

**Universidade Estadual de Maringá
Centro de Tecnologia
Departamento de Informática**

**Utilização de Técnicas de IA para Implementação de um
Agente Operador de Sistema de Acesso Remoto**

Leandro Magno Correa da Silva

TG-03-96

Brasil

Universidade Estadual de Maringá
Centro de Tecnologia
Departamento de Informática

**Utilização de Técnicas de IA para Implementação de um
Agente Operador de Sistema de Acesso Remoto**

Leandro Magno Correa da Silva

TG-03-96

Trabalho de Graduação apresentado ao Curso de
Ciência da Computação, do Centro de Tecnologia, da
Universidade Estadual de Maringá.
Orientador: *Prof. Nilson Moutinho dos Santos*

**Maringá - Paraná
1996**

Leandro Magno Correa da Silva

**Utilização de Técnicas de IA para Implementação de um
Agente Operador de Sistema de Acesso Remoto**

Este exemplar corresponde à redação final da monografia aprovada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação da Universidade Estadual de Maringá, pela comissão formada pelos professores:

Orientador: Prof. Nilson Moutinho dos Santos
Departamento de Informática, CTC, DIN

Prof. Antônio Mendes da Silva Filho
Departamento de Informática, CTC, DIN

Prof. Ademir Carniel
Departamento de Informática, CTC, DIN

Maringá, Dezembro de 1996

Agradecimentos

Agradeço primeiramente a Deus que, concedeu-nos o dom de aprender e produzir, inseriu-me num contexto de mundo onde foi possível chegar até o término não só deste trabalho mas também ao final do curso de graduação, e principalmente, ilumina-nos dia-a-dia em tudo o que fazemos.

Agradeço à minha noiva Rejane que, além de dar um sentido a minha vida, mesmo tantas vezes privada de minhas atenções, incentivou-me durante toda esta fase de desenvolvimento, acreditando sempre em minhas capacidades.

Agradeço aos meus pais pela confiança, dedicação e carinho com que criaram seus filhos, não medindo esforços para que houvesse toda a infra-estrutura necessária que possibilitasse o desenvolvimento deste trabalho.

Agradeço a ótima atuação do professor orientador Nilson Moutinho dos Santos. Além de dar atenção e apoio, sua presença foi fundamental na realização deste trabalho. Mais do que um colega de trabalho, um ser humano formidável.

Agradeço aos *sysops* do Instant BBS de Maringá, e em especial à pessoa do Sr. Marco Antônio Santos de Moraes Leme, que gentilmente possibilitou todo o estudo do ambiente de aplicação, dando toda a força e o apoio necessário. Uma excelente pessoa e profissional extremamente dedicado.

Agradeço carinhosamente a todos os meus colegas de curso e em especial aos meus amigos Marcelo Piccoli Crivelli, Gill Sandro Santos, Hermes Francisco Vecchi, Maurício Teixeira Pinheiro, Edicezar Leandro Nanni, Wellington Siqueira Marinho, Franklin Cesar Flores, Flávia Lumi Matuzawa e Carla Renata Bissaro pelas suas preciosas companhias tanto nos bons momentos quanto nos mais difíceis. Amigos estes que, não só possibilitaram uma boa produção em muitos trabalhos, mas também tornaram muito mais alegre e amena esta etapa de graduação.

Agradeço por fim, ao meu irmão Cassiano que ajudou a revisar esta monografia.

Resumo

Com o surgimento e a proliferação dos sistemas de acesso remoto (como BBS's e Provedores de Informação, *sites* Internet comerciais, etc.) veio à tona toda uma cultura de comunicação e, inserido neste contexto social, o *chat*.

O chat é uma forma de conversa entre duas (ou mais) pessoas através de computadores e meios de acesso remoto — como linhas de rede (locais ou globais) ou vias telefônicas. O que cada pessoa digita em seu computador é ecoado para os outros participantes em tempo real. Dessa forma, é possível uma conversa ao vivo (mas não pessoalmente), diferindo na velocidade entre fala e digitação e no fato das pessoas não se vêem, apenas lêem o que as outras participantes digitam. O chat pode ser realizado desde entre pessoas presentes fisicamente numa mesma sala até entre pessoas ao redor do mundo.

Para manter-se grandes estruturas funcionais, surgiram os operadores (ou *System Operators*). Com tamanha responsabilidade e carga de serviços, dificilmente um operador possui tempo para gastar num *chat* com um usuário do sistema.

De outro lado, a Inteligência Artificial (IA) nos mostra duas tecnologias aplicáveis neste problema em sistemas de acesso remoto: Sistemas Especialistas e Agentes Inteligentes.

Neste sentido este trabalho traz uma proposta de utilização das referidas técnicas de IA para examinar a possibilidade de criar uma interface que possa manter uma conversa razoável com o usuário remoto.

Abstract

Besides the remote access systems growth and proliferation (like BBS', Information Providers and comercial internet sites) appears a whole comunication culture and, inserted on this social context, you find the chat.

The chat is a kind of talking between two (or more) person through computers and remote access ways — like (local or global) networks and telephone lines. What each person types in his/her computer is repeated to others participators in real time. This way, it's possible to talk in live, differing on talk and type velocities and in the fact that there is no face-to-face talking. The persons only read what the other ones type. The chat can be realized since between two persons on a same room till between persons around the world.

The operators with this responsability and charge of service, a operator rarely does have time to spend on chat with a user.

By other side, the Artificial Inteligence (AI) field provides two other kinds of technologies that can be applied to remote access systems: Expert Systems and Inteligent Agents.

So these AI technologies will be used as a tool to developpe an interface that supports a reasonable talking to the remote user.

ÍNDICE

1. INTRODUÇÃO

2. O AMBIENTE DE APLICAÇÃO

2.1 DEFINIÇÃO DE BBS

2.2 FUNCIONAMENTO INTERNO DE UM BBS

2.2.1 PLATAFORMA

2.2.2 SOFTWARE

2.3 RELAÇÃO BBS'S X INTERNET

2.4 TIPOS DE BBS'S

2.4.1 BBS CASEIRO

2.4.2 BBS COMERCIAL

2.4.3 BBS DE EMPRESAS

2.4.4 SISTEMAS DE INFORMAÇÃO GLOBAL (SIG)

2.5 RECURSOS

2.6 ATIVIDADES REALIZADAS POR UM OPERADOR DE BBS

2.6.1 TAREFAS ESPORÁDICAS

2.6.2 TAREFAS ROTINEIRAS

2.7 SERVIÇOS COMUMENTE PRESTADOS POR BBS'S À COMUNIDADE

3. IDOCTOR: A INTERFACE EXISTENTE

3.1 O QUE É O IDOCTOR

3.2 TÉCNICA DE IA E AS CARACTERÍSTICAS DO IDOCTOR

4. SISTEMAS ESPECIALISTAS

4.1 O QUE SÃO SISTEMAS ESPECIALISTAS

4.2 ARQUITETURA GENÉRICA DE UM SISTEMA ESPECIALISTA

5. AGENTES INTELIGENTES

5.1 O QUE SÃO AGENTES INTELIGENTES ?

5.2 CARACTERÍSTICAS QUE TORNAM UMA APLICAÇÃO APROPRIADA PARA AGENTES

6. IDENTIFICANDO A APLICAÇÃO

7. DOCUMENTANDO REQUISITOS PARA A APLICAÇÃO

7.1 EMBASAMENTO

7.2 ESPECIFICAÇÃO DE REQUISITOS

7.2.1 INTRODUÇÃO

- 7.2.2 DESCRIÇÃO DE INFORMAÇÕES
- 7.2.3 DESCRIÇÃO FUNCIONAL
- 7.2.4 DESCRIÇÃO COMPORTAMENTAL
- 7.2.5 CRITÉRIOS DE VALIDAÇÃO

8. O PROTÓTIPO FUNCIONAL EM SEU AMBIENTE

9. CONCLUSÕES

- 9.1 SOBRE OS RUMOS TRAÇADOS
- 9.2 FUTURAS IMPLEMENTAÇÕES
- 9.3 CONSIDERAÇÕES FINAIS

REFERÊNCIAS BIBLIOGRÁFICAS

BIBLIOGRAFIA

APÊNDICE A **A.**

APÊNDICE B **B.**

APÊNDICE C **C.**

C.1 CLASSIFICAÇÃO DE SISTEMAS ESPECIALISTAS **C.**

APÊNDICE D **D.**

D.1 TERMINOLOGIA PARA AGENTES INTELIGENTES **D.**

D.2 TIPOS DE AGENTES **D.**

APÊNDICE E **E.**

APÊNDICE F **F.**

APÊNDICE G **G.**

Lista de Ilustrações

Figura 4.1 - Elementos básicos de um Sistema Especialista.....	
Figura 7.1 - Diagrama de Fluxo de Dados de Nível 0.....	
Figura 7.2 - Diagrama de Fluxo de Dados de Nível 1.....	
Figura 7.3 - Diagrama de Fluxo de Controle de Nível 1.....	
Figura 7.4 - Quadro das Funções dos Processos.....	
Figura 7.5 - Diagrama de Transição de Estados.....	
Figura 7.6 - Quadro de Eventos e Ações.....	
Figura 7.7 - Fluxograma de Blocos do Mecanismo de Inferência.....	
Figura 8.1 - O XPERTSYS em seu ambiente.....	
Figura A.1.1 - Uma rede de troca de mensagens entre BBS's.....	A.
Figura A.1.2 - Formação de endereços de rede de BBS.....	A.
Figura E.1 - A interface do XPERTSYS.....	E.
Figura F.1 - Quadro de testes com a Base de Conhecimentos em <i>Download</i>	F.
Figura F.2 - Quadro de testes com a Base de Conhecimentos em Mensagens.....	F.

Capítulo 1

Introdução

A cada dia o número de Sistemas de Acesso Remoto (SAR) vem aumentando aceleradamente, seja em BBS's (Bulletin Board System), seja em sites da Internet, seja em Sistemas de Acesso a Serviços Específicos (como bancos, centros meteorológicos, etc.).

Cada sistema desses conta com pelo menos um operador, esteja ele disponível por 24 horas ao dia ou não. É difícil encontrar algum desses operadores que já não precisou se dar à angustiante tarefa de instruir um usuário remoto que insiste em pedir ajuda sem ter lido toda a documentação à qual tem acesso. Muitas vezes, o usuário apenas deseja “bater um papo”, enquanto que as tarefas do lado interno do sistema se acumulam para desespero do operador.

Para auxiliar o operador foi desenvolvido um “Agente” [Fag94] que realiza tarefas como: entreter o usuários até que possam ser atendido pelo operador, resolver o problema (responder às “questões”) do usuário se for algo relativamente simples e freqüentemente requisitada.

Isso está de acordo com as mais recentes pesquisas sobre a utilização de Agente de *software* para apoiar o homem nos mais variados campos do conhecimento.

No capítulo seguinte trouxemos uma imagem do ambiente de aplicação, com suas definições, funcionamento, classificação, recursos, serviços prestados e problemas enfrentados pelo operador. Neste capítulo trazemos um embasamento importante para o entendimento das as necessidades do mundo real e das possibilidades de implementação.

No capítulo 3 descrevemos uma interface de comunicação já existente para este ambiente como complemento à motivação do desenvolvimento.

Nos capítulos 4 e 5 trouxemos dois embasamentos em tecnologias de IA: Sistemas Especialistas e Agentes Inteligentes respectivamente, proporcionando uma base teórica à implementação que segue.

No capítulo 6 identificamos o que realmente precisa ser implementado frente o que já dispomos no ambiente de aplicação a nível de características.

No capítulo 7 tentamos dar aos requisitos de implementação uma aparência semi-formal. Desta maneira esperamos obter uma melhor documentação da aplicação.

No capítulo 8 descrevemos como foi o processo de implementação, os erros e acertos de programação. Descrevemos também o protótipo tal qual ele fica(rá) em seu ambiente.

Finalmente no capítulo 9 trouxemos uma visão geral do que foi o trabalhos em seus três momentos de implementação: passado, presente e futuro.

Alguns aprofundamentos e curiosidades que tangem à “espinha dorsal” do trabalho encontram-se nos apêndices.

Capítulo 2

O Ambiente de Aplicação

2.1 Definição de BBS

BBS (Bulletin Board System - Sistema de Mural de Boletins) - segundo [Ins94] trata-se de um sistema computadorizado dedicado a atender (via telefone) um ou mais usuários colocando à disposição destes uma série de serviços tais como: “mail” (troca de mensagens locais e remotas - redes) e troca de programas e dados de toda e qualquer espécie. Uma outra definição [Lem96] considera um BBS uma central de transferência, armazenamento e disponibilização de informações computadorizadas por meio de telefone.

Para poder acessar um BBS, precisa-se de no mínimo: uma linha telefônica, um microcomputador XT, um Modem (modulador/demodulador), e um programa de comunicação. Não é necessário “grande” aparato computacional como: monitor de vídeo VGA, 4 Mb de memória, mouse, placa de som, etc.

O que é mais específico dentre o mínimo necessário, é o modem (custa em torno de R\$ 70,00), portanto não é um investimento dispendioso, além disso um modem permite também a comunicação entre seu computador e outros computadores, como por exemplo, provedores de acesso à Internet, BBS's, terminais bancários, o computador de outros usuários que possuam modem, etc.

Só no Brasil, atualmente existem cadastradas (segundo a NACIONAL.LST encontrada no Instant BBS [Ins96] - lista nacional de BBS's de março deste ano) em torno de 644 BBS's em 106 cidades. Mais da metade opera 24 horas por dia e a grande maioria não usa mais do que uma linha telefônica, no entanto, eles possuem no mínimo 300 usuários cada um (em média de 800). No mundo, só a lista de BBS's participantes de uma das mais conhecidas redes de mensagens mundiais, relaciona um número superior a 35 mil BBS's. Há informações não oficiais (de operadores de outros BBS's nacionais) que dizem existir acima de 100 mil em todo o mundo.

O jornal de circulação nacional "O Estado de São Paulo" publica uma lista de BBS nacionais (não contém todas) nas edições de segunda-feira em seu Caderno de Informática, que costuma ser a informação inicial para novos usuários de BBS's.

Em São Paulo, um exemplo para todo o país: a Mandic BBS, possui mais de 20.000 usuários cadastrados, um número superior a 300 linhas telefônicas, o que demanda uma quantia considerável de equipamento e pessoal, sendo que uma grande parte destas linhas é do tipo 0800 (gratuita) para usuários pagantes.

Através do serviço de mail, os usuários de qualquer cidade onde haja um BBS ligado a uma rede de mensagens, podem se comunicar com qualquer usuário de outro BBS, na mesma ou em outra cidade com um atraso de no máximo 24 horas. Os pacotes de mensagens e/ou programas são distribuídos para os BBS's conectados numa hora específica, diária e automaticamente, e permitem acesso a várias redes regionais, nacionais e internacionais (como a Internet). Informações adicionais sobre redes de mensagens em BBS's podem ser encontradas no Apêndice A.

Os BBS's são uma boa opção para trocar programas, manter-se informado sobre diversos assuntos, conversar com pessoas de todo o mundo, usufruir de suporte técnico de algumas empresas e universidades, ou, no mínimo, ter sempre a última versão do seu programa predileto.

2.2 Funcionamento Interno de um BBS

2.2.1 Plataforma

Existem basicamente duas formas de se compor a plataforma de um BBS:

A primeira, requer apenas um microcomputador (PC) que gerencia um ou mais modem's. Isto é possível através de um sistema operacional que possibilite um funcionamento multi-

tarefa seguro, como UNIX ou OS/2. O número de *modem's* internos é limitado ao número de 4. O número de *modem's* externos é limitado apenas pelo número de portas disponibilizadas pelas interfaces multi-seriais. Hoje existem placas expansoras multi-seriais de até 32 portas de comunicação. Devido a esta exigência de multi-tarefa, uma máquina com configuração mínima recomendada é um 486 DX 66, com 8 Mb de memória. É claro que à medida que cresce o número de *modem's* ligados a uma mesma máquina, cresce a necessidade de um equipamento mais robusto (com maiores velocidade e memória), podendo chegar até num Pentium 166, com 64 Mb de memória.

A segunda não requer máquinas de porte tão grande como a primeira, no entanto exige uma rede ligando dois ou mais microcomputadores. Um servidor de arquivos como um 486 DX 2 66 se enquadra bem, sendo que e as estações podem ser 286's ou 386's sx. Como não é necessário além de 2 Mb de memória nas estações, qualquer microcomputador acima destes dois últimos modelos já é considerado dispendioso. Mesmo porque o que custa mais nestas estações são os 2 Mb de memória e um modelo 386 DX 40 ou superior já exige 4 Mb de memória RAM.

Esta segunda possibilidade geralmente é a adotada quando se acredita que o sistema virá a crescer muito. Ela permite que o número de *modems* seja limitado apenas pela velocidade da rede e do servidor de arquivos. Cada estação é composta de: gabinete, placa-mãe (com memória), placa de rede e placa *modem* (ou *modem* externo) tendo um custo mínimo de 250 dólares.

2.2.2 Software

Existe disponível no mercado um número razoável de produtos para o gerenciamento do BBS. Entende-se por gerenciamento funções como: controle de *logins*, controle de tempo diário de utilização das contas, controle de vencimento de contas, controle de ponteiros de mensagens (última mensagem lida de cada área), provimento de menus e interface, gerenciamento de arquivo (armazenamento, envio e recebimento, etc.).

Cada desenvolvedor de software de gerenciamento define rotinas até um certo ponto, depois disponibiliza a outros programadores os principais esquemas dos bancos de dados do gerenciador para que sejam produzidos módulos auxiliares (*doors*) para tarefas mais específicas. Para mais detalhes, veja a definição de *door*, no Apêndice B.

Alguns produtores de *software* gerenciador de BBS incluem rotinas para quase todas as funções, dando pouca margem aos *doors*, como acontece no software PCBoard. Outros *softwares* são, de certa forma, menos completos, exigindo uma quantidade maior de *doors*, como o Remote Access [Wan94]. Porém as vantagens de se ter um software bastante completo na hora da montagem do BBS (necessitando-se a instalação de poucos módulos), pode se tornar um problema quando se deseja um sistema um pouco mais personalizado e obviamente (muito) mais maleável do que aquele que um sistema completo proporciona.

2.3 Relação BBS's x Internet

De forma grosseira [Lem96] pode-se diferenciar um BBS da Internet da seguinte forma: enquanto a Internet é o meio para se acessar informações espalhadas pelo mundo, o BBS é um resumo do mundo pré-selecionado de informações prontas para serem acessadas.

Para poder-se comparar melhor a Internet com os BBS's, deve-se lembrar sempre que a Internet é representada localmente por um Provedor de Acesso e é neste ponto que os dois mais se assemelham, pois a grande maioria dos BBS's tem escopo local.

Comercialmente falando, é corretíssimo afirmar que os BBS's tendem a uma especialização oferecendo serviços cada vez mais direcionados às necessidades da comunidade local. Alguns exemplos seriam: anúncios comerciais de empresas locais, áreas de transferência de dados para empresas, informações urbanas de interesse geral, incluindo chamadas de licitações, etc. Quanto aos provedores, eles devem diversificar-se oferecendo mais do que simples acesso à rede. Apesar de haver espaço comercial para todos, o crescimento do número de Serviços de Acesso Remoto (SAR) descreve uma curva exponencial no decorrer do tempo. Assim, para que todos se mantenham é necessário que escolham as fatias definidas do mercado, especializando-se.

Sabendo-se disto, é inevitável o desaparecimento de Provedores de Acesso Internet que pensam em faturar somente vendendo o acesso. Os BBS's também se enquadram nesta regra de sobrevivência, no entanto os custos de manutenção de BBS's são muitas vezes menor que os dos Provedores, podendo conferir aos BBS's um tempo precioso em seus processos de especialização.

Após alguns anos teremos muitos sistemas, todos especializados. Seus sustentos sairão da venda do acesso para garantir sobrevivência e de serviços exclusivos para obter lucro e atingir o sucesso. No entanto, todos tendem a se ligar à Internet, seja *on-line* 24 horas (como os Provedores e *Hosts*) ou *off-line* (como os BBS's e usuários).

Pensando assim, os BBS's, desde já, procuram ser clientes de Provedores para executar a tarefa que os Provedores acreditam não ser deles: agregar valor ao nome 'Internet'.

Ainda segundo Marco Leme [Lem96], isto só vem a confirmar o que todos já desconfiavam: não existe um negócio (como Provedor de Acesso Internet) que seja tão lucrativo ao ponto de retornar o capital de investimento e obter lucro sozinho, sem muito investimento e trabalho árduo. Todos os serviços se resumem em muito trabalho e dedicação.

2.4 Tipos de BBS's

2.4.1 BBS Caseiro

Caracterizado por possuir apenas uma linha telefônica e estar situado fisicamente na casa do operador. Este operador geralmente possui uma idade que varia entre 11 e 18 anos e não se preocupa em obter lucro comercialmente. O único objetivo deste sistema é fazer com que o operador possua laços fortes de amizade através de outros operadores e usuários. De certa forma, não passa de uma brincadeira de adolescente(s). Eventualmente existem outros operadores que se limitam a ajudar o operador principal a cuidar do sistema e mantê-lo

atualizado. Sua principal característica está no fato de ser o resultado produzido pelo operador enquanto ele pratica seu *hobby*.

2.4.2 BBS Comercial

Geralmente de um porte um pouco maior, situado fisicamente em um local de comércio, podendo ter de 2 ou 3 linhas ou até uma centena delas. Quase todos são constituídos de empresas e todos se caracterizam, principalmente, por visarem ao lucro. Esse lucro é obtido de atividades que vão desde a cobrança do acesso de usuários particulares até a montagem de BBS's de empresas.

2.4.3 BBS de Empresas

Usado como ferramenta de comunicação da empresa com seus clientes, seus fornecedores e suas filiais, portanto, trata-se de uma atividade meio, e não fim, como nos outros dois tipos citados acima. Costumam ser fechados ao público, ou seja, as contas de acesso precisam ser criadas pelo operador antes que a primeira conexão possa ser realizada.

2.4.4 Sistemas de Informação Global (SIG)

São constituídas pelos BBS de grande porte, com mais de uma centena de linhas telefônicas para acesso. A princípio, parece tratar-se de um simples BBS Comercial de Grande porte. No entanto, seus serviços atingem desde o “*micreiro*” (que também acessa o BBS Caseiro), até trabalhadores das mais diversas atividades (como o caminhoneiro, o agricultor, etc) chegando mesmo aos grandes comerciantes, representantes de vendas, investidores e outros usuários de grande porte. Embora não aconteça necessariamente, até hoje todos os SIGs também são provedores de acesso Internet a usuários finais e constituem sites de grande representatividade dentro da rede. Os sistemas deste porte são tão raros no mundo que podemos relacionar os que existem nas Américas: Compuserve, America On-Line, Dialdata e Mandic.

2.5 Recursos

Praticamente tudo o que se pode imaginar em termos de comunicação e informática pode ser feito dentro de um BBS. Isto é possível graças a quantidade de software modular que existe para cada gerenciador diferente. Para quase todos os tipos de gerenciadores estão disponíveis dados técnicos (como, por exemplo, o seu esquema de banco de dados) que possibilitam a qualquer programador a criação de *doors* (software modular) para executar uma tarefa para a qual até hoje ninguém havia se dado ao trabalho, estendendo assim as funções do gerenciador.

Dentre os recursos mais interessantes usados encontram-se os empacotadores de mensagens. Num sistema do porte de um BBS Comercial circula diariamente entre 1000 e 5000 mensagens. Mas isto não quer dizer que os usuários lêem todas as mensagens enquanto estão conectados ao BBS (isto teria um custo altíssimo). Para evitar esta explosão na conta telefônica dos usuários, existem *doors* do tipo empacotadores de mensagens.

A função desses módulos é criar um arquivo empacotado com todas as mensagens que determinado usuário ainda não leu (de acordo com seus ponteiros de áreas de mensagens) e enviar ao usuário. Depois de desconectado, o usuário, calmamente em um horário que mais lhe convém, usa um outro tipo de programa (leitor off-line) para poder ler e responder a todas as mensagens (tanto particulares como as públicas - fóruns). Posteriormente na próxima conexão o usuário envia um pequeno pacote com as suas respostas às mensagens. Então, o door empacotador de mensagens também desempacota as respostas do usuário e as coloca em suas devidas áreas e fóruns existentes no BBS.

Outro recurso bastante usado em BBS's Comerciais são os *menus particulares de empresas*. Consistem-se em menus de serviços criados especificamente para uma determinada empresa contratante. Geralmente compõe-se de opções diferenciadas, com termos técnicos característicos da atividade da empresa. Eventualmente os operadores também desenvolvem *doors* que só podem ser acessados a partir destes menus para proporcionar um serviço único à empresa.

2.6 Atividades Realizadas por um operador de BBS

O conjunto de atividades realizada por um operador de BBS pode ser dividido em duas categorias: tarefas esporádicas e tarefas de rotina.

2.6.1 Tarefas esporádicas

Quando se trata de um sistema relativamente grande, com mais de um *node* e com mais de um acionador de CD-ROM, uma vez ou outra podem ocorrer os seguintes tipos de problema:

Ocorrências na rede

Verificação de cabos, teste de placas a nível de hardware, sobrecarga de atividades para o servidor como: serviço de impressão e serviço de disco simultâneos, acesso simultâneo a 2 unidades de CD-ROM por usuários diferentes, etc.

Problemas causados por doors

Eventualmente um usuário chama um *door* que, além de acionar o *swap out* do gerenciador do BBS, pode usar a memória de forma desordenada, causar acesso indevido ou malsucedido a dispositivos, *runtime errors*, etc.

Alguns dos problemas acima podem ocasionar desde um simples *halt* em algumas tarefas dos terminais de rede, deixando o BBS com um ou mais *nodes* a menos temporariamente, até provocar um estado de pane geral em que todo o BBS fica inútil até que o operador venha a intervir no sistema. Posteriormente, o operador poderá vir a tentar descobrir a causa de tal erro no sistema, que pode ser um detalhe simples de configuração, ou uma atividade que lhe tome horas, e nem sempre é bem sucedida.

Problemas com unidades de disco rígido (HD)

Funcionando 24 horas por dia, ininterruptamente, um HD tem duração, de um a dois anos (com a tecnologia atual). Apesar dos constantes *backups* em fita *streamer* e em outras unidades de disco, inevitavelmente o operador, num determinado dia, será surpreendido com a “ausência” da unidade em questão. Este evento irá ocasionar inúmeros atropelos para recuperação dos arquivos que não foram perdidos, para a volta do backup, para o transporte dos dados para o HD novo e verificações extras para certificar-se que está tudo novamente em ordem. Caso não exista um HD de reserva, é necessário ainda comprar um novo HD, o que gera um inevitável atraso atribuído ao setor de compras da empresa.

Inclusão de novas seções

Doors: Esta tarefa ocupa de uma hora até 2 ou 3 dias de serviço do operador, dependendo da qualidade do software (módulo) que está sendo implementado, principalmente no tocante a documentação.

Anúncios de patrocinadores: em geral consiste na criação de uma tela de 80x23 caracteres arranjando-se o logotipo do patrocinador com quaisquer outros dizeres que ele desejar. Esta tarefa não é muito difícil tecnicamente, no entanto requer uma boa dose de criatividade. Esta atividade toma do operador, cerca de 15 minutos a 3 horas (caso o logotipo seja um tanto quanto complexo).

Menus de serviços: esta tarefa consiste da análise de requisitos, análise da viabilidade do serviço requisitado pelo cliente (geralmente outra empresa), criação do lay-out do menu (geralmente o mesmo padrão dos demais menus do BBS), implementação dos recursos verificados na análise de requisitos e posteriores ajustes a serem feitos de acordo com novas necessidades do cliente. Uma atividade como esta ocupa de 30 minutos a 2 horas (se contínuas), dependendo principalmente do bom entendimento entre o cliente e o operador.

2.6.2 Tarefas rotineiras

Verificação de cadastro

A maioria dos BBS's costumam ligar por voz para cada novo usuário que se cadastra no sistema, para confirmar seus dados (endereço e telefone), fazer uma espécie de *telemarketing* e obter *feed-back* da atuação do BBS. Isto vem a ser necessário uma vez que os BBS's são sistemas abertos (em que cada usuário se cadastra pessoalmente) e estão sujeitos a informações falsas com intuítos não-saudáveis.

Inclusão de novos arquivos

Constantemente os usuários enviam ao BBS (*upload*) arquivos que julgam ser de interesse comuns a outros usuários. Apesar da tarefa de checagem contra vírus e “cavalos-de-tróia” ser feita automaticamente pelo sistema, resta saber de que tipo de arquivo se trata: se for um código executável, deve-se verificar se funciona. Se funcionar, verificar se não é repetido, ultrapassado ou inútil (para manter a qualidade de arquivos disponíveis para *download*). Isto feito, resta completar o processo de classificação de arquivos, movendo-os para as áreas que possuam arquivos similares.

Atualização de Boletins

Em alguns casos, esta tarefa é terceirizada para um outro tipo de “operador” - o Coordenador. Ele é encarregado de produzir, com certa frequência (geralmente semanal), textos dos mais variados gêneros (como: explicativo, informativo, de entretenimento, etc) que serão inclusos em seções preestabelecidas do sistema, para que sejam lidos on-line pelos usuários, e eventualmente passíveis de download para leitura off-line. Já em outros casos, o próprio operador encarrega-se desta tarefa, o que lhe custa tanto tempo quanto redigir uma edição de um desses tipos de jornal informativo (como o Informativo da UEM), buscando informações em diversos locais, produzindo matérias, criando textos explicativos e colecionando textos de cunho humorístico.

Atendimento ao Usuário via *Chat*

Existe um recurso possível dentro de praticamente todos os BBS's que é a conversa *on-line* com o *sysop*. Geralmente usa-se um *door* que além de promover o espelhamento de igual teor tanto na tela do *sysop* como na do usuário, também divide a tela em duas partes: uma contém o que é “dito” (na verdade **escrito**) pelo *sysop* e outra o que é “dito” pelo usuário.

Um usuário costuma chamar o *sysop* por vários motivos, desde um simples “bate-papo” com o usuário até para pedir orientação sobre recursos do próprio BBS ou sobre algum software conhecido do mercado.

Esta atividade pode ser rapidamente executada — como por exemplo fornecendo uma “dica” rápida — em menos de cinco minutos, ou pode levar horas se o usuário quer apenas trocar idéias com o *sysop* sobre assuntos diversos, ou seja, “bater um papo”.

Aqui se enquadra a aplicação que está sendo proposta: um *door* que seja capaz de tirar as dúvidas mais comuns entre os usuários, substituindo assim o *sysop* para que este não seja afastado de qualquer outra atividade a que esteja dedicado (como as descritas anteriormente). Ver-se-á que esta proposta também serve para outras interessantes finalidades.

2.7 Serviços Comumente Prestados por BBS's à Comunidade

Além dos serviços convencionais de um BBS, como provimento de e-mail local/Internet, distribuição de arquivos/programas e disponibilização de informações gerais (boletins), alguns BBS's prestam também alguns serviços extras que, não necessariamente, estão ligados às suas atividades principais. Serviços como implantação de sistemas de comunicação matriz-filiais para empresas, intercâmbio de arquivos empresa/clientes, disponibilização de áreas/seções exclusivas para empresas/profissionais autônomos,

representação de *software* ou empresas, anúncios classificados, anúncios comerciais e criação de HTML's (*Home Pages*) para a Internet.

A seguir, uma breve descrição de uma interface que atende alguns dos requisitos de um BBS — precisamos saber o que já existe neste sentido.

Capítulo 3

IDoctor: A Interface Existente

3.1 O que é o IDoctor

IDoctor é um *door* desenvolvido [Fag94] para cumprir duas funções dentro de um BBS. A primeira e mais simples é a de servir como um *chat door* (módulo/aplicação para conversação) entre o *sysop* e o usuário. A segunda é para estabelecer com um usuário uma conversa simples baseada no casamento de padrões, assim como o ELIZA [Wei66] desenvolvido na década de 60.

Esta segunda função é fruto da mesma idéia de dispensar o *sysop* da cansativa, repetitiva e muitas vezes inoportuna tarefa de atender os *chats* dos usuários. Para isto, o IDoctor simula ser o “*co-sysop*”, uma espécie de co-operador do sistema também possível no ambiente do BBS.

Para o perfeito funcionamento dentro do ambiente de um BBS, este *door* possui todas as características necessárias, tais como: tratamento de caracteres em baixo nível diretamente na porta serial, checagem do tempo máximo permitido ao usuário, constante verificação da presença de sinal *carrier* na porta de comunicação serial, total compatibilidade com o sistema de passagem de parâmetros de vários gerenciadores de BBS, etc.

3.2 Técnica de IA e as características do IDoctor

A única técnica de IA presente na implementação do IDoctor é o casamento de padrões.

A grande vantagem desta técnica é que apesar do usuário poder interagir em linguagem natural, o sistema não precisa interpretar e realmente “entender” sobre o que está se falando. Simplesmente o sistema procura uma ou mais palavras chaves (*keyword*) na frase que o usuário digitou e responde com outras frases também pré-definidas.

Para que as respostas não sejam tão repetitivas, pode-se programar mais de uma possível resposta para cada *keyword*. Assim a resposta será escolhida aleatoriamente num conjunto delas.

Outra característica é o uso de sinônimos. Desta forma mais de uma *keyword* pode selecionar um mesmo conjunto de respostas.

O IDoctor conta com todos estes recursos que enriquecem o uso desta técnica. Seus limites são: até 200 *keywords* diferentes, até 10 possíveis respostas para cada *keyword* e até 9 sinônimos para cada *keyword*.

A Base de Conhecimentos contida no IDoctor é alterada manualmente pelo *sysop*, com um editor ASCII comum, e pré-compilada para permitir acesso rápido. Como o IDoctor só abre a Base de Conhecimentos pré-compilada em modo *read-only*, várias cópias (cada uma em uma tarefa de máquina) podem ser executadas simultaneamente, possibilitando seu uso em redes.

No próximo capítulo trazemos um breve embasamento em IA que compartilha algumas poucas características do IDoctor: Sistemas Especialistas

Capítulo 4

Sistemas Especialistas

4.1 O que são sistemas especialistas

Sistemas Especialistas (SE) são aqueles que atuam em uma área restrita e bem definida do conhecimento humano. A exemplo de especialistas humanos um SE possui uma área de atuação e conhecimentos específicos dentro desta área. Um SE é capaz de emitir decisões apoiadas em conhecimento justificado [Cun87]. Para tanto, possui uma base de informações, comumente chamada de Base de Conhecimento (BC), que contém regras que representam conhecimento.

Um especialista, para tomar uma decisão sobre um determinado assunto, usa alguns insumos básicos: fatos encontrados sobre o assunto, hipóteses formuladas por ele mesmo, conhecimento prévio armazenado em sua memória, experiências acumuladas anteriormente. Durante esta busca de uma decisão ou solução, o especialista vai comparando os fatos com as informações já conhecidas, criando novas hipóteses e concluindo novos fatos. Este processo influencia diretamente o processo de raciocínio. Por contar com insumos nem sempre disponíveis, corretos, suficientes e utilizáveis, o especialista pode não chegar a conclusão alguma, ou finalmente concluir algo errado. No entanto estas limitações do especialista se justificam com os fatos que encontrou, com suas experiências anteriores, etc.

Além de emitir conclusões, um especialista deve ser capaz de aprender novos conhecimentos para desenvolver seu conhecimento como um todo e melhorar a qualidade de suas decisões.

Existem basicamente três formas em que o especialista pode fornecer um resposta: delimitando um universo de respostas, fornecendo uma ou poucas respostas possíveis e/ou apenas orientando o usuário humano com a finalidade de que ele tome uma decisão mais fundamentada e consciente acerca do assunto.

Sistemas Especialistas são recomendados em aplicações em que o problema não pode ser algoritmizado e/ou o algoritmo demandaria recursos de hardware ainda inexistentes no momento para se ter uma solução confiável e ainda dentro do tempo útil.

No Apêndice C pode-se encontrar a classificação de Sistemas Especialistas, de acordo com [Cun87].

4.2 Arquitetura genérica de um sistema especialista

Um sistema especialista é formado de três partes: mecanismo de inferência, base de conhecimento e quadro-negro.

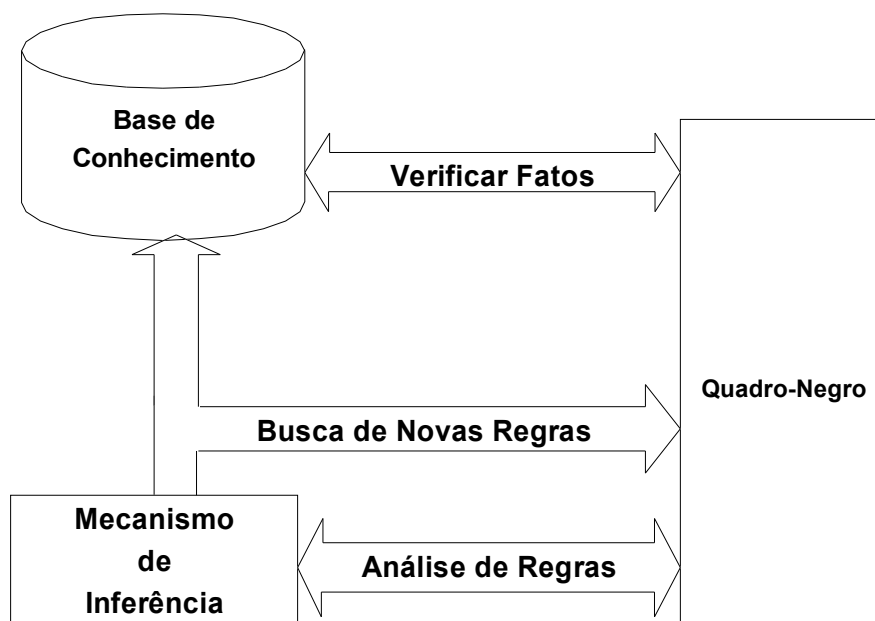


Figura 4.1 - Elementos básicos de um Sistema Especialista

Mecanismo de inferência é o elemento capaz de verificar os fatos, buscar regras e ordenar todos de uma forma lógica, de maneira que, baseando-se em tais fatos e regras, possa chegar a outros fatos conclusivos.

De forma geral, é um mecanismo que compara pedaços de “*strings*” com padrões. Se tal comparação for positiva, então hipóteses podem ser tomadas como fatos ou ainda pode ser feita a busca de novas regras, percorrendo outros caminhos para se chegar a um determinado objetivo. O processo mais usado para se inferir é o da avaliação de regras.

A base de conhecimento é o componente responsável sobre as características de funcionamento do sistema. Nela estão contidos fatos e regras que representam de forma geral o conhecimento de um especialista humano. O conhecimento pode ser passado para a base de conhecimentos de diversas formas, desde um editor convencional, passando a editores especializados, até chegar a interfaces altamente futurísticas.

O quadro-negro, consiste de estruturas de dados específicas em memória para o constante processo de inferência escrever e apagar quando necessário, ou literalmente “rascunhar”. Analogamente à mente humana, esta área de memória corresponderia a memória de curta duração de uma pessoa, em que uma informação só permanece lá enquanto for necessária, geralmente frações de segundo.

No próximo capítulo veremos uma outra tecnologia emergente de IA: Agentes Inteligentes. Esta tecnologia tem sido muito estudada, já existindo inclusive algumas aplicações implementadas, como o SSAA de Moniz [Mon94] e outras destinadas ao uso em *sites* Internet (também **sistemas de acesso remoto**), como as presentes no <http://www.ftp.com> via web.

Capítulo 5

Agentes Inteligentes

5.1 O que são agentes inteligentes ?

Ainda não foi possível encontrar uma definição formal do conceito de Agente que possa abranger todo o universo dos mesmos. O que se pode fazer [DeC95] é identificar as características esperadas em um agente.

Analogamente ao mundo real, agentes trazem a noção de conhecimentos especializados, como por exemplo agentes financeiro, de viagem, de empregos, etc. Extendendo esta idéia ao mundo do *software*, cada Agente vem a ser uma entidade ativa, constantemente ao lado do usuário, e que possui conhecimento específico sobre um certo domínio.

Quando a tecnologia de agentes for suficientemente desenvolvida para tal feitos, os agentes poderão executar atividades de auxílio ao usuário, de forma transparente, ou seja, sem a necessidade do usuário saber como eles trabalham. Um Agente então operará de modo que esconda a complexidade e os detalhes de tarefas, formulando objetivos, iniciando ações, trabalhando independentemente, reconhecendo situações de autoativação, usando base de conhecimento e mecanismo de raciocínio.

Em tal estado da arte, os agentes ensinarão os usuários a utilizar os recursos disponíveis — fornecendo assim, uma forma mais apropriada e “amigável” para a interação homem-computador; executarão tarefas e ocuparão cargos antes restritos apenas aos humanos — como professores, ajudantes ou gerentes; interagirão com as pessoas em uma forma quase humana, utilizar-se-ão de linguagem natural, gráficos, vídeos animados, etc. Serão também úteis em ambientes onde existam trabalhos cooperativos envolvendo vários usuários, sendo-lhes atribuída uma gama de atividades, como escalonamento, monitoração, comunicação, observação e manipulação.

Uma terminologia foi especificada por Andrew Wood [Woo93] nas representações de agentes inteligentes e foi adotada por De Carlo [DeC95] numa breve descrição dos tipos de Agentes. Ambas se encontram presentes no Apêndice D.

5.2 Características que Tornam uma Aplicação Apropriada para Agentes

Tomando como base as características comuns dos agentes, pode-se identificar um conjunto de características que tornam uma tarefa ou aplicação suscetível a uma abordagem com base na tecnologia de agentes. As seguir as propriedades especificadas por Andrew Wood [Woo94] e simplificada por De Carlo [DeC95]:

- **Adaptação:** tarefa que requer um certo grau de adaptabilidade; o agente necessita desenvolver habilidades para executá-la aprendendo melhores ou novos meios. O que também inclui métodos para evitar falhas e se adaptar às próprias necessidades, desejos e objetivos pessoais do usuário.
- **Pesquisa:** a tarefa não é completamente definida; o agente deve considerar uma grande quantidade de possíveis soluções, escolhendo uma das mais adequadas de acordo com sua experiência.
- **Demonstração:** a tarefa envolve aprendizado e treinamento. Isto inclui ensinar os usuários a usar as ferramentas de *software* de maneira mais eficaz e também, por outro lado, fornecer explicações do que o próprio agente está fazendo.
- **Ajuda:** a tarefa requer um certo grau de cooperação entre o usuário e o Agente. O Agente poderia fazer críticas construtivas ao modo de trabalhar do usuário, ou dar “dicas” sobre como utilizar melhor os recursos do sistema.
- **Autonomia:** a tarefa requer atenção constante ou regular, mas pouca ou nenhuma entrada ou interação. Dessa forma, delegar esta tarefa seria muito útil e benéfico. Um

exemplo seria o monitoramento de sistemas simples, onde uma mudança no comportamento poderia gerar a execução automática de alguma tarefa ou ação.

- **Assincronia:** a tarefa tem um intervalo significativo entre seu início e término. Este intervalo poderia ser devido ao tempo de processamento de grandes quantidades de informação ou mesmo à falta de informações vitais em um determinado momento.

Mais detalhes sobre Agentes, veja [DeC95] e [Cri96].

Com este embasamento que apresentamos já podemos identificar algumas características implícitas no ambiente de aplicação e discutir algumas idéias de implementação. Tudo isso no próximo capítulo.

Capítulo 6

Identificando a Aplicação

Inicialmente, observando-se as situações que um *sysop* enfrenta diariamente, verificou-se que estas exigiam deste profissional uma forma de atuação dinâmica, amigável e ao mesmo tempo reservada no trato com os usuários. Reservada no sentido de não se comportar apenas como um usuário experiente, agindo realmente como um *sysop*: mantendo uma certa distância necessária e aceitável no convívio profissional.

Outro fator bastante importante é a falta de tempo para dar a todas as suas atividades a atenção necessária. Desta forma, nem sempre ele pode atender o usuário que, ora quer tirar uma dúvida, ora quer apenas “bater um papo”.

Em seguida, observando-se as idéias acerca do que é possível com a tecnologia de agentes inteligentes, pretendeu-se criar um **agente operador de um sistema de acesso remoto**. No entanto, após uma reanálise das características da aplicação, percebeu-se que um Sistema Especialista integrado com o IDoctor no ambiente do BBS seria suficiente para obter-se resultados imediatos satisfatórios. Daí, então, uma vez que tanto os Sistemas Especialistas quanto os Agentes utilizam-se de conhecimentos específicos sobre determinado domínio, decidiu-se pela implementação de um Sistema Especialista, o qual preparado para um *chat*, poderia, com um custo muito menor, cobrir um ponto chave da problemática encontrada no ambiente da aplicação: o atendimento ao usuário do BBS.

Tendo em vista que o IDoctor possui características que cobrem a necessidade do usuário de se comunicar com o *sysop* (“bater um papo”), resta ajudar o usuário na solução de seus problemas no que tange ao uso de recursos do sistema.

Observando-se as características distintivas de Agentes e Sistemas Especialistas, já mencionadas no capítulo 4 e 5, verifica-se que elas podem ser resumidas em cinco características essenciais [Cri96], são elas: atuação em paralelo, atuação de forma transparente, autoativação, uso de bases com conhecimentos específicos e aprendizagem.

Pretende-se mostrar até o final do capítulo que estas características podem ser identificadas no conjunto IDoctor e XPERTSYS (que será descrito no próximo capítulo) inseridos no ambiente de SAR do BBS.

Para a produção de um protótipo funcional, considerando as condições da aplicação, nota-se que a característica de autoativação é atendida pelo IDoctor quando este toma o controle do *chat*, funcionando como um agente de comunicação, sempre que o *sysop* não tome a iniciativa de atender o usuário depois de decorrido um tempo, o qual é pré-ajustável por um parâmetro no ambiente.

Com relação à característica de atuação transparente, observa-se que, no caso do IDoctor, seu modo de operar será tanto menos distinto do *sysop*, e portanto mais transparente para o usuário, quanto melhor for o desempenho da conversação baseada no casamento de padrões. Com algum cuidado pode-se desenvolver uma interface de conversação que simule com bastante realidade o comportamento do *sysop* humano.

A característica de paralelismo em agentes está presente, no ambiente onde o protótipo opera, quando vários usuários conectam-se simultaneamente no BBS e chamam o *sysop* para um chat. Assim, o IDoctor pode atender todos de forma paralela, cada um em uma tarefa (*task*), o mesmo podendo acontecer quando em seqüência de execução o XPERTSYS é ativado. Estas atividades poderiam ser vistas como se se tivesse disparado várias *tasks* num Sistema Operacional multi-tarefa (por exemplo Unix).

A Base de Conhecimento, no domínio pretendido, foi elaborada tomando-se como especialista um dos *sysops* do Instant BBS [Ins96], utilizando-se um editor de texto simples, como ferramenta de auxílio na aquisição de dados, através do engenheiro de conhecimento. Mais detalhes sobre a implementação da Base de Conhecimento podem ser encontrados no capítulo 8.

Por fim, a característica de aprendizagem no protótipo está relacionada à evolução da Base de Conhecimento. À medida que o engenheiro de conhecimento, consultando o *log* do

XPERTSYS, verifica as deficiências ou incompletudes da base e toma providências para que a próxima consulta tenha mais chance de ser bem sucedida, verifica-se que o sistema como um todo “aprende”.

Neste caso, não seria possível a aprendizagem direta na interação com o usuário do BBS, dado que justamente o usuário não possui o conhecimento necessário para resolver seu problema e muito menos para inserir conhecimento na Base.

De uma forma geral, podemos visualizar o protótipo proposto como uma aplicação que usa técnicas de Sistemas Especialistas e Agentes, implementada diretamente sobre o ambiente, a qual poderá ser objeto de futuras melhorias usando-se técnicas mais refinadas e explícitas de Agentes. Sugestões para futuras implementações encontram-se no capítulo 9.

A seguir, no próximo capítulo, traremos uma documentação dos requisitos da implementação que tornarão mais fácil entender exatamente como é e como funciona nosso protótipo.

Capítulo 7

Documentando requisitos para a aplicação

7.1 Embasamento

Para poder documentar com maior clareza em que consiste a aplicação proposta, tomamos como base para formalização e referência de documentação um Modelo de Especificação de Requisitos proposto por Pressman [Pre92] e simplificado por Reder [Red94].

A iniciativa de incluir uma especificação de requisitos foi realizada no sentido de possibilitar um melhor entendimento das características do software e como ele se comporta, possibilitando uma posterior continuidade na implementação do protótipo.

É importante notar que a inclusão de uma especificação de requisitos não implica que esta monografia segue rigidamente os preceitos de engenharia de software e toda sua formalidade envolvida. Além do mais, segundo a própria Reder [Red94], o documento de especificação de requisitos é um modelo, uma abstração de alguma situação real, e é por isto mesmo que não é necessariamente completo. Neste sentido procurou-se fazer com que o próximo item (7.2) seja um modelo, uma abstração desta situação real.

Neste sentido, permitindo uma certa flexibilidade, foi possível incluir também alguns diagramas de bloco — do tipo fluxograma — que vem a contribuir em muito no entendimento do funcionamento e dinâmica de alguns processos.

7.2 Especificação de Requisitos

7.2.1 Introdução

Referência do sistema

O protótipo proposto é um Sistema Especialista (SE) destinado a interagir com o usuário sobre um determinado assunto preestabelecido. O sistema será chamado de XPERTSYS.

Descrição geral

Trata-se de um módulo com características de Sistema Especialista. Características estas similares as do sistema especialista do tipo para instrução. Este módulo poderá tanto ser incorporado pela aplicação IDoctor [Fag94], como por um sistema gerenciador de BBS como o Remote Access [Wan94] ou similares.

Trata-se de um door que possibilitará ao usuário encontrar, por si mesmo, dicas e soluções para suas dificuldades no uso dos recursos do BBS ou informações sobre assuntos os quais o *sysop* disponibilizar bases de conhecimentos.

Restrições do projeto de software

Para documentação deste projeto não se utilizou uma metodologia específica, apenas esta especificação de requisitos foi criada baseando-se em Reder [Red94], o que já dá uma idéia razoável de como é o protótipo.

A principal restrição desta implementação é que o módulo especialista que recebe parâmetros passados pelo IDoctor não é parte integrante deste último como inicialmente se pretendia. Os parâmetros passados pelo IDoctor ao XPERTSYS definem, dentre outros dados, principalmente sobre qual assunto tratará a interação especializada com o usuário.

7.2.2 Descrição de informações

Representação do fluxo da informação

- Fluxo de dados

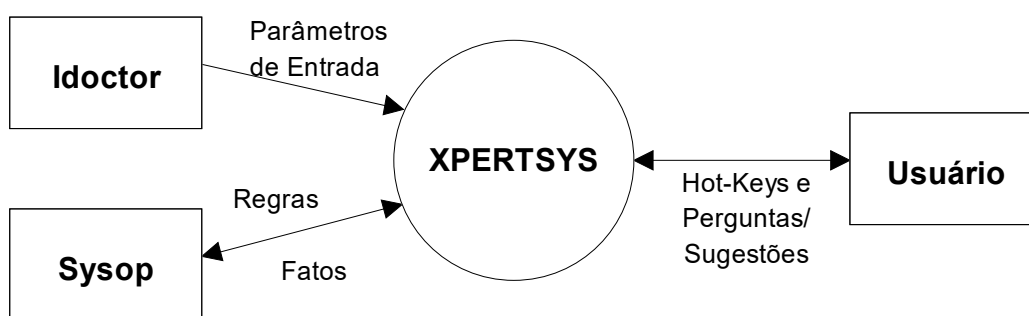


Figura 7.1 - Diagrama de Fluxo de Dados de Nível 0

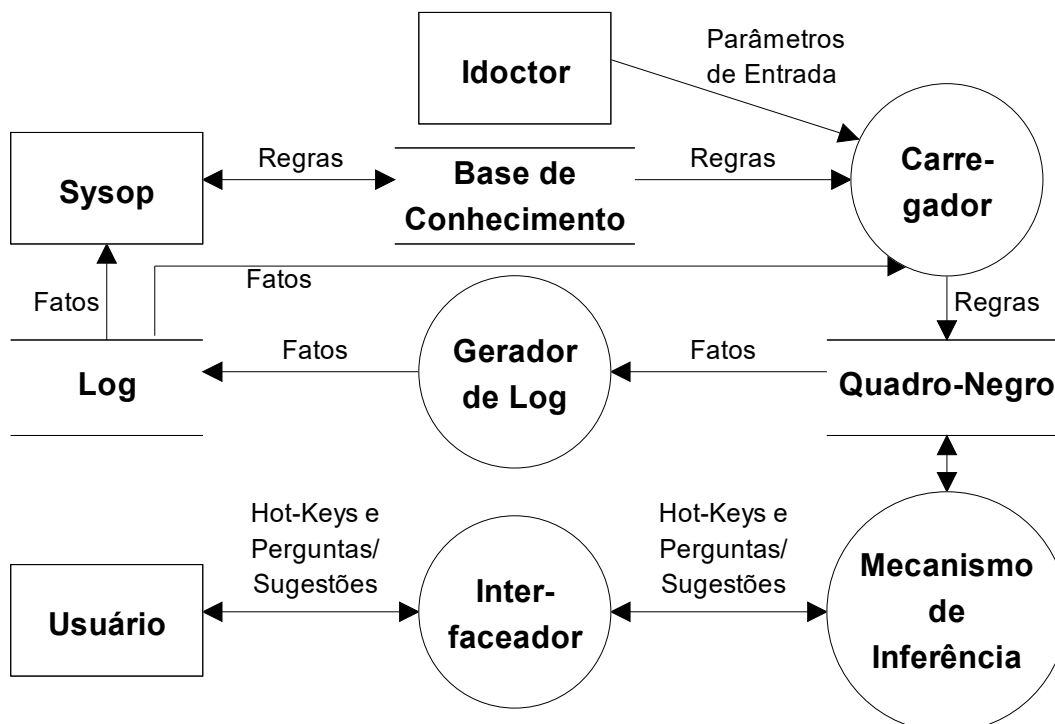


Figura 7.2 - Diagrama de Fluxo de Dados de Nível 1

• Fluxo de controle

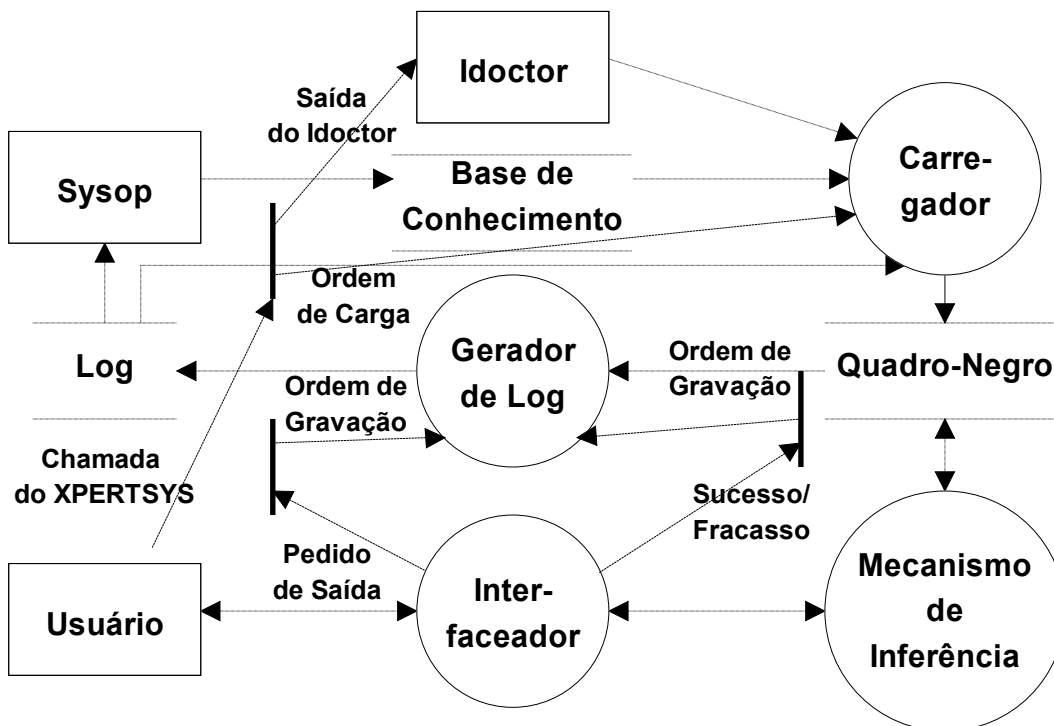


Figura 7.3 - Diagrama de Fluxo de Controle de Nível 1

Representação do conteúdo da informação

Tanto <predicado>, <fato> como <condição_n> são informações do tipo cadeia de caracteres de tamanho máximo 70.

Já <hot-key> e <default> são do tipo 1 (um) caracter, que pode conter “S”, “N”, etc.

Regras: cada regra possui o formato:

<predicado>; <default>:<condição_1>, <condição_2>, <condição_3>, ..., <condição_10>

Onde <predicado> são fatos em potencial, ou seja, se as condições <condição_1>, <condição_2>, etc. forem provadas.

Fatos: <fato> é um predicado que já foi provado e é tomado como verdadeiro.

Parâmetros de entrada: consistem em <assunto>, <usuário> e <log>; onde: <assunto> é o nome da base de conhecimentos a ser carregada; <usuário> é o nome do usuário que está interagindo com o especialista e <log> é o nome do arquivo de log

Hot-Keys: pode possuir os valores “S”, “N”, “!”, “?”, ou “A” e representam respostas ou comandos do usuário. Respectivamente: “SIM” para a pergunta em questão, “NÃO” para a pergunta em questão, “NÃO SEI” para pergunta em questão, comando “POR QUE esta sendo feita esta pergunta ?” e “ABANDONA”.

Perguntas e sugestões: consistem de perguntas expostas para a avaliação do usuário (antes de ler a *hot-key* do teclado), ou sugestão(ões) do especialista sobre como proceder corretamente.

Descrição da interface do sistema

Para manter a interface simples e de fácil portabilidade para futuras implementações (descritas do capítulo 8), preferiu-se usar uma única tela simples, sobre o padrão texto. Para a interação do usuário foi usado apenas o teclado. Veja no Apêndice E uma possível situação da interface do sistema.

7.2.3 Descrição funcional

Partição funcional

Função	Processo
Interfaceamento com o usuário	interfaceador
Controle de inferência	mecanismo de inferência
Carga de fatos ocorridos em seções anteriores do mesmo usuário sobre o mesmo assunto	carregador
Gravação de fatos ocorridos em <i>log</i> para que possam ser lidos tanto pelo próprio XPERTSYS como pelo <i>sysop</i> posteriormente	gerador de <i>log</i>

Figura 7.4 - Quadro das Funções dos Processos

Descrição funcional

• Narrativa do processamento

Carregador: assim que o XPERTSYS é chamado, o carregador é quem recebe e interpreta os parâmetros de entrada, carrega a base de conhecimento indicada, envia para o interfaceador o nome do usuário que irá interagir, e verifica se há alguma seção pendente daquele usuário com essa mesma base de conhecimento (assunto) no *log*. Se esse assunto for pendente, então o carregador carrega também os fatos que já haviam sido provados na seção anterior, deixando o último fato sem carregar, para que o mecanismo de inferência inicie o processo tentando provar justamente o último fato da seção anterior. Isto possibilita que, depois que o usuário perceba em que ponto errou, de acordo com as instruções (sugestões) do XPERTSYS. Caso não haja seção anterior com esse assunto pendente, inicia-se uma nova seção desde seus preliminares.

Interfaceador: consiste de um conjunto de procedimentos que possibilita: expor ao usuário a tela de interação, ler a *hot-key* do usuário, expor as perguntas ao usuário, informar

quando uma possível solução foi encontrada e expor seus resultados (sugestões), informar quando o sistema não conhece nenhuma situação similar e não apresenta resultado algum — apenas diz que conversará com o *sysop* e verá o que se pode fazer, fornecer a explicação do por quê de uma pergunta, etc.

Mecanismo de inferência: trabalha internamente com as estruturas (no DFD presentes como um depósito de dados) referentes ao quadro-negro e consiste num ciclo que a cada iteração tenta provar um determinado predicado ou condição, tornando-o um fato que será tomado como base para as próximas iterações. O processo se encerra quando se encontra um fato conclusivo ou não se encontram regras para determinados fatos.

Gerador de Log: encarregado de acrescentar no arquivo *log* os fatos provados nesta seção, junto com parâmetros usuário, assunto (base de conhecimento), *status* (se sucedeu, falhou ou ainda está pendente), e data e hora da seção.

• Restrições/limitações

O processo interfaceador estabelece com o usuário um padrão de conversação no modo texto, não permitindo uma interação em ambientes gráficos como o Windows, OS/2 ou X-Windows. Compensando esta limitação está a possibilidade de interação através de terminais TTY, VT-100, ANSI, etc.

O *sysop* não pode ficar constantemente (por exemplo diariamente) dando manutenção às bases de conhecimento. De tempos em tempos (por exemplo mensalmente) esta atividade é executada.

O XPERTSYS não possui suporte de leitura, escrita e controle da porta serial — como possui o IDoctor — limitando-se a ser executado sob um nível de shell que permita a sua operação remota, como o DoorWay [Dud93].

• Requisitos de desempenho

Para obter um bom desempenho do sistema, é necessário um microcomputador do tipo IBM-PC com processador 386 sx ou superior, com monitor de vídeo CGA ou superior, com 2 megabytes de memória e de um espaço mínimo de 500 kilobytes em um disco rígido.

• Restrições de projeto

O quadro-negro, por se tratar de uma área de memória onde o sistema “risca e rabisca”, é representada por um conjunto de estruturas criadas para possibilitar o tratamento de regras e fatos.

Tais estruturas estão limitadas de forma que sejam possíveis no máximo 50 regras, cada predicado/condição ou fato esteja limitado até no máximo 70 caracteres e sejam possíveis até 10 condições para cada predicado.

Por se tratar de um protótipo, não foi implementado o mecanismo de aprendizado, onde o SE pede ao usuário que forneça a solução do problema, em caso de fracasso na busca. Não teria sentido a existência de tal mecanismo, já que o usuário realmente não sabe a solução. Assim, o aprendizado dá-se quando o *sysop* facilmente edita a base de conhecimentos, usando um editor ASCII comum.

Descrição de controle

• Especificação de controle

Todo o XPERTSYS é controlado basicamente por dois eventos:

Iniciar: a chamada executada pelo IDoctor dispara o carregador. Depois que este executa suas tarefas, passa o controle para o mecanismo de inferência.

Sair: O mecanismo de inferência atuará até que o interfaceador receba a ordem final do usuário ou seu processo de busca acabe. Em seguida o controle é passado para o gerador de *log* que grava a seção e encerra a execução do protótipo.

- **Restrições de projeto**

Em se tratando de controle, uma restrição é a do XPERTSYS não ser parte integrante do IDoctor como já mencionado anteriormente.

7.2.4 Descrição comportamental

Estados do sistema

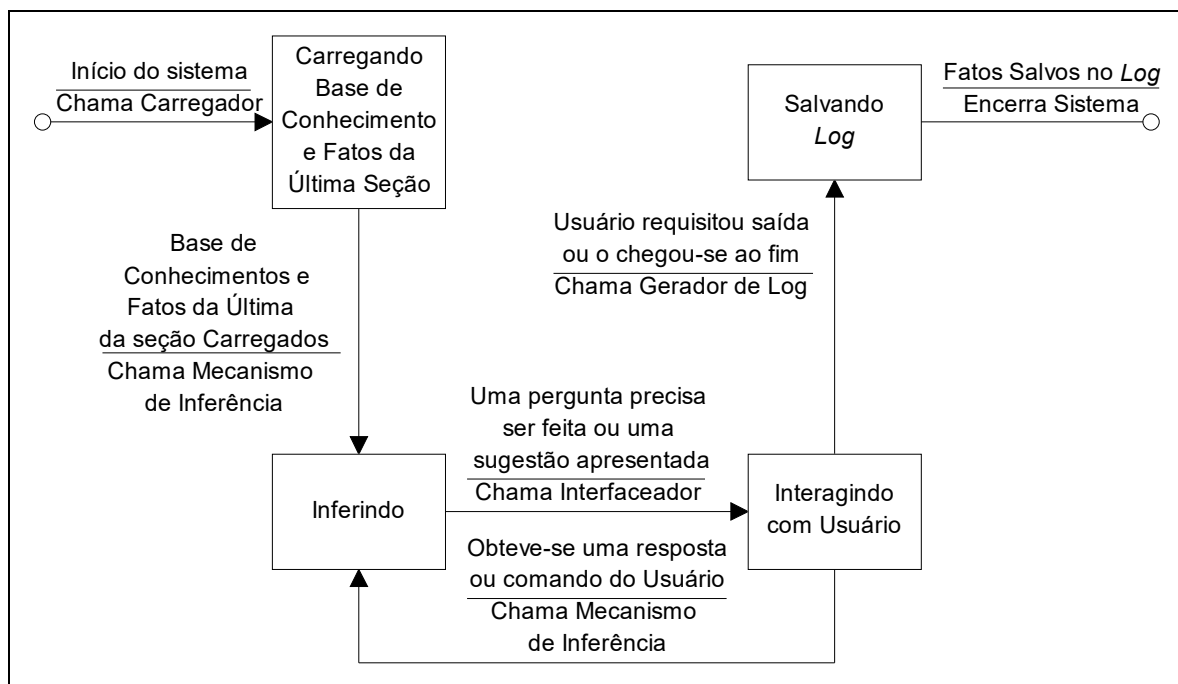


Figura 7.5 - Diagrama de Transição de Estados

Eventos e ações

Evento(causa)	Ação(consequência)	Descrição
Início do sistema	Chama Carregador	Disparo de execução pelo Usuário seguido de carga dos depósitos necessários
Base de Conhecimentos e Fatos da Última Seção Carregados	Chama Mecanismo de Inferência	Início do processo de inferência (do início ou do ponto em que o usuário havia deixado a última seção)
Uma Pergunta Precisa ser Feita ou uma Sugestão Apresentada	Chama Interfaceador	Invocação do Interfaceador, que expõe uma Pergunta ou Apresenta uma Sugestão
Obteve-se uma Resposta ou Comando do Usuário	Chama Mecanismo de Inferência	Retorno da resposta do usuário
Usuário Requisitou Saída ou Chegou-se ao Fim	Chama Gerador de Log	Encerramento da seção, início da geração de Log
Fatos Salvos no Log	Encerra o Sistema	Retornando ao nível anterior de shell

Figura 7.6 - Quadro de Eventos e Ações

• Diagrama de Blocos

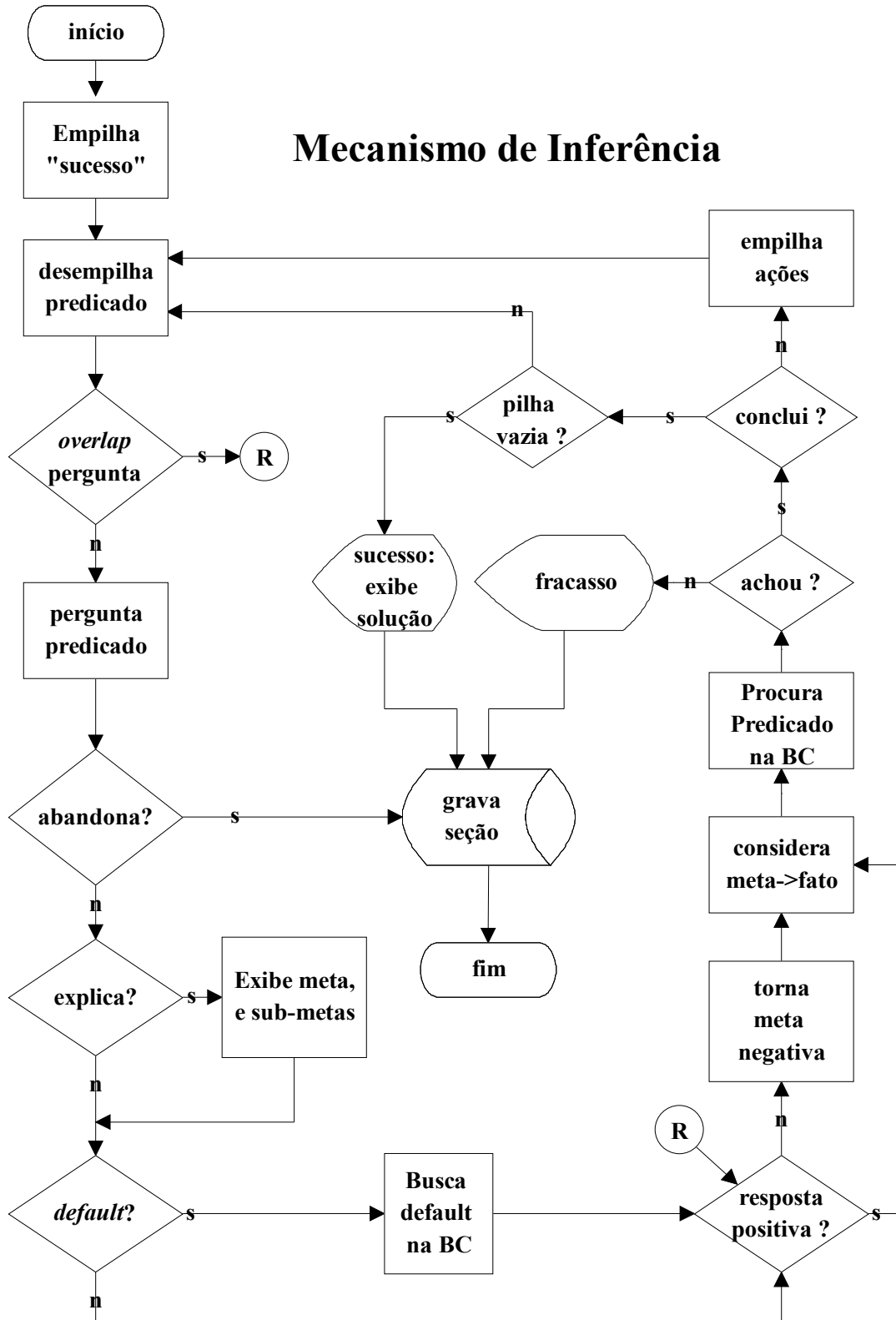


Figura 7.7 - Fluxograma de Blocos do Mecanismo de Inferência

7.2.5 Critérios de validação

O critério de validação baseia-se essencialmente em Cunha & Ribeiro [Cun87].

Limites de desempenho

As formas determinadas para representar os conceitos e premissas do conhecimento do especialista devem favorecer o desempenho do sistema. Neste protótipo, deve ser usado um tipo de dado que, ao mesmo tempo que simples, seja representativo: o tipo *string*. Assim, tanto os predicados e condições como as metas e fatos são bem representados por este tipo de estrutura de dados, provendo, além da compatibilidade interna entre as estruturas do quadro-negro e o depósito *log*, também a externa com as entidades *sysop* (ou Engenheiro do Conhecimento) e usuário.

Nenhuma menção a limite de tempo se faz necessária dado que na plataforma de implementação atual raramente será possível respostas além do limite de tempo real.

Já no que tange ao espaço em memória (RAM) ocupado pelo protótipo em funcionamento, deve-se ressaltar que a implementação deve respeitar os limites do sistema operacional (no DOS 640kb) e de distribuição de memória (no DOS blocos de 64k) na plataforma escolhida.

Deve-se usar um editor de textos de ASCII puro para introduzir o conhecimento na Base de Conhecimento como parte operacional do processo de aquisição. Não deve haver respostas demasiadamente longas ou extremamente curtas. Não deve haver regras que produzam resultados ambíguos ou confusos.

Classes de testes

Nesta fase avalia-se tanto a velocidade em que as respostas são exibidas quanto a qualidade delas. Como se trata de um protótipo e os microcomputadores de hoje são

incomparavelmente mais velozes que os da época em que os critérios de teste foram criados, dificilmente não haverá uma resposta em tempo hábil. Quanto à qualidade das respostas, foi criada uma bateria de testes (Apêndice F), com respostas conhecidas, para medir o desempenho do protótipo. Isto permitirá a verificação de sua eficiência e praticidade.

A implementação deste protótipo conta com a freqüente (ao menos bimestralmente) manutenção às Bases de Conhecimento por parte do *sysop*, como forma de teste constante da consistência e eficiência dos conhecimentos das Bases.

Respostas esperadas do software

Identificam-se as respostas do XPERTSYS como as respostas do terceiro modo de técnica de emissão, tal qual descrito no item 4.1. Assim, o objetivo das respostas do protótipo não é emitir um resultado, mas interagir com o usuário do BBS funcionando como em um diálogo entre o Especialista e o leigo, obrigando este último a refletir e entender, por exemplo uma determinada seqüência de passos necessária ao uso de determinado recurso do BBS como um todo, ou qualquer que seja o assunto (Base de Conhecimento implementada pelo *sysop* ou Engenheiro de Conhecimento).

Uma seção do XPERTSYS pode terminar em três diferentes situações: o usuário cansou-se do “diálogo” e resolveu encerrá-lo incondicionalmente, o XPERTSYS obteve sucesso na busca heurística e sugeriu ao usuário que experimentasse um conjunto de possíveis soluções, ou então o protótipo não obteve sucesso na busca e informa ao usuário que não tem nenhuma sugestão imediata e irá informar o *sysop*.

No primeira das três saídas, o XPERTSYS grava a seção para que seja retomada posteriormente, e nas outras duas grava a seção com um *status* ou de sucesso (“OK”) ou de falha (“FAIL”) — o que significa que ou não se sabe porque o usuário está com problemas ou não existe uma aparente solução imediata para o problema.

No capítulo seguinte traremos o relato da experiência no desenvolvimento do protótipo e como ele foi implementado em seu ambiente de aplicação.

Capítulo 8

O protótipo funcional em seu ambiente

Foi criada uma integração entre o casamento de padrões do IDoctor e o mecanismo de inferência do sistema especialista (XPERTSYS), de tal forma que o IDoctor seja usado para detectar uma necessidade do usuário e, em seguida, passe o para o especialista, que fornecerá informações que podem estar faltando para o usuário usufruir adequadamente dos recursos do BBS. Abaixo o esquema implementado.

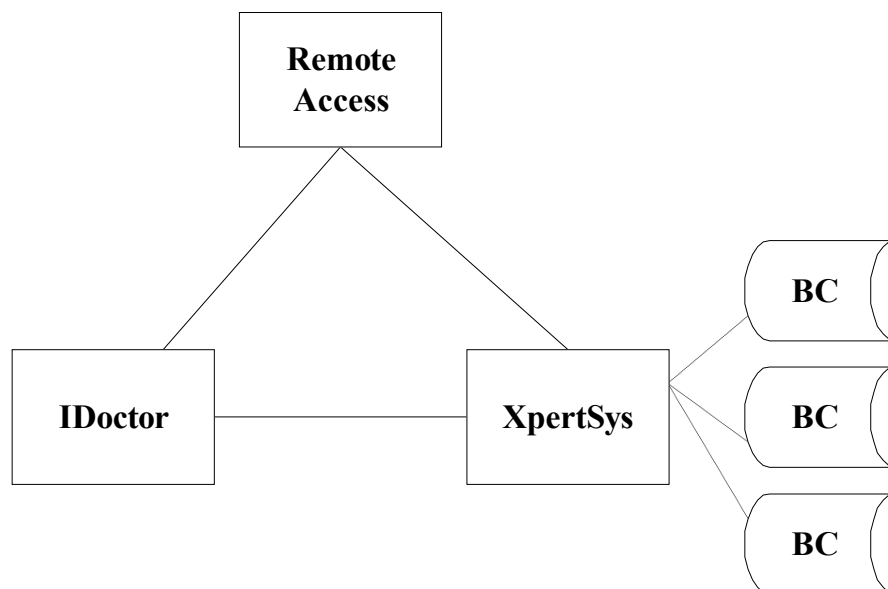


Figura 8.1 - O XPERTSYS em seu ambiente

A implementação foi executada usando o ambiente de programação em C 3.0 da Borland. Foi escolhida a linguagem C por três motivos: portabilidade para futuras implementações, domínio da linguagem, e pelo IDoctor também ter sido implementado em C (já que no início pretendia-se uma implementação totalmente integrada).

Foram criadas três bases de conhecimento: *download*, mensagens e internet. Cada uma possui de 20 a 30 regras nos respectivos assuntos ligados ao BBS. Cada base de conhecimento foi desenvolvida em cerca de uma semana, acompanhando seções de *chat* entre o *sysop* do Instant BBS e usuários. De cerca de 50 seções acompanhadas, 16

iniciaram-se com estes assuntos e no total 22 delas passaram por orientações do *sysop* para o usuário nestes assuntos. Após o término dos *chats* foram discutidas as regras e a maioria delas era bastante clara. Assim as poucas que restaram foram ajustadas posteriormente.

O mecanismo de inferência foi implementado seguindo-se Cunha e Ribeiro [Cun87] e usado um processo de refinamento do tipo teoria - tentativa - resultado. Neste processo percebemos duas características marcantes:

- C não é uma linguagem perfeita para tal implementação (no entanto não abrimos mão da compatibilidade e portabilidade)
- Um mecanismo de inferência na prática não é exatamente como a teoria descreve. Muitas vezes dois ou mais processos tiveram suas ordens de chamada invertida para possibilitar o resultado pretendido, contrariando a teoria.

Muitas vezes o processo de criação das bases de conhecimento foi mais trabalhoso que o próprio mecanismo de inferência. Isto é atribuído ao fato de o mecanismo de inferência possuir um ciclo de desenvolvimento que chega a um fim, e o ciclo das bases de conhecimento não necessariamente termina.

No próximo capítulo as conclusões do trabalho.

Capítulo 9

Conclusões

9.1 Sobre os rumos traçados

Apesar do protótipo ser relativamente modesto em comparação com tudo o que vislumbra a Inteligência Artificial em termos de Agentes, pode-se reconhecer a implementação do protótipo, no seu ambiente, as principais características preconizadas para um Agente.

9.2 Futuras implementações

Em próximas implementações o XPERTSYS pode ser acrescido de inúmeras bases de conhecimentos, podendo assim ser usado em contextos diversos.

Será possível a integração completa do Idoctor com o XPERTSYS utilizando algum ambiente de definição de Agentes como o SSAA de Moniz [Mon93] facilitado por Crivelli [Cri96].

A implementação do XPERTSYS em C possibilita que a aplicação seja portada para o C Unix (por exemplo o GCC) e adaptada para o uso através de *telnet* na Internet. Isto vem a acrescentar em muito sua possibilidade de utilização em *sites* acadêmicos e em SARs do tipo provedor de Acesso Internet e torna totalmente explícitas as características de Agente.

Além de portar o sistema para o ambiente Windows ou X-Windows, é possível também a criação de ferramentas para a aquisição de conhecimento, que podem ser desde editores inteligentes até módulos de aprendizado, passando por interfaces mais elaboradas com recursos de rede e multimídia.

9.3 Considerações Finais

Ao verificar-se a simplicidade do protótipo implantado, comparado com o que se espera da emergente tecnologia de Agentes, pode-se pensar que o passo dado é pequeno demais. Porém, tendo em vista o contexto didático imediato do trabalho desenvolvido, e reconhecendo que o produto obtido não só pode ser utilizado comercialmente, como na realidade o será, é possível então se sentir tão gratificado com ele como talvez, Weizembaun se sentiu com o singelo ELIZA, nos anos 60.

Referências Bibliográficas

- [Cri96] Crivelli, M. P. e Santos, N. M. “*Análise dos Requisitos Necessários para Implementação de um Protótipo de uma Bancada para Simulação de Ambientes e Agentes Heterogêneos*”, Trabalho de Graduação, Universidade Estadual de Maringá, Paraná, Brasil, (Dez. 1996).
- [Cun87] Cunha, H. e Ribeiro, S. “*Introdução aos sistemas especialistas*”, Livros Técnicos e Científicos Editora S.A., 1987.
- [DeC95] De Carlo, R. e Santos, N. M., “*Estudo e Aplicação de Agentes Inteligentes no Processo de Engenharia de Software*”, Trabalho de Graduação, Universidade Estadual de Maringá, Paraná, Brasil, (Dez. 1995).
- [Dud93] Dudley, M. - O software “*DoorWay - DoorWay to Unlimited Doors* ©”, © Copyright 1987-1993 TriMark Engineering & Marshall Dudley, 1993.
- [Fag94] Fagundes, S. A. O software “*Doctor Chat Door v1.04 Shareware Edition* ©” - © Copyright 1994 Abyss Software & Sérgio André Fagundes, 1994.
Fidonet: 4:802/27, Rede Brasileira de Telemática (RBT)12:1221/19.
- [Ins94] Instant BBS, Boletim para leitura on-line “*O que é um BBS?*”. Acesso por modem pelo telefone (044)226-1450 em Maringá, Paraná, Brasil. Setembro de 1994.
- [Ins96] Instant BBS, Contato - Serviços de Informação por Computador Ltda. ME.
Acesso por modem pelo telefone (044)226-1450 em Maringá, Paraná, Brasil.
E-mail: sysop@instant.kanopus.com.br
- [Lem96] Leme, M. A. S. M., “*Contrato Social do Instant BBS*”, *Contato - Serviços de Informações por computador Ltda. ME*, Brasil, maio de 1996.

- [**Mon93**] Moniz, L., “*SSAA : Sistema de Simulação de Agentes e Ambientes*” Msc. Thesis, Lisboa : Instituto Superior Técnico/Universidade Técnica de Lisboa, Portugal, Junho de 1993.
- [**Pre92**] Pressman, R. S., *Software Engineering - A Practitioner’s Approach*, 3a. Edição, McGraw-Hill International Editions, 1992.
- [**Red94**] Reder, A. C., *Uma metodologia para análise de requisitos orientada a objetos*, Universidade Estadual de Maringá, 1994.
- [**Wan94**] Wantree Development & Milner A. O software *Remote Access 2.02* © - 1989-1994 Wantree Development & Andrew Milner, 1994.
- [**Wei66**] Weizenbaum, J. “*ELIZA*” - Msc. Thesis, MIT, 1964-1966.
- [**Woo93**] Wood, A. & Dr Beale R. “*Desktop Agents*” Scholl of Computer Science, The University of Birmingham, abril de 1993.
- [**Woo94**] Wood, A., “*Towards a Medium for Agent-Based Interaction*”, Thesis Proposal, University of Birmingham, School of Computer Science, (Out.1994)

Bibliografia

- [Car95] Carvalho, J. “*Tecnologia de agentes ganha mercado*” Revista ComputerWorld, (Infoworld/EUA), edição de Maio de 1995.
- [Ind95] Indermaur, K. “*Baby Steps*” - *State of the Art*, Revista Byte (USA), edição de março de 1995.
- [Ric93] Rich, E. e Knight K., “*Inteligência Artificial*”, 2a. Edição, Makron Books do Brasil Editora Ltda, 1993.
- [Sch93] Shieldt, H. “*C completo e total*”, McGraw-Hill & Makron Books do Brasil Ltda., 1990.
- [Shn92] Shneiderman, B. *Designing the user interface: strategies for effective human-computer interaction*. Segunda edição. Addison Wesley 1992.
- [Sim83] Simon, H. A. “Why should machines learn?”. In *Machine Learning. An Artificial Intelligence Approach*, ed. R. S. Michalski, J. G. Carbonell e T. M. Mitchell. Palo Alto, CA, Tioga Press, 1983.
- [Way95] Wayner, P. “*Free Agents*” - *State of the Art*, Revista Byte (USA), edição de março de 1995.

Apêndice A

Funcionamento de uma Rede de Mensagens entre BBS's

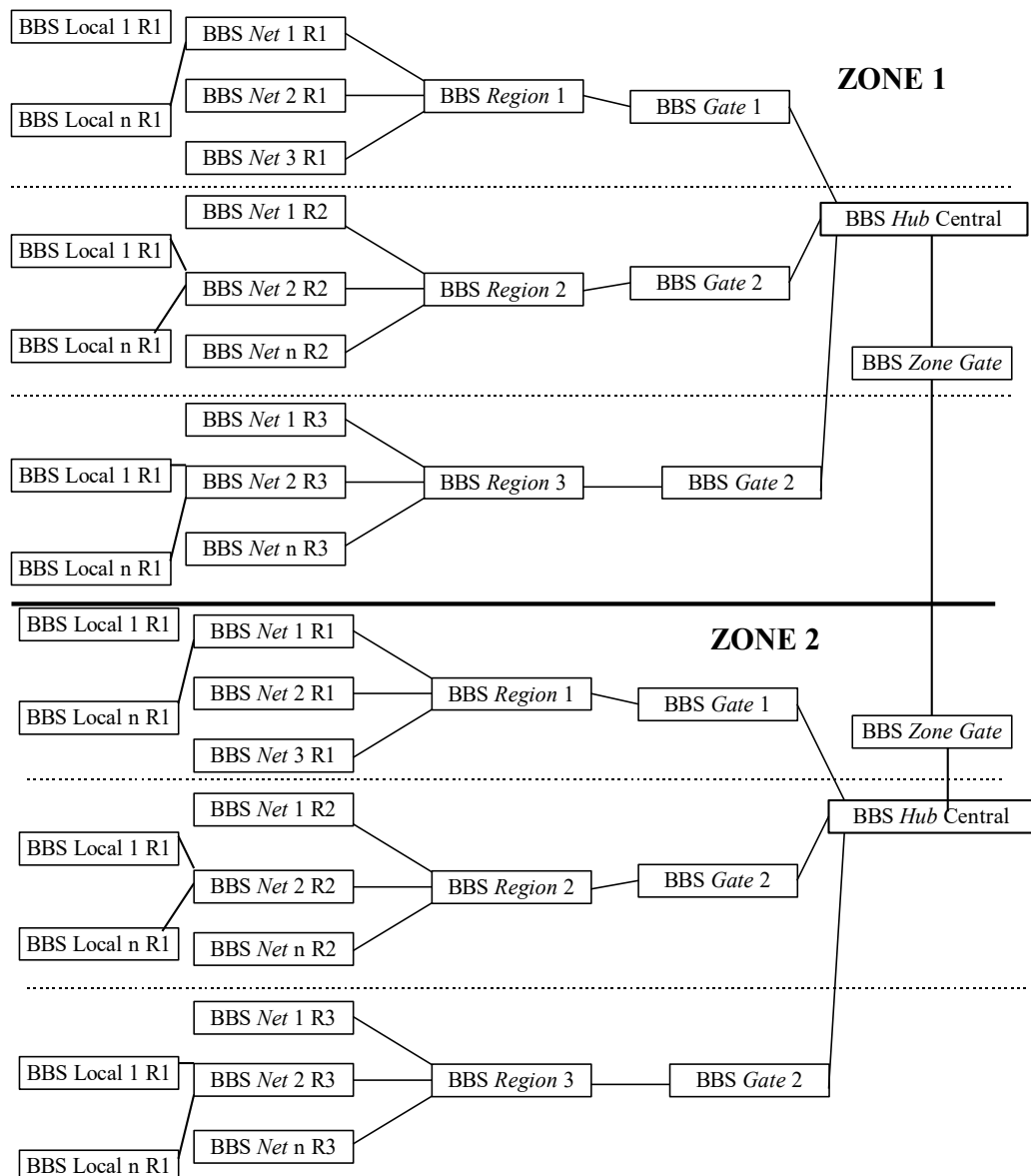


Figura A.1.1 - Uma rede de troca de mensagens entre BBS's

Na figura A.1.1 as linhas finas representam as ligações telefônicas entre um BBS e outro. Geralmente acontece todas durante a madrugada e são programadas para um horário fixo, não requerendo um operador para executá-la.

Temos duas zonas distintas: *Zone 1* e *Zone 2*. (Estas zonas poderiam ser exemplificadas por América no Norte e Europa). Em cada zona existe um BBS *Zone Gate* (ponte) o qual faz a ligação intercontinental.

Cada região representa um país. Dentro de cada país temos um *gate* que faz a ligação internacional e os “*nets*”, geralmente, um para cada estado. Dentro de cada estado temos vários BBS’s distribuídos nas cidades e um BBS *Net* que centraliza o estado. Se houver mais de um BBS em uma cidade, na qual não esteja localizado o *Net* do estado, então podemos ter *Hubs* que centralizem a distribuição nessa cidade. E um desses *Hubs* faria as ligações interurbanas vindo a ser, dessa maneira, apenas um distribuidor.

No caso do Paraná, um BBS situado no interior tem o mesmo nível na rede que um outro BBS qualquer em Curitiba, desde que não seja o BBS *Net*.

Para rotear mensagens por toda esta rede, tem-se endereços para identificar os BBS’s. São formados desta maneira:

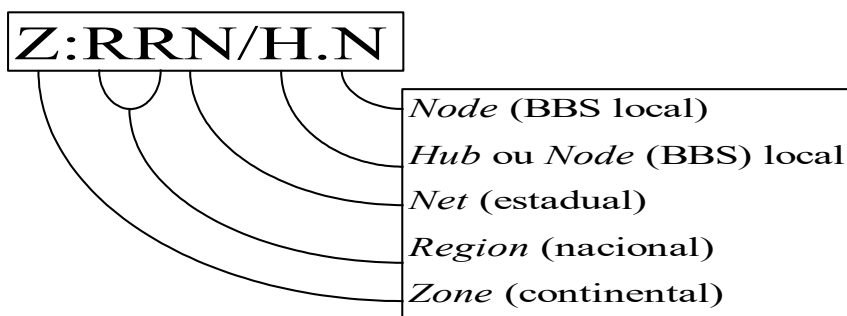


Figura A.1.2 - Formação de endereços de rede de BBS

No caso específico do Instant BBS de Maringá:

<i>Zone 4</i>	(América Latina)
<i>Region 80</i>	(Brasil)
<i>Net 4</i>	(Paraná - DDD 04)
<i>Hub e Node 44</i>	(o Instant BBS, BBS local e Hub da cidade de Maringá (DDD 044))
<i>Node</i>	(ainda não existem outros nodes em Maringá)

Assim tem-se:

4:804/44 representa o Instant BBS de Maringá na rede Fidonet, de alcance mundial.

Pode-se concluir assim, que trata-se de uma rede mais lógica do que física, já que o meio só é estabelecido quando ocorrem os eventos de trocas de mensagens e os BBS's se ligam para executar a transação.

Desta forma, os BBS's fazem um papel importante no desenvolvimento da informática, "coleccionando" informações sobre diversas áreas, promovendo conferências (*on-line* ou via mensagens) entre informatas em geral e transitando conhecimento entre os usuários.

Apêndice B

Glossário

- **Sysop**: *SYStem OPerator*, operador do sistema. A pessoa encarregada de todas as tarefas técnicas em um BBS (a operação).

- **Chat**: conversação entre o usuário e o *sysop* ou entre dois usuários em que a troca de informações se dá através da linha telefônica. Geralmente a tela é dividida em tantas vezes quantas forem as pessoas na conversa e todas as pessoas tem uma imagem (tela idêntica) da tela da conversa. O que cada um escreve é impresso numa dessas diferentes partes da tela.

- **News**: seção em que todo usuário do sistema passa na entrada, a cada vez que ‘loga’ no BBS. O *news* é uma sucessão de textos-notícia mostrados ao usuário que lhe informam acerca de novas características do sistema, novos recursos, eventos promocionais do BBS, cuidados necessários em algumas seções, normas do sistema, etc.

- **Door**: é considerado um *door*, todo tipo de módulo adicional que pode ser incluído no sistema com exceção do gerenciador principal do BBS. Assim, existem uma infinidade de utilitários que se somam ao sistema para acrescentar um novo recurso ou melhorar um já existente. Mais recentemente algumas ferramentas de programação e *browsers* para Windows têm incluído este mesmo conceito com o nome de *Plug-In* ou *Add-On*. Como exemplo podemos citar: jogos *on-line*, *chat* para usuários, *chat* operador-usuário, editor de mensagens, empacotador de mensagens, procurador de vírus, gerador de *logs*, apresentador de *news*, criador de endereços Internet, etc.

- **Download**: é o processo de transferência de um ou mais arquivos de um computador remoto para o computador que está sendo usado localmente. Recepção de arquivo.

- ***Microiro***: [Lem96] classe de usuário de microcomputadores que, com características autodidatas, procura adquirir conhecimento em informática que vai muito além do que lhe é necessário, ou seja, seus conhecimentos estropolam suas reais necessidades de aplicação emediata.

Apêndice C

Classificação de SE's

C.1 Classificação de Sistemas Especialistas

Pode-se classificar os sistemas especialistas [Cun87] de acordo com suas características de funcionamento nas seguintes categorias:

C.1.1 Interpretação

Faz análise de dados e procura determinar as relações e seus significados. Também deve saber como se comportar quando existem informações erradas, distorcidas, ambíguas ou parcialmente ausentes.

C.1.2 Diagnósticos

Destinado a detectar falhas conseqüentes da interpretação de dados, também detecta falhas com origem no próprio diagnóstico (que não detectou antes por ter falhado). Deve permitir ao diagnosticante decidir sobre que medida tomar, haja visto que os dados podem ser inacessíveis, caros ou de perigosa recuperação. Este sistema engloba o sistema de interpretação de dados.

C.1.3 Monitoramento

Deve estar constantemente monitorando sinais sobre o comportamento do monitorado, observando limites e avisando quando tais limites forem ultrapassados. Sinais idênticos

podem ser interpretados de maneira diferente devido ao estado dos fatos globais em momentos diferentes.

C.1.4 Predição

Com base em fatos do passado, o sistema deve permitir uma determinada visão do futuro. Deve ter recursos que lhe permitam considerar os vários futuros possíveis, a partir da análise do comportamento dos dados, usando raciocínios hipotéticos e verificando a tendência de acordo com a variação dos dados de entrada.

C.1.5 Planejamento

Capaz de trabalhar com grandes problemas, é encarregado de dividir de forma adequada o problema em problemas menores e definir metas (intermediárias) e/ou submetas para se chegar ao (grande) objetivo. Em caso de metas conflitantes, deve definir prioridades.

C.1.6 Projeto

Além de possuir características similares às do Planejamento, este sistema tem a incumbência de confeccionar especificações que possibilitem o atendimento aos objetivos dos requisitos particulares. Deve ser capaz de justificar alternativas tomadas ao projeto final e, nas etapas de alteração do projeto, alterar todos os pontos em que a atual alteração venha a afetar.

C.1.7 Depuração

Possui mecanismo fornecedor de soluções para as falhas provocadas por distorções de dados. Automaticamente verifica diversas partes do projeto, validando cada etapa do projeto.

C.1.8 Reparo

Este tipo de sistema segue um plano para administrar os resultados fornecidos por um sistema de diagnóstico. Assim, desenvolve e executa planos de reparos a serem realizados.

C.1.9 Instrução

Possui recursos para verificar e corrigir o comportamento do aprendizado de estudantes. Tomando como base uma informação sobre o conhecimento hipotético do aluno, interage com treinando, podendo apresentar pequenas explicações intermediárias para serem analisadas pelo mesmo. De acordo com o seu grau de conhecimento, pode aumentar ou diminuir o nível de dificuldade, se o aluno domina ou não determinada etapa, respectivamente.

C.1.10 Controle

Engloba e controla todos os demais sistemas, levando o conjunto de sistemas como um todo a atingir o grande objetivo comum.

Apêndice D

Classificação de Agentes

D.1 Terminologia para Agentes Inteligentes

Agente Humano: qualquer pessoa que interage direta ou indiretamente com um sistema de computador - geralmente chamada de usuário ou usuário final.

Agente Inteligente: uma entidade que se comporta como um agente.

Ferramenta de *Software* ou Aplicação: qualquer pedaço de *software* que não seja um agente. *Softwares* utilizados no processo de informações, como editores de texto, planilhas e agendas eletrônicas, programas de desenho e editoração, etc.

Comunicação ou Interação: a seta indica que controle ou informação podem ser transmitidas entre as entidades.

Observação: a seta indica uma entidade está obtendo informações sobre o estado de outra entidade.

Interface: o ponto pelo qual duas entidades comunicam-se. Com relação a interação homem-*software*, essa é tradicionalmente conhecida como a interface do usuário.

D.2 Tipos de Agentes

Pode-se classificar os agentes inteligentes em várias categorias condizentes com suas características. A seguinte classificação apresentada foi por Andrew Wood [Woo93] e simplificada por De Carlo [DeC95].

Conselheiro: oferece ajuda e treinamento. Ensina os passos iniciais para usar um determinado sistema. Fornece suporte contínuo, observando todas as ações do usuário, as quais ele pode interceptar e pedir confirmação. Pode ser consultado para mostrar como executar uma atividade particular, ou sugerir métodos alternativos e mais rápidos para executá-la.

Guia: ajuda a navegação em bancos de dados e hipermídia. Classifica, recupera e filtra grandes quantidades de informações, apresentando somente os dados relevantes e importantes ao usuário, no formato personalizado. Fornece caminhos apropriados para o usuário navegar pelo banco de dados e auxilia-o, caso se sinta “perdido”.

Empregado: executa as atividades tediosas e repetitivas, o conhecido “serviço braçal”. Um tipo de *feedback* pode ser fornecido pelo usuário ou pelo agente.

Representante: trabalha na ausência do usuário. Parecido ao agente Empregado citado anteriormente, diferenciando-se no fato de que as atividades não precisam ser imediatamente executadas, podendo ser executadas após eventos específicos. Alguns exemplos seriam: fazer *backups* de arquivos de madrugada, fazer pedidos de compras — caso algum produto atinja o limite mínimo no estoque.

Comunicador: trabalha com outros usuários e seus agentes, para conseguir executar as atividades às quais é incumbido. Por exemplo, poderia organizar reuniões, alocando recursos, convidando as pessoas, ou juntar um grupo de agentes para executar uma atividade mais complexa.

Apêndice E

Interface do XPERTSYS

Abaixo uma possível situação do XPERTSYS, interagindo com um usuário sobre um assunto (Base de Conhecimento) específico.

```
Agente Especialista para Sistemas de Acesso Remoto
Base de Conhecimentos em Ação: DOWNLOAD
Interação com o Usuário: FULANO_DE_TAL
*-----*
*
CONSEGUIU FAZER DOWNLOAD ?
[(S)im, (N)ão, Não Sei(!), Por que(?), (A)bandona]:
*-----*
*
```

Figura E.1 - A interface do XPERTSYS

Apêndice F

Bateria de Testes

Como uma forma de complementar a validação da implementação do protótipo XPERTSYS, foram relacionados algumas situações reais trazidas diretamente do ambiente de aplicação.

O ambiente, como um todo, deve conter as características de Agente, e as respostas fornecidas pelo XPERTSYS, de acordo com suas Bases de Conhecimento, devem demonstrar as características de Sistema Especialista.

BC	Fatos ocorridos com o Usuário	Resposta do XPERTSYS
D O	<ul style="list-style-type: none"> • Não conseguiu fazer <i>download</i> • Não sabe se recebeu mensagem de “nenhum arquivo enviado” • Chegou a acionar o <i>download</i> • Não sabe se usou protocolo correto 	Sugere o uso do protocolo “ZMODEM”
W N L O	<ul style="list-style-type: none"> • Não conseguiu fazer <i>download</i> • Não sabe se recebeu mensagem de “nenhum arquivo enviado” • Chegou a acionar o <i>download</i> • Usou protocolo correto • Acionou o <i>download</i> local 	Não possui conhecimento necessário para sugerir uma possível solução
A	<ul style="list-style-type: none"> • Conseguiu fazer <i>download</i> • O arquivo está no disco • Não sabe se olhou no diretório certo 	Sugere possíveis nomes de diretórios onde podem estar os arquivos
D	<ul style="list-style-type: none"> • Conseguiu fazer <i>download</i> • O arquivo está no disco <perguntou “porque”> • Olhou o diretório certo 	Não possui conhecimento necessário para sugerir uma possível solução

Figura F.1 - Quadro de testes com a Base de Conhecimentos em *Download*

BC	Fatos ocorridos com o Usuário	Resposta do XPERTSYS
M E N	<ul style="list-style-type: none"> • Não sabe se conseguiu postar mensagens • Foi até o menu mensagens • Não sabe se selecionou a área de mensagens <pede para responder novamente> • Digitou o nome do destinatário • Não sabe se enviou mensagem preparada 	Sugere leitura de um boletim informativo sobre mensagens preparadas
S A G E N	<ul style="list-style-type: none"> • Não sabe se conseguiu postar mensagens • Foi até o menu mensagens • Não sabe se selecionou a área de mensagens <pede para responder novamente> • Digitou o nome do destinatário • Não enviou mensagem preparada • Digitou o nome do destinatário • Digitou a mensagem • Não salvou ao sair 	Sugere que a mensagem seja salva e indica como fazê-lo
S	<ul style="list-style-type: none"> • <retorna> • Salvou a mensagem ao sair 	Conclui que esta foi a solução do problema

Figura F.2 - Quadro de testes com a Base de Conhecimentos em Mensagens

Com o XPERTSYS respondendo tal qual mostrado nas figuras F.1 e F.2 pode-se validar o XPERTSYS como Sistema Especialista.

Apêndice G

Código Principal do Protótipo

```
int main(int argc, char *argv[])
{
    int ret=0;
    int resp=0;
    int pr=0;
    int r=0,result=0;
    int not_end=1;
    char *meta;
    char *nao_meta;
    int abandono=0;
    if(argc < 4)
    {
        printf("\nUse: \n");
        printf("XPERTS2 <base de conhecimento> <nome_do_usuario>
        <arquivo_log>\n");
        return(1);
    } else
    {
        clrscr();
        strcpy(log,argv[3]);
        strcpy(fulano,argv[2]);
        strcpy(bc,argv[1]);
        strcpy(bci,argv[1]);
        carrega_bc(argv);
        carrega_fatos(argv);
        mostra_tela();
        empilha("SUCESSO");
        do //faca -- enquanto tem algo na pilha
        {
            //mecanismo de inferencia
            meta=desempilha();
            nao_meta=nega(meta);
            if ((overlap_pergunta(meta)) || (overlap_pergunta(nao_meta)))
            //ja foi feita esta pergunta
            {
                if (eh_fato(meta))
                {
                    resp=0;
                }
                if (eh_fato(nao_meta))
                {
                    resp=1;
                }
                if ((!eh_fato(meta)) && (!eh_fato(nao_meta)))
                {
                    if (overlap_pergunta(meta))
                    {
                        resp=0;
                    }
                    if (overlap_pergunta(nao_meta))
                    {
                        resp=1;
                    }
                }
            }
        } else //esta pergunta ainda n,,o foi feita
        {
            if (!strcmp(meta,"SUCESSO"))
            {
                resp=0;
            } else
            {
                resp=pergunta_predicado(meta);
            }
        }
        if (resp==9) abandono=1;
        if (!abandono)
```

```

{
    //default -- j implementado
    pr=eh_conhecimento(meta);
    //if (resp==4) {retorna(pr);}
    while (resp==3)
    {
        explica(pr);
        resp=pergunta_predicado(meta);
    }
    if (resp==2)
    {
        resp=get_default(pr);
    }
    if (resp==1)
    {
        meta=nega(meta);
        pr=eh_conhecimento(meta);
    }
    considera_fato(meta);
    if (pr==-1) //n,"o est na BC
    {
        not_end=0;
        result=0;
    } else //est na BC
    {
        result=1;
        r=conclui(pr); //
        if (!r) //n,"o , um conhecimento conclusivo
        {
            empilha_todos(pr);
        }
        if ((r)&&(r!=9)) //, um conhecimento conclusivo
        {
            not_end=!pilha_vazia();
        }
        if (r==9)
            abandono=1;
    }
}
if (abandono)
{
    abandona();
    grava_fatos(3);
    ret=0;
    printf("\n Pressione qualquer tecla.");
    getch();
    clrscr();
    exit(ret);
}

} while (not_end);

ret=result;
if (ret)
{
    grava_fatos(1);
    solucao();
} else
{
    grava_fatos(2);
    mostrã_fracasso();
}
}
printf("\n Pressione qualquer tecla.");
getch();
clrscr();
return(ret);

```

Universidade Estadual de Maringá
Departamento de Informática
Av. Colombo 5790, Maringá-PR
CEP 87020-900
Tel.: (044) 261-4219 e 261-4324
Fax: (044) 223-2676