UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

(Bacharelado)

SISTEMA DE APOIO PARA OTIMIZAÇÃO DAS ATIVIDADES DE SUPORTE TÉCNICO DE UMA EMPRESA DE DESENVOLVIMENTO DE SOFTWARE, UTILIZANDO RACIOCÍNIO BASEADO EM CASOS

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDO À UNIVERSIDADE REGIONAL DE BLUMENAU PARA A OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA COM NOME EQUIVALENTE NO CURSO DE CIÊNCIAS DA COMPUTAÇÃO — BACHARELADO

CARLOS EDUARDO DE SOUZA E SILVA

BLUMENAU, DEZEMBRO/2002.

2002/2-9

SISTEMA DE APOIO PARA OTIMIZAÇÃO DAS ATIVIDADES DE SUPORTE TÉCNICO DE UMA EMPRESA DE DESENVOLVIMENTO DE SOFTWARE, UTILIZANDO RACIOCÍNIO BASEADO EM CASOS

CARLOS EDUARDO DE SOUZA E SILVA

ESTE TRABALHO DE CONCLUSÃO DE CURSO FOI JULGADO ADEQUADO PARA OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA DE TRABALHO DE CONCLUSÃO DE CURSO OBRIGATÓRIA PARA OBTENÇÃO DO TÍTULO DE:

BACHAREL EM CIÊNCIAS DA COMPUTAÇÃO

_____

Prof. Roberto Heinzle — Orientador na FURB

_____

Prof. José Roque Voltolini da Silva — Coordenador do TCC

BANCA EXAMINADORA

_____

Prof. Roberto Heinzle

_____

Prof. Jomi Fred Hübner

_____

Prof. Dr. Oscar Dalfovo

AGRADECIMENTOS

Agradeço a Deus por ter me dado uma família, onde fui bem cuidado e sempre tive o que precisei, principalmente amor e carinho.

Agradeço a toda minha família, em especial a meus pais que sempre me ajudaram em toda minha vida, muitas vezes se privando de seu próprio conforto, para que eu pudesse estudar, sem eles certamente eu não teria conquistado mais esta vitória.

Ao meu orientador, Roberto Heinzle pelo auxílio no desenvolvimento deste trabalho. Também a todos os demais professores que se empenharam ajudar os alunos a adquirir conhecimento no decorrer do curso.

A todos os meus amigos que de uma forma ou de outra contribuíram para a realização deste trabalho, em especial à minha grande amiga Silvanira Cervi, também a amigos como: Alan Augusto Lira, Cristina, André, Simone entre outros que contribuíram de forma expressiva na realização deste trabalho.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Este trabalho de conclusão de curso tem por objetivo um estudo sobre sistemas de informação e aplicação dos conceitos de raciocínio baseado em casos (RBC). Sistemas de RBC simulam o ato humano de relembrar um episódio prévio para resolver um determinado problema em função da identificação de afinidades entre os mesmos, para tanto se utilizou uma fórmula matemática para calcular a similaridade. Trata-se sobre o domínio do problema, aquisição, representação, indexação e recuperação dos casos. Fornece-se uma visão geral da técnica de RBC empregada. Para exemplificar estes conceitos especificou-se e implementou-se um sistema aplicado a uma empresa de desenvolvimento de sistemas com objetivo de ajudar o técnico da área de atendimento ao usuário no fornecimento de soluções para problemas com software.

ABSTRACT

This work is a study related to Case Based Reasoning (CBR) in problems solving. CBR systems simulate the human act of remembering a previous episode in order to solve a determined problem according to the identification of affinities between them. It´s used a mathematics formula to calculate the similarity on it. This work presents explanations to control the problem, cases and its representation, and also a general technique view on using the system (CBR). To show the concepts it was specified and implemented a system for a developing system enterprise so that the on call technician can be helped with solutions for problems in softwares.

1 INTRODUÇÃO

1.1 APRESENTAÇÃO

Esta monografia é o resultado de estudos realizados na área de IA para tentar deixar mais fácil o trabalho realizado pelos técnicos da área de suporte a usuários.

Segundo Delpizzo (2002), a Inteligência Artificial pode ser definida como "o campo de estudo da ciência que persegue a meta de fazer o raciocínio do computador similar ao raciocínio humano" O objetivo de desenvolver modelos e sistemas capazes de emular aspectos do comportamento inteligente do ser humano existe desde o surgimento dos primeiros computadores.

Os pesquisadores da Ciência da Computação vêm utilizando características do ser humano, como memória, aprendizado, raciocínio dedutivo e raciocínio analógico modelando-os ao longo das últimas três décadas em diferentes ferramentas da Inteligência Artificial (IA). Para dar inteligência ao computador, os cientistas da Inteligência Artificial desenvolveram programas que contém conhecimento sobre determinada área, denominados de Sistemas Especialistas (SE) (Delpizzo, 2002).

Observando-se que na maioria das empresas que desenvolvem sistemas, é comum a preocupação voltada somente com o desenvolvimento dos sistemas para seus clientes, sistemas de apoio ao técnico da área de suporte ao cliente representaria um diferencial tanto para o técnico que contaria com o apoio de uma ferramenta para auxiliá-lo no fornecimento de soluções, como para a empresa que aumentaria a rapidez e qualidade no atendimento de seus clientes.

Para tanto propõe-se a criação de um sistema tipo *help desk*, que utiliza a técnica de Raciocínio Baseado em Casos (RBC). Esta técnica possibilita armazenar o conhecimento de um especialista da área de suporte, possibilitando definir procedimentos a serem aplicados em função de uma dada situação. Mais especificamente a técnica de recuperação utilizada foi a do vizinho mais próximo. Nesta técnica utiliza-se uma soma ponderada das características entre

um novo caso e um armazenado no banco de dados. A aplicação da técnica de RBC em sistemas de informações agiliza a busca de soluções para problemas. Estas soluções podem ser respostas para as dúvidas dos clientes com relação a procedimentos ou a correção de problemas. O processo de similaridade em sistemas de RBC refere-se à comparação do caso de entrada com os casos que constam na base de casos do sistema. Esta avaliação é executada no nível dos atributos, associando-se valores cuja natureza determina a função de combinação a ser empregada (Abel, 1996).

1.2 OBJETIVOS

1.2.1 OBJETIVO GERAL

O objetivo do trabalho é desenvolver de um sistema de apoio ao departamento de suporte técnico de uma empresa que desenvolve sistemas (*softhouse*), visando informatizar suas atividades, no que diz respeito ao registro de chamadas, agilizar o atendimento ao cliente, utilizando recursos da IA para melhorar a qualidade no processo de manutenção e suporte.

1.2.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos do trabalho são:

a) registrar as chamadas dos clientes;
b) agilizar o atendimento ao cliente;
c) armazenar o conhecimento adquirido pelo técnico da área;
d) utilizar o sistema para treinamento de novos técnicos;
e) facilitar os serviços do técnico da área de suporte técnico.

1.3 MOTIVAÇÃO E IMPORTÂNCIA DO TRABALHO

Como o orientando tem alguma experiência com a atividade de suporte técnico, propôs-se desenvolver um sistema para esta área. Procurou-se desenvolver um TCC que fosse capaz de abranger tanto o lado acadêmico como o profissional.

A preocupação em encontrar um diferencial para o sistema levou o orientando a considerar a idéia de aplicar ao sistema conceitos de inteligência Artificial. Surgiu então a

idéia de desenvolver um sistema tipo *hep desk* utilizando RBC, para auxiliar o técnico de suporte na busca de soluções para problemas dos clientes. A abordagem de RBC proposta no presente trabalho origina-se do fato de que esta técnica representa computacionalmente “a ação humana de relembrar uma experiência anterior ao deparar-se com uma nova situação muito semelhante e conduzir a situação nova a partir da informação e conhecimento contidos na experiência passada” (Lee, 2002).

Além de auxiliar o técnico na busca de soluções para problemas, a utilização de técnicas de Inteligência Artificial, em geral, proporciona benefícios importantes: um deles é o de captar para dentro de um sistema, o "*know-how*" de vários especialistas. O especialista adquire conhecimento partindo de um aprendizado formal teórico e, agregado à esta experiência, modela o conhecimento até se tornar um especialista, baseado nas limitações das experiências vivenciadas. O poder de agregar experiências de vários especialistas torna-se então, um dos grandes benefícios da IA.

1.4 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado em 8 capítulos, sendo que o capítulo 1, apresenta uma introdução à organização e ao sistema em geral, os objetivos, a motivação e a estrutura do mesmo.

No capítulo 2 apresenta-se uma fundamentação ao tema Sistemas de Informação, demonstrando alguns conceitos básicos, porém necessários para o seu entendimento.

No capítulo 3 mostra-se uma visão geral sobre sistemas *help desk* e a área de suporte técnico, como funciona o atendimento ao cliente e detalhes do que se pretende fazer com o sistema proposto.

No capítulo 4 detalha-se a técnica de RBC, trata-se sobre definição, aquisição, representação, indexação dos casos, o ciclo de um sistema de RBC e todas as etapas necessárias para seu desenvolvimento.

No capítulo 5 relata-se as ferramentas utilizadas para especificar e desenvolver o sistema, tais como: análise essencial de sistemas, ferramenta CASE, banco de dados Access, a ferramenta Genexus e Visual Basic 6.0.

No capítulo 6 trata-se da especificação e modelagem do sistema, o que o sistema se propõe, que técnicas foram utilizadas em relação ao RBC e ainda neste capítulo define-se a modelagem do sistema em questão, identificando processos, fluxos, eventos e dados presentes no sistema.

No capítulo 7 trata-se da implementação do sistema em si, mostrando as telas do mesmo e detalhando-as, identificando onde estão os princípios do RBC.

E para finalizar, encontra-se no capítulo 8 conclusões, sugestões para continuidade em trabalhos futuros.

2 SISTEMAS DE INFORMAÇÃO

De acordo com Dalfovo (2000), “está havendo uma substituição da sociedade industrial pela sociedade da informação, na qual o principal recurso não é mais o capital e sim, a informação”. Quem está fazendo esta mudança na nova sociedade é a tecnologia da informação. Anteriormente as empresas, para conquistar uma fatia de mercado, utilizavam o capital como um “trunfo” e investiam cada vez mais em recursos para aumentar e melhorar a qualidade de seus produtos, caracterizando assim a competitividade entre elas. Nos tempos atuais o capital perdeu seu lugar no mercado competitivo para a informação, onde o uso da inteligência da tecnologia da informação é voltado para a obtenção de resultados.

Torna-se uma necessidade para as organizações a missão de administrar as informações, “porque existe uma crescente demanda e sofisticação na tecnologia da informação de *software* e *hardware*, em que esse recurso será de vital importância para a sobrevivência das empresas” (Dalfovo, 2000).

O propósito básico da informação é o de habilitar a empresa a alcançar seus objetivos pelo uso eficiente dos recursos disponíveis, nos quais se inserem as pessoas, materiais, equipamentos, tecnologia, dinheiro, além da própria informação. Os sistemas de informação através da geração de informações de caráter decisorial contribuem para a eficácia do executivo no exercício das suas funções de planejamento, organização, direção e controle na gestão das empresas (Oliveira, 1996).

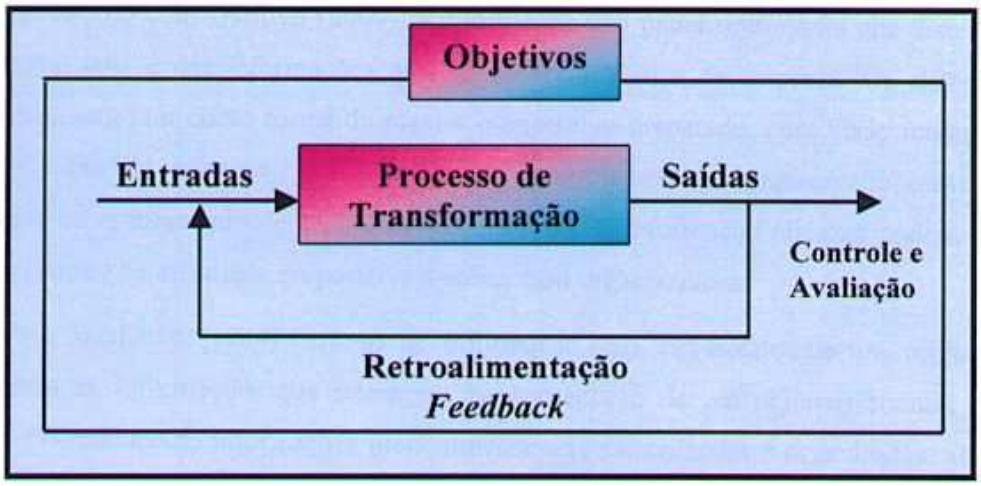
2.1 DEFINIÇÃO DE SISTEMAS DE INFORMAÇÃO

Quando se procura por definição de sistema de informação, são encontradas várias definições, dos mais diversos autores, mas todos eles tratam este sistema como de estrema importância para o executivo na tomada de decisões. Entre várias definições serão destacadas as seguintes:

Conforme Yourdon (1990), um sistema pode ser definido como sendo “um conjunto organizado de doutrinas, idéias ou princípios, habitualmente previsto para explicar a organização ou funcionamento de um conjunto sistemático”. Na concepção de Oliveira (1996), “é um conjunto de partes integrantes e interdependentes que conjuntamente formam um todo unitário com determinado objetivo e efetuam determinada função”. Segundo a

definição de Stair (1998), “é um conjunto de elementos ou componentes que interagem para se atingir objetivos”. Os componentes de um sistema são as entradas, o processamento e as saídas, visualizados conforme Figura 1.

Figura 1: Ambiente de um sistema



Fonte: Adaptado de Dalfovo (2000, p.17).

O sistema de informação dentro de uma organização seja ela de fins lucrativos ou não, é de suma importância. Através dele é possível obter um melhor gerenciamento de todos os processos organizacionais, bem como, obter uma melhor administração de todos os recursos disponíveis. Na atual conjuntura, os sistemas de informação tornam-se indispensável para obtenção de resultados positivos e bem elaborados, dentro de um planejamento estratégico para que a organização possa atingir seus objetivos com satisfação e sucesso (Dalfovo, 2000).

2.2 TIPOS DE SISTEMAS DE INFORMAÇÃO

Os Sistemas de Informação foram divididos de acordo com as funções administrativas, que pelas suas características próprias, foram sendo tratadas de uma forma individualizada, resultando na criação de vários sistemas para ajudarem os executivos, nos vários níveis hierárquicos, a tomar decisões.

Os tipos de sistema de informação segundo Stair (1998) são:

a) **Sistema de Processamento de Transações**: Desde sua primeira geração os computadores têm o objetivo de facilitar o trabalho humano, e reduzir custos. Isto era feito pela automatização de muitas rotinas e sistemas empresariais de trabalho

intenso, um dos primeiros sistemas empresariais a ser computadorizado foi o de folha de pagamento. Como estes primeiros tratavam e processavam trocas diárias de negócios ou transações, foram chamados de sistemas de processamento de transações. Estes sistemas representam a aplicação dos conceitos e tecnologia da informação em transações rotineiras, repetitivas e geralmente comuns de negócios.

b) **Sistemas de Informações Gerenciais**: Um sistema de informações gerenciais (SIG), é um agrupamento organizado de pessoas, procedimentos, banco de dados, e dispositivos usados para fornecer informações de rotina aos administradores e tomadores de decisões. O SIG focaliza a eficiência operacional. Marketing, produção, finanças e outras áreas funcionais são apoiadas pelos sistemas de informações gerenciais e ligadas através de um banco de dados comum. Os sistemas de informações gerenciais fornecem relatórios pré-programados gerados com dados e informações do sistema de processamento de transações.

c) **Sistemas de Apoio a Decisão**: Um sistema de apoio a decisão (SAD) é um grupo organizado de pessoas, procedimentos, banco de dados e dispositivos usados para dar apoio a tomada de decisões referentes a problemas específicos. O foco do SAD é a eficácia da tomada de decisão, o SAD ajuda o administrador a tomar a decisão certa, enquanto um SIG ajuda a organização à “fazer a coisa direito”. Os sistemas de apoio à decisão são usados quando o problema é complexo e a informação necessária à melhor solução é difícil de ser obtida e usada.

d) **Inteligência Artificial e Sistemas Especialistas**: As organizações freqüentemente usam sistemas baseados na noção de Inteligência Artificial (IA), na qual um sistema de computador toma as características da inteligência humana. O campo da IA tem vários subcampos, e os sistemas especialistas são um deles. Um sistema especialista (SE), é um sistema de informações que pode fazer sugestões e podem chegar a conclusões de um modo bem semelhante ao de um profissional especialista.

2.3 BENEFÍCIOS DOS SISTEMAS DE INFORMAÇÃO

Um sistema de informação eficiente pode ter um grande impacto na estratégia corporativa e no sucesso da organização. Este impacto pode beneficiar a organização, os usuários do sistema de informação e qualquer indivíduo ou grupo de que interagir com o

sistema de informação. De acordo com Stair (1998), entre os benefícios que as empresas procuram obter através dos sistemas de informação estão:

1) vantagens competitivas;
2) menos erros;
3) produtos de melhor qualidade;
4) maior eficiência e produtividade;
5) custos reduzidos;
6) administração mais eficiente;
7) maior e melhor controle sobre as operações
8) tomada de decisões gerenciais superiores;
9) tomada de decisões financeiras superiores;
10) mais oportunidades.

Mais informações sobre o tema sistemas de informação podem ser encontradas em: Dalfovo(2000), Oliveira(1996), Stair(1998).

3 SISTEMAS *HELP DESK* E A ÁREA DE SUPORTE TÉCNICO

Neste capítulo apresenta-se um pouco sobre sistemas *help desk* e sobre a área de suporte técnico nas empresas.

Segundo Ferreira (1999), suporte é “o que suporta algo, que dá apoio”, e técnico, “peculiar a uma arte ou ciência, perito em determinada função”, trazendo esta definição para a área de informática, suporte técnico representa um grupo de pessoas que dão apoio aos usuários de um determinado sistema, (software), ou a algum produto específico da empresa, em outras palavras eles ajudam os usuários quando estes têm dúvidas ou problemas em relação ao produto ou serviço que adquiriram da empresa fornecedora.

A vida do consumidor de tecnologia é dura, as queixas dos consumidores são pesadas na área de tecnologia. A área de suporte técnico vem sendo motivo de descontentamento para muitos consumidores de tecnologia, que muitas vezes se revoltam com serviços de má qualidade e totalmente desestruturados.

Por mais que o sistema de atendimento seja falho, as pessoas devem entender que do outro lado da linha ou do correio estão pessoas, que dão duro, penam para ter respostas na ponta da língua, as vezes com treinamento precário e estão longe de ganhar salários milionários, explosões de raiva no e-mail ou no telefone raramente ajudam; Paciência e gentileza talvez seja a chave para solução.

Os clientes estão cada vez mais críticos quanto ao tratamento que recebem, a preocupação com a qualidade dos serviços prestados tem aumentado cada vez mais, não somente pelo aumento da competitividade entre as empresas, mas também porque o desempenho das organizações está diretamente ligado a qualidade das informações, dos produtos e serviços prestados.

Para Cougo (2002), atualmente a grande maioria das ferramentas voltadas ao ambiente de suporte técnico ou *help desk* para a área de informática tem focado sua atuação fortemente nos aspectos de disponibilização de recursos para a preservação e recuperação da estrutura técnica. Desse modo passam a estar disponíveis nestas ferramentas meios para que:

1) o acervo de hardware e software seja controlado;
2) os diagnósticos e execução de procedimentos de reparo sejam executados, inclusive remotamente;
3) a distribuição de novos recursos na rede possa ser automatizada;
4) o gerenciamento das intervenções solicitadas possa ser executado.

Neste panorama se enquadram praticamente a totalidade dos softwares voltados à implementação de um *help desk* convencional para a área de informática. Porém estes sistemas ficam restritos a alguns fatores que levam ao bom atendimento das necessidades de suporte, são eles:

1. a disponibilidade de técnicos;
2. a disponibilidade de conhecimento específico dos técnicos;
3. a possibilidade de intervenções remotas, sem o deslocamento dos técnicos;
4. os dados sobre as características do equipamento que apresentou o problemas e o histórico de problemas desse equipamento;

Mas o que fazer quando:

1. técnicos estiverem ausentes?
2. técnicos estiverem ocupados com a demanda de maior prioridade?
3. problemas repetitivos estiverem sendo canalizados à equipe de suporte?
4. procedimentos de resolução estiverem sendo “gerados” a cada nova ocorrência?

A produtividade no atendimento de chamados e suporte técnico operacional depende basicamente o fator “conhecimento”. Conhecimento de soluções, conhecimento de alternativas, conhecimento de procedimentos, etc. Entretanto, isto não significa que a área de suporte técnico de uma empresa precise ficar “nas mãos” das pessoas que detenham este conhecimento, estando elas presentes ou ausentes, teoricamente a estabilidade e a continuidade de atendimentos deve ser preservada. Mas como conseguir isso, se a “base de conhecimento” de um técnico está intimamente ligada as suas experiências vividas no dia-a-dia?

Em resposta a esta questão, é que o sistema proposto possa buscar meios para que o conhecimento adquirido por um técnico possa ser explicitado, mantido e disponibilizado em

uma “memória de casos”, visando ser uma ferramenta para a disponibilização de conhecimento, que atue no auxílio ao técnico da área, no fornecimento de soluções para problemas repetitivos.

Uma ferramenta com esta capacidade pode representar um diferencial em termos de qualidade no processo de manutenção e suporte técnico de um ambiente corporativo. A principal contribuição da utilização de RBC, nesta situação, é auxiliar a corrigir os problemas que impedem o usuário de desempenhar normalmente suas tarefas, visando a redução do tempo no atendimento ao cliente fornecendo um serviço de maior qualidade, proporcionando assim uma maior satisfação do cliente em relação ao serviço prestado pela área de suporte técnico.

4 RAIOCÍNIO BASEADO EM CASOS

Raciocínio Baseado em Casos (RBC) ou do Inglês CBR que significa *Case Based Reasoning*, é uma tecnologia de representação e processamento de conhecimento que utiliza a experiência passada para resolver problemas. A idéia é descrever e acumular descrições de "casos" na área do conhecimento especializado e tentar descobrir, por analogia, quando um determinado problema é similar a um outro já resolvido. Desta forma a solução já aplicada ao problema pode ser utilizada novamente ou adaptada para o caso em questão. Portanto o princípio básico no qual se fundamentam os sistemas de RBC é o armazenamento organizado de problemas com suas soluções e sua utilização para resolver novos problemas similares aos já resolvidos.

Um sistema de RBC recupera o caso cuja descrição mais se aproxima daquela do problema apresentado e adapta-o (modifica-o) para ajustar-se ao problema e apresentar a solução mais adequada, o novo caso, assim modificado é apreendido pelo sistema e passa a fazer parte do banco de casos disponíveis.

4.1 HISTÓRICO

Segundo Watson[1], *apud* Silva (1999), a história do RBC começa com a investigação do filosofo Wittgentein em 1953 e leva ao trabalho de Roger Schank em memória dinâmica, em 1982.

Conforme Abel (1996), o estudo de CBR tem seu marco inicial na teoria da memória dinâmica de Schank, que foi uma das importantes contribuições para a pesquisa na área. A teoria baseou-se na idéia de que não é possível separar experiência, compreensão, memória e aprendizado, e propôs o conceito de pacotes de organização de memória ou MOPs (*Memory Organization Packets*). Eles utilizam a lembrança de experiências passadas associadas a estereótipos de situações para a solução de problemas e aprendizado.

Embora as raízes filosóficas da teoria de RBC possam se espalhar por trabalhos de diversos pesquisadores no campo da psicologia, medicina ou ciência da computação, foram sem dúvida os trabalhos do grupo de Schank, na universidade de Yale, no inicio dos anos 80

1 WATSON, Ian. "The Case for Case-Base Reasoning". Disponível em: <http://turing.une.edu.au/~jirapun/watson.html>. Acesso em: 01 abr. 1999.

que produziram o modelo cognitivo de RBC e as primeiras aplicações baseadas nesse modelo. Pesquisadora desse grupo Janet Kolodner, desenvolveu o primeiro sistema utilizando RBC em 1983, chamado CYRUS. O sistema continha as viagens e encontros do ex-secretário do estado dos EUA, Cyrus Vance, descritos na forma de casos e implementado com os MOPs de Schank. O modelo de casos do sistema CYRUS serviu como base para o grupo de Yale desenvolver diversos outros sistemas de RBC. Esses trabalhos e conceitos evoluíram rapidamente para inúmeras aplicações de sistemas baseados em casos, especialmente nos domínios de direito, medicina e engenharia (Abel, 1996).

4.2 DEFINIÇÃO

Para Heinrich (2000), o ser humano ao tentar compreender o que está vendo e ouvindo, busca em sua memória algo que possa ajudá-lo nesta compreensão, ou seja, ele se recorda de algo que já foi compreendido no passado que lhe é útil para compreender a situação atual. De acordo com Abel (1996), um sistema de RBC age de forma semelhante. Ele visa usar os resultados dos casos passados para analisar ou resolver um novo caso. Os problemas a serem resolvidos tendem a ser recorrentes e repetir-se com pequenas alterações em relação a sua versão original. Desta forma é possível reaplicar soluções anteriores com pequenas modificações.

Conforme (Lee, 2002), Raciocínio Baseado em Casos, é uma técnica de Inteligência Artificial que reproduz aspectos da cognição humana para resolver problemas especialistas. Os sistemas de RBC simulam o ato humano de relembrar um episódio prévio para resolver um determinado problema em função da identificação de afinidades entre os mesmos.

Quanto maior o número de casos de sucesso armazenados e eficientemente indexados, maior será a chance de que um novo caso possa ser tratado com a mesma solução ou com uma pequena adaptação de uma solução já utilizada. O primeiro passo para utilizar uma solução já aplicada com sucesso anteriormente é determinar qual das experiências passadas mais se assemelha ao problema atual. Para ser possível realizar esta comparação é necessário que as experiências sejam analisadas e armazenadas de forma organizada (Abel, 1996).

De acordo com Delpizzo (2002), muitas das inspirações originais para o desenvolvimento do RBC surgiram dos conceitos de memória do raciocínio humano de

Schank, o entendimento da técnica de RBC está implícito em assumir alguns princípios da natureza do mundo:

1) regularidade: O mundo é na maioria das vezes regular, as ações executadas nas mesmas condições tendem a ter os mesmos, ou similares, resultados. Conseqüentemente, soluções para problemas similares são utilizáveis para o início da resolução de outro;
2) tipicalidade: os tipos de problemas tendem a repetir; as razões para as experiências são provavelmente as mesmas para as futuras ocorrências;
3) consistência: Pequenas mudanças ocorridas no mundo requerem apenas pequenas mudanças na maneira como nós interpretamos o mundo, e conseqüentemente, pequenas mudanças nas soluções de novos problemas;
4) facilidade de adaptação: As coisas não se repetem exatamente da mesma maneira; as diferenças tendem a ser pequenas e pequenas diferenças são fáceis para compensar;

4.3 CASOS

Caso é uma descrição completa de um problema do domínio com a respectiva solução aplicada, mais uma avaliação da eficácia dessa solução. É descrito através da enumeração e qualificação e valores dos objetos que ocorrem naquele episódio. O conjunto dos casos é indexado por seus atributos mais significantes de forma a agilizar a busca e recuperação de casos similares (Abel 1996).

Em sistemas de RBC a forma para representar e armazenar a experiência é através de “casos”, é o caso que guarda todos os atributos e as características relevantes desse evento passado. Dessa forma a recuperação desse caso, também se dará em função dessas características, de acordo com a combinação, no futuro, da especificação do novo problema com as características do caso armazenado em uma Base de Casos. Essas características servirão de índices para recuperação do caso.

Para Abel (1996), caso é o que representa o conhecimento associado a uma determinada situação, é tornado explícito como uma determinada tarefa foi executada e que estratégias foram utilizadas para se atingir o objetivo. Em um sistema para a área médica um

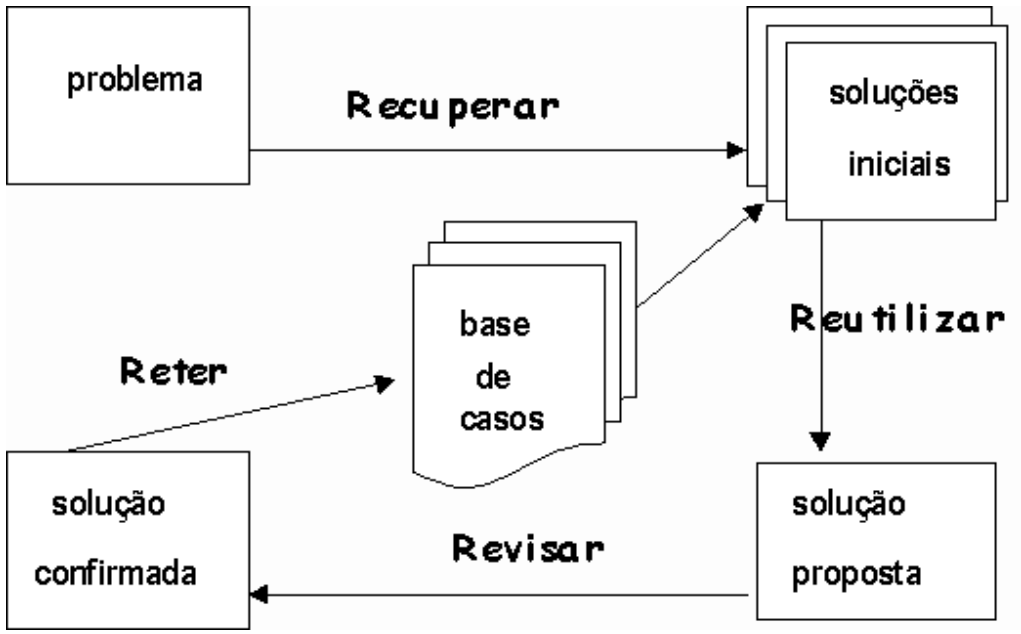
caso seria a descrição de um paciente e seu diagnóstico, essa descrição irá incluir as características e sintomas relativos à doença e irá omitir os que não forem. Por exemplo, se o diagnóstico do paciente for fratura do fêmur, o médico não irá incluir o hábito de fumar como uma característica relevante do caso, mas se o problema for um enfarto essa característica se torna muito importante.

Um caso pode assumir diferentes formas de representação. O exemplo mais simples de um caso é uma experiência descrita através de atributos devidamente valorados. O caso está modelado para o sistema RBC somente quando indexado. A falta de indexação não descaracteriza um caso. Assim, um texto pode ser um caso, uma idéia, um fato qualquer. Deve-se garantir a contextualização do mesmo e a modelagem adequada para sua utilização computacional (Alves Júnior, 1998).

4.4 CICLO DE UM SISTEMA DE RACIOCÍNIO BASEADO EM CASOS

De acordo com Abel (1996), “um sistema de RBC funciona por comparar a descrição de um problema a ser resolvido com os caso descritos em uma base de casos”. O sistema recupera então o caso mais parecido avalia a necessidade de adaptar a solução associada e aplica essa solução ao novo problema. Esse procedimento pode ser representado através de um ciclo. A figura 02 apresenta o ciclo de um sistema de RBC proposto por Lee (2002), em seguida cada uma destas etapas será detalhada.

Figura 02: Ciclo de um sistema RBC



Fonte: (Lee,2002).

1) **recuperar:** é o processo de retornar um ou mais casos da base de casos em resultado à comparação de um novo caso (caso alvo) com cada um dos casos da base (casos candidatos). Esta comparação é feita através de uma avaliação de similaridade. O resultado desta comparação é a seleção de um caso (ou uma combinação de casos) que sugere uma solução ao caso alvo;
2) **reutilizar:** é a etapa pertinente ao aproveitamento do conteúdo presente no caso recuperado (adaptado ou não) no sentido de resolver o caso alvo;
3) **revisar:** Nesta etapa a solução proposta é avaliada;
4) **reter** refere-se à adição desta nova experiência ou das experiências que inicialmente compõem a memória de conhecimento, podendo a adição de novos casos representar um mecanismo de aprendizagem.

De acordo com Aamodt[2], *apud* Silva (1999), a utilização de RBC envolve quatro passos que devem ser observados:

a) construir modelos que representem de forma satisfatória a abstração dos problemas ou objetos envolvidos;
b) selecionar um problema ou objeto conhecido a fim de compará-lo a um desconhecido;
c) mapear os atributos do problema ou objeto conhecido com o desconhecido;
d) estender o mapeamento com o objetivo de gerar uma solução que seja considerada válida para ser aplicada ao desconhecido.

4.5 ETAPAS DE DESENVOLVIMENTO DE UM SISTEMA DE RBC

O desenvolvimento de um sistema de RBC é uma tarefa variável e não se encaixa facilmente em uma metodologia genérica. Mesmo assim tenta-se apresentar algumas das principais etapas que devem ser realizadas no desenvolvimento de um sistema de RBC, são elas:

1) representação dos casos;
2) recuperação dos casos;
3) adaptação dos casos;
4) retenção dos casos ou Aprendizagem.

Pode-se imaginar este conjunto de etapas como áreas de problemas a serem modelados no desenvolvimento de um sistema de RBC. Entretanto algumas destas etapas são dispensáveis, dado que se pode desenvolver um sistema de RBC que não realize adaptação ou aprendizagem (retenção do caso). Para estar se tratando de um sistema de RBC, basta que seja utilizado o paradigma da reutilização de uma experiência passada para resolver, ou diagnosticar uma experiência similar atual.

2 AAMODT, Agnar; PLAZA, Enric. “Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches”. 3 jul. 1996. Disponível em: <http://www.iiia.csic.es/People/enric/AICom.html>. Acesso em: 19 mar. 1999.

4.5.1 REPRESENTAÇÃO DOS CASOS

Um caso é uma parte contextualizada de um problema que representa uma valiosa experiência de onde se pode tirar boas lições no futuro. Determinar o que é um caso, é o primeiro problema na modelagem do RBC (Lee, 2002). São os casos que contém elementos para que a solução do problema proposta seja alcançada.

Esta seção é composta por:

1) Modelagem dos Casos;
2) Modelagem de memória;
3) Indexação.

4.5.1.1 MODELAGEM DOS CASOS

De acordo com Abel (1996), “a aquisição de casos pode ser uma tarefa quase tão complexa quanto à construção de modelos, um sistema baseado em casos só é viável se houverem casos corretos, completos e representativos do domínio e em número adequado para serem coletados e organizados com o mínimo de intervenção do especialista”.

Uma medida da disponibilidade dos casos pode indicar o grau de dificuldade na construção de um sistema. Os casos registram experiências concretas que podem auxiliar a alcançar um determinado objetivo, porém nem todos os casos devem ser selecionados para serem incluídos no sistema. Apenas os casos que ampliam a capacidade de resolução de problemas devem ser armazenados, isto é casos que repetem uma situação anterior apenas com pequenas modificações não deveriam ser incluídos, já que estas diferenças podem ser compensadas pelos algoritmos de adaptação. Ao mesmo tempo não devem divergir muito do problema resolvido pelo sistema, sob o risco de o sistema conter casos que nunca serão utilizados (Abel 1996).

Na representação dos casos há dois componentes básicos: a descrição do problema e a descrição da solução. A descrição do problema é realizada através da atribuição de características que descrevem o problema de entrada. A descrição da solução consiste em determinar quais características descrevem a solução do caso, apontando a solução do problema de entrada e informando qual o resultado da aplicação desta solução ao problema de

entrada. A representação dos casos também se refere ao formalismo a ser adotado no programa. Muitos formalismos da IA podem ser empregados, tais como *frames*, representação tipo formulário e redes semânticas. A escolha adequada depende da consideração de tópicos pertinentes à aquisição e às demais etapas do processo de desenvolvimento, tais como recuperação, adaptação e aprendizagem (Lee, 2002).

De acordo com Abel (1996), o formato mais utilizado para se representar casos é o de objeto ou tabelas do modelo relacional, embora muitos sistemas utilizem mais de uma forma de representação combinadas, além desse formato faz parte da tarefa de representação definir como os casos serão organizados e ligados entre si, compondo a memória de casos.

O problema de representação em RBC refere-se, fundamentalmente, em o que guardar de um caso, encontrando uma estrutura apropriada para descrever o conteúdo do caso e decidindo como a memória de casos deve ser organizada e indexada para uma efetiva recuperação e reutilização.

4.5.1.2 MODELAGEM DE MEMÓRIA

Dentre os componentes de um sistema RBC a memória de casos é um dos mais importantes. Esta memória é formada pelas experiências na resolução de problemas pelos especialistas. Cada experiência representa um caso, esses casos devem ser identificados (indexados) pelo que eles têm de útil para que só seja recuperado no momento certo.

Para Abel (1996), dois modelos de memória de casos influenciaram historicamente o modo como os casos são organizados na memória de casos, vistos a seguir:

a) **modelo de memória dinâmica:** Criado por Schank em 1982, é composto de pacotes de organização de memória, que são *frames* que compõem uma unidade básica de memória dinâmica. Este modelo é chamado de dinâmico porque novos pacotes de organização de memória são criados no momento da inserção de novos casos. Para diferenciá-los dos anteriormente armazenados;
b) **modelo de categoria de exemplares:** desenvolvido por Portes e Bareiss em 1986, considera que os casos do mundo real podem ser vistos como exemplares de acontecimento. Cada caso é associado a uma categoria e suas feições têm importância para enquadra-lo ou não na categoria. Para armazenar um novo caso, é

buscado um caso semelhante na memória de casos. Se houver pequenas diferenças entre os dois, apenas um é armazenado, ou é feita uma combinação dos dois.

4.5.1.3 INDEXAÇÃO DOS CASOS

Para Abel (1996), indexar casos corresponde a atribuir índices aos casos de forma a facilitar sua recuperação. Isto inclui colocar rótulos nos casos no momento de sua inclusão na base de casos, para que possam ser posteriormente recuperados, organizar os casos para facilitar a busca e recuperação e definir os algoritmos de recuperação mais eficientes.

A indexação é a essência do raciocínio baseado em casos porque orienta a avaliação da similaridade. O conjunto de descritores que são usados como índices modelam a resposta para a pergunta, "*o que faz um caso ser similar a outro*?" - representando a relevância dos casos. A indexação determina o que deve ser comparado entre os casos para avaliar sua similaridade no intuito de recuperar casos que conduzam à tarefa principal - permitir a recuperação dos casos mais úteis para resolver ou interpretar o novo caso (Lee, 2002).

Segundo Abel (1996), indexar casos depende da compreensão do conteúdo e finalidade da informação que eles armazenam. Um bom índice permite reconhecer similaridades úteis entre os casos recuperados, e essa utilidade só pode ser percebida se os índices forem escolhidos com base em uma boa compreensão do problema.

Kolodner (1993), aponta algumas qualidades necessárias para um bom índice:

a) Prever a futura utilização da informação para solução de diferentes problemas;
b) Endereçar as similaridades úteis entre os casos;
c) Ser abstrato o suficiente para tornar um caso útil em uma variedade de diferentes situações;
d) Ser concreto o suficiente para ser facilmente reconhecido em futuras situações.

As características a serem utilizadas como índice devem ser criteriosamente escolhidas para que recupere apenas os casos mais úteis para a solução de um novo caso. Para selecionar as características deve-se primeiramente analisar as tarefas e domínios para descobrir os descritores relevantes que serão utilizados na descrição dos casos. Feito isto, é necessário selecionar entre estes descritores quais serão atribuídos como índices.

4.5.2 RECUPERAÇÃO DOS CASOS

Consiste em fazer uma busca na memória de casos e selecionar quais poderão ser aproveitados, a partir de um problema a ser resolvido (caso de entrada). A busca por casos é feita por algoritmos que selecionam casos com determinada similaridade com relação ao caso de entrada. O resultado da busca é chegar a um conjunto de casos (casos candidatos) cujas características combinem com as do problema de entrada, de acordo com as definições de similaridade estabelecidas. Finalmente um dentre os casos candidatos deve ser escolhido como sugestão para a resolução do caso de entrada (Lee, 2002).

De acordo com Watson[3], *apud* Silva (1999), a base de casos deve estar organizada de tal forma que facilite a recuperação dos casos quando necessário. Os índices devem simplificar o acesso e a recuperação dos casos pertinentes. Geralmente, os casos são armazenados como dados de arquivos em uma estrutura simples, ou dentro de uma estrutura de banco de dados, utilizando-se índices para referenciar os casos. Existem várias técnicas de RBC, como vizinho mais próximo, método de recuperação indutiva, algoritmo de indução, indução guiada por conhecimento, recuperação de padrões, *flat memory*, entre outros. Em aplicações comerciais, atualmente têm-se utilizado a técnica do vizinho mais próximo e o método de recuperação indutiva.

A técnica do vizinho mais próximo baseia-se na comparação entre um novo caso e os casos armazenados no banco de dados utilizando uma soma ponderada de suas características. Para isso é necessário atribuir um peso a cada uma das feições que descrevem o caso e que serão utilizadas na recuperação (Abel, 1996). Técnica esta que será utilizada no desenvolvimento deste trabalho.

A técnica de recuperação indutiva determina que feições são mais eficazes em discriminar casos e utiliza estas feições para gerar uma árvore de decisões que organiza a memória de casos. Esta técnica é eficiente quando os casos são comparados através de uma única feição que determina a solução (Abel, 1996).

As tarefas envolvidas na etapa de recuperação de casos são:

3 WATSON, Ian. “The Case for Case-Base Reasoning”. Disponível em: <http://turing.une.edu.au/~jirapun/watson.html>. Acesso em: 01 abr. 1999.

1) Avaliação e Métrica da Similaridade;
2) Recuperação;
3) Seleção.

4.5.2.1 AVALIAÇÃO E MÉTRICAS DE SIMILARIDADE

A avaliação também é considerada conhecimento especialista no momento em que é o resultado do valor numérico dado para avaliar a similaridade entre os dois casos, número este que representa o conhecimento do especialista. No momento em que se tem a avaliação, deve-se fazer uma segunda aquisição do conhecimento com o especialista para definição dos pesos. Assim, o especialista pode definir qual índice é mais importante. Neste momento, deve-se utilizar algumas heurísticas para fazer aquisição do conhecimento na tentativa de captar qual peso que vai ser suportado, até o momento em que todos os aspectos que envolvem a recuperação estão representando realmente o que o especialista deseja. Todas estas variáveis são interligadas e devem guiar o desenvolvimento da métrica da similaridade (Delpizzo, 2002).

Uma das maneiras de se fazer a aquisição do conhecimento com objetivo de saber o peso dos índices, é solicitar que o especialista faça uma lista em ordem de importância. A similaridade é o ponto crucial de RBC, pois a partir desta etapa, todo processo de raciocínio que fundamenta esta técnica torna-se viável (Lee, 2002).

A métrica de similaridade é uma função que mede numericamente os graus de similaridade entre dois casos. Uma métrica é normalmente necessária em sistemas nos quais os casos são comparados um a um e a medida de sua similaridade é o meio de distinguir entre os casos candidatos similares e não similares. A métrica de similaridade sintetiza a similaridade ao nível de cada atributo através de uma medida da importância que é usada para modelar a relevância de cada atributo na avaliação sintética da similaridade. Conseqüentemente, qualquer cálculo matemático orientado para avaliações sintéticas é válido, tais como a média ponderada e as integrais difusas. Contudo, ainda é possível encarar a avaliação de similaridade como um problema de reconhecimento de padrões dentro do espaço dos casos. Desta forma, algoritmos que realizam estes tipos de avaliação também podem ser usados, como os algoritmos de vizinho mais próximo (Lee, 2002).

4.5.2.2 RECUPERAÇÃO

A partir da comparação dos casos da base com os casos de entrada, dá-se um valor numérico à similaridade, que utiliza a função do cálculo vizinho mais próximo. Conforme Abel (2002), "nesta técnica de recuperação utiliza-se uma soma ponderada das características entre um novo caso e um caso armazenado na base de dados, sendo que cada um dos atributos que compõe o caso possui um peso, de acordo com sua relevância". Abaixo listamos a função de cálculo do vizinho mais próximo segundo Watson[4], *apud* Silva (1999):

$$\text{Similaridade}\ (T, S) = \sum_{i=1}^{n} \mathbf{f}\,(T_i, S_i)\ \mathbf{x}\ \mathbf{W}_i$$

Onde: *T* é o caso de entrada ;

S é o caso da base;

n é o número de atributos de cada caso;

i é um atributo individual ;

f é a função de similaridade para o atributo i nos casos *T* e *S*;

W é o peso dado ao atributo *i.*

Outro exemplo de algoritmo de vizinhança segundo Abel (1996), é aquele utilizado pelo sistema de ReMind (cognitive systems 1992) e mostrado abaixo:

$$\frac{\sum_{i=1}^{n} W_i \text{ x Sim}(f_i^{\,1}\, f_i^{\,R})}{\sum_{i=1}^{n} W_i}$$

Onde: W corresponde ao peso de uma feição i qualquer que descreve o caso, Sim é a função de similaridade e $f_i^{\,1}\ f_i^{\,R}$ são os valores da feição i para o novo caso e o caso recuperado.

4 WATSON, Ian. "The Case for Case-Base Reasoning". Disponível em: <http://turing.une.edu.au/~jirapun/watson.html>. Acesso em: 01 abr. 1999.

A maioria das ferramentas de RBC utiliza algoritmos como estes. Normalmente o resultado deve ser entre zero (0) e um (1), onde zero não tem similaridade e um é exatamente similar.

Nas tabelas a seguir vê-se exemplos de cálculos de similaridade para recuperação de casos:

Tabela 1: Exemplo de casos da base de casos

Casos / Atributos	A	B	C
X1	Raciocínio	Sistemas	Inteligente
X2	Inteligente	Inteligente	Métricas
X3	Análise	Robótica	Similaridade
X4	Casos	Computador	Análise
X5	Baseado	Análise	Prototipagem

Fonte: Varela (1998).

Tabela 2: Novo caso a ser comparado

Atributos / Casos	X1	X2	X3	X4	X5
Caso Novo	Raciocínio	Inteligente	Análise	Casos	Sistemas

Atribuindo 1 para atributos coincidentes e 0 para não coincidentes:

Tabela 3: Valores atribuídos

Atributos / Casos	X1	X2	X3	X4	X5
Caso Novo => A	$1w_1$	$1w_2$	$1w_3$	$1w_4$	$0w_5$
Caso Novo => B	$0w_1$	$1w_2$	$1w_3$	$0w_4$	$1w_5$
Caso Novo => C	$0w_1$	$1w_2$	$0w_3$	$1w_4$	$0w_5$

Considerando todos os atributos com o peso igual a 1, a comparação entre os casos será:

Sim (caso novo, A) = (1+1+1+1+0)/5 = 4/5 = 0,8;
Sim (caso novo, B) = (0+1+1+0+1)/5 = 3/5 = 0,6;
Sim (caso novo, C) = (0+1+0+1+0)/5 = 2/5 = 0,4;

O caso A é o caso mais semelhante, pois é o que mais se aproxima de 1.

4.5.2.3 SELEÇÃO

Seleção é a última etapa da fase de recuperação, ela é mais específica do que a busca pelo conjunto de casos mais similares. A seleção pode ser implementada de várias formas, a mais óbvia é revisar os elementos que determinam a similaridade entre os casos, visando uma comparação mais apurada. A importância desta etapa se dá pelo fato de que ela gera o resultado para a solução do problema, ou seja, será a saída do sistema (Delpizzo, 2002).

4.5.3 ADAPTAÇÃO DOS CASOS

A tarefa final do Raciocínio Baseado em Casos é adaptar a solução associada a um caso recuperado para as necessidades do problema corrente. Normalmente o caso selecionado não casa perfeitamente com a descrição do problema do usuário. Existem diferenças entre o problema do usuário e o caso contido na memória de casos que devem ser levadas em conta. O processo de adaptação procura por diferenças salientes entre as duas descrições e aplica regras de forma a compensá-las (Abel, 1996).

A reutilização da solução dos casos recuperados no contexto dos novos casos tem foco em dois aspectos: as diferenças entre o caso passado e o caso corrente, e que parte do caso recuperado pode ser transferida para o novo caso. Idéia esta fundamentada também por Kolodner (1993), “a etapa de adaptação pode tomar várias formas entre elas: incluir parte do novo caso na solução do caso antigo; e a exclusão de parte da solução do caso antigo; a substituição do valor de algum item, ou a transformação de alguma parte da solução do caso antigo”.

Segundo Abel (1996), existem dois tipos de adaptação:

a) **Adaptação estrutural:** as regras de adaptação são aplicadas sobre a solução armazenada junto aos casos. É o mecanismo possível de ser utilizado quando as soluções associadas aos casos não são bem compreendidas.
b) **Adaptação derivacional:** o algoritmo reusa os algoritmos, métodos ou regras que geraram a solução que consta no banco de casos para gerar uma nova solução para o problema corrente. Neste método a seqüência que construiu a solução original deve ser armazenada juntamente com o caso. Esta adaptação somente poderá ser utilizada para domínios que são bem compreendidos.

De acordo com Abel (1996), definir a melhor forma de adaptação para alguns problemas pode ser uma tarefa extremamente complexa que pode comprometer a confiabilidade do sistema. Em muitas aplicações, a solução é simplesmente apresentar o melhor caso e deixar o problema da adaptação para o usuário do sistema. Esta estratégia vem sendo utilizada na maior parte dos sistemas comerciais.

4.5.4 RETENÇÃO DO CASO (APRENDIZADO)

A retenção do caso é o processo de reter o que é útil do problema resolvido. O aprendizado do sucesso ou das falhas da solução proposta é efetuado depois da avaliação e possíveis reparos. Isto envolve selecionar que informação do caso deve-se reter, a forma de retê-las e, como indexar o caso para posterior recuperação de problemas similares e como integrar o novo caso na estrutura da memória.

De acordo com Lee (2002) a base de casos pode crescer com novos casos a partir de um pequeno conjunto. A geração destes novos casos se dá através de novos casos informados pelos usuários ou a partir de uma fonte externa. Na aprendizagem ao nível dos casos, mantém-se no caso o registro de seu desempenho ao ser utilizado. Desta forma, tanto sucessos como fracassos são informados, incrementando o conhecimento e as lições embutidas no caso. Tal registro também serve para prevenir o usuário com relação às possíveis conseqüências de seu uso. Assim, compensa-se a inclusão de informações no caso, evitando a reutilização de sugestões menos favoráveis, o que resulta no aumento da qualidade da recuperação.

Para Abel (1996), a retenção de novos casos na memória deve acontecer somente quando o novo caso apresentar uma lição útil em relação ao demais casos do sistema, ou seja,

somente os casos que ampliam a capacidade de solução de problemas. Com este processo a atualização do banco de casos é constante e evolutiva possibilitando assim que cada vez mais casos possam ser solucionados por experiências passadas e semelhantes.

4.6 VANTAGENS DE RBC

A técnica de Raciocínio Baseado em Casos diferencia-se das outras técnicas de IA por uma série de características. Estas diferenças terminam por se traduzir em vantagens desta técnica sobre outros métodos equivalentes de resolução de problemas. Abaixo são listadas algumas vantagens segundo Abel (2002):

a) Fornecem uma amostragem significativa do tipo de problemas que o sistema deve resolver;
b) Facilita a aquisição de conhecimento, especialmente em domínios pouco estruturados e muito complexos, mesmo antes de uma perfeita compreensão do domínio;
c) Permitem o reuso de conhecimento armazenado em bancos de dados e outras fontes;
d) Permite o encapsulamento da descrição do conhecimento com a solução aplicada;
e) Realizam a manutenção do banco de conhecimento por aprendizagem automática de novos casos com pouca ou nenhuma interferência de um engenheiro de conhecimento;
f) A reutilização de soluções implementadas geralmente exige menos processamento do que a geração de uma solução através de um modelo.

4.7 APLICAÇÕES DE RBC

De acordo com Abel (2002), as áreas potenciais para aplicação de sistemas de RBC podem ser classificadas em duas grandes classes: tarefas de classificação e tarefas de síntese.

As tarefas de classificação buscam verificar qual a classe do problema em questão para reutilizar uma solução já aplicada. Incluem:

a) Diagnósticos de falhas em equipamentos ou diagnósticos de doenças;
b) Previsão de falhas em equipamentos ou de renovação de estoques;

c) Avaliação, como análise de investimentos de risco, de seguros ou estimativa de custos de projetos;
d) Controle de processos, como controle de processos industriais.

As tarefas de síntese criam uma nova solução por combinar partes de soluções aplicadas anteriormente. São elas:

a) Projeto, um novo produto é criado por adaptar elementos de outros;
b) Planejamento, onde novos planos são criados a partir da combinação de elementos de planos anteriores;
c) Configuração, novas configurações são propostas a partir de modificações de anteriores;

4.8 TRABALHOS CORRELATOS

Dentre os vários trabalhos utilizando RBC encontrados podemos citar os seguintes:

a) Em Heinrich (2001), a tecnologia de RBC foi aplicada no diagnóstico de defeitos em equipamentos eletrônicos aplicado as oficinas eletrônicas. Aplicando-se a técnica de similaridade são pesquisados soluções de problemas com equipamentos anteriormente consertados.
b) Em Varela (1998), a tecnologia de RBC foi aplicada no auxílio à recuperação de projetos do instituto de pesquisas ambientais. Com uso da técnica de similaridade são retornados os projetos que mais se aproximam dos parâmetros informados;

5 FERRAMENTAS E TECNOLOGIAS UTILIZADAS

Para a realização deste trabalho foram utilizadas algumas ferramentas e tecnologias, a fim de especificar e desenvolver o sistema. Neste capítulo alguns itens são abordados: análise essencial, ferramenta CASE, Microsoft Access, a ferramenta Genexus e Visual Basic 6.0.

5.1 ANÁLISE ESSENCIAL DE SISTEMAS

Relata Shiller (1992) que, Análise Essencial de Sistemas relaciona-se diretamente com eventos, eventos estes que causam a reação do sistema, e por sua vez o sistema possui um conjunto de reações, que responderão aos eventos.

Segundo Pompilho (1994), a análise essencial é composta por Lista de Eventos, Diagrama de Contexto, Diagrama de Fluxo de Dados (DFD), Modelo Entidade-Relacionamento (MER físico e lógico) e Dicionário de Dados. Os componentes da Análise Essencial são descritos a seguir:

a) lista de eventos é o primeiro passo na especificação de um sistema, que ajuda a delimitar as fronteiras do problema de que estamos tratando. Então a lista de eventos é uma lista textual dos estímulos no ambiente externo aos quais o sistema deve responder;
b) diagrama de contexto: é construído baseado na lista de eventos, que significa representar o sistema por um único processo e suas interligações com as entidades externas, mostrando as interfaces do sistema com o ambiente em que ele está inserido;
c) DFD: é o diagrama de contexto separado por eventos (processos), este representa os processos e o fluxo de dados entre eles. Os dados fluem de um nódulo de processamento para outro, onde se modificam;
d) MER fornece uma visão simples e gráfica do sistema para os usuários que não necessitam saber dos detalhes funcionais do sistema;
e) dicionário de dados provém do MER que é uma listagem de informações sobre componentes dos sistemas. Os dicionários de dados fornecem a informação em forma de texto a fim de auxiliar a informação gráfica mostrada no DFD.

Mais informações sobre análise essencial de sistemas podem ser encontradas em: Gane(1984), Keller(1990), Martin(1991), Yourdon(1990), Shiller(1992), Demarco(1989).

5.2 FERRAMENTA CASE

Ferramenta CASE, (*Computer Aided Software Engineering*), ou seja, engenharia de software auxiliada por computador, impulsiona o engenheiro de software e os profissionais da área a especificar e projetar o software. Evoluiu a partir das metodologias estruturadas, alcançando ima aceitação comercial maior por volta dos anos 80 quando ficou evidente que ferramentas gráficas tais como diagrama de fluxo de dados (DFD), diagrama de entidade-relacionamento(DER), poderiam ser úteis em análise e projeto de sistemas (Fisher 1990).

Em vários aspectos, as ferramentas CASE, são um desenvolvimento direto das antigas metodologias estruturadas feitas no papel, estas metodologias estruturadas são a espinha dorsal da engenharia de software computadorizada, oferecem a estrutura rigorosa essencial à especificação e ao projeto completo do software de aplicação. Uma das definições de engenharia de software computadorizada é a utilização de ferramentas que oferecem auxílio nas etapas de análise dos requisitos e especificação do projeto de software.

O objetivo principal da tecnologia CASE é separar o projeto da implementação do código, pois uma das principais causas de fracasso nos projetos de software é que muitas organizações não seguem as metodologias de desenvolvimento, pulando algumas de suas etapas, muitas vezes por causa de restrições de orçamento ou pela pressão de se lançar algo novo no mercado, com isso a implementação do código do projeto acaba por vir antes das etapas de análise dos requisitos e especificação, “nunca se tem tempo para fazer o certo, mas sempre se tem para fazer duas vezes!”. A ferramenta CASE surge como uma proposta para remediar estas falhas, e reduzir o risco de fracasso do projeto (Fisher, 1990).

5.3 BANCO DE DADOS ACCESS

O Microsoft Access é um sistema de gerenciamento de banco de dados relacional (SGBD) para a criação de aplicações de banco de dados para Windows e para sistemas cliente-servidor. O Access foi projetado para ser usado tanto como um SGBD isolado em uma estação de trabalho única quanto em um modo compartilhado cliente/servidor em rede,

comportando um sistema de segurança que impede que pessoas não autorizadas visualizem ou modifiquem informações guardadas no banco (Jeninngs, 1997).

O Access utiliza tabelas para armazenar os dados do usuário, sendo assim, o primeiro passo a ser dado é criar todas as tabelas de dados que farão parte do banco de dados, em seguida deve-se criar os relacionamentos entre as tabelas e as planilhas de consulta de dados, as telas de entradas de dados e os relatórios podem ser criados por último.

A base de dados do sistema proposto neste trabalho será gerada automaticamente pela ferramenta de desenvolvimento Genexus utilizando o aplicativo de gerenciamento de base de dados Access. Toda a estrutura definida pela ferramenta Genexus será criada no Access, com suas consistências e relacionamentos entre as tabelas.

Mais informações sobre banco de dados podem ser encontradas em: Chu(1983), Mercado-Gardner(1995) e sobre Microsoft Access em: Jeninngs(1997).

5.4 A FERRAMENTA GENEXUS

Conforme ARTech[6], existem várias ferramentas de desenvolvimento de sistemas que, de alguma maneira, pretendem aumentar a produtividade do desenvolvimento de aplicações.

O esquema tradicional de desenvolvimento de programas consiste em combinar todas as ações e regras do negócio, organizá-los em um algoritmo e depois, programar em alguma linguagem. A programação é procedural.

As primeiras ferramentas foram as 4GL que, apesar de usarem o mesmo esquema procedural, tinham uma forte capacidade de expansão de código, o que permitia escrever muito menos, e também cometer menos erros, para obter os mesmos resultados. O impacto que estas ferramentas tiveram em cima de produtividade de desenvolvimento foi considerável. Porém, o impacto que as 4GL tiveram em cima de custos de manutenção foi muito baixo: estas ferramentas não ofereceram inteligência e, como conseqüência, era impossível uma análise mais ampla do impacto das mudanças na base de dados sobre os programas e muito menos a propagação automática destas mudanças.

Outras ferramentas importantes foram os geradores de código. Neste caso o sistema "entende" as especificações e pode gerar o programa correspondente. As primeiras versões eram muito rígidas, mas com o tempo, foram incorporadas linguagens processadoras de diagrama de ação - conceitualmente linguagens procedurais, muito semelhantes às 4GLs. Estas ferramentas passaram a oferecer níveis altos de produtividade para o desenvolvimento do que as 4GLs.

Com relação à manutenção, o aporte era pequeno, pois os geradores de código dependem da base de dados, onde o diagrama E-R é apenas uma entrada, e modificações nesses diagramas de E-R podem invalidar procedimentos múltiplos. Em outras palavras, estas ferramentas não oferecem inteligência nesta área e, como uma conseqüência, a análise de impacto das mudanças e sua eventual propagação eram muito limitadas.

Embora elas não sejam consideradas como relevantes, do ponto de vista de mercado, existem ferramentas semelhantes às anteriormente mencionadas, que em vez de gerar em código, interpretam as especificações. Isto não muda nada, quando uma especificação perde sua validade devido a mudanças feitas na base de dados, o fato de que são gerados programas ou a própria especificação é interpretada, não é a essência do problema. Um dos problemas que estes tipos de ferramentas têm é que elas não são capazes de propagar automaticamente as especificações das mudanças da base de dados. Infelizmente, elas estão baseadas em uma hipótese incorreta: "a base de dados é estável".

Problemas semelhantes existem com programação orientada a objeto: quando vem a lógica, muito complexa às vezes, porém, que não precisam acessar o banco de dados, com diálogos sofisticados, tudo funciona muito bem. Quando é necessário acesso à base de dados, reaparecem os problemas mencionados acima.

GeneXus é uma ferramenta para desenvolvimento de aplicações. Seu objetivo é auxiliar os analistas de sistema na implementação de aplicações no menor tempo e com a melhor qualidade possível. Em linhas gerais, o desenvolvimento de uma aplicação implica tarefas de análise, projeto e implementação. O caminho seguido por GeneXus para atingir estes objetivos é o de liberar as pessoas das tarefas automatizáveis, por exemplo a implementação, permitindo a elas concentrar-se nas tarefas realmente difíceis e não automatizáveis, como compreender o problema do usuário. Do ponto de vista teórico,

GeneXus é uma ferramenta associada a uma metodologia de desenvolvimento de aplicações. Como metodologia, tem alguns pontos em comum com as metodologias tradicionais, porém com enfoques diferentes em outros. Em comum com as metodologias tradicionais, se mantém a ênfase da análise e projeto sobre a implementação. Com Genexus, este enfoque é mais destacado, já que o desenvolvedor estará a maior parte do tempo realizando tarefas de análise e o GeneXus em si, tarefas de projeto e implementação, por exemplo, normalização da base de dados, geração de programas e outras. Por outro lado, alguns dos enfoques metodológicos são bastante diferentes que os habituais, como, por exemplo, começar a análise da aplicação pelas interfaces do usuário, a quase nula referencia da implementação física do sistema, e outros.

No desenvolvimento de aplicações com GeneXus, os objetos de usuário são usados como o começo da análise. Os tipos de objetos utilizados são:

Transações: visões do usuário que têm um diálogo associado e que podem modificar o conteúdo da base de dados. São a entrada básica do conhecimento da Ferramenta GeneXus. O conhecimento da vida real é capturado através de transações definidas pelo usuário construindo-se uma base de conhecimento, a partir da qual cria-se uma base de dados e os programas que permitem modificações e consultas. Neste caso, por uma transação entende-se a modificação interativa da base de dados, permitindo ao usuário criar, modificar ou eliminar informação.

O desenvolvimento da base de dados baseia-se na Teoria da Base de Dados Relacionais, e a mesma cumpre a 3ª Forma Normal. As transações são os eventos que modificam a informação de forma interativa. O desenvolvimento da base de dados se realiza a partir das transações.

Relatórios: são os objetos que permitem realizar consultas, geradas pela aplicação. Os relatórios são consultas batch (não interativas), que não modificam o conteúdo da base de dados.

Procedimentos: estes objetos têm todas as características dos relatórios, só que permitem ainda criar ou modificar as informações da base de dados, mas sem necessariamente, envolver diálogo, ou seja, tipicamente processos batch.

Painéis de Trabalho: são consultas interativas que permitem a recuperação de informações de forma dinâmica, orientando a pesquisa em tempo de execução. Os painéis de trabalho conhecidos como work-panel permitem ao usuário o encadeamento por evento para capturar informações, navegar livremente através de telas, elegendo as ações que serão disparadas e executadas sobre elementos selecionados. Indiretamente chamando procedimentos adequados, podem determinar modificações na base de dados.

Web Objects: possuem as mesmas características das Work Panels, porém com diálogo assíncrono, via Internet ou Intranet.

Mais informações sobre a ferramenta de desenvolvimento Genexus podem ser encontradas em: http://www.genexus.com/main/hmain.aspx.

5.5 VISUAL BASIC 6.0

O ambiente de programação Visual Basic 6.0, desenvolvido pela Microsoft, será utilizado neste trabalho apenas para compilar o programa gerado pela ferramenta de desenvolvimento Genexus. Através da ferramenta gera-se o código fonte e o executável do aplicativo é gerado utilizando o ambiente Visual Basic.

Mais informações sobre Visual Basic pode ser encontradas em McKelvi(1997).

6 ESPECIFICAÇÃO E MODELAGEM DO SISTEMA

Quando uma *softhouse* coloca um de seus produtos no mercado, torna-se explícita a necessidade de criar mecanismos para atender a demanda de chamadas dos clientes, que pedem suporte para o esclarecimento de dúvidas e resolução de problemas em relação ao produto que adquiram. O fato de os técnicos da área de suporte freqüentemente utilizarem a experiência adquirida na resolução de problemas anteriores na resolução de novos problemas, torna o raciocínio baseado em casos particularmente apropriado para sistemas *"help desk"*.

O sistema realiza a mesma tarefa que técnicos da área de suporte ao procurar soluções para seus clientes. Quando esta tarefa é desempenhada por especialistas humanos, eles conduzem a busca comparando a interpretação de uma dada situação atual com experiências vividas no passado. Os especialistas procuram lembrar situações similares que possam oferecer caminhos para a nova situação.

O uso de RBC permite o armazenamento do conhecimento do especialista em uma base de casos o que facilita a resolução técnica do problema em questão, através da busca de um caso anterior correspondente por similaridade em uma base de problemas resolvidos no passado.

6.1 DOMÍNIO DO TRABALHO

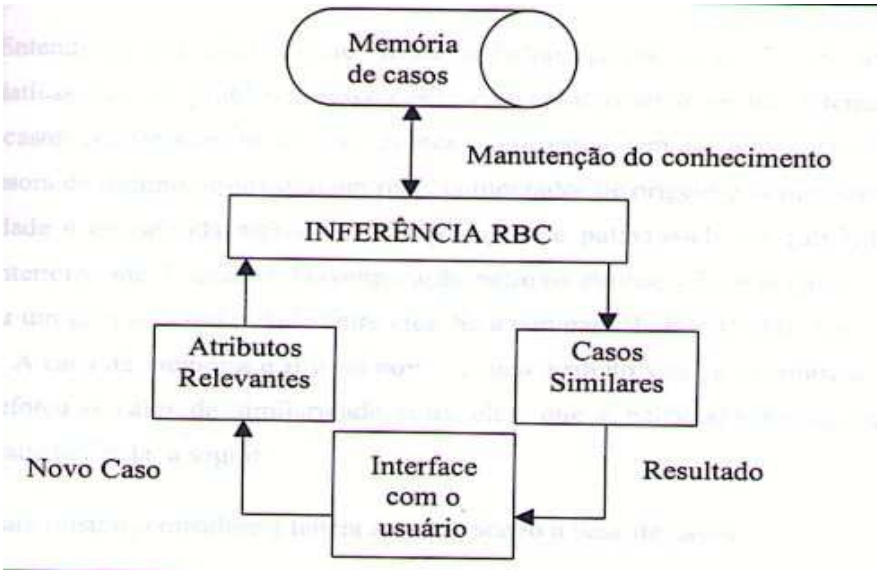
O domínio a ser considerado no sistema implementado é aquele que inclui os problemas que impedem os usuários de um determinado sistema, de realizar suas tarefas normalmente. Desde erros de implementação, mensagens de erro do sistema, falta de conhecimento do próprio usuário quanto a forma de proceder para realizar determinada tarefa etc.

6.2 ARQUITETURA DO SISTEMA

O raciocínio no sistema proposto inicia com a identificação de um novo problema. Através da chamada de um cliente o técnico se depara com uma nova situação que exige uma pesquisa. Este profissional acessa o sistema com sua interpretação com relação a esta nova situação, informando ao sistema atributos relevantes para a pesquisa. O técnico infere esses atributos em uma representação tipo formulário, modelando a nova situação da mesma forma

que os casos na base de casos. Então, o sistema pode comparar a nova situação (caso de entrada) com cada caso da base de casos (casos candidatos). Uma métrica de similaridade mede o valor de cada comparação para ordenar os casos candidatos para serem oferecidos como resultado da iteração. Conforme descrito no item 4.5.2.1, uma métrica de similaridade é uma função que mede numericamente o grau de similaridade entre dois casos. Para a implementação deste trabalho utilizou-se o cálculo do vizinho mais próximo para medir a similaridade entre os casos, as próximas seções descrevem as etapas do desenvolvimento do sistema. A arquitetura básica sobre a qual desenvolvemos o protótipo está esquematizada na figura 03:

Figura 03: Arquitetura do sistema



Fonte: Alves Júnior, P38.

6.3 AQUISIÇÃO DO CONHECIMENTO

Para o desenvolvimento deste sistema foi necessário o conhecimento de especialistas para domínio da aplicação. Para a aquisição de tal conhecimento especialista, faz-se necessária a utilização de um método de extração do conhecimento junto àqueles que o detêm. Portanto, antes da implementação das etapas do ciclo do RBC, propriamente ditas, foi necessária a realização da tarefa de aquisição do conhecimento.

Assim sendo, a aquisição dos casos para a implementação do protótipo se deu através de várias entrevistas com um técnico da área de suporte de uma empresa da região de

Blumenau e de algumas anotações sobre históricos de resoluções de problemas de alguns técnicos da área.

6.4 REPRESENTAÇÃO DO CONHECIMENTO

Uma etapa importante no desenvolvimento de sistemas que utilizem RBC é a forma de representar o conhecimento do especialista, neste caso, o técnico da área de suporte. Na procura desta representação, chegou-se a conclusão de que a melhor forma é utilizar os "sintomas" e algumas características do caso para atributos sobre os quais será calculada a similaridade entre os casos. Esses sintomas serão representados na forma de atributos relevantes (palavras-chave).

Uma vez definidos os atributos (características relevantes de cada caso) a serem considerados pelo sistema, a preocupação passou a ser quanto à forma de representação do mesmo. Como o RBC utiliza-se de representação de memória de casos velhos para a comparação com um novo caso, estes atributos devem estar representados em uma base de casos, devidamente indexados, possibilitando assim sua recuperação. Para a implementação do sistema, os casos foram armazenados seqüencialmente em uma tabela, pois de acordo com o item 4.5.1.1 onde Abel (1996) relata que "o formato mais utilizado para se representar casos é o de tabelas do modelo relacional".

6.5 INDEXAÇÃO DOS CASOS

No item 4.5.1.3 Lee (2002) relata que "a indexação determina o que deve ser comparado entre os casos para avaliar sua similaridade". Sendo assim neste trabalho a indexação foi realizada através de alguns atributos como o código do caso, e através da atribuição de palavras-chave para cada caso.

6.6 RECUPERAÇÃO

Como foi dito na seção 4.5.2 a recuperação consiste em fazer uma busca na memória de casos e selecionar quais poderão ser aproveitados, ocorre através de um caso a ser resolvido (caso de entrada), onde a busca por casos é feita por algoritmos que selecionam casos com determinada similaridade com o caso de entrada através dos índices descritos acima.

A similaridade compreendida neste trabalho consiste em encontrar algum grau de igualdade entre as características de um problema atual com as características de um caso da base de casos. Dentro desse contexto as características relevantes correspondem a versão do sistema que o cliente utiliza, o ambiente no qual está instalado, palavras contidas nas mensagens de erro do sistema, etc. No sistema implementado, essas características são definidas através da escolha de palavras chave para cada caso, e pesos que diferenciam cada palavra-chave. É sobre os pesos de cada palavra-chave que vai ser calculada a similaridade entre os casos.

A similaridade é encontrada através da comparação entre as características entre dois casos, (A e B) sendo caso A o caso da memória e caso B o novo caso, cada atributo similar encontrado reforça o valor da similaridade entre eles, que é calculado através da fórmula da similaridade, que foi descrita no capitulo 4.5.2.1.

Uma vez feita a comparação (casonovo, casobase), e qualquer um dos atributos for similar, então este caso passa a ser candidato a melhor caso, após concluído o processo de recuperação.

Este método diferencia-se do método tradicional de pesquisa de banco de dados porque, na pesquisa tradicional o caso recuperado através da indexação do atributo, levando em consideração os atributos citados, este deverá ser idêntico ao atributo procurado, ou seja, todos os atributos do caso novo devem estar no caso da base de casos. Com RBC se no momento da procura qualquer um dos atributos do caso novo estiver em um dos casos da base este caso então será recuperado não importando o valor da similaridade que vai estar entre 0 (não similar) e 1 (totalmente similar). Assim este método mostra-se mais interessante uma vez que qualquer atributo coincidente entre os casos em questão possibilita na recuperação de um caso da base que pode ser analisado pelo técnico para complementação de seus trabalhos.

6.7 ADAPTAÇÃO

Na seção 4.5.3 Abel (1996) relata que normalmente o caso recuperado não casa perfeitamente com as necessidades do problema do usuário, o processo de adaptação procura por meio de algoritmos amenizar estas diferenças. No protótipo implementado quando os casos mais similares são recuperados, o próprio técnico de suporte analisa cada um e se

necessário faz as devidas adaptações para o problema atual, não havendo intervenção do sistema.

6.8 RETENÇÃO

A etapa de retenção é a etapa de reter o que é útil de um problema novo resolvido. De acordo com Abel (1996), na seção 4.5.4, a retenção de novos casos na memória deve acontecer somente quando o novo caso amplia a capacidade de resolução de problemas. Sendo assim quando a similaridade retornar zero, ou muito inferior, fica a critério do técnico do suporte se inclui o novo caso ou não. No protótipo implementado a base de casos somente é alimentada com o consentimento do técnico, não há intervenção do sistema.

6.9 MODELAGEM PARA IMPLEMENTAÇÃO

6.9.1 LISTA DE EVENTOS

De acordo com Shiller (1992), os eventos são a pedra fundamental dos sistemas, e a especificação de um sistema deve começar pela lista de eventos. Nesta pode-se identificar os eventos para os quais o sistema deve responder.

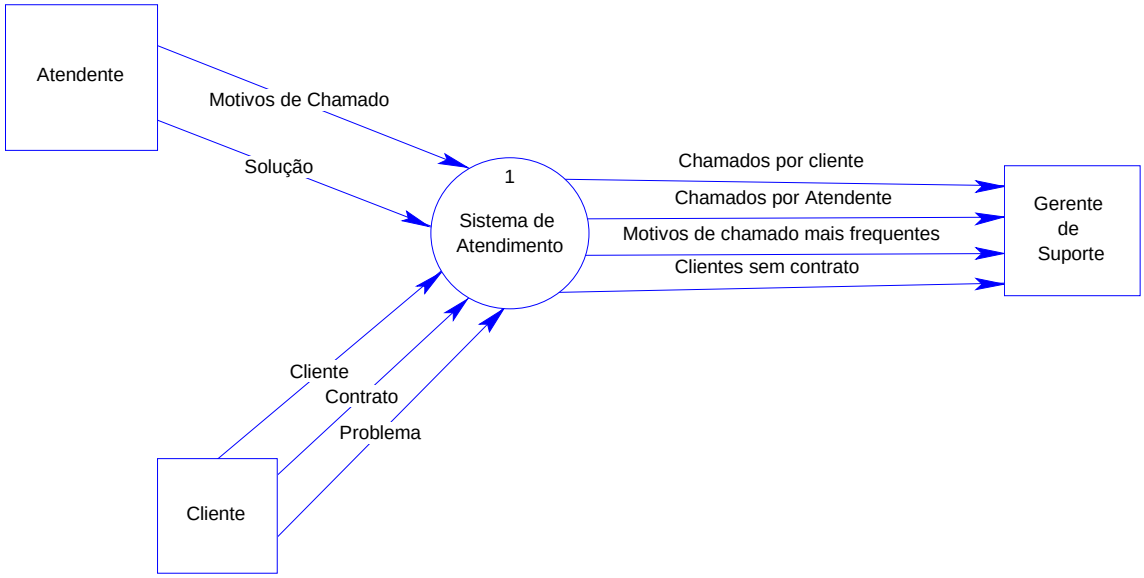
Os eventos são enumerados a seguir:

1) Cliente é cadastrado;
2) Cliente efetua contrato de manutenção;
3) Atendente Cadastra motivo de chamado;
4) Atendente registra chamado do cliente;
5) Atendente registra procedimentos para chamados;
6) Gerar chamados por cliente;
7) Gerar chamados por atendente;
8) Gerar clientes sem contrato;
9) Gerar motivos de chamados mais freqüentes;

6.9.2 DIAGRAMA DE CONTEXTO

O Diagrama de Contexto define com o que ou com quem o sistema faz interface, e qual o conteúdo dessa interface. O diagrama de contexto representa o sistema em um único processo. Com base na lista de eventos anteriormente relacionada desenvolveu-se o diagrama de contexto, que é mostrado na figura 04.

Figura 04: Diagrama de contexto



6.9.3 MODELO ENTIDADE RELACIONAMENTO (MER)

Figura 05: MER – modelo conceitual

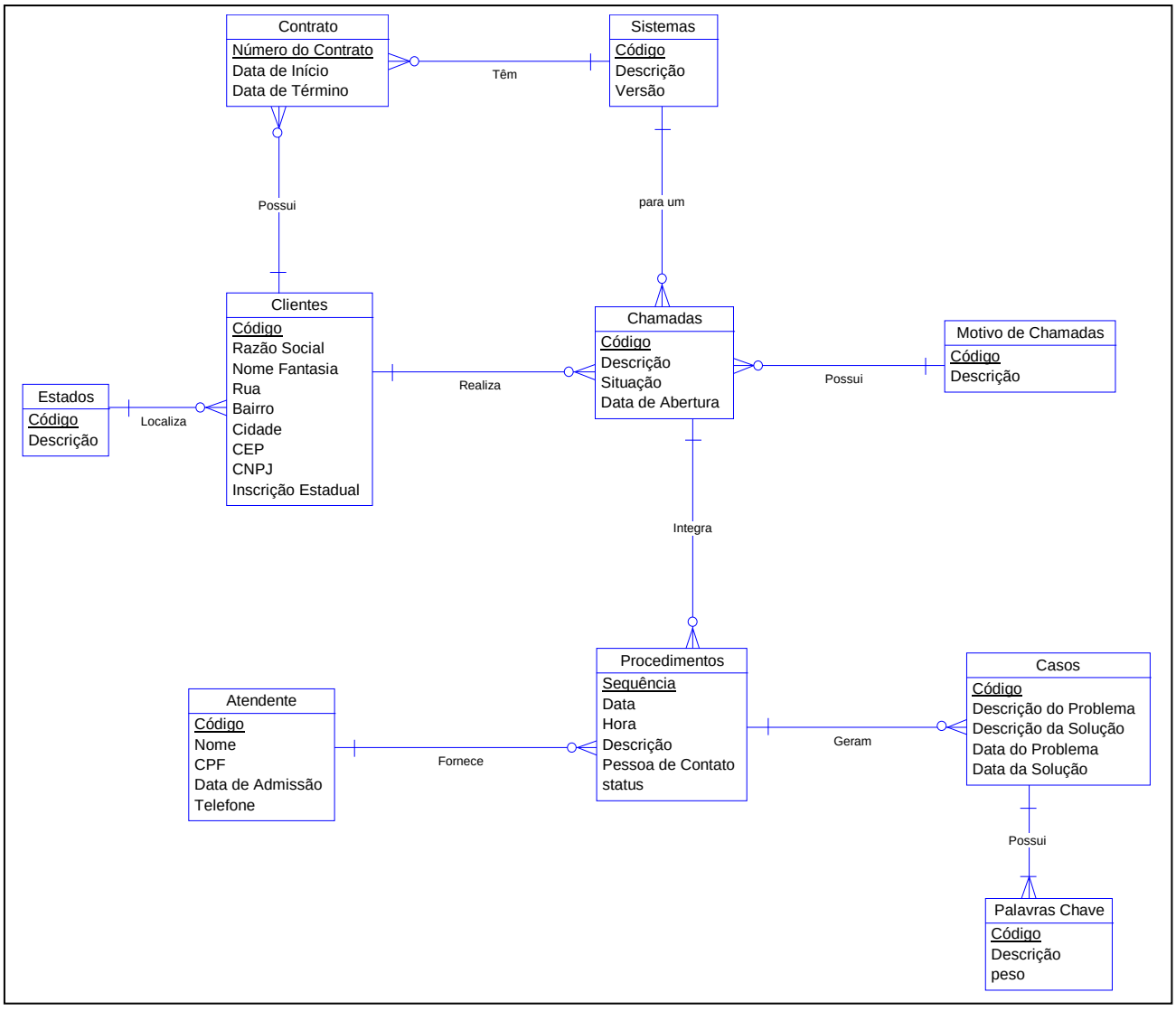
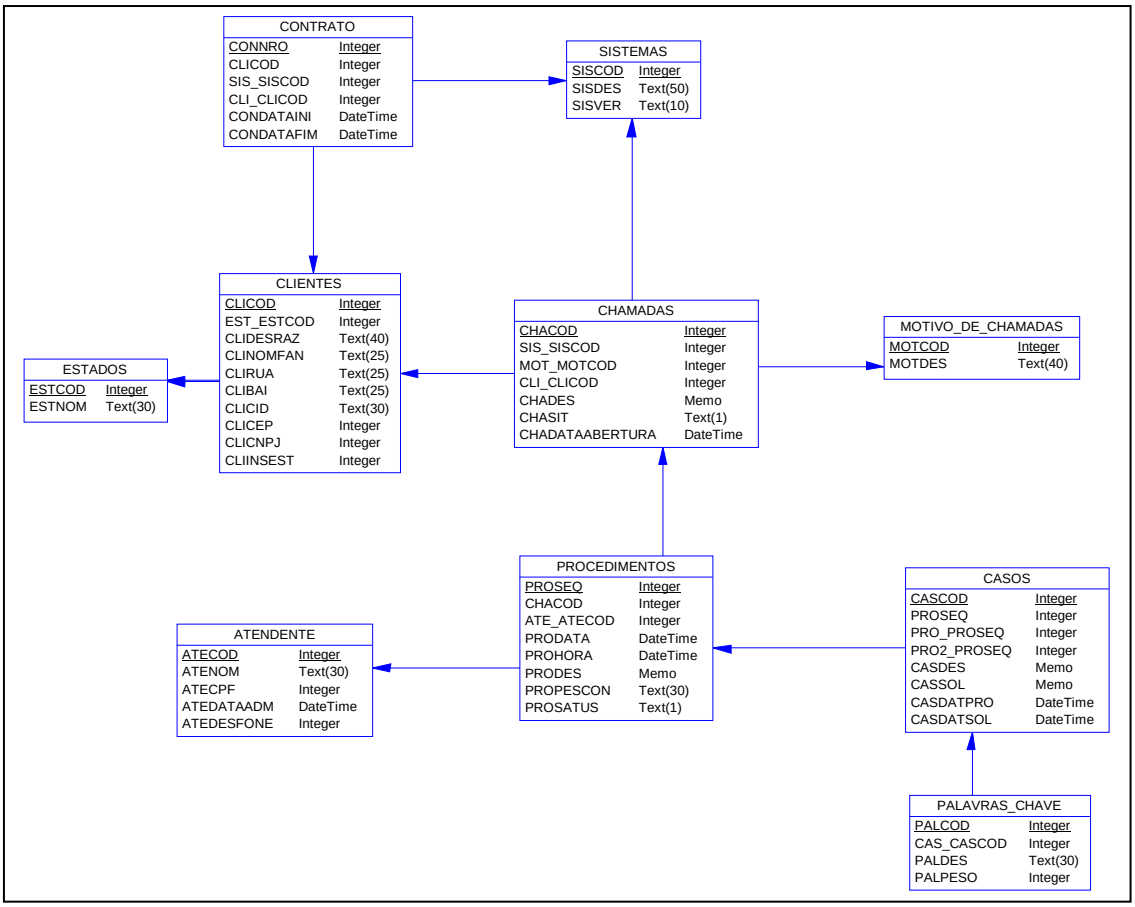


Figura 06: MER – Modelo físico

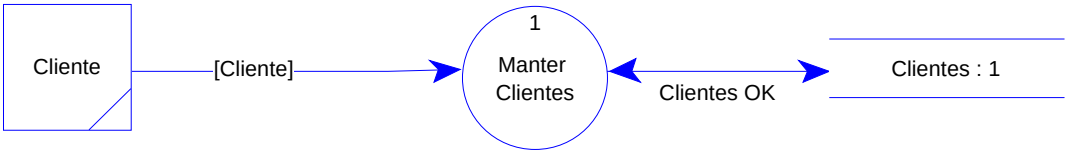


6.9.4 DIAGRAMA DE FLUXO DE DADOS

Pressman (1995), descreve o DFD como “uma técnica gráfica que descreve o fluxo de informação do sistema”. Para a implementação do protótipo os DFD`s foram separados por eventos, com uma breve explicação sobre cada um.

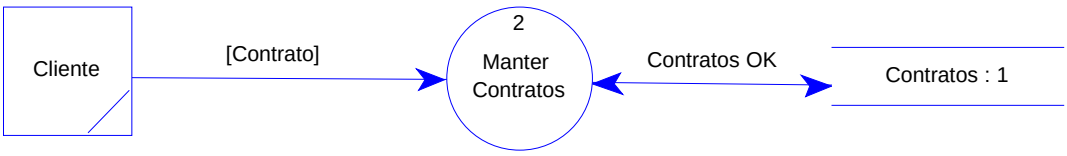
A figura 07 mostra o DFD do evento número um, neste evento acontece um simples cadastro de clientes, pois para que um chamado de um cliente possa ser registrado o cliente já deve ter estar previamente cadastrado.

Figura 07: DFD do evento número 1:



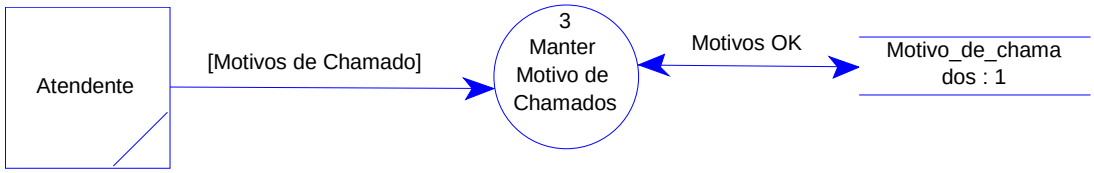
No evento número dois, ilustrado na figura 08, é efetuado o cadastro de contratos de manutenção, pois para que se tenha um controle dos clientes que pagam ou não manutenção deve-se realizar um contrato.

Figura 08: DFD do evento número 2:



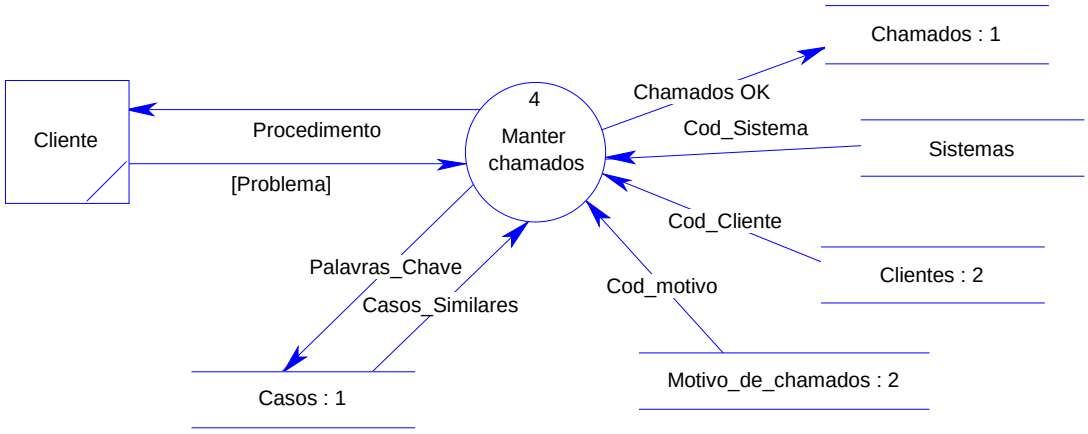
Quando se registra um chamado de um cliente, deve-se registrar o motivo pelo qual o cliente está chamado, exemplo: mensagem de erro do windows, banco de dados corrompido, dúvidas etc. Então o evento de número três realiza o cadastro desses motivos de chamados, conforme mostrado na figura 09.

Figura 09: DFD do evento número 3:



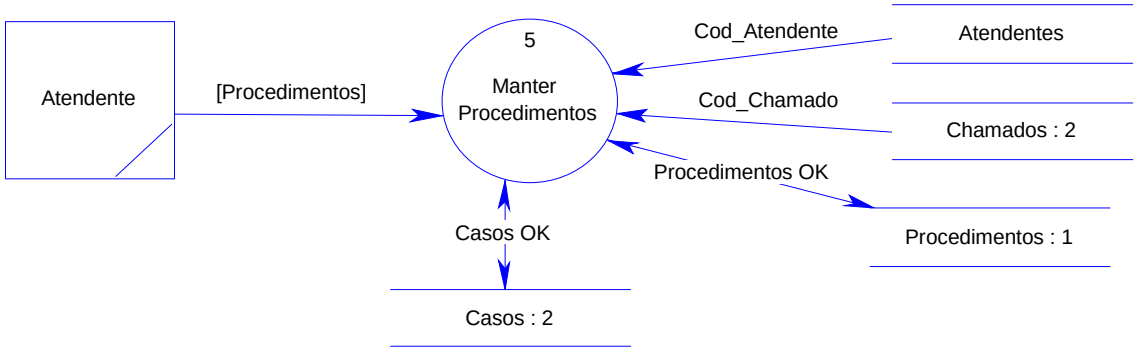
No evento número quatro, acontece o registro da chamada do cliente, ele informa qual problema está acontecendo e para qual sistema está pedindo suporte, e o atendente (técnico de suporte) registra o chamado. É também nesse momento que acontece o ciclo do RBC, pois se o atendente não souber o não lembrar no momento qual é o procedimento a ser informado ao cliente para resolver o problema, ele pode acessar a base de casos em busca de algum caso similar que lhe ofereça algum caminho para resolver o problema atual. Todo esse processo é ilustrado na figura 10

Figura 10: DFD do evento número 4:



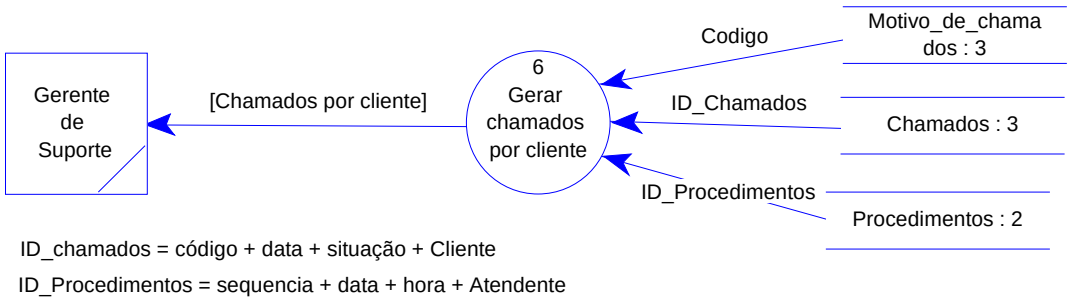
No evento número cinco, mostrado pela figura 11, o técnico de suporte registra o procedimento que foi passado para o cliente. Um chamado de um cliente pode se estender por vários dias até que se resolva o problema, e vários técnicos podem atender o mesmo chamado, registrando-se vários procedimentos até que se chegue na solução. Quando finalmente se chega a uma solução para um problema novo, se o técnico assim desejar essa nova experiência pode passar a integrar a base de casos.

Figura 11: DFD do evento número 5:



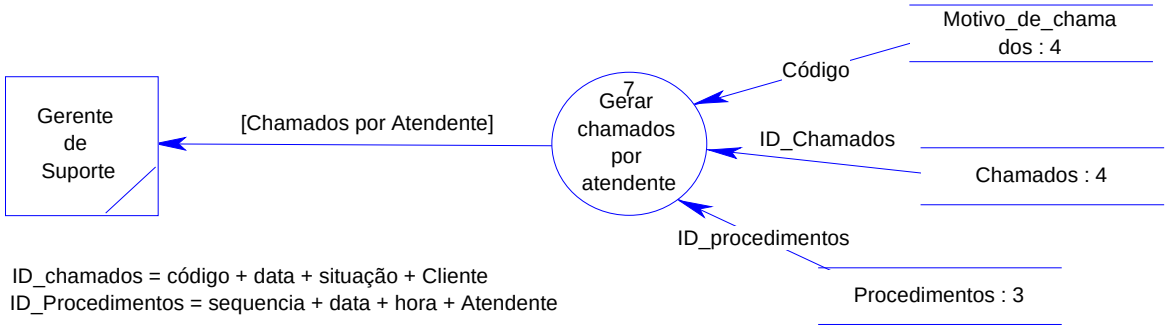
No evento número seis, mostrado na figura 12, o gerente de suporte recebe um relatório de chamados por cliente, dessa forma ele consegue saber em determinado período os clientes que pediram suporte e por quais motivos.

Figura 12: DFD do evento número 6:



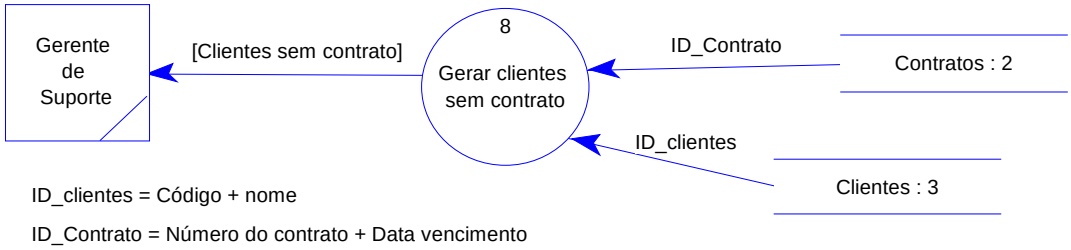
No evento número sete, mostrado na figura 13, o gerente de suporte recebe um relatório de chamados por atendente, assim ele fica informado de quais clientes determinado técnico atendeu, e por quais motivos.

Figura 13: DFD do evento número 7



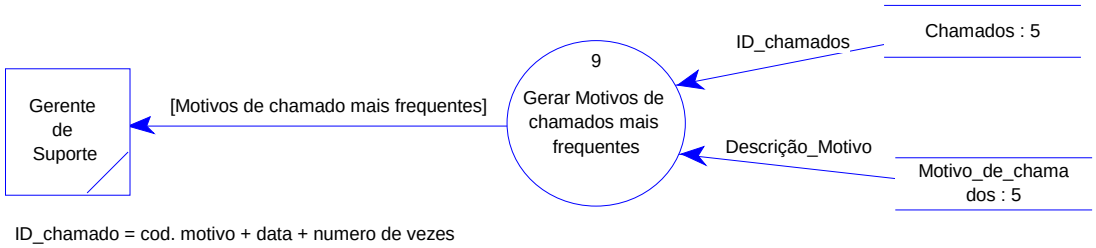
No evento número 8, ilustrado na figura 14, o gerente recebe um relatório dos clientes cujos contratos de manutenção já venceram ou clientes que ainda não tem contrato, dessa forma ele pode entrar em contato com cada cliente e tomar as providências cabíveis de acordo com a política da empresa, como pro exemplo enviar propostas de contrato.

Figura 14: DFD do evento número 8



No evento de número nove, o gerente de suporte recebe um relatório dos motivos de chamados mais freqüentes em determinado período, com esse relatório ele pode procurar melhorias, como por exemplo, se o motivo mais freqüente é Dúvidas quanto a operacionalidade do sistema, o gerente pode chegar a conclusão de que é necessário investir em mais treinamento e procurar rever a operacionalidade do sistema buscando melhorias. O evento número nove é ilustrado na figura 15.

Figura 15: DFD do evento número 9



6.9.5 DICIONÁRIO DE DADOS

A notação utilizada na descrição das tabelas mostradas na seqüência foi: CP = chave primária, CE = chave estrangeira e AN = atributo normal.

Tabela 4: Atendente

Nome	Código	Tipo	Identificação
Código	ATECOD	N3	CP
Nome	ATENOM	A30	AN
CPF	ATECPF	N18	NA
Data de Admissão	ATEADM	D	AN
Telefone	ATEFONE	N15	AN

Tabela 5: Casos

Nome	Código	Tipo	Identificação
Código	CASCOD	N6	CP
Descrição do Problema	CASDES	TXT250	AN
Data	CASDATA	D	AN
Descrição da solução	CASDESSOL	TXT250	AN
Data da solução	CASDATSOL	D	AN

Tabela 6: Chamado

Nome	Código	Tipo	Identificação
Código	CHACOD	N8	CP
Descrição	CHADES	TXT250	AN
Situação	CHASIT	A1	AN
Data de Início	CHADATINI	D	AN

Tabela 7: Clientes

Nome	Código	Tipo	Identificação
Código	CLICOD	N6	CP
Razão social	CLIRAZSOC	A50	AN
Nome fantasia	CLINOMFAN	A50	AN
Rua	CLIRUA	A50	AN
Bairro	CLIBAI	A30	AN
Cidade	CLICID	A30	AN
Estado	CLIEST	A2	AN
Cep	CLICEP	A9	AN
CNPJ	CLICNPJ	N18	AN
Inscrição Estadual	CLIINSEST	N18	AN
Contrato Manutenção	CLICONMANUT	A1	AN

Tabela 8: Contrato

Nome	Código	Tipo	Identificação
Número do Contrato	CONCOD	N8	CP
Data de Início	CONDATINI	D	AN
Data de Término	CONDATFIM	D	AN

Tabela 9: Motivos de chamado

Nome	Código	Tipo	Identificação
Código	MOTCOD	N2	CP
Descrição	MOTDES	A30	AN

Tabela 10: Procedimentos

Nome	Código	Tipo	Identificação
Seqüência	PROSEQ	N2	CP
Data	PRODATA	D	AN
Hora	PROHORA	T	AN
Descrição	PRODES	TXT250	AN
Pessoa de Contato	PROPESCON	A25	AN
Status	PROSTATUS	A10	AN

Tabela 11: Sistemas

Nome	Código	Tipo	Identificação
Código	SISCOD	N2	CP
Descrição	SISDES	A25	AN
Versão	SISVER	A10	AN

Tabela 12: Palavras-Chave

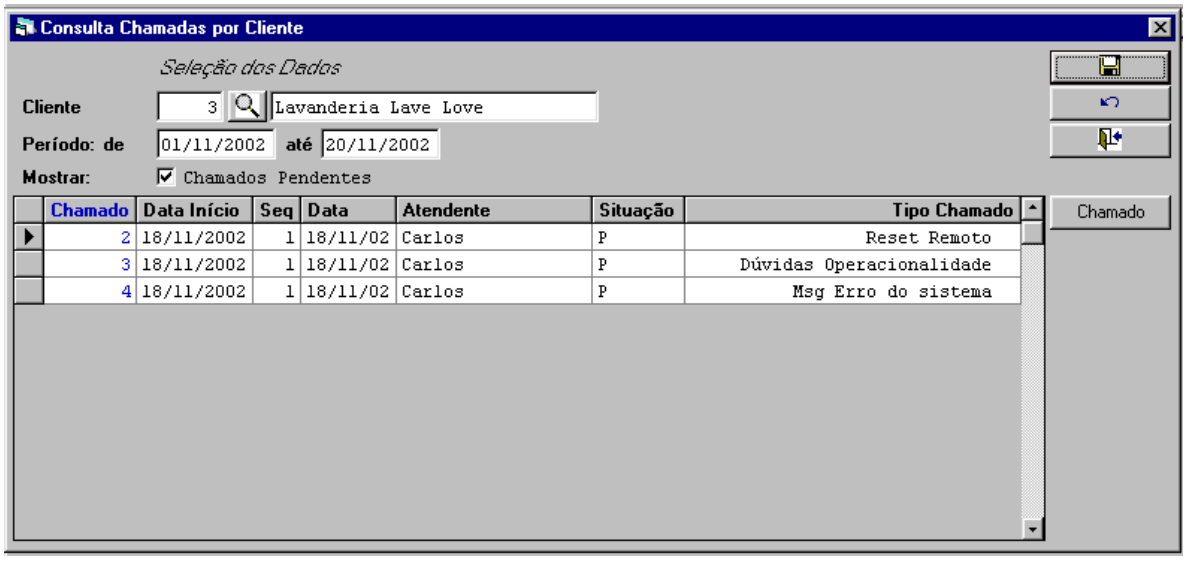
Nome	Código	Tipo	Identificação
Código	PALCOD	N2	CP
Descrição	PALDES	A25	AN
Peso	PALPESO	N6.2	AN

7 IMPLEMENTAÇÃO DO SISTEMA

Neste capítulo, mostra-se a funcionalidade do sistema, exemplificando o processo de busca com RBC, serão mostradas apenas algumas de suas principais telas.

Sempre que algum cliente que está com dificuldades e liga pedindo suporte técnico, o atendente deve antes de abrir um novo chamado para este cliente, verificar se já existe algum chamado pendente, através da consulta de chamados por clientes, conforme mostrado na figura 17.

Figura 17: Tela de consulta chamada por cliente

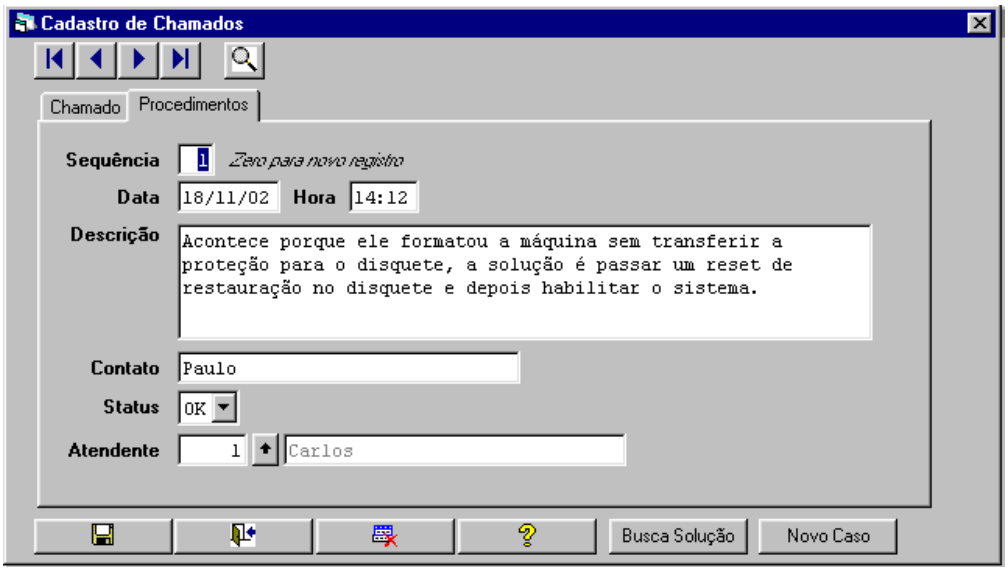


Caso exista algum chamado pendente, se tratando do mesmo problema deve-se dar continuidade a este chamado. Clicando no código do chamado (aparece em azul) a tela de cadastramento de chamados se inicia possibilitando a continuidade do chamado, a inserção de novos procedimentos e a realização de alterações. Se não existir chamados pendentes o técnico cadastra um novo chamado, clicando no botão “Chamado” que aparece à direita da tela. A tela para cadastramento de chamados é mostrada conforme visto na figura 18.

Figura 18: Tela para Cadastro de chamados

Depois de preencher os dados do chamado, o técnico deve informar ao cliente qual é o procedimento para resolver o problema, a figura 19, mostra a tela para o cadastro de procedimentos.

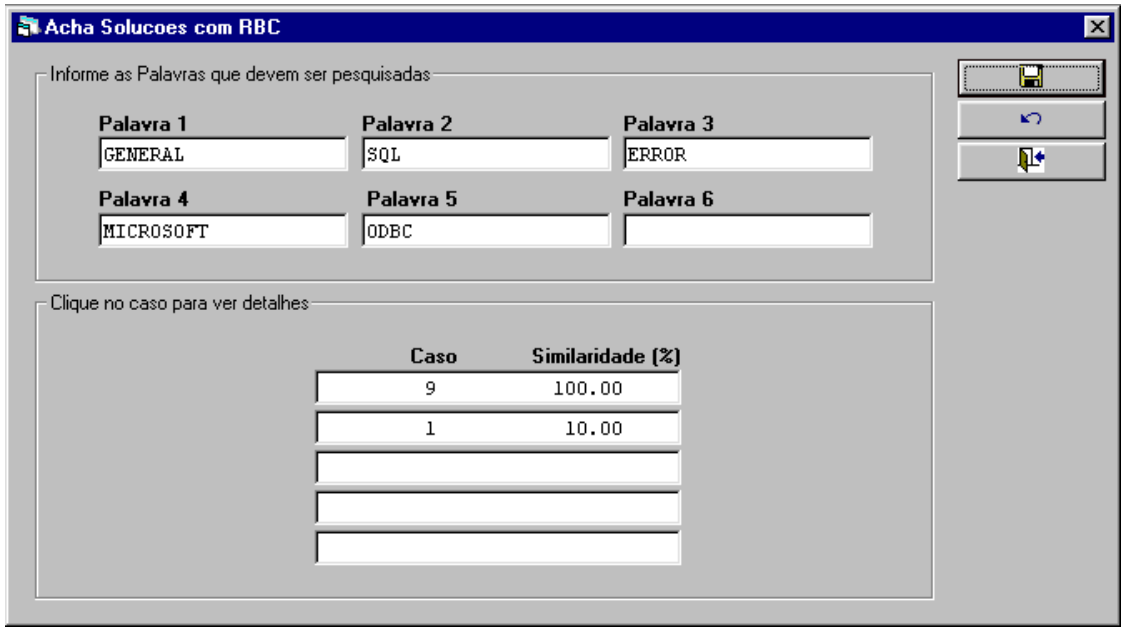
Figura 19: Tela para cadastro de Procedimentos para chamados



Para incorporar o módulo de Raciocínio Baseado em Casos ao sistema de atendimento, foi criado um botão na tela de cadastramento de chamados. Quando um cliente que está com dificuldades liga pedindo suporte técnico, o técnico que o atender deverá então registrar o chamado do cliente, conforme ilustrado anteriormente.

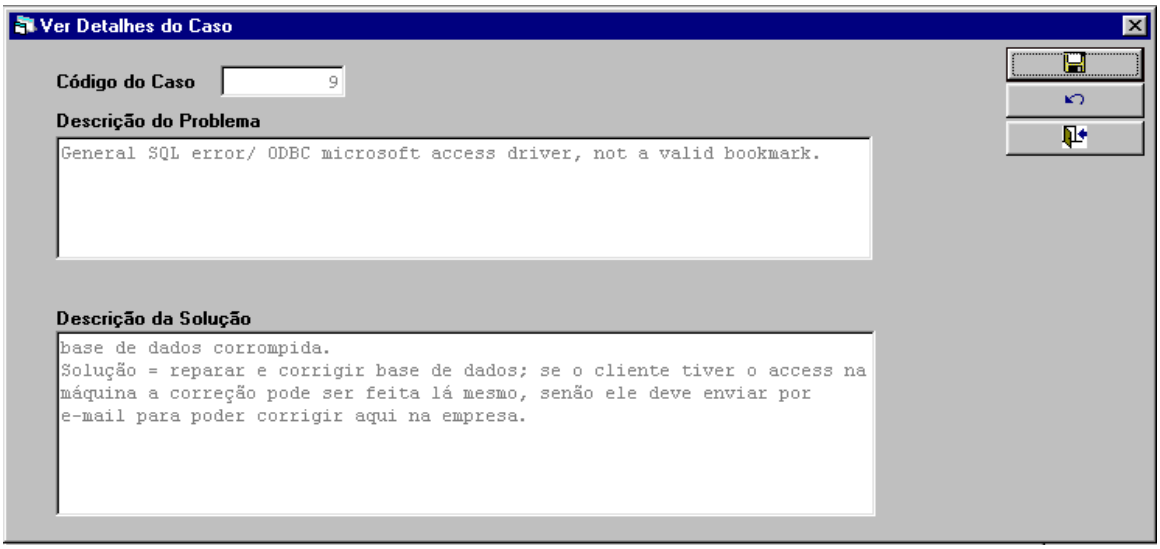
Caso o atendente (técnico de suporte) não lembre ou não conheça o procedimento que deve ser passado ao cliente para que o problema seja resolvido, este pode executar uma busca por algum caso similar cadastrado na memória de casos, que possa oferecer algum caminho para que o problema atual possa ser solucionado. Clicando no Botão “busca solução”, dá-se início ao ciclo do RBC, conforme mostrado na figura 20. Nesta tela o módulo de RBC fará a pesquisa através da definição de palavras-chave que o técnico informará ao sistema. Nesta mesma tela são retornados os casos que possuírem a maior similaridade com o caso atual.

Figura 20: Interface de entrada RBC, palavras-chave de um caso



Clicando no caso selecionado, o sistema mostra os detalhes deste caso, Conforme ilustrado na figura 21.

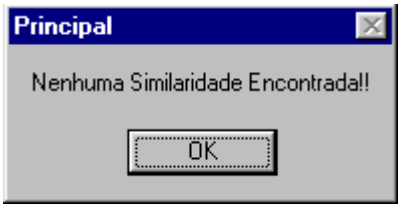
Figura 21: Detalhes do caso



O técnico então analisa o caso recuperado e se necessário faz as devidas adaptações para o problema atual, não havendo intervenção do sistema.

Quando é realizada uma pesquisa que não retorna similaridade, o sistema informa que não foi encontrado caso similar, conforme visto na figura 22.

Figura 22: Quando o sistema não encontra similaridade

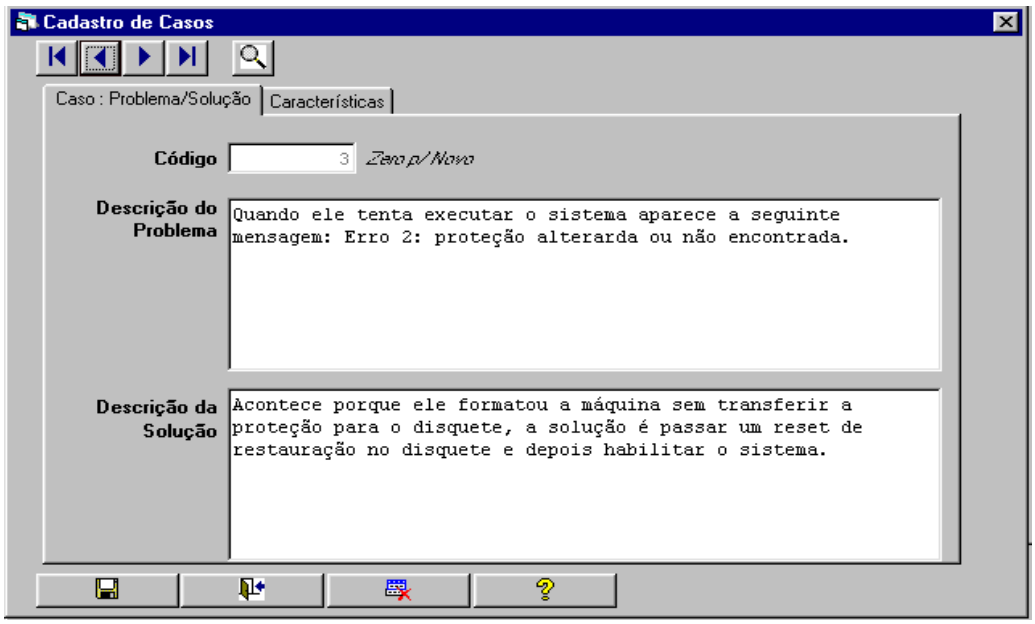


A experiência tem demonstrado que, quando surge um problema que os técnicos ainda não tinham o conhecimento, o problema é estudado e resolvido na técnica da “tentativa e erro”, sendo assim a solução pode levar dias ou semanas para ser encontrada e validada, pois para que um procedimento possa ser considerado como solução para um problema novo, precisa ser comprovado que funciona. Então os técnicos registram o chamado do cliente que está com o problema novo, e ficam trabalhando na procura da solução para o problema.

Quando finalmente se chega a uma solução comprovada para o problema, a base de casos pode ser alimentada com essa nova experiência, no protótipo implementado a base de casos somente é alimentada com o consentimento do técnico, não há intervenção do sistema.

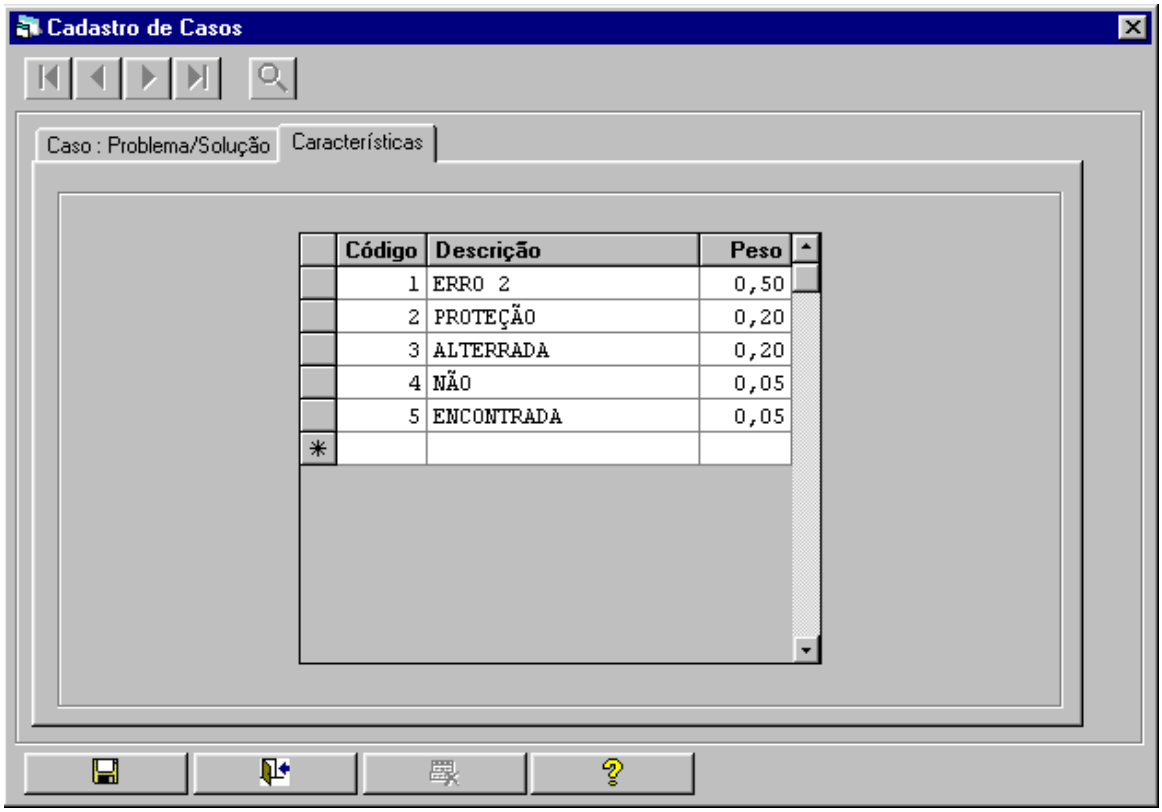
O técnico clicando no botão “novo caso” (na tela de cadastro de chamados), transfere os dados desse chamado que registrou uma nova experiência para a memória de casos, conforme mostra a figura 23.

Figura 23: cadastro de novo caso na memória de casos.



É necessário que se informe os atributos relevantes, (palavras-chave) do novo caso, para que se possa calcular a similaridade. É o técnico que escolhe as palavras-chave de cada caso, geralmente escolhe palavras que constam na descrição da mensagem de erro e atribui a estas palavras pesos, sobre os quais será calculada a similaridade. O técnico pode atribuir pesos diferenciados para cada palavras-chave, de acordo com a importância de cada uma. A figura 24 mostra o cadastro de palavras chave para o novo caso.

Figura 24: Tela para atribuição de palavras chave para o caso novo



8 CONCLUSÃO

Durante este trabalho procurou-se ilustrar a potencialidade de um dos paradigmas da inteligência artificial (IA), a tecnologia de raciocínio baseado em casos (RBC) como ferramenta para auxiliar os técnicos da área de suporte no fornecimento de soluções para seus clientes.

O sistema desenvolvido foi instalado a título de experiência na empresa onde trabalho e pode-se concluir que teve seus principais objetivos alcançados, pois o sistema possibilita o registro da chamada o cliente, agiliza o atendimento, armazena o conhecimento na memória de casos para eventuais consultas e sendo assim facilita o serviço do técnico da área. Não foi realizada nenhuma comparação com dados estatísticos para avaliar a situação antes, e depois do sistema, mas a melhora já é perceptível.

Este trabalho possibilitou-me iniciar estudos na área de inteligência artificial, além de ter despertado o interesse pela área, fortaleceu a vontade de continuar estudando inteligência artificial.

Percebeu-se que a potencialidade da técnica de RBC em sistemas de help desk podem significar vários benefícios, o mais importante deles talvez seja que partindo-se de uma base de casos (modelados para determinado domínio), pode-se diagnosticar e resolver problemas sem que necessariamente o técnico esteja presente, isso possibilita uma continuidade nos atendimentos e uma redução no custo do suporte técnico. Outro benefício que podemos citar é a possibilidade de se disponibilizar a base de casos para o treinamento de novos técnicos.

8.1 SUGESTÕES PARA TRABALHOS FUTUROS

Percebendo a grande potencialidade da técnica de RBC na resolução de problemas e principalmente na combinação com sistemas *help desk*, sugere-se a continuidade do presente trabalho a implementação de uma rotina que possibilite ao sistema detectar as palavras-chave através da descrição do problema, não sendo necessário que o técnico as informe no momento de realizar uma pesquisa por casos similares e nem informa-las na hora de cadastrar um novo caso.

A técnica de recuperação utilizada neste trabalho foi a do vizinho mais próximo, uma outra sugestão para a continuação desse trabalho é a implementação de outra técnica de recuperação, para compara-las e avaliar qual é mais veloz.

Uma outra sugestão seria a criação de uma *home page* para a disponibilização de uma base de conhecimento para consulta via internet, para que o comportamento da técnica fosse estudado e avaliado. Além disso, esta opção representaria uma comodidade ainda maior para o cliente

REFERÊNCIAS BIBLIOGRÁFICAS

ABEL, Mara. **Raciocínio baseado em casos – CBR.** Instituto de Informática. Universidade Federal do Rio Grande do Sul. [Porto Alegre], [2002?]. Disponível em: <http://marabel.inf.ufrgs.br/publico/disciplinas/SistEspecialistasN/CBR-Mara.pdf>. Acesso em: 04 set. 2002.

ABEL, Mara. **Um estudo sobre raciocínio baseado em casos.** Porto Alegre: UFRGS, 1996.

ALVES JÚNIOR, Osni Pereira. **Sistema de apoio ao processo decisório relativo à manutenção de hardware/ software para minimizar o tco em uma lan utilizando raciocínio baseado em casos.** 1998. Trabalho de conclusão de curso (Bacharelado em Ciências da Computação). Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

ARTech Consultores SRL. **Designing applications with GeneXus**. Montevideo: 2002.

CHU, Shao Yong. **Banco de dados:** organização, sistemas, administração. São Paulo: Atlas, 1983.

COUGO, Paulo Sérgio. **O papel do helpdesk-3 no suporte técnico e operacional em ambientes corporativos.** Tree Tools Informática. [Curitiba],[1999]. Disponível em: <http://www.treetools.com.br/artigos/arthd3.htm>. Acesso em 22 out 2002.

DALFOVO, Oscar; AMORIM, Sammy Newton. **Quem tem informação é mais competitivo.** Blumenau: Acadêmica, 2000.

DELPIZZO, Vanessa L. Francalacci. **Prescrição de atividades físicas através do uso da inteligência artificial**. Programa de pós-graduação em engenharia de produção. Universidade Federal de Santa Catarina. [Florianópolis], [1997]. Disponível em: <http://www.eps.ufsc.br/disserta98/delpizzo>. Acesso em: 22 set. 2002.

DEMARCO, Tom. **Analise estruturada e especificação de sistema.** Rio de Janeiro: Campus, 1989.

FAYYAD, U.M.; SHAPIRO, G.P.; UTHURUSAMY, R. **Advances in knowledge discovery and data mining**. Massachusetts: The MIT Press, 1996.

FERREIRA, Aurélio Buarque de Holanda, et al. **Novo Aurélio século XXI**: o dicionário da língua portuguesa. 3. ed. Rio de Janeiro: Nova Fronteira, 1999.

FISHER, Alan S. **CASE:** utilização de ferramentas para desenvolvimento de software. Tradução Info-Rio. Rio de Janeiro: Campus, 1990.

FOURNIER, Roger. **Guia pratico para desenvolvimento e manutencao de sistemas estruturados.** Sao Paulo : Makron Books, 1994.

FREITAS, Henrrique, LESCA, Humbert. Competitividade empresarial na era da informação. **Revista de Administração**, São Paulo: v.27, n.3, p.92-102, set.1992.

FURLAN, José Davi. **Sistemas de informação executiva:** como integrar executivos ao sistema informacional das empresas, fornecendo informações úteis e objetivas para suas necessidades estratégicas e operacionais. São Paulo: Makron Books, 1994.

GAEBLER, Ana Cristina. **Sistema de controle da qualidade para a produção de manufatura utilizando raciocínio baseado em casos**. 1999. Trabalho de conclusão de curso (Bacharelado em Ciências da Computação). Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

GANE, Chris; SARSON, Trish. **Analise estruturada de sistemas.** Rio de Janeiro: Livros Tecnicos e Cientificos, 1983.

GEN INFORMÁTICA. **Curso básico de Genexus.** Blumenau, 1999. Apostila.

HEINRICH, Luciane Tondorf. **Sistema de informação aplicados a lojas de confecções do Alto Vale do Itajaí – SC utilizando raciocínio baseado em casos**. 2000. Trabalho de conclusão de curso (Bacharelado em Ciências da Computação). Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

JENNINGS, Roger. **Usando Microsoft Access 97**: o guia de referência mais completo. Rio de Janeiro: Campus, 1997.

KELLER, Robert. **Análise estruturada na pratica:** desmistificando mitos. São Paulo: Makron, McGraw-Hill, 1990.

KOLODNER, Janet. **Case-based reasoning**. San Mateo, California:Morgan Kaufman Publishers, 1993.

LEE, Rosina Weber. **Pesquisa Jurisprudencial Inteligente.** Programa de pós-graduação em engenharia de produção. Universidade Federal de Santa Catarina. [Florianópolis], [1998]. Disponível em: <http://www.eps.ufsc.br/teses98/rosina>. Acesso em: 25 set. 2002.

MACHADO, Carlos. Como dar o tiro certo na hora de decidir. **Exame Informática,** São Paulo, v. 11, n. 120, p. 27-29, mar. 1996.

MARTIN, James; MCCLURE Carma. **Técnicas estruturadas e case.** São Paulo: Makron, McGraw-Hill,1991.

MCKELVY, M. et al. **Usando Visual Basic 5:** o guia de referência mais completo. Rio de Janeiro: Campus, 1997.

MERCADO-GARDNER, Juanita. **Projetando bancos de dados com Access 2**. Tradução Elisa M. Ferreira. São Paulo: Berkeley, 1995.

MOREIRA, Maria Isabel; ARRUDA Roberto; PREUSSLER Rodrigo. Suporte Técnico: Um guia para você se defender nessa selva. **Revista Info.** São Paulo, n.187, p. 51-58, out. 2001.

OLIVEIRA, Djalma de Pinho Rebouças de. **Sistemas de informações gerenciais: estratégicas, táticas, operacionais.** São Paulo: Atlas, 1996.

POMPILHO, S. **Análise essencial: g**uia prático de análise de sistemas. Rio de Janeiro: Infobook, 1994.

PRESSMAN, Roger S. **Engenharia de software.** São Paulo: Makron Books do Brasil Editora Ltda, 1995.

SILVA, Reginaldo Rubens da. **Sistema inteligente para apoio a identificação de possíveis suspeitos de crimes**. 1999. Trabalho de conclusão de curso (Bacharelado em Ciência da Computação). Centro de Educação Superior de Ciências Tecnologia, da Terra e do Mar, Universidade do Vale do Itajaí, Itajaí.

SHILLER, Larry. **Excelência em software.** São Paulo: Makron Books, 1992.

STAIR, Ralph M. **Princípios de sistemas de informação:** uma abordagem gerencial. Rio de Janeiro: Livros Técnicos e Científicos, 1999.

VARELA, Geraldo Menegazzo, LUNA, Paulo de Tarso Mendes. **Utilização de raciocínio baseado em casos no auxilio a recuperação de projetos do instituto de pesquisas ambientais.** 1998. Trabalho de conclusão de curso (Bacharelado em Ciências da

Computação). Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

YOURDON, Edward. **Análise estruturada moderna.** Rio de Janeiro: Campus, 1990.

WETHERBE, James C. **Análise de sistemas:** para sistemas de informação por computador. 3 ed. Rio de Janeiro: Editora Campus. 1987.

ANEXO A

Parte da codificação do cálculo da similaridade entre os casos da base

```
// verificar se pelo menos uma das palavras foi informada

For each CasCod
defined by PalDes

 &vSomPeso = 0
 &vPeso = 0
  for each
   if  .not. null(PalCod)
       &vSomPeso = &vSomPeso + PalPes
       if   PalDes = &Vpalavra1 .or. PalDes = &Vpalavra2 .or. PalDes = &Vpalavra3
.or.  PalDes = &Vpalavra4 .or. PalDes = &Vpalavra5 .or. PalDes = &vPalavra6
          &vPeso = &vPeso + PalPes
          &VAchouSim = 'S'
       endif
    endif
  endfor

  if &vSomPeso > 0
   if &VAchouSim = 'S'
      &VSim = round(&vPeso/&vSomPeso,2)
   endif
  endif

   if  &VSim <> 0
      &VContVet = &VContVet + 1
      &VetCaso(&VContVet) = CasCod
      &VetSim(&VContVet)  = &VSim
   endif
 endfor

 //=== rotina que ordena o vetor para mostrar somente os maiores ===
 &VContVet = 1
 &Vtrocou = 'S'
 &Vfim= 'N'
 do while &Vfim = 'N'
    if  &VetSim(&VContVet + 1) = 0
       if  &Vtrocou = 'S'
          &Vtrocou = 'N'
          &VContVet = 1
       else
          &Vfim = 'S'
          exit
       endif
    else
```

```
      if  &VetSim(&VContVet) < &VetSim(&VContVet+ 1)
         &VCasoAux = &VetCaso(&VContVet)
         &VsimAux = &VetSim(&VContVet)

         &VetCaso(&VContVet) = &VetCaso(&VContVet + 1)
         &VetSim(&VContVet) = &VetSim(&VContVet+ 1)

         &VetCaso(&VContVet + 1) = &VCasoAux
         &VetSim(&VContVet+ 1) = &VsimAux
         &Vtrocou = 'S'
      endif
      &VContVet = &VContVet +  1
   endif
enddo
```