MODELAGEM DO SISTEMA Minhoc0:
um simulador orientado a objeto para o ensino de redes de computadores

Henrique **ALMEIDA** - **Serge LALANNE** - Mauro **OLIVEIRA***

Universidade Estadual do Ceará - UECE
Escola Técnica Federal do Ceará - ETFCe

Av. Paranjana, 1700 - Campus do Itaperi
CEP 60715 Fortaleza - CE
e-mail: lhp@bruece.bitnet

RESUMO

MinhocO é um sistema para o ensino de redes de computadores e sistemas distribuídos que faz uso de recursos de CAL ("Computer Aided Learning"). Trata-se de um sistema orientado à objeto, constituído de três componentes: uma rede local didática, um método de utilização desta rede e um ambiente de desenvolvimento de protótipos. Este trabalho apresenta a modelagem do sistema MinhocO baseada na notação ASN.1 e discute aspectos da nova arquitetura do sistema.

ABSTRACT

MinhocO is a CAL(Computer Aided Learning) system to be used in a technical course about computer network and distributed systems. It's an object-oriented system with three main components: a local area network, a method for better use of this network and a prototype development environment. This work presents the modelling with ASN.1 notation and features of the new architecture of this system.

Palavras chave: CAL, Protótipos, Arquitetura, Modelagem Orientada a Objeto, ASN.1, Geração de Protótipos.

1. INTRODUÇÃO

Nas atividades de ensino ainda utiliza-se, técnicas e recursos tradicionais (quadro negro, transparências, etc). Esses recursos nem sempre são eficazes no processo ensino-aprendizagem de assuntos técnicos.

O Sistema Minhoc0, objeto deste trabalho, faz parte do Projeto Minhoca [Oli86] cuja proposta é uma ferramenta para auxiliar o ensino de redes de computadores e sistemas distribuídos. Ele utiliza recursos de CAL ("Computer Aided Learning") e tem evoluído em decorrência de sua utilização e de sua potencialidade como ambiente de pesquisa [Oli87].

---

* Autor financiado pelo CNPQ e ETFCe

As versões do Projeto Minhoca são as seguintes:

1. Versão Minhoca:

A versão Minhoca é basicamente uma rede local sem componentes eletrônicos (Zero Slot LAN) [Coe89]. Esta rede possui um programa (protocolo de acesso ao meio físico e diversas aplicações) e conectores (padrão EIA-RS232c) ligados a um par trançado (topologia em barramento). Esta primeira versão foi implementada em Pascal, com microcomputadores IBM-PC (compatíveis).

2. Versão Minhonix:

A versão Minhonix [Oli91] acrescenta um método didático à Rede Minhoca, propondo um conjunto de procedimentos para a utilização desta rede. Além disso, outros protocolos foram desenvolvidos e adicionados ao sistema, tendo uma parte sido implementada com a linguagem C sob MINIX [Tan89] (um sistema operacional compatível com o UNIX, concebido para ensino).

Minhonix faz uso de técnicas de "prototipagem" [Kri92]. Os Protótipos Minhonix são programas (serviços/protocolos e aplicações) que permitem a construção de uma rede de computadores passo a passo. Estes protótipos são "o coração" do sistema e apresentam as seguintes características pedagógicas:

- Podem ser usados, pelo professor, como uma ferramenta para a demonstração de conceitos clássicos (anexo 1) ou pelo aluno como sistema de desenvolvimento, de acordo com as atividades propostas no citado método.

- Permitem a simulação de problemas clássicos em uma rede local (Minhonix não é uma simulação em uma única máquina). O sistema fornece um ambiente interativo, onde a construção de soluções para os problemas simulados é trabalhada cooperativamente pelo professor e pelo aluno, proporcionando um aprendizado mais eficaz.

3. Versão MinhocO:

Esta versão tem uma concepção orientado à objeto [Mey88] e está sendo implementada na linguagem C++ em microcomputadores IBM-PC (compatíveis). O Sistema MinhocO (Minhoca Orientada à Objeto) incorpora os elementos de base e o método do Sistema Minhonix. Assim, ele fornece varios protótipos para a realização das atividades práticas e teóricas de laboratório, as quais são organizadas de acordo com a arquitetura OSI/ISO, descritas no Guia do Usuário [Oli92] (um guia de instalação da rede também é disponível).

Este trabalho apresenta o Sistema MinhocO, sua modelagem [Alm92] baseada na notação ASN.1 [TIM88] e uma nova arquitetura. Exemplos de utilização e uma descrição do ambiente (atividades e agentes) do sistema são também apresentados.

2. EXEMPLOS DE UTILIZAÇÃO

O Método pedagógico proposto para a utilização do Sistema MinhocO se decompõe em várias atividades de ensino, distribuídas em três momentos ou fases: **Sensibilização, Análise** (problemática, análise de uma solução, implementação) e **Síntese.**

Estes momentos ou fases pedagógicas são ilustrados, a seguir, com analogias. Cada fase é acompanhada de uma frase que sugere o objetivo pedagógico que se pretende atingir:

- **Sensibilização (O QUE É UMA** rede de computadores?):

Neste primeiro momento, o cenário é um em que o comportamento do aluno assemelha-se ao de um ***Comprador (Cliente)*** *e o do professor a um* ***Vendedor (Fornecedor)*** *de um produto* ***(Rede Local).***

- **Análise (COMO FUNCIONA ESSA** rede?):

Neste segundo momento, o aluno e o professor ***são membros*** *de uma mesma empresa (de compra ou venda), chefiada inicialmente pelo professor.*

- **Síntese (ESTA É A MINHA** solução):

Neste terceiro momento, os papeis do aluno e do professor são invertidos em relação ao primeiro momento: agora, aluno e professor comportam-se como ***Fornecedor e Cliente*** *de um produto, respectivamente.*

Em seguida, são apresentados exemplos de utilização do Sistema MinhocO (rede local e método), para cada um dos momentos acima:

* **Exemplo da fase Sensibilização:** "O que é uma rede de computadores?"

O professor apresenta aos alunos as primeiras idéias de uma rede de computadores. Para tanto, ele realiza demonstrações com um certo protótipo da Rede MinhocO, apresentando aplicações típicas: correio eletrônico, transferência de arquivo, aplicação distribuída, jogos, etc. O objetivo nesta fase é, principalmente, motivar os alunos, esclarecendo os conceitos básicos e criando expectativas para os próximos assuntos. Os alunos podem participar das demonstrações como usuários do sistema.

Espera-se, no final, que o aluno seja capaz de:

-Definir uma rede de computadores;
-Conceituar os principais componentes de uma rede;
-Caracterizar as redes locais e redes de longa distância (LAN x WAN);
-Distinguir aplicações de serviços e protocolos de comunicação.

* **Exemplo da fase Análise:** "Enlace de dados."

O professor faz demonstrações com determinados protótipos da Rede Minhoc0, tendo como objetivo ilustrar o conceito de um protocolo de comunicação no nível de enlace. Ele realiza troca de mensagens entre estações da rede, com hipóteses simplificadoras (por ex. suporte físico ideal) que abstraem a complexidade do experimento. Erros são produzidos intencionalmente. Em seguida, o professor refaz as demonstrações introduzindo, gradativamente, elementos que evidenciam todas as etapas necessárias ao envio de uma mensagem em uma situação real: identificação de um posto receptor, multiplexagem do suporte físico de comunicação para os quadros (mensagens), detecção e correção de erros, etc.

Depois da discussão dos problemas abordados e das soluções possíveis, as experiências devem ser refeitas, agora com as versões de programas (protótipos) que possuem as soluções aos problemas identificados anteriormente.

Espera-se, ao final, que o aluno seja capaz de:

-Descrever os processos de envio e de recepção de uma mensagem,
-Descrever mecanismos de detecção e correção de erros,
-Descrever mecanismos de acesso ao suporte físico de comunicação,
-Definir um protocolo de comunicação a nível de enlace.

* **Exemplo da fase Síntese:** "Protocolo & aplicação"

Nesta fase, o professor sugere aos alunos trabalhos práticos baseados ou não nas experiências que já foram realizadas. Por exemplo: a implementação de um outro tipo de protocolo de acesso, um servidor de nomes simples, um sistema de correio eletrônico, a simulação de incoerência em banco de dados distribuídos, etc.

Espera-se, no final, que o aluno seja capaz de:

- Implementar algoritmos e protocolos simples,
- Melhorar e implementar aplicações de redes.

MinhocO apresenta-se também como um simulador. Quatro tipos de simulação são identificados no sistema:

• O primeiro tipo de simulação corresponde às abstrações feitas a partir de hipóteses simplificadoras do sistema, já comentadas. Desta forma, um problema específico pode ser abordado abstraindo-se a complexidade do sistema como um todo, conforme a descrição da fase de **Análise** acima.

• O segundo tipo de simulação corresponde às situações onde as hipóteses abstraídas acima são consideradas (passo à passo). Neste caso, o sistema evolui em direção à uma rede convencional (como ilustrado no clássico exemplo do protocolo de enlace no livro "Computer Networks" [Tan91]).

• O terceiro tipo de simulação corresponde a problemas que aparecem em uma rede de computadores por acaso e cujas soluções são, naturalmente, exigidas. Com o Minhoc0, essas situações podem ser produzidas, isto é, os problemas podem ser provocados e suas soluções analisadas em detalhes.

• O quarto tipo de simulação é a que permite a configuração dos parâmetros a fim de se observar e de se avaliar os resultados. Um exemplo simples é a configuração da velocidade de transmissão/recepção de dados sobre o suporte físico. Em um outro exemplo mais complexo, pode-se configurar o tamanho dos pacotes (frame) e analisar-se o desempenho das diferentes políticas de acesso ao suporte de comunicação, em função de parâmetros (também configuráveis) tais como tamanho do pacote, taxa de erros, etc.

3. A ARQUITETURA DO SISTEMA MinhocO

3.1. Descrição do Ambiente

O Sistema MinhocO possui o mesmo ambiente das versões precedentes (Minhoca e Minhonix): três Agentes (figura 1) e quatro Atividades (figura 2), descritos a seguir :

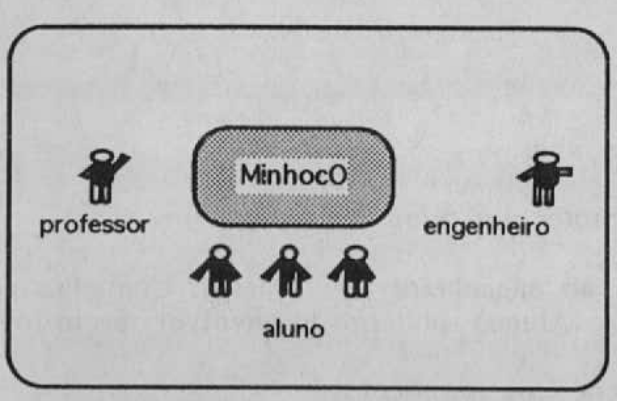

Figura 1: Agentes do sistema

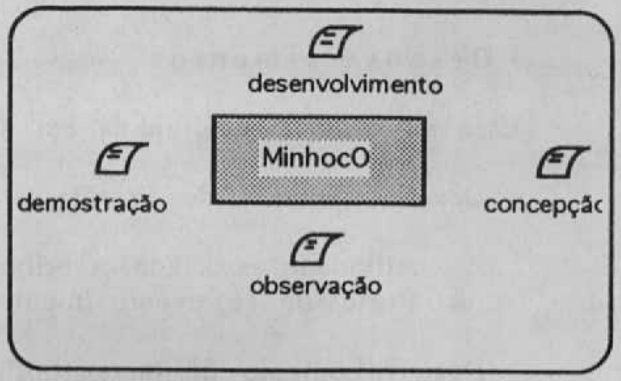

Figura 2: Atividades do sistema

AGENTES:

• O Engenheiro (sistema):

É o agente que implementa os protótipos do sistema. Ele conhece todos os detalhes de baixo nível do sistema ("hardware" e sistema operacional), indispensáveis à concepção dos protótipos.

• O Professor:

É um usuário do sistema. Ele utiliza os protótipos para demonstrações e para proposição de trabalhos práticos aos alunos, de acordo com o Método MinhocO proposto no Guia do Usuário.

• O Aluno:

É outro usuário do sistema. É o agente que observa as demonstrações e usa o MinhocO como sistema refazendo experiências ou como desenvolvimento em trabalhos orientados.

ATIVIDADES:

• Observação:

Essa atividade é a manifestação mais evidente do sistema dentro do seu objetivo educativo. Ela se verifica com mais frequência no primeiro momento pedagógico do método MinhocO (**Sensibilização**);

• Demonstração:

Trata-se de uma atividade dual a atividade de observação, comentada acima. Compete ao professor realizar as demonstrações, sendo previsto no método proposto a repetição desta demonstrações pelo aluno.

• Concepção:

Essa é uma atividade exclusivamente de planejamento dos protótipos que envolve o Professor e o Engenheiro do sistema: o modelo pedagógico de um protótipo proposto pelo professor é viabilizado (ou ajustado) em função das características do sistema (competência do Engenheiro). O agente Professor comporta-se como um arquiteto do protótipo na interação com o engenheiro.

• **Desenvolvimento:**

Essa atividade é subdividida em dois itens.

- Desenvolvimento de Protótipos (protocolos e/ou serviços):

  Atividade associada, a priori, ao engenheiro do sistema. Contudo, o Professor (e eventualmente o Aluno) poderá desenvolver protótipos.

- Desenvolvimento de novas aplicações ou protocolos:

  Atividade prevista para o aluno no terceiro momento pedagógico no método (**Síntese**) .

3.2 A Arquitetura do Sistema

A arquitetura funcional do Sistema MinhocO (bem como das versões precedentes) é composta dos seguintes módulos e interfaces (figura 3):

• **Modulos**

MSC: Módulo Suporte de Comunicação,
MPM: Módulo de Protótipos MinhocO.

• **Interfaces:**

icon: Interface de Concepção,
iobs: Interface de Observação,
idem: Interface de Demonstração.

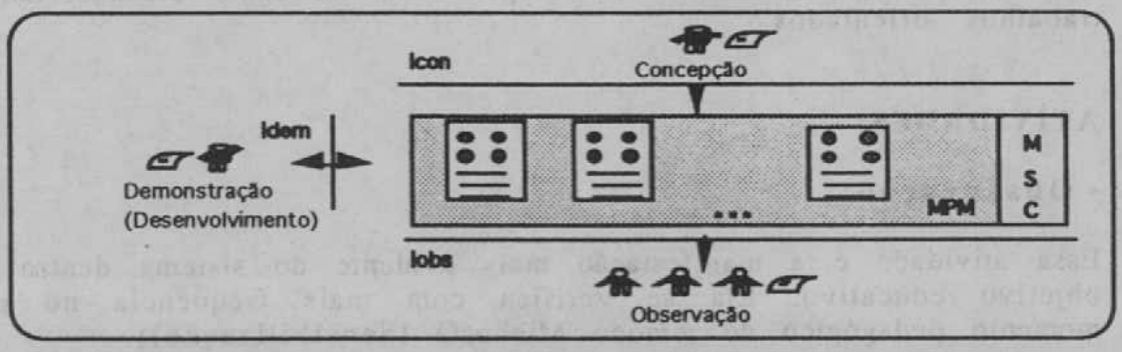

Figura 3: arquitetura atual

O mapeamento entre os Agentes e as Atividades no sistema é mostrado a seguir, na tabela 1:

	Concepçãc	Desenvolvimento	Demonstração	observação
Aluno				•
Professor	•		•	
Engenheiro	•	•		

Tabela 1: mapeamento Agente/Atividade da arquitetura atual

4 MODELAGEM DO SISTEMA MinhocO

4.1 Os protótipos do Sistema MinhocO

Os sistemas operacionais tornaram-se elementos primordiais nos ambientes computacionais. Atualmente, dizer que se está trabalhando "sob MSDOS ou UNIX" é mais frequente do que dizer "sob a máquina X ou Y". Em consequência, os vários protótipos desenvolvidos no Sistema MinhocO são organizados em função do sistema operacional sob o qual foram desenvolvidos. São eles:

- Protótipos MhDOS (desenvolvidos sob MSDOS) et
- Protótipos MhNIX (desenvolvidos sob MINIX)

Em seguida, são apresentadas a descrição informal e a definição de classes do modelo correspodente aos protótipos MhDOS. Mais simples que os protótipos MhNIX, os protótipos MhDOS ilustram bem os elementos de base da arquitetura da Rede MinhocO.

4.2 Modelagem Estática

Um modelo de um sistema é simplesmente um meio de se organizar conhecimentos, com o intuito de se explicar como este sistema funciona. Assim, um modelo descreve o comportamento externo do sistema, independente de uma particular implementação.

Na técnica de modelagem utilizada neste trabalho, nós usamos como paradigma o documento normalizado do grupo de estudo T1M1.5[T1M1.88] (organismo de normalização americano) que define uma metodologia por etapas: o modelo é descrito pela definição das classes, pelo seu nome, seu tipo e os atributos (informações) que essas classes possuem.

4.2.1 Descrição Informal (etapa 1):

A primeira etapa corresponde à identificação das classes pelo diagrama Entidade/Relação (fig.4). A descrição informal pode ser assim expressa:

> *A **Rede** MinhocO contém um **suporte** físico de comunicação e **máquinas** (microcomputadores) que são interconectadas por esse suporte para a troca de **mensagens**. Deve-se executar em cada máquina as **aplicações** de rede e os **protocolos** de comunicação necessários a realização de um experimento. MinhocO aborda, principalmente, as camadas de **nível 2** e de **nível 4** correspondentes a arquitetura ISO. A partir de **hipóteses** feitas sobre o suporte de comunicação, dispõe-se diversos protocolos de **enlace** de dados (nível 2) de acordo com essas hipóteses. O protocolo de **acesso** (também do nível 2) exerce uma importante função no Sistema.*

Assim, os recursos do sistema podem ser modelados pelas entidades **rede, suporte, máquina, mensagem, protocolo, nível2, nível4, enlace, acesso.**

O fato de que ao **suporte** de comunicação podem ser atribuídas diversas hipóteses e, por consequência, ser associado a várias versões de protocolo de **enlace**, implica que a relação que associa estas classes (enlace/suporte), possui propriedades. Portanto, uma classe objeto relação **hipótese** é criada tendo **hipo** como atributo.

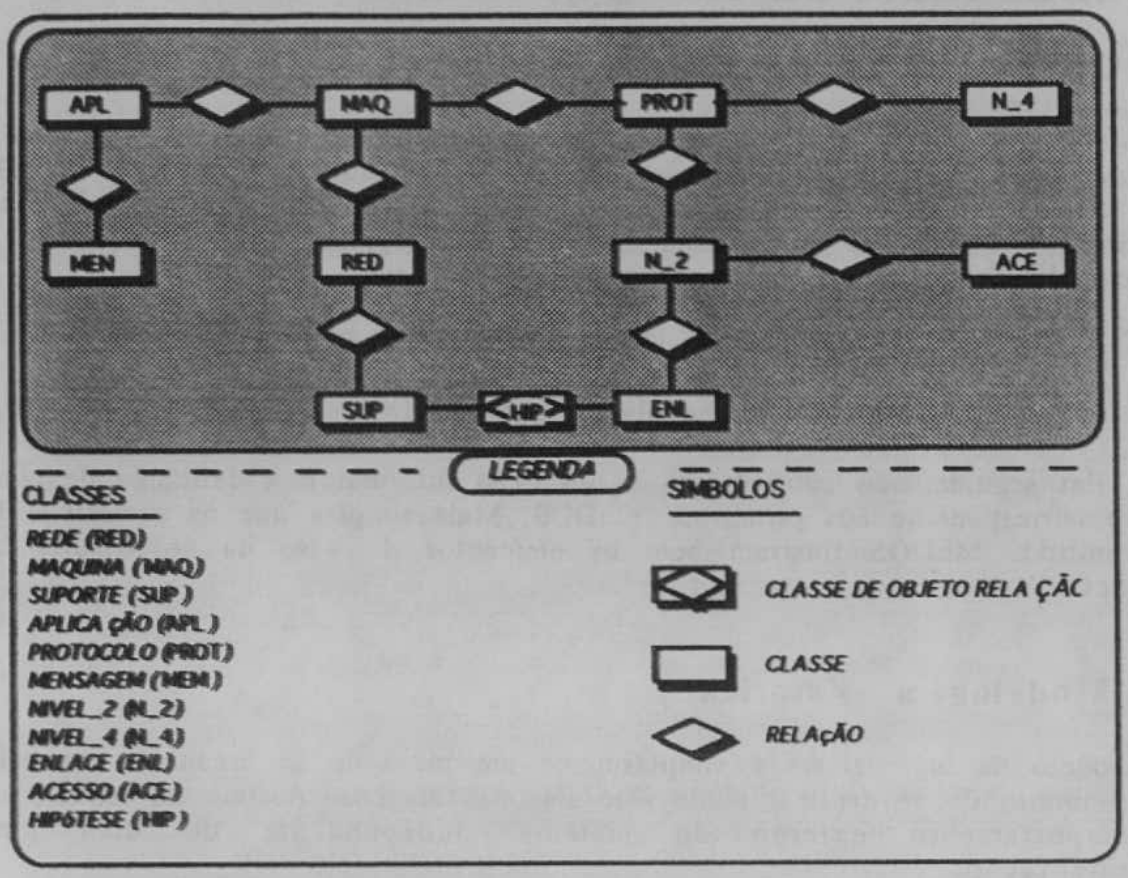

Figura 4: diagrama de entidade/relação

4.2.2 Definição de Classes (etapa 2)

A segunda etapa é a definição de classes de objetos:

- A classe **rede** modela o sistema e possui atributos especificando sua localização geográfica (*localização*), as versões (*versões*) e as línguas em que o sistema se apresenta ao usuário (*línguas*);

- A classe **máquina** modela a noção de estação, e possui atributos especificando seu estado (*estado*), a identificação da estação (estação) e o sistema operacional sob o qual a versão foi implementada (*sistema*);

- A classe **suporte** modela o suporte físico de comunicação, e possui atributos especificando as hipóteses feitas sobre o suporte (*hipo*) e a taxa de transmissão e de recepção de octetos (*taxa*);

- A classe **protocolo** modela as camadas (bascada na arquitetura OSI), e possui atributos especificando o tipo de camada (*camada*) e o tipo de conexão (*tipo-conexão*);

- A classe **aplicação** modela os programas de aplicação de rede que utilizam os protocolos de comunicação, e possui atributos especificando o tipo de aplicação (*tipo-aplicação*) e os protocolos utilizados (*tipo-protocolo*);

- A classe **mensagem** modela as mensagens trocadas entre estações, e possui atributos especificando a rede destino (*rede-destino*), a rede origem (*rede-origem*), e a estrutura de dados (*pacote*);

• A classe **nível4** modela a camada transporte, e possui atributos especificando o tipo de conexão (*tipo-conexão*) e a quantidade de conexões ativas (*numero-conexão*) ;

• A classe **nível2** modela a camada ligação, e possui atributos especificando o tipo de enlace (*tipo-enlace*) e o tipo de acesso ao suporte de comunicação (*tipo-acesso*);

• A classe **enlace** modela os protocolos de enlace de dados de nível 2, e possui atributos especificando as hipóteses sobre o suporte físico e a máquina (*hipótese*);

• A classe **acesso** modela os protocolos de acesso de nível 2, e possui atributos especificando os tipos de acesso (*tipo-acesso*);

• A classe **hipótese** representa a relação entre as entidades **enlace** e **suporte**, e possui atributos especificando as hipóteses sobre o suporte físico e a máquina (*hipo*).

4.2.3 Especificação Formal (etapa 3)

A terceira etapa na metodologia usada é a representação formal do modelo (fig 5). Ela faz uso de um subconjunto da notação ASN.1.

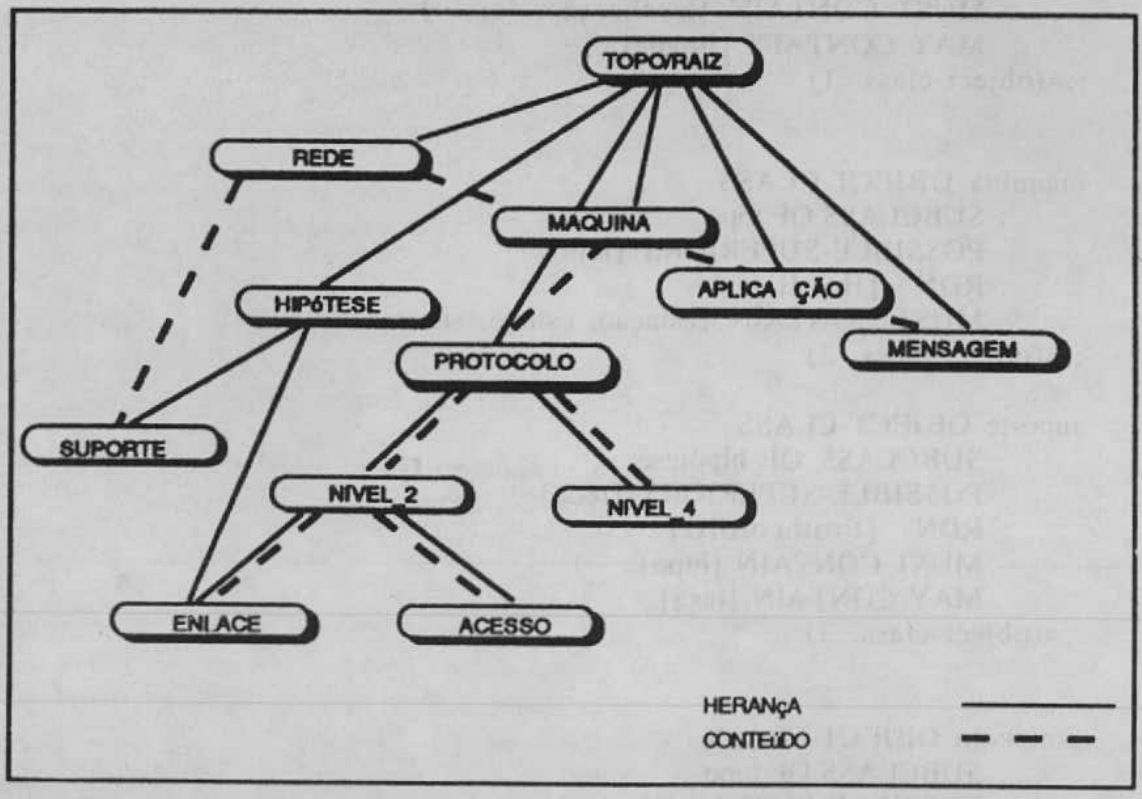

Figura 5: estrutura de herança e de conteúdo

A macro OBJECT-CLASS pode ser utilizada para a definição de classes de objetos [Ros90].

As classes são definidas pelos seguintes elementos:

•A referência do objeto seguida pela palavra chave OBJECT-CLASS;

•A palavra chave SUBCLASS OF seguida pela classe mãe (arvore de herança);

•A palavra chave POSSIBLE SUPERIOR seguida pelas classes superiores (arvore de "contenance");

•A palavra chave RDN seguida por uma lista de referências que correspondem ao "relative distiguished name" ;

•A palavra chave MUST CONTAIN seguida por uma lista de atributos que são mandatários .

•A palavra chave MAY CONTAIN seguida por uma lista de atributos que são opcionais.

;;;Classes de objetos

```
rede OBJECT-CLASS
        SUBCLASS OF topo
        POSSIBLE-SUPERIORS {raiz}
        RDN   {RedeID}
        MUST CONTAIN {localização, versão}
        MAY CONTAIN {língua}
::=(object-class  1)


máquina OBJECT-CLASS
        SUBCLASS OF topo
        POSSIBLE-SUPERIORS {rede}
        RDN   {HostID}
        MUST CONTAIN {estação, estado, sistema}
::=(object-class  2)

suporte OBJECT-CLASS
        SUBCLASS OF hipótese
        POSSIBLE-SUPERIORS {rede}
        RDN   {ProtocoloID}
        MUST CONTAIN {hipo}
        MAY CONTAIN {taxa}
::=(object-class  3)


protocolo OBJECT-CLASS
        SUBCLASS OF topo
        POSSIBLE-SUPERIORS {máquina}
        MUST CONTAIN {camada, tipo-conexão}
::=(object-class  4)
```

```
aplicação OBJECT-CLASS
        SUBCLASS OF topo
        POSSIBLE-SUPERIORS {rede, máquina}
        RDN {AplicaçãoID}
        MUST CONTAIN {tipo-aplicação}
        MAY CONTAIN {tipo-protocolo}
::=(object-class 5)

mensagem OBJECT-CLASS
        SUBCLASS OF topo
        POSSIBLE-SUPERIORS {rede, máquina, aplicação}
        RDN {ProtocoloID}
        MUST CONTAIN {pacote}
        MAY CONTAIN {rede-destino, rede-origem}
::=(object-class 6)


nível_4 OBJECT-CLASS
        SUBCLASS OF protocolo
        POSSIBLE-SUPERIORS {protocolo, máquina, rede}
        RDN {TransporteID}
        MUST CONTAIN {tipo-conexão}
        MAY CONTAIN {numero-conexão}
::=(object-class 7)


nível_2 OBJECT-CLASS
        SUBCLASS OF protocolo
        POSSIBLE-SUPERIORS {protocolo, máquina, rede}
        RDN {LigaçãoID}
        MUST CONTAIN {tipo-enlace, tipo-acesso}
::=(object-class 8)


enlace OBJECT-CLASS
        SUBCLASS OF hipótese, nível_2
        POSSIBLE-SUPERIORS {nível_2, protocolo, máquina, rede}
        RDN {EnlaceID}
        MUST CONTAIN {hipo}
::=(object-class 9)

acesso OBJECT-CLASS
        SUBCLASS OF nível_2
        POSSIBLE-SUPERIORS {niveau2, protocolo, machine, reseau}
        RDN {AccesID}
        MUST CONTAIN {tipe-acesso}
        MAY CONTAIN {taxa}
::=(object-class 10)


hipótese OBJECT-CLASS
        SUBCLASS OF topo
        MUST CONTAIN {hipo}
::=(object-class 11)
```

Os atributos correspondentes encontram-se especificados no ANEXO II.

4.3 modelagem Dinâmica

Como dito na descrição informal (item 4.2.1), dentre as classes do Sistema MinhocO identificadas no modelo estático, destaca-se a classe **nível2.**

Em seguida são descritos os protocolos de envio e recepção de pacotes (ou quadros) correspondentes à sub-camada MAC do Sistema MinhocO (figuras 6 e 7).

- **Recepção de um pacote (quadro):**

O protocolo de recepção utilizado nos protótipos MhDOS usa a interrupção da saída serial: Cada octeto (dado) que chega à estação produz uma interrupção de "hardware", a qual é ativada pelo circuito serial (UART) do microcomputador. Essa interrupção provoca a execução de um programa (figura 6) previamente instalado (residente) no endereço correspondente ao vetor desta interrupção.

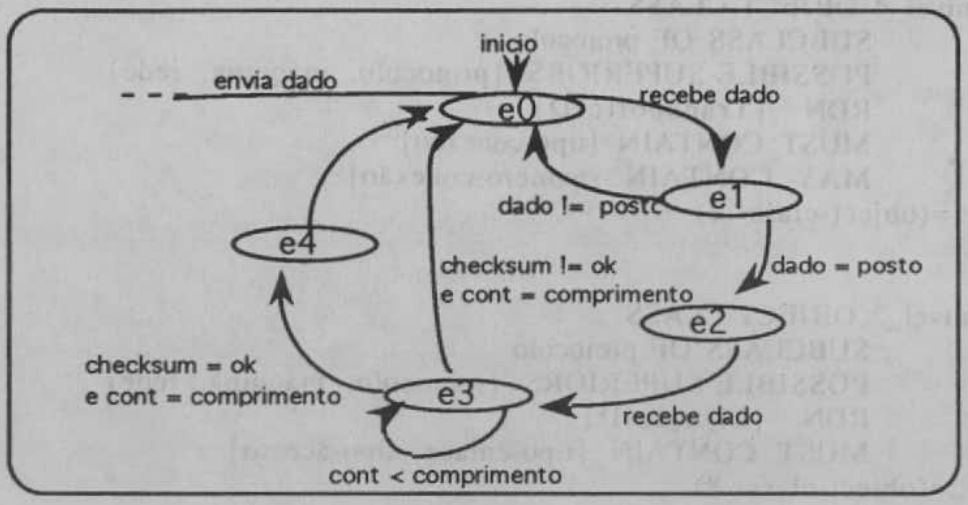

Figura 6: diagrama de estado da recepção

- **Envio de um pacote (quadro):**

A técnica utilizada no compartilhamento do suporte de comunicação da Rede MinhocO é baseada no método NP-CSMA (Non Persistent Carrier Sense Multiple Access). Protocolos de acesso CSMA/CD e Passagem de ficha (anel virtual) são também disponíveis no sistema.

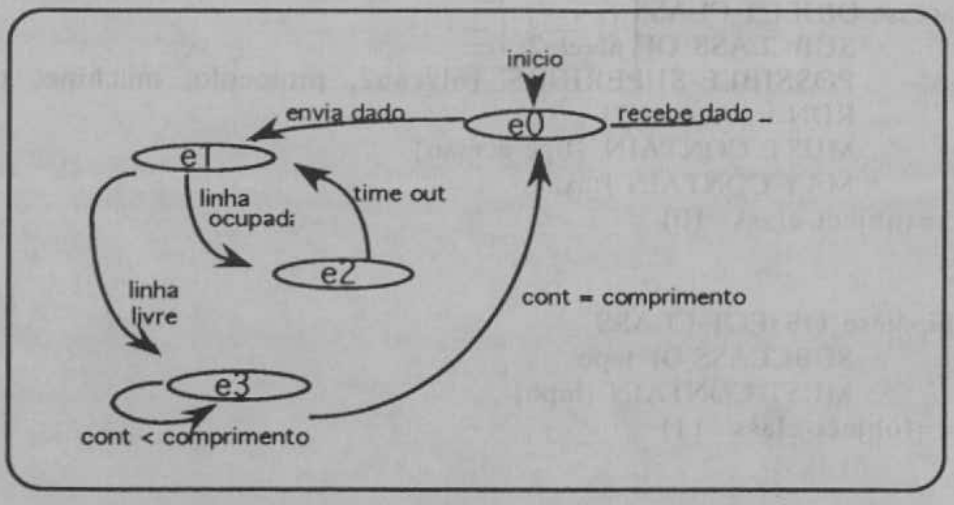

Figura 7: diagrama de estado da transmissão

5. EM DIREÇÃO A UMA NOVA ARQUITETURA

5.1 Ambiente de Desenvolvimento

Como observado anteriormente, a atividade **Desenvolvimento** no Sistema MinhocO pode, eventualmente, ser exercida pelos agentes Aluno e Professor. A dificuldade inerente ao desenvolvimento de protótipos na arquitetura atual (figura 3) decorre da necessidade do agente conhecer detalhes de "hardware" do microcomputador e do sistema operacional utilizados: os protótipos na atual arquitetura foram concebidos como blocos únicos, integrando em tempo de compilação elementos de base (protocolos e serviços) e aplicações (correio eletrônico, transferência de arquivos, jogos, etc).

A nova arquitetura (figura 8), visa favorecer a atividade de Desenvolvimento no sistema. A idéia de base é fornecer ao Agente Professor (e eventualmente ao Aluno), facilidades permitindo ao mesmo desenvolver seu próprio modelo conceitual (protótipos) de uma forma transparente, ou seja, independente das características de baixo nível do sistema. A nova arquitetura funcional do sistema apresenta os seguintes módulos e interfaces:

- **Módulos**
  - **MSC:** Módulo Suporte Físico de Comunicação,
  - **MPM:** Módulo de Protótipos MinhocO.
  - **MOB:** Módulo de Objetos de base
  - **MGP:** Módulo Gerador de Protótipos,

- **Interfaces:**
  - **icon :** Interface de Concepção,
  - **iobs :** Interface de Observação,
  - **idem:** Interface de Demonstração,
  - **ides :** Interface de Desenvolvimento.

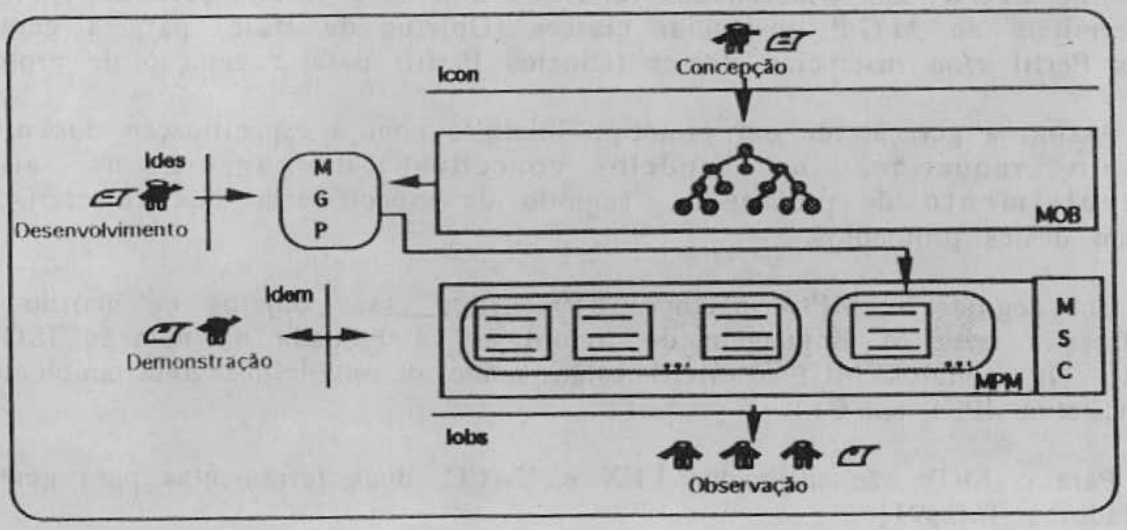

Figura 8: nova arquitetura proposta

O novo mapeamento entre agentes e as atividades são mostrados na tabela 2.

	Concepçãc	Desenvolvimento	Demonstração	Observação
Aluno		•		•
Professor	•	•	•	
Engenheiro	•	•		

Tabela 2: mapeamento Agente/Atividade da nova arquitetura proposta

5.2 Geração de Protótipos

Ao compararmos a atual arquitetura (fig. 3) com a proposta neste item (fig.8), verificamos a presença de dois novos módulos e de uma nova interface: o Módulo Gerador de Protótipos (**MGP**) [Oli93], o Módulo de Objetos de base (**MOB**) e a Interface de Desenvolvimento (**ides**). A proposta consiste em permitir que a um agente na atividade **Desenvolvimento** especificar o seu próprio protótipo através da **ides**. A partir desta especificação o **MGP** instancia os objetos necessários, contidos na **MOB**, produzindo o protótipo especificado.

A geração de protótipos é baseada numa linguagem de "templates" [Lal92]. Uma "template" é um molde que permite a especificação parcial ou total do modelo de um Protótipo MinhoncO. Esta linguagem funciona como uma interface entre o agente na atividade de Desenvolvimento e os processos de geração de protótipos. Ela trata com objetos em tres níveis de granularidade, distribuídos em duas fases [Pen92]:

Fase de modelagem

- Objeto de Base:
  corresponde a uma classe de objetos de pequena granularidade;

- Objeto Perfil ("profile"):
  modelo coerente de objetos de base;

Fase de instanciação

- Objeto Protótipo:
  configuração coerente ou não de Objetos Perfil

No módulo **MOB** são armazenados todos os objetos e informações de configuração que permitem ao **MGP** instanciar classes (Objetos de Base) para a geração de Objetos Perfil e/ou instanciar classes (Objetos Perfil) para a geração de protótipos.

Assim, a geração de um protótipo inicia-se com a especificação dos níveis de protocolo requeridos no modelo conceitual do agente na atividade **Desenvolvimento** de protótipos, seguido da especificação das características de cada um destes protocolos.

Em seguida o MPG instancia e reagrupa esses objetos de acordo com a especificação feita. A linguagem de "templates" é baseada na notação ISO-ASN.1 [ISO91]. Os modulos MGP e MOB estão sendo desenvolvidos num ambiente SUN (SPARCstation IPC), em C++.

Para o MGP são utilizados LEX e YACC, duas ferramentas para geração de compiladores [Mas91]:

- LEX lê um arquivo de especificação contendo expressões (regulares) gerando uma rotina em C. Esta rotina será capaz de realizar a análise léxica de uma entrada de caracteres em função das especificações do arquivo lido, produzindo identificadores ("tokens").

- YACC lê um arquivo de especificação que codifica a gramática de uma linguagem gerando uma rotina em C ("parsing routine"). Esta rotina agrupa "tokens" dentro de uma sequência lógica, coerente com a especificação da gramática.

6. OUTROS DESENVOLVIMENTOS

• NETBIOS

O Sistema MinhocO conta com uma implementação do NETBIOS ("Network basic input output system"), uma interface padrão para recursos de redes locais IBM-PC. NETBIOS estabelece uma conexão e trata solicitações de recursos compartilhados de redes tais como servidores de disco, impressoras, etc. Esta versão foi desenvolvida na PUC-RJ [Juc91], tendo sido adaptada ao Sistema MinhocO. Neste contexto, vale também observar a disponibilidade no Sistema MinhocO, dos didáticos protocolos de enlace e transporte propostos no livro "Computer Network" [Tan91].

• Configuração remota

Atualmente, esforços estão sendo concentrados na implementação da nova arquitetura apresentada neste trabalho, em especial o gerador de protótipos [Oli93b]. Esta nova arquitetura permitirá ao Agente professor não só a concepção mais fácil de protótipos, de acordo com seu próprio modelo conceitual, como também a instalação destes protótipos à distância, nas estações da rede.

Para melhor explicitar a idéia, imaginemos várias redes MinhocO em um mesmo laboratório, onde os alunos devem utilizar diferentes protótipos do sistema (fase de **síntese**). O professor poderá, a partir de uma estação (central), comandar à distância, a instalação destes protótipos em cada uma das 4 estações de cada rede, em funcão, por exemplo, do desempenho de cada equipe. Isso representa uma enorme vantagem operacional comparada com o metódo manual de instalação dos protótipos a partir de disquetes. Aspectos pedagógicos são também beneficiados com a possibilidade de configuração remota, oferecida pela nova arquitetura.

• Versão MinhOSI

No contexto desta nova arquitetura, também busca-se a abordagem de conceitos correspondentes às camadas de alto nível que não são tratadas (ainda) no sistema MinhocO (ver anexo I). Para tanto, um protótipo que utiliza primitivas de serviço OSI, disponíveis no "pacote" ISODE [Ros91], está em vias de implementação.

ISO/DE é uma implementação não proprietária de protocolos definida pela ISO/IEC. O objetivo do projeto ISO/DE é, em fornecendo livremente este "pacote", acelerar o processo de desenvolvimento de aplicações OSI. Outra motivação é o desenvolvimento de serviços (transporte) OSI em termos de circuitos virtuais TCP. ISO/DE é um projeto em evolução. A versão mais recente (versão 8) oferece não só as principais implementações OSI (FTAM, CMISE, X400, etc) como também um ambiente completo para desenvolvimento de aplicações OSI e sistemas distribuídos baseado em compiladores ASN.1 [Ros91].

A idéia central da utilização de ISODE no sistema MinhocO é encapsular as rotinas que implementam as primitivas de serviço de alto nível OSI, de acordo com a arquitetura e a filosofia (método) apresentadas no início deste trabalho. Desta forma, conceitos do nível de aplicação ("Application process & application entity Information, ASE & ACSE, ROSE, RTSE, CMISE, etc") e do nível de apresentação ("Abstract Syntax & Transfert Syntax Notation, Context management & Syntax matching, etc") poderão ser ensinados (aprendidos) através do Sistema MinhocO.

Com a utilização ISO/DE, o Sistema MinhocO passará a abordar praticamente todos os conceitos de redes de computadores correspondentes às camadas da arquitetura OSI, além de melhor explorar o ensino de sistemas distribuídos.

CONCLUSÃO

O Sistema MinhocO tem sua origem associada a trabalhos desenvolvidos no primeiro semestre de 1985 durante o Curso de Redes Locais da PUC-RJ [Soa87].

O primeiro protótipo (Rede Minhoca) tornou-se operacional em dezembro de 1987, tendo sido utilizado nos anos seguintes em cursos sobre redes de computadores e em atividades de iniciação científica na ETFCe e na UFC.

O sistema, já na sua versão Minhonix (1989), foi apresentado em cinco universidades francesas. Desde então, ele vem servindo de suporte para cursos no "IUT Informatique" das Universidades de Lyon, Velizy, Paris IV e no "DESS Informatique" da Universidades Paris VI.

Os excelentes resultados referentes ao processo ensino/aprendizagem, obtidos desde o início com o sistema, motivou-nos a intensificar estudos no mesmo, transformando-o num tema de pesquisa. Como decorrência, o sistema foi modelado segundo o paradigma orientado à objeto, facilitando a concepção da nova arquitetura proposta neste trabalho.

O uso da notação ASN.1 na modelagem do Sistema MinhocO, condiz com o estado da arte no que diz respeito ao desenvolvimento de protocolos e de aplicacões em redes de computadores e sistemas distribuídos. A modelagem do Sistema MinhocO oferece uma especificação clara do mesmo para os usuários, além de outras vantagens clássicas dos sistemas orientados a objetos.

Duas linhas de pesquisa estão previstas para um futuro próximo: a primeira refere-se a aspectos cognitivos no tocante à interface aluno-sistema [Cou90]; a segunda está relacionada à utilização de técnicas de Inteligência Artificial, mais precisamente, ao uso de um sistema de "help" ativo e inteligente através de uma base de regras ativada por um motor de inferência [Lim92]. Este sistema de "help" permitirá a identificação do Agente do ambiente (Professor ou Aluno) que necessita de ajuda e dar-lhe-á essa ajuda dentro do contexto que ele, o Agente, se encontra envolvido [Oli93a]..

AGRADECIMENTOS

Gostaríamos de expressar nosso reconhecimento ao relevante serviço prestado por todos aqueles que participaram e/ou colaboraram com o Projeto Minhoca em suas versões precedentes, razão de ser do Sistema MinhocO. Em especial ao **Engenheiro** Luiz Fernando Gomes Soares (PUC-RJ), à **Professora** Marly Rodrigues Gurgel (ETFCe) e ao **aluno** Jose Maria Coelho Filho (UFC), **AGENTES** importantes neste projeto. Finalmente, nosso agradecimento a ETFCe e ao CNPq que financiam este trabalho.

REFERENCIAS

[Alm92] Almeida H, Oliveira M, Claude JP; L'évolution de l'architecture Minhonix. Lab.MASI-Paris VI. Equipe Réseaux Intelligents. Rapport Interne. Paris 1992.

[Coe89] Coelho Filho J.M., Silva Jr J.C.T., Vieira M.H.B. Oliveira A.M.B. MINHOCA PLUS, uma rede local para fins didáticos. VII SBRC. Porto Alegre, 1989.

[Cou90] Coutaz J. Interface Homme-Ordinateur. DUNOP, Paris, 1990.

[IS087] Information processing systems. Specification of ASN.1. ISO8824 doc.1987.

[Juc91] Juca P., Sanchez M.L., Colcher S.; Curso de Redes Locais. PUC-RJ. 1991.

[Kri92] Krief Ph. Langages objets pour le prototypage. MASSON Paris, 1992.

[Lal92] Lalanne S. Description of the INDB. PEMMON Project. Lab.MASI. Paris 1992.

[ISO91] OSI/NM Forum. Modelling Pronciples. Forum TR102. Issue 1.0. 1991.

[Lim92] Lima J.C.M. SINTONIA: Vers un système intelligent pour la construction des diagnostics dans les tuteurs d'informatiques. Tese de Doutorado, Paris VI, 1992.

[Mas91] Mason T., Brown D. Lex & Yacc. O'Reilly 1 Associates, Inc. 1991.

[Mey88] Meyer B. Object-Oriented Software Construction. Prentice-Hall, 1988.

[Oli86] Oliveira M. Gurgel M.R Projeto Info 2000. II SEMINFA-UFMA. São Luiz, 1986.

[Oli87] Oliveira A.M.B. Soares L.F.G. Um sistema de computação interredes para aplicação em um ambiente educacional. Tese de Mestrado, DEE- PUC/RJ. Rio, 1987.

[Oli91] M., Peyrin J.P; MINHONIX, a distributetd system for teaching. IX SBRC. 1991.

[Oli92] Oliveira M, Saleh M. MinhocO User's Guide. Lab. MASI. Paris VI. 1992.

[Oli93a] Oliveira M.; Paget M-M.; Lima C. Towards an Intelligent System in the teaching of Computer Networks. CAL-93. York, UK, 1993.

[Oli93b] Oliveira M; Lalanne Serge, Claude J-P. A Generator of prototypes for a teaching computer system. Submetido ao XX SEMISH. Florianopolis, agosto/1993.

[Pen92] Penna Neto M.C.O. Mandataire Generique.: un concept pour l'integration d'environnement hétérogenes distribués. Tese de Doutorado, Paris VI, 1992.

[Ros90] Rose M. The open Book - Prentice Hall, 1990.

[Ros91] Rose M. The ISO/DE:User's Manual,Vol 4:Application Cookbook.PSI, Inc.1991.

[Soa87] Soares L.F.G. Redes Locais. Editora Campus. 1987.

[Tan89] Tanenbaum A. Systèmes d'Exploitation. InterEditions. Paris. 1989.

[Tan91] Tanenbaum A. Computer Networks. Prentice-Hall, second edition,1991.

[T1M88] T1M1.55/88-014R4 Modelling Guidelines. CCITT document. 1988.

ANEXO I: MinhocO System User's Guide

CHAPTER III: How to use MinhocO System?

3.1 Physical Layer

Experiment 1.1:	What is a computer network?
Experiment 1.2:	Asynchrounous Byte transfer
Experiment 1.3:	Adressing Frame Broadcast and Reception?
Experiment 1.4:	Aplication: A simplex electronic mail

3.2. Data-Link Layer

3.2.a Medium Access Control

Experiment 2.a.1:	What is a Communication Protocol?
Experiment 2.a.2:	CSMA medium-access Protocol?
Experiment 2.a.3:	Token-bus medium-access Protocol?
Experiment 2.a.4:	Aplication: A simplex Medium-acces protocol

3.2.b Logical Link Control

Experiment 2.b.1:	What is a Data-Link Protocol?
Experiment 2.b.2:	Stop-and-wait Protocol?
Experiment 2.b.3.	Sliding Window Protocol
Experiment 2.b.4:	Aplication: A simplex Data-Link protocol

3.3 Network Layer

Experiment 3.1:	What is a logical connexion?
Experiment 3.2:	Circuit-switched x Packet-switched
Experiment 3.3:	Datagrame model xVirtual circuit model
Experiment 3.4:	Aplication: A simplex internetworking (gateway)

3.4 Transport Layer

Experiment 4.1:	What is a end-to-end transport service?
Experiment 4.2:	Fragmentation and Reassembly
Experiment 4.3:	Flow control and buffering
Experiment 4.4 :	Application: A simplex transport protocol

3.5 Session Layer

Experiment 5.1:	What is a session?
Experiment 5.2:	Naming
Experiment 5.3:	Synchronization services
Experiment 5.4 :	Application: A simplex session protocol

3.6 Distributed System

Experiment 6.1:	What does a transparency system means?
Experiment 6.2:	Client -server model
Experiment 6.3:	Remote Procedure Call
Experiment 6.4:	Aplication: A simplex Distributed Data Base

ANEXO III: Atributos da modelagem do sistema

A macro ATTRIBUTE pode ser utilizada para a definição do tipo do atributo. Os atributos sao definidos pelos seguintes elementos:

- Uma referência ao tipo do atributo seguida pela palavra chave ATTRIBUTE;
- A palavra chave WITH ATTRIBUTE SYNTAX seguida por uma definição de tipo (e.g. string);

```
;;;  Attributs

localização ATTRIBUTE
        WITH ATTRIBUTE SYNTAX Printable String
::=(attribute   1)

língua  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER {
                                        anglais(0),
                                        français(1),
                                        portugais  (2)}
::=(attribute  2)

versão ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER {
                supportS1+bufferB1+connexionC1   (1)
                supportS2+bufferB2+connexionC2  (8)  }
::=(attribute  3)

estado ATTRIBUTE
        WITH ATTRIBUTE SYNTAX Status
::=(attribute  4)

estação ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER
::=(attribute  5)

sistema  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX  INTEGER {
                                        MSDOS (0),
                                        MINIX (1)}
::=(attribute  6)

hypo ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER {
                        supportS1+  Buffer1(0),
                        supportS1+  Buffer2(1),
                        supportS2+  Buffer1(2),
                        supportS2+  Buffer2(3)}
::=(attribute  7)

taxa ATTRIBUTE
        WITH ATTRIBUTE SYNTAX  Configuration
::=(attribute  8)ATTRIBUTE

camada ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER{
                        niveauliaison  (0),
                        niveautransport(1)}
::=(attribute  9)
```

```
tipo-conexão  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER {
                        connection-oriented(0),
                        connection-less(1)}
::=(attribute  10)
tipo-aplicação  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER {
                                eletronicmail(0),
                                filetransfer(1),
                                TicTacToe(2)}
::=(attribute  11)

rede-destino  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER
::=(attribute  12)

rede-origem  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER
::=(attribute  13)

pacote  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX SEQUENCE {
                                destination OCTET STRING,
                                longuer OCTET STRING,
                                source OCTET STRING,
                                donnees SEQUENCE OF}
::=(attribute  14)

numero-conexão  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER
::=(attribute  15)

tipo-enlace  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX INTEGER {
                                SuportIdeal(0),
                                CorrectionErreur(1),
                                Controleflux(2)}
::=(attribute  16)

tipo-acesso  ATTRIBUTE
        WITH ATTRIBUTE SYNTAX  INTEGER {
                                CSMA(0),
                                CSMA/CD(1),
                                Token(2),
                                ALOHA(3)}
::=(attribute  17)

;;;   Paramètres
status ::=INTEGER{
                        available  (0),
                        unavailable  (1)}

Configuration ::= INTEGER{
                        110(0),
                        1200(1),
                        9900(2),
                        19600(3)}
```