ou sublinhado

* incluir um outro arquivo texto, no momento da formatação

* definir listas, com definição de numeração ou caracter no início de cada elemento e espacejamento entre os elementos

* gerar índices automaticamente, seja pela estrutura do texto, seja um índice remissivo (as palavras sendo indicadas pelo usuário)

Exemplo:

```
$
$ TYPE TEXTO.RNO
.PS 60,50
.P
Este e um exemplo do formatador de textos. Neste
exemplo será mostrado o arquivo original e o
arquivo formatado.
.B
.C; Este e um texto centralizado
.B
.LS "*"
.LE
Este e o primeiro elemento de uma lista;
.LE
Este e o segundo.
.ELS
$
$ RUNOFF TEXTO
$ TYPE TEXTO.MEM
```

Este e um exemplo do formatador de textos. Neste exemplo sera mostrado o arquivo original e o arquivo formatado.

Este e um texto centralizado

* Este e o primeiro elemento de uma lista;

* Este e o segundo.

6.9 Utilitário de Mensagens

Este utilitário permite a definição, manutenção e acesso a mensagens.

Toda aplicação possui mensagens que devem ser mostradas a um usuário. Estas mensagens, no VMS, podem estar separadas do programa, em um arquivo externo.

A grande vantagem das mensagens estarem separadas do programa é que elas podem ser modificadas sem que haja necessidade de modificação no programa-fonte.

As mensagens são definidas através de uma linguagem específica. Toda mensagem é composta de quatro partes:

* nome da aplicação
* nível da mensagem (sucesso, aviso, informativo, erro, erro severo)
* mnemônico da mensagem
* texto da mensagem

Após a definição das mensagens, elas são compiladas (comando MESSAGE) e ligadas (comando LINK) para formar uma tabela de símbolos. O acesso às mensagens é feito através de rotinas. As rotinas recuperam as partes de uma mensagem através do seu mnemônico.

Em geral os utilitários do VMS usam arquivos de mensagem.

7. ARQUITETURA DE INFORMAÇÃO

A arquitetura de informação para o VAX (VIA - VAX Information Architecture) é um conjunto de produtos, totalmente integrados entre si, que visa suprir todas as necessidades de um usuário em termos de informações. Existem produtos para preparar, manter e apresentar informações.

As áreas às quais estes produtos pretendem atender são as seguintes:

* DD - Dicionário de Dados

  Descrição centralizada de dados, relatórios, telas e procedimentos usados por uma aplicação.

* Gerência de Dados (GD)

  Organização, controle e relação entre os dados de uma aplicação.

* Acesso a Dados (AD)

  Manipulação dos dados armazenados.

* Gerência de Terminal (GT)

  Controle de entrada e saída interativa de dados.

* Gerência de Aplicações (GA)

  Controle de acesso e de uso de aplicações. Otimização do processamento em aplicações multi-usuário.

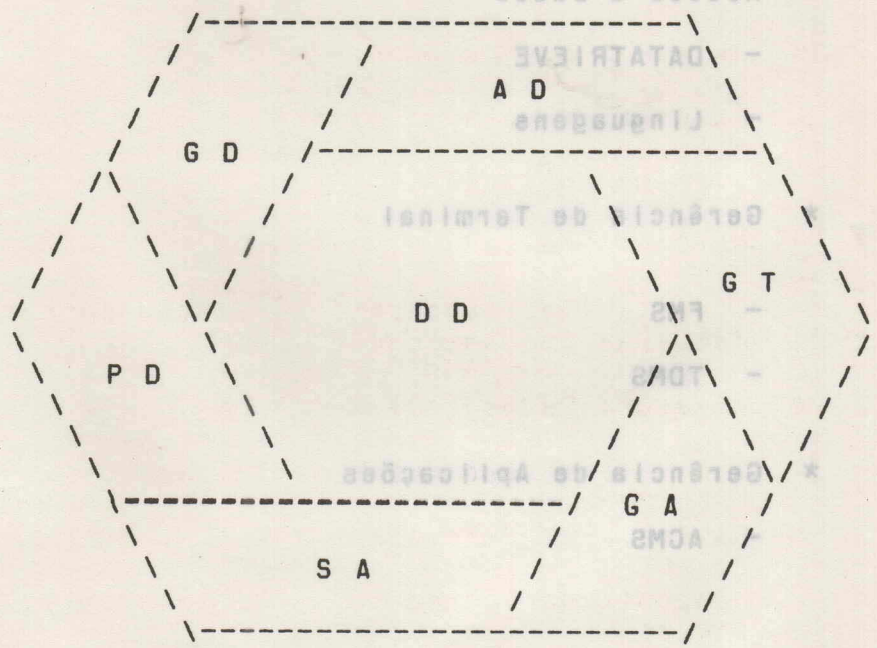
* Serviços de Apresentação (SA)

  Formatação, diagramação e apresentação de dados.

* Processamento Distribuído (PD)

  Controle da distribuição de processamento entre vários computadores. Uso de dados distribuídos.

O símbolo a seguir é usado para identificar os produtos relacionados à estrutura de informação:



Os produtos da DIGITAL atualmente disponíveis para cada área são os abaixo relacionados.

* Dicionário de Dados

  - CDD

* Gerência de Dados

  - RMS

  - DBMS

  - RDB

* Acesso a Dados

  - DATATRIEVE

  - Linguagens

* Gerência de Terminal

  - FMS

  - TDMS

* Gerência de Aplicações

  - ACMS

* Serviços de Apresentação
  - DECGRAPH
  - DECSLIDE
  - DATATRIEVE
  - VTX
* Processamento Distribuído
  - DATATRIEVE
  - DECNET

A seguir, uma descrição sucinta dos produtos da arquitetura de informação do VAX. Os produtos para comunicação entre computadores serão descritos no próximo capítulo.

7.1 CDD (Common Data Dictionary)

O CDD é um dicionário de dados físico, ou seja, ele guarda apenas descrição de dados (não possui o mesmo conceito utilizado na metodologia de Análise Estruturada).

Os principais produtos da arquitetura de informação (DATATRIEVE, DBMS, RDB, TDMS e ACMS) guardam as suas definições (dados, telas, relatórios, etc) no CDD. Isto permite que, para uma aplicação, seja possível executar facilmente alterações.

Por exemplo, um arquivo de dados só precisa ser definido uma vez. Como a descrição é única, não há necessidade de programas de conversão de dados, para que o mesmo arquivo possa ser usado por outros produtos da arquitetura de informação.

Várias características estão relacionadas ao uso das descrições armazenadas no CDD. Por exemplo, a cada descrição pode ser associado um histórico de utilização (quem, quando e para que foi usada uma descrição). Isto permite que o impacto na modificação de uma descrição possa ser avaliado.

O CDD está organizado logicamente sob a forma de dicionários. Cada raiz pode possuir definições e/ou outras raízes (subdicionários). Esta organização é análoga à organização de arquivos e diretórios no VMS.

O controle de acesso às descrições pode ser feito da seguinte forma: proteção do arquivo físico que contém as descrições e listas de controle de acesso para cada descrição ou subdicionário.

As descrições que são armazenadas no CDD podem ser preparadas via DATATRIEVE ou DBMS. Elas são armazenadas de tal forma que é possível acessá-las por todos os produtos da arquitetura de informação, inclusive pelas linguagens de programação. No caso das linguagens, o compilador, ao buscar uma definição no CDD, transforma a descrição para a sintaxe da linguagem.

Existem utilitários que auxiliam na definição e na manutenção do CDD. São eles:

* Gerência do Dicionário
  (DMU - Data Management Utility)

  Permite criar, copiar, salvar e proteger (controlar o acesso) as descrições.

* Linguagem de Definição de Dados (CDDL - CDD Data Definition Language)

  Permite a descrição de dados, em uma linguagem genérica.

* Verificação e Correção do Dicionário (CDDV - CDD Verify/Fix Utility)

  Permite corrigir erros (devido a falhas de processamento) na estrutura do CDD, reduzir o espaço ocupado e aumentar a performance de acesso às descrições.

7.2 DATATRIEVE

O ambiente para desenvolvimento de programas em linguagem de quarta geração no VMS chama-se DATATRIEVE. Ele está orientado para consulta a dados e desenvolvimento de aplicações.

As definições dos dados e das aplicações são feitas interativamente. Todas estas descrições são armazenadas no CDD.

Os dados são descritos em uma linguagem semelhante a COBOL, porém com mais recursos. Estas descrições de dados podem ser usadas para criar "visões" de dados, ou seja, definição de apenas alguns campos que sejam de uso de um determinado usuário. Todos os arquivos de dados criados pelo sistema de gerência de arquivos RMS ou pelos bancos de dados DBMS e RDB podem ser acessados. Também podem ser acessados arquivos em outro computador (em uma rede).

Os procedimentos são descritos na própria linguagem do DATATRIEVE. Entre os principais comandos podemos citar a seleção de dados, cruzamento de dados de dois ou mais arquivos, geração de relatórios, geração de gráficos e uso de telas FMS ou TDMS para entrada de dados.

Os principais conceitos de relação entre os dados em DATATRIEVE são os seguintes:

* Arquivo

  Arquivo físico, gerado pelo RMS ou banco de dados, com os dados a serem manipulados. Pode ser de qualquer tipo de organização: seqüencial, relativa ou indexada.

* Registro

  Descrição dos dados de um arquivo. Esta descrição pode ser total ou parcial (visão).

* Domínio

  Associação de um arquivo a uma definição de registro. Todas as operações com os dados são realizadas através de domínios. Um arquivo pode ter várias definições de registro associadas, ou seja, pode participar de vários domínios.

7.3 DECGRAPH

Este sistema permite a confecção de gráficos, a partir de dados numéricos.

Os dados podem ser informados via teclado, arquivos de dados, arquivos especiais e, ainda, arquivos gerados pela matriz eletrônica DECALC.

Todas as características do gráfico são definidas interativamente, via menu de símbolos. As principais características de um gráfico são o formato, a cor e os padrões de preenchimento de áreas.

Os gráficos são apresentados em vídeo, porém eles podem ser gerados opcionalmente em arquivo, para posterior impressão.

7.4 DECSLIDE

O DECSLIDE é um software para geração de slides (apresentações). Esses slides podem combinar texto e diagramas (gráficos), e podem ser gerados em preto & branco ou coloridos.

O slide é preparado interativamente, sendo que os comandos são informados através de teclas especiais. Estes comandos permitem:

* Desenhar linhas

* Colocar um texto, e definir atributos como tamanho, rotação e estilo (itálico, por exemplo)

* Selecionar, mover, preencher e reduzir/aumentar formas geométricas básicas.

A saída pode ser gerada em arquivo, para posterior impressão.

7.5 VTX

O vídeo-texto tem por principal objetivo a recuperação e a divulgação de informações. Ele foi projetado para ser usado por pessoas sem experiência em informática e cujo contato com o sistema é pouco freqüente.

As informações no vídeo-texto são organizadas em páginas. Em cada página existem informações sob a forma de texto e/ou gráficos, com uso de cor. O acesso às páginas é feito através de menus.

Como exemplos para o uso de vídeo-texto podemos citar treinamento, notícias (jornais), listas telefônicas, oferta de serviços, relação de produtos e atividades e transações bancárias (consulta e movimentação).

O sistema VTX de vídeo-texto está totalmente baseado nas funções do VMS. Ele não é um sistema dedicado, podendo ser integrado às funções de automação de escritórios e processamento de dados.

Ele está dividido em três módulos:

* Controle de Terminal
* Recuperação em Banco de Dados
* Atualizador

Outra característica importante é que todo o seu processamento pode ser distribuído. Cada módulo pode estar sendo executado em um nó de um cluster, por exemplo.

Este sistema pode ser integrado ao sistema de automação de escritórios "ALL-IN-1", da DIGITAL.

7.6 FMS (Forms Management System)

Este utilitário tem por objetivo a geração e manutenção de telas.

As telas são preparadas interativamente em um editor específico. Também pode ser utilizada uma linguagem de definição.

Podem ser atribuídas diversas características para a tela como um todo, e para cada campo individualmente.

Em relação à tela pode ser definido vídeo reverso, cores, conjunto de caracteres, área que a tela ocupa (linha inicial e final), outra tela com texto explicativo (ajuda), etc.

Para cada campo existem características como numérico, letras maiúsculas, preenchimento obrigatório, valor inicial (se nada for digitado), fim de campo automático e texto de auxílio, entre outros. A crítica a nível de campo é bem simples. Para críticas mais complexas podem ser associadas rotinas a cada campo. Essas rotinas são preparadas pelo próprio usuário, em qualquer linguagem.

A comunicação entre uma tela e um programa é feita através de rotinas. Estas rotinas permitem mostrar uma tela, ler uma tela, ler um campo.

7.7 TDMS (Terminal Data Management System)

O TDMS é semelhante ao FMS. Ele possui um editor para confecção das telas, é usado através de rotinas, e permite a definição de atributos para telas e campos.

A principal diferença está no conceito de requisição. Uma requisição associa uma tela a um ou mais registros de dados. Cada campo da tela

pode estar associado a vários campos, em registros diferentes. Nem todos os campos dos registros precisam estar associados a campos da tela.

Em uma requisição podem ser incluídos comandos para verificar se o valor de um campo está em uma faixa de valores ou se o valor está em uma lista de valores válidos.

As descrições de requisição ficam armazenadas no CDD. Isto permite que toda a entrada e saída de dados de um programa possa ser alterada sem que seja necessário alterar o programa-fonte.

Tanto o FMS quanto o TDMS tratam as diferenças existentes entre terminais. Por exemplo, alguns terminais mais antigos (VT52) não permitem o uso de campo reverso. Esta característica não elimina a execução de um programa que use telas com campos em vídeo reverso: ela é simplesmente ignorada. Toda a movimentação do cursor durante a entrada de dados também é controlada pelo utilitário.

7.8 DBMS (Database Management System)

Este banco de dados implementa a arquitetura em rede (modelo CODASYL).

Este modelo de banco de dados deve ser usado quando existe um grande volume de dados, as relações entre os dados são complexas e estáveis (pouco sujeitas a modificações) e existe um grande volume de operações.

Todas as definições de dados são armazenadas no CDD. A estas definições, bem como aos dados, podem ser associados privilégios de uso e acesso.

Os dados podem ser manipulados simultaneamente. Existe também um registro de todas as operações efetuadas (antes/depois), que permite a recuperação dos dados em caso de falha.

O acesso aos dados pode ser feito via DATATRIEVE, COBOL (os comandos estão embutidos na linguagem) e outras linguagens, através do uso de pré-compiladores.

Existem, ainda, dois utilitários: o primeiro para consulta interativa e o segundo para manutenção do banco de dados.

7.9 RDB (Relational Database)

RDB é um banco de dados relacional. O modelo relacional representa os dados na forma de tabelas. Existe uma correspondência entre cada linha da tabela e um registro, e cada coluna da tabela com os valores de um campo.

Este modelo de banco de dados deve ser usado quando a aplicação não está bem definida (muito sujeita a modificações, uso de prototipação) e os seus usuários possuírem pouco conhecimento em banco de dados.

Possui as mesmas características de utilização do DBMS, porém não existem linguagens de programação com comandos embutidos para manipulação do banco de dados. É necessário o uso de pré-compiladores, que existem para linguagens como COBOL, PASCAL, BASIC e FORTRAN. As linguagens que não possuem pré-compiladores devem usar uma interface interpretativa.

7.10 ACMS (Application Control and Management System)

Este sistema visa o controle do uso de aplicações "on-line".

Uma aplicação é composta por uma seqüência de transações. Cada transação possui uma parte de entrada e saída de dados (usando requisições TDMS) e outra parte de processamento (comandos DCL, programas ou rotinas).

O processamento pode ser distribuído, ou seja, ser executado em um outro computador que esteja ligado ao computador local. A parte de entrada e saída de dados não pode ser distribuída.

A cada usuário de uma aplicação pode ser associado um menu com as transações permitidas. O sistema controla quais usuários tem acesso a que aplicação e a que transações de uma aplicação.

Para cada transação executada podem ser armazenados dados com o objetivo da transação e os recursos do VMS consumidos. Pode manter, ainda, um registro do volume de transações por usuário.

7.11 RMS (Record Management System)

Este é o sistema de gerência de dados do VMS. Ele é composto por um conjunto de rotinas que manipulam arquivos e registros. As operações de entrada e saída podem ser feitas tanto a nível de registro quanto a nível de bloco (512 bytes).

Todos os utilitários e linguagens, que executam sob o VMS, usam o RMS para implementar as suas operações de entrada e saída.

8. COMUNICAÇÕES

A comunicação entre computadores é, em geral, feita através de uma rede. Uma rede de computadores é uma configuração de dois ou mais computadores, chamados nós, que estão interligados com pelo menos um dos seguintes objetivos:

* Comunicação remota

  Comunicação entre os nós, principalmente para troca de dados.

* Compartilhar recursos

  Recursos computacionais de alto custo (impressoras de alta qualidade, discos, etc) quando compartilhados têm o seu custo por usuário diminuído.

* Processamento distribuído

  Partes de um sistema modular podem estar sendo processadas ao mesmo tempo em computadores diferentes.

A DIGITAL possui uma arquitetura de redes, que se chama DNA (DIGITAL Network Architecture). Esta arquitetura é um conjunto de protocolos dividido em níveis. Estes níveis são, do maior para o menor, os seguintes:

* Usuário
* Gerência de rede
* Aplicação
* Sessão
* Transporte
* Dados
* Físico

Estes níveis são semelhantes aos níveis definidos pela ISO, para o que é chamado de "Open Systems Interconnect" (OSI).

Existem quatro sistemas desenvolvidos pela DIGITAL para comunicação com os seus computadores:

* DECNET
* INTERNET
* PACKETNET
* ETHERNET

Um outro modo de comunicação bem mais primitivo é o de "emulação de terminal", em geral aplicado a comunicação entre microcomputadores e computadores maiores.

8.1 DECNET

Este é o sistema usado para comunicação entre equipamentos da DIGITAL.

Permite transferência de arquivos entre os nós e comunicação entre aplicações que estão em nós diferentes.

Os arquivos podem ser traduzidos de/para ASCII durante a transmissão.

As aplicações podem acessar a rede dos seguintes modos:

* Transparente

  A aplicação pode usar todos os comandos de acesso a arquivos, sendo que a especificação do arquivo é incorporada a indicação do nó;

* Não transparente

  Acesso por programas/rotinas codificados na linguagem MACRO ("assembler"). Estas aplicações têm acesso a informações sobre o estado da rede e características da transmissão.

Uma aplicação também pode ser carregada para execução em qualquer nó da rede. Isto permite que um sistema seja desenvolvido em um computador com mais recursos de "hardware" e "software", e seja executado em outro computador com menos recursos, ou vice-versa.

A gerência da rede (Network Control Program) se encarrega de monitorar o comportamento das comunicações. Por exemplo, o caminho (rota) entre dois nós pode ser determinado pela melhor relação custo/benefício ou quanto ao gasto de recursos.

Os terminais podem estar conectados à rede como um todo. O controle de acesso a cada nó é feito segundo os arquivos de autorização do nó (relação dos usuários que podem ter acesso ao nó). Isto permite a associação a cada usuário o direito de usar um ou mais nós, de forma seletiva. Um usuário que está em um nó pode abrir uma sessão em um outro nó através do comando

"$ SET NODE <novo nó>"

8.2 INTERNET

É uma família de produtos que interliga computadores e redes da DIGITAL a sistemas de outros fabricantes.

Entre os produtos existentes podemos citar:

* Gateway SNA

  Permite a comunicação com computadores IBM que estejam em uma rede SNA (System Network Architecture). Esta comunicação permite emulação de terminal 3270 e RJE (submissão remota de tarefas).

* Emulador do protocolo IBM 2780 ou 3780 (BISYNC)

  Permite a transferência de arquivos de/para computadores que recebem/transmitem dados segundo estes protocolos, podendo ser usado em conjunto com o DECNET.

* Emulador do protocolo 3271 IBM

  Permite a ligação de programa a programa entre um computador VAX e outro IBM (System/370 sob CICS ou IMS).

No VAX 11/750 o emulador pode se comunicar com até dois IBM simultaneamente. Pode fazer automaticamente a tradução ASCII/ EBCDIC.

* Emulador multiterminal MUX200

Permite comunicação com CDC-6000 Cyber ou computador capaz de usar o protocolo 200UT modo 4A.

8.3 PACKETNET

Este sistema permite usar comutação de pacotes no padrão X.25. Neste sistema, os dados a serem transmitidos são agrupados em "pacotes". Os pacotes são transmitidos em uma rede comutada (rede de transmissão de dados onde existem vários caminhos entre dois nós). Eles possuem informações de controle que permitem que os dados cheguem ao seu destino.

Este padrão é o mesmo usado no serviço de comunicação por comutação de pacotes TRANSDATA, da EMBRATEL.

Existe ainda um conjunto de rotinas que permite acesso à rede, chamadas de PSI (Packetnet System Interface).

Os terminais remotos podem estar conectados, comunicando-se com o sistema através dos protocolos estabelecidos nas recomendações X.3, X.28 e X.29, da CCITT.

8.4 ETHERNET

Este é um sistema para rede local. Uma rede local é caracterizada pela pequena distribuição geográfica dos nós a ela ligados, e pela alta taxa de transmissão de dados. Este sistema foi desenvolvido pela DIGITAL em conjunto com a INTEL e a XEROX.

O meio físico para transmissão de dados é um cabo coaxial. Os computadores são ligados ao cabo através de transreceptores (H4000). O transreceptor é quem controla as transmissões (entrada e saída) do computador. A taxa de transmissão desta rede pode chegar a até 10 megabits por segundo.

Existe "software" para comunicação entre UNIBUS ("barra de dados" do VAX) e ETHERNET.

8.5 Emulação de terminal

Várias empresas, principalmente de "software", desenvolvem emuladores de terminal. Estes emuladores simulam um terminal em um microcomputador, ou seja, um microcomputador ligado a um VAX pode ser utilizado como se fosse um terminal.

Um dado especialmente interessante (dada a situação nacional atual de informática na área de microcomputadores) é que existem emuladores para terminal VT100, VT125 e até VT240 que são executados em microcomputadores da linha IBM PC. Estes emuladores permitem ainda transferência de arquivos, nos dois sentidos (micro -> VAX e VAX -> micro).

9. CLUSTER

O Cluster é um sistema cujos objetivos são interligar processadores VAX por um meio físico que tenha alta velocidade de transmissão, e manter aspectos de gerência do sistema operacional uniformes em todos os processadores conectados.

Os objetivos da interligação de processadores são idênticos aos citados no capítulo COMUNICAÇÕES.

Este sistema combina, com baixo acoplamento (dependência), dois ou mais processadores VAX e controladores de armazenamento (disco).

Cada processador ou controlador de armazenamento é chamado de nó: o processador é um nó ativo e o controlador é um nó passivo. Um Cluster pode ter no máximo 16 nós. Todos podem ser ativos, mas apenas 15 deles podem ser passivos.

Todos os nós de um Cluster são ligados a um conector (em forma de estrela) através de uma interconexão de computador. Para cada interconexão existe um controlador inteligente que trata falhas.

Em um Cluster, a mesma versão do sistema operacional VMS tem que estar sendo executada em todos os nós ativos.

Existem dois tipos de sistemas de multiprocessamento:

* Tipo 1

  Fortemente acoplado (dois processadores, com uma cópia do sistema operacional, compartilhando memória).

* Tipo 2

  Muito pouco acoplado (vários processadores, cada um com a sua cópia do sistema operacional, interligados em rede).

	Tipo 1	Cluster	Tipo 2
Os sistemas são carregados e falham em	conjunto	separado	separado
O sistema de arquivos é	integrado	integrado	separado
O domínio de gerência do sistema operacional é	igual	igual	separado
O potencial de crescimento é	limitado	grande	muito grande
A configuração ocupa	um gabinete	uma instalação	uma grande área

Os processadores VAX que podem ser combinados são os seguintes:

* VAX-11/750 (MX 850)
* VAX-11/780
* VAX-11/782
* VAX-11/785

9.1 Hardware

O hardware de um Cluster (excluindo os processadores) possui as seguintes partes:

* Conector em estrela (Star Coupler)

  Este conector cria uma combinação em estrela (ou radial) de até 45 metros. Ele pode suportar até 16 nós (ativos ou passivos).

  A ele são conectados pares de cabos coaxiais, sendo dois pares para cada nó. Para cada par de cabos coaxiais existe um tratamento, de tal forma que os nós podem ser conectados ou desconectados sem afetar o funcionamento do Cluster.

  Todo sinal recebido por um cabo coaxial transmissor é enviado a todos os cabos coaxiais receptores que estejam conectados. O sistema operacional em cada nó ativo reconhece se a transmissão é para aquele nó ou não.

* Interconexão de computador
(Computer Interconnect)

São os cabos coaxiais que são ligados ao conector em estrela. Cada par de cabos forma um "caminho". Existem sempre dois "caminhos".

Em cada par de cabos coaxiais um deles é transmissor e o outro receptor.

A taxa de transmissão de dados pode chegar a 70 megabits por segundo.

* Controlador de conexão
(Interface)

Existe um controlador para cada interconexão. A sua principal tarefa é transferir todo o tráfego para somente um caminho, quando um dos caminhos apresentar defeito.

O sistema VMS testa, periodicamente, o caminho defeituoso, de tal forma que, quando este estiver novamente disponível, o controlador passa a usá-lo normalmente.

* Controlador de armazenamento
(HSC - Hierarchical Storage Controller)

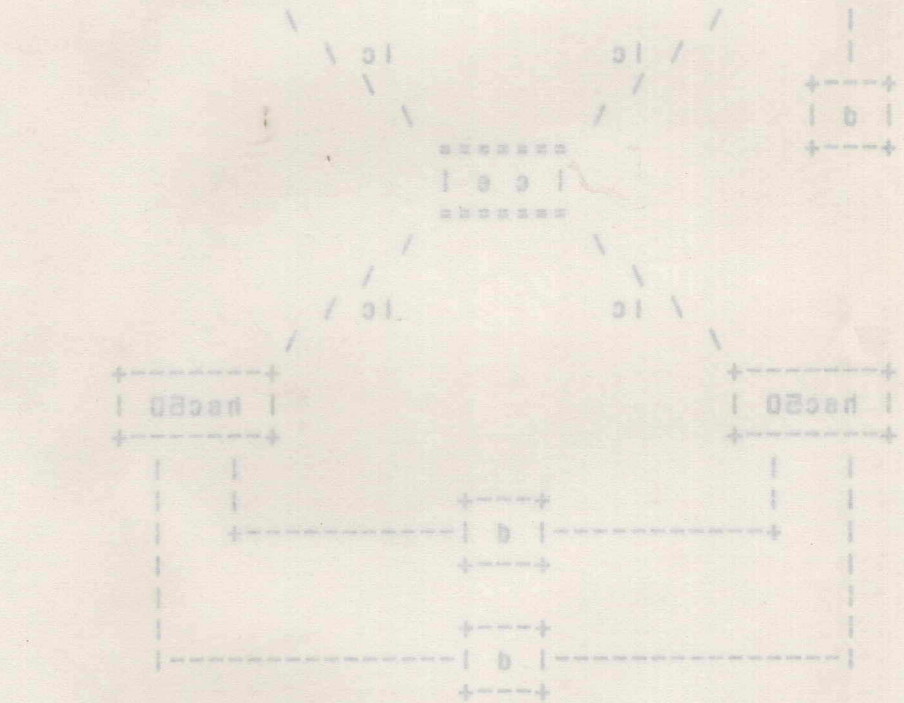
Controla o acesso a unidades de disco com especificação DSA (DIGITAL Storage Architecture).

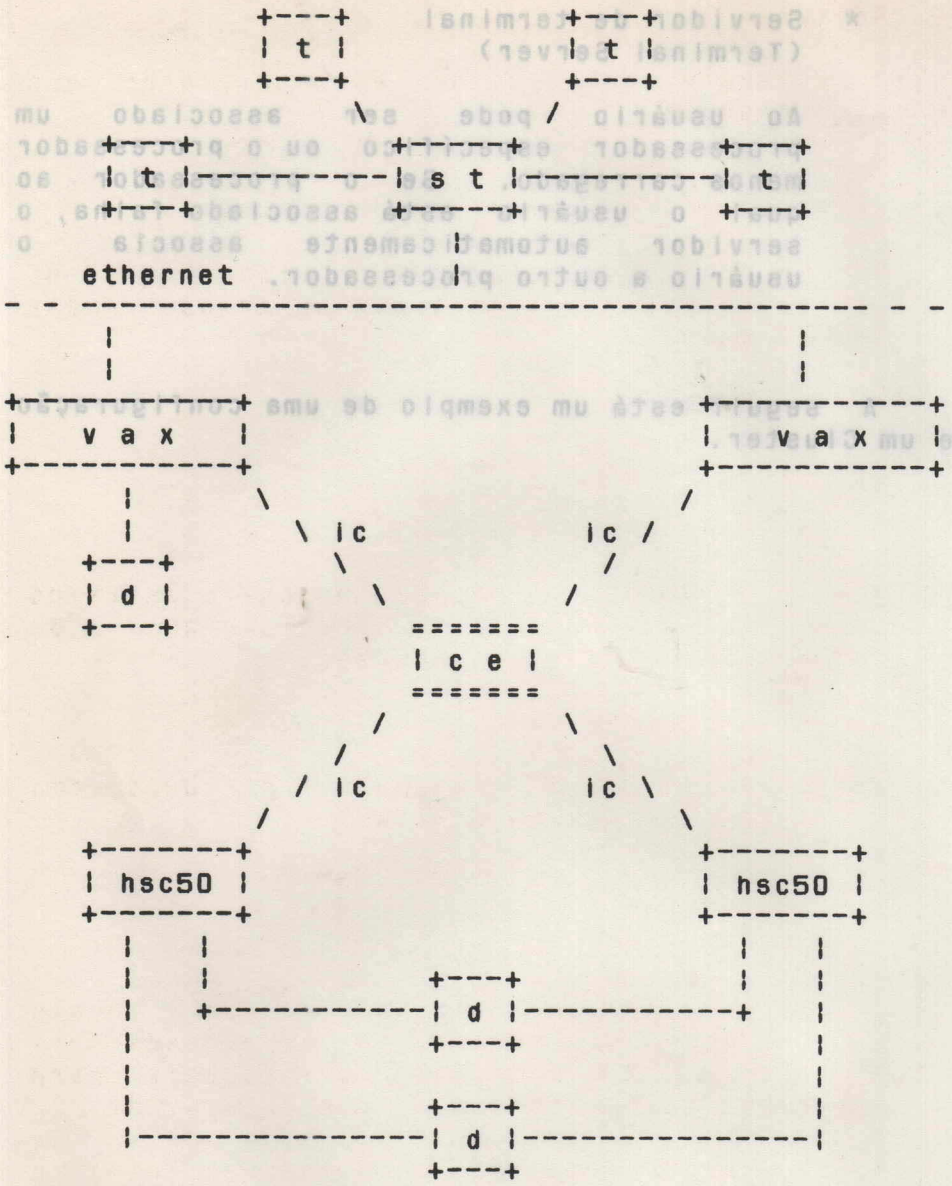
Uma unidade de disco pode ser conectada simultaneamente a dois HSC50. Neste caso, se um dos HSC50 falhar, o outro assume todas as suas operações.

* Servidor de terminal
(Terminal Server)

Ao usuário pode ser associado um processador específico ou o processador menos carregado. Se o processador ao qual o usuário está associado falha, o servidor automaticamente associa o usuário a outro processador.

A seguir está um exemplo de uma configuração de um Cluster.




t
t
t
s t
ethernet
vax
vax
ic
ic
d
c e
ic
ic
hsc50
hsc50
d
d

9.2 Software

O software do Cluster está dividido nos seguintes módulos:

* Serviços de comunicação do sistema

  Responsável pela comunicação entre os nós. Suporta três tipos de transferência de dados:

  - Datagrama

    Usado pelo DECNET

  - Mensagem seqüencial

    Usado para pequenas mensagens entre os nós (exemplo: requisições do controlador de recursos)

  - Blocos de dados

    Usado para transporte de dados como e/s de discos e fitas

* Servidor MSCP

  Este servidor torna discos locais (ligados diretamente a um processador) tipo MASSBUS e UNIBUS, disponíveis para todos os outros nós ativos do Cluster.

  O protocolo de comunicação entre o processador e o controlador de dados chama-se MSCP (Mass Storage Control Protocol).

  Este servidor transforma as mensagens em MSCP para pedidos de e/s MASSBUS ou UNIBUS, e vice-versa.

* Gerente de conexão

Ele é o responsável pelo controle dinâmico do Cluster.

Uma das suas tarefas é verificar o estado dos nós do Cluster para mantê-lo "consistente" (via QUORUM).

* Gerente de bloqueio de recursos distribuído

Este gerente sincroniza o uso dos recursos do Cluster, através de rotinas primitivas (disponíveis ao usuário) para loquear e liberar estes recursos. Se um nó falha, todos os seus recursos bloqueados são liberados.

A verificação de "deadlock" (impasse na alocação de recursos) é feita automaticamente.

Este gerente é usado pelo sistema de arquivos e pelo controlador de tarefas.

* Sistema de arquivos distribuído

Os registros de um arquivo podem ser compartilhados por todo o Cluster. Para permitir este acesso compartilhado existe uma serialização das requisições de entrada e saída.

* Controlador de tarefas distribuído

As filas de impressão e "batch" podem ser compartilhadas por todo o Cluster.

As tarefas ("batch" ou impressão) podem ser submetidas em filas específicas ou em filas genéricas.

Quando uma tarefa é submetida em uma fila genérica, será escolhido o nó menos atarefado para a sua execução.

* Quorum

Este algoritmo é usado para determinar quando um Cluster pode estar em funcionamento.

Ele está baseado em duas informações:

- Votos

  A cada nó é atribuída uma certa quantidade de votos (geralmente apenas um).

- Quorum

  Soma dos votos dividido por dois, mais um, truncado para inteiro.

O Cluster só pode funcionar enquanto o número de votos for maior ou igual ao quorum.

No caso de um Cluster com apenas dois processadores, se um falhar, o processamento de todo o Cluster pára.

Uma relação de importância pode ser atribuída aos processadores do Cluster através do número de votos de cada processador. Por exemplo, uma relação mestre-escravo pode ser estabelecida em um Cluster com dois processadores (um não tem voto, e o outro um voto).

As tarefas ("batch" ou impressão) podem ser submetidas em filas específicas ou em filas genéricas.

Quando uma tarefa é submetida em uma fila genérica, será escolhido o nó menos atarefado para a sua execução.

* Quorum

Este algoritmo é usado para determinar quando um Cluster pode estar em funcionamento.

Ele está baseado em duas informações:

- Votos

  A cada nó é atribuída uma certa quantidade de votos (geralmente apenas um).

- Quorum

  Soma dos votos dividido por dois, mais um, truncado para inteiro.

O Cluster só pode funcionar enquanto o número de votos for maior ou igual ao quorum.

No caso de um Cluster com apenas dois processadores, se um falhar, o processamento de todo o Cluster pára.

Uma relação de importância pode ser atribuída aos processadores do Cluster através do número de votos de cada processador. Por exemplo, uma relação mestre-escravo pode ser estabelecida em um Cluster com dois processadores (um não tem voto, e o outro um voto).

Impressão e acabamento
(com filmes fornecidos):
EDITORA SANTUÁRIO
Fone (0125) 36-2140
APARECIDA - SP

VAX

UMA INTRODUÇÃO

Este livro apresenta o sistema VAX para gerentes de processamento de dados, programadores de aplicações, estudantes de Informática e usuários não especializados de sistemas de computação.

O texto apresenta as principais características dos recursos de "software" e "hardware" do equipamento, os conceitos de tecnologia VAX, os comandos disponíveis, a operação, e a gerência do sistema.

O texto não tem como objetivo servir de manual de utilização de um sistema VAX (apesar de apresentar inúmeros comandos). Para esse fim devem ser utilizados os manuais da DEC (Digital Equipment Corporation) — fabricante dos computadores VAX no exterior — ou da Elebra Computadores, fabricante no Brasil.

EDMUNDO BASTOS TORREÃO / MARCOS VIANNA VILLAS

são professores do Departamento de Informática da PUC/RJ e possuem larga experiência em aplicações e equipamentos VAX.

ISBN 85-7001-403-1