Mestrado Integrado em Engenharia Química

Sistemas Especialistas no Apoio ao Projecto de Equipamento em Industrias de Processo Químico

Tese de Mestrado

de

Sara Pinto Soares dos Reis

Desenvolvida no âmbito da unidade curricular de Dissertação

realizado na

Faculdade de Engenharia da Universidade do Porto

Orientador na FEUP: **Prof. Fernando Gomes Martins**

Departamento de Engenharia Química

Setembro de 2010

Agradecimentos

Quero agradecer ao Professor Fernando Gomes Martins por todo o acompanhamento e orientação durante a realização deste projecto. Ao tempo dispendido, ajuda, compreensão e paciência.

A execução deste projecto foi pautada por imprevistos e limitações a nível pessoal. Chega por isso agora a altura de agradecer a todos os que de alguma forma, mais do que colaborarem com o meu trabalho, me ajudaram a ultrapassar o período mais difícil da minha vida.

À Ana, a minha amiga de sempre, obrigada por todo o apoio e força. Aos meus amigos desde a infância que reapareceram, é bom saber que estão sempre por perto. Ao Francisco que sempre me disse que eu era capaz de tudo e me apoiou incondicionalmente.

À Isabel, Marta, Carolina, Sandra e Ana por estes 5 anos de amizade e por estarem sempre disponíveis.

Ao Tiago e ao Guilherme pelo sofá alugado durante tantas horas quando o dia era mais difícil. Ao José Miguel, Beatriz, Miguel, João e ao Pedro pela preocupação e toda a força. Ao Bruno por toda a ajuda que me deu. A todos os amigos que têm estado sempre ao meu lado. À Raquel por tudo o que tem feito por mim.

Ao Rodrigo por estar sempre disponível para me ouvir e por continuar sempre por perto.

A toda a minha família e amigos que me deram tanto ânimo para acabar este projecto.

Ao meu irmão João e à Catarina por estarem sempre disponíveis e me ajudarem tanto.

À minha Mãe por sempre me apoiar em tudo e estar sempre ao meu lado.

Ao meu Pai que apesar de não ter conseguido ver-me chegar ao fim do curso sempre me deu força e motivação para estudar e sempre me dizia que eu era capaz.

E a todos que de alguma forma se cruzavam no meu caminho neste período e sempre me disseram para ter força e andar para a frente com o trabalho.

Resumo

O desenvolvimento de qualquer processo da indústria química implica o projecto e dimensionamento de um número elevado de equipamentos. Uma vez que no projecto é necessário ter em conta vários factores e heurísticas torna-se pertinente ter essa informação bem organizada e acessível. Os sistemas especialistas, por terem a capacidade de manter o conhecimento organizado e separado, podem ser ferramentas importantes no apoio ao projecto de equipamento utilizado na indústria química.

Os sistemas especialistas são aplicações informáticas que apresentam, dentro de um domínio específico, um grau de experiência na resolução de problemas que é comparável com o dos especialistas humanos.

Os sistemas especialistas são usados em variados campos para facilitar o trabalho de quem os utiliza de modo a ter a informação acessível e responder às necessidades do utilizador. Estes sistemas funcionam normalmente como auxiliares à tomada de decisões e na instrução.

Assim, este trabalho tem como objectivo estabelecer as linhas base do desenho de sistemas especialistas no apoio ao projecto base de equipamento da indústria de processo químico. Foram desenvolvidos vários exemplos usando duas plataformas diferentes: Visual Prolog e Ms-Access. Foram demonstradas as vantagens e desvantagens de cada uma das aplicações.

Em Visual Prolog desenvolveu-se um exemplo sobre sistemas de separação. O utilizador é questionado sobre as características do processo e conforme as suas respostas o sistema especialista segue para uma solução.

No Ms-Access foram desenvolvidos três exemplos. Um exemplo sobre sistemas de separação, outro sobre tanques de armazenamento e um sobre o cálculo de permutadores de calor.

Comparando as duas aplicações utilizadas, chegou-se à conclusão de que o Ms-Access é a melhor alternativa, relativamente ao Visual Prolog.

Palavras Chave (Tema): Sistema especialista, Projecto de equipamento, Visual Prolog, Ms-Access, *Visual Basic for Aplications*

Abstract

The development of any process in chemical industry involves the design of a high number of equipments. Once that it has to be taken in account many factors and heuristics it becomes pertinent to have that information well structured and easily reachable. The expert systems, because they have the ability of keeping the knowledge organized and separated, can be important tools in backing up the project design. Expert systems can be defined as being the informatics applications that have, in a specific field, a high degree of experience solving problems that is equivalent with the one from human experts.

Expert systems are used in many fields to make easier the work of the users in a way that they have accessible information and response to their needs. These systems usually work as support to make decisions and instruction.

The objective of this work is to create the base guide lines to the design of expert systems for supporting the design of chemical industry equipments. Several examples were developed using two different platforms: Visual Prolog and Ms-Access. After using these two programs there can be explained some advantages and disadvantages using one or other.

In Visual Prolog it was developed and example about separation systems. The user answers some questions about the process characteristics and the expert system gives the user a solution by analyzing the answers.

In Ms-Access there were three different developed examples. One about separation systems also, another about storage tanks and a last one that calculates an heat exchanger.

Comparing the two used applications, it can be concluded that Ms-Access is a better choice than Visual Prolog.

Índice

1 Introdução

1.1 Enquadramento e Apresentação do Projecto

Segundo Walas et al (1990), o projecto de um processo químico implica, na realidade, o projecto de uma sequência de operações físicas e químicas. Definem-se as condições operatórias, especificações de equipamentos, materiais de construção dos equipamentos utilizados no processo, a localização dos equipamentos na instalação industrial de forma a obter o melhor funcionamento destes, etc. O projecto do processo é resumido num diagrama de processo, em balanços materiais e energéticos e num conjunto de especificações de equipamentos.

A estratégia de projecto preliminar do equipamento pode seguir duas linhas: desenho proprietário ou desenho personalizado. O desenho proprietário é aquele que é projectado pelo fabricante de modo a atingir as especificações de desempenho requeridas pelo utilizador. Nesta categoria inclui-se normalmente equipamento que possui partes amovíveis como bombas, compressores, torres de arrefecimento, secadores, filtros, misturadores e agitadores, tubagens, válvulas e até mesmo aspectos estruturais de permutadores de calor, fornalhas, e outros equipamentos. O desenho personalizado é utilizado na definição de especificações de reactores químicos, na maioria dos reservatórios, nos destiladores/fraccionadores, e em outros equipamentos que tenham medidas estandardizadas no mercado.

Durante o projecto preliminar de um processo apenas algumas características do equipamento são importantes. Por exemplo, numa bomba o projecto do processo identifica as condições operatórias, capacidade, diferença de pressão, NPSH, os materiais de construção, etc, não sendo prioritário o cálculo da espessura da parede, do revestimento e do tamanho da embocadura.

O desenho proprietário deve ser evitado, pois acarreta normalmente custos muito elevados. Assim, o projecto de equipamento deve, sempre que possível, seguir as medidas padrões existentes. Mesmo o equipamento que é, muitas vezes, desenhado por medida, como os reservatórios, deve ser ajustados aos padrões, como por exemplo a medidas normais de diâmetros de cabeças, bocas de tubos, tipos de tabuleiros e enchimentos. O governo, as agências, as companhias de seguros estipulam muitas normas e padrões. Alguns padrões adoptados por alguns fabricantes são escolhas feitas para simplificar a construção, manutenção e a reparação. Como exemplos tem-se o limitar do tamanho dos tubos de um permutador de calor, a padronização das bombas centrífugas e as restrições a nível do

equipamento de controlo de um processo. É da competência dos engenheiros que projectam o processo e o equipamento fazerem todas as restrições.

A existência de uma aplicação informática que auxilie a tomada de decisões a nível da escolha do equipamento e dimensões padrão respectivas é uma ferramenta importante no apoio ao projecto de equipamento.

Durante séculos a população enriquecia graças ao esforço físico do seu povo. A riqueza estava dependente da quantidade e qualidade de trabalho que as pessoas executavam. Depois deu-se a revolução industrial onde as máquinas alimentadas a vapor e óleo foram desenvolvidas para auxiliar nas tarefas que exigiam mais esforços, aumentando exponencialmente a produtividade. Historicamente é possível verificar que as nações que hoje são líderes são as que no passado utilizaram estas tecnologias (Durkin, 1994).

Hoje em dia, a riqueza vem dos recursos intelectuais. Os povos com pessoas dotadas para a ciência, medicina, negócios e engenharia produzem inovações que levam a uma melhor qualidade de vida. Para ser possível explorar ao máximo estes recursos, procura-se agora novas máquinas capazes de adquirir o conhecimento destes indivíduos especialistas. Ou seja, hoje em dia procuram-se máquinas que não trabalhem a vapor mas sim a *conhecimento*. Deste modo, onde os especialistas não existem ou não se podem deslocar, existem máquinas capazes de resolver os problemas que só estes indivíduos seriam capazes.

De acordo com Durkin (1994), a procura destas máquinas inteligentes fez com que se desenvolvesse um novo campo de estudos: a inteligência artificial.

Inteligência artificial: Uma área de estudos das ciências computacionais que tem como objectivo conseguir com que um computador “pense” de um modo semelhante aos seres humanos.

Ou seja, em inteligência artificial procura-se desenvolver aplicações informáticas que funcionem com um grau de inteligência similar à do ser humano. Um objectivo mais prático da inteligência artificial é tornar os computadores mais úteis para os humanos. Ajuda também a perceber melhor a inteligência humana uma vez que ao desenvolver um sistema computacional inteligente é necessário entender como é que os seres humanos obtêm, organizam e usam o conhecimento durante a resolução de problemas.

Os sistemas especialistas são aplicações informáticas que contêm conhecimento de peritos para resolver determinado tipo de problemas. Por exemplo, os sistemas especialistas são utilizados em aplicações de diagnóstico tanto destinadas a seres humanos como a máquinas. Também podem jogar xadrez, tomar decisões de planeamento financeiro, controlar sistemas em tempo real e muitos outros serviços que necessitam de conhecimento especializado.

Um sistema especialista está organizado de modo a ter o conhecimento sobre determinado domínio (na base de conhecimentos) separado do outro conhecimento mais geral para resolver problemas ou do código para interagir com o utilizador (no motor de inferência). As aplicações com o conhecimento organizadas desta forma são designadas por sistemas baseados em conhecimento. Pela Figura 1 é possível constatar que os sistemas especialistas são sistemas baseados no conhecimento, enquanto o contrário não é necessariamente verdade. Outros sistemas baseados no conhecimento são, por exemplo, sistemas baseados em regras e sistemas de gestão de bases de dados.

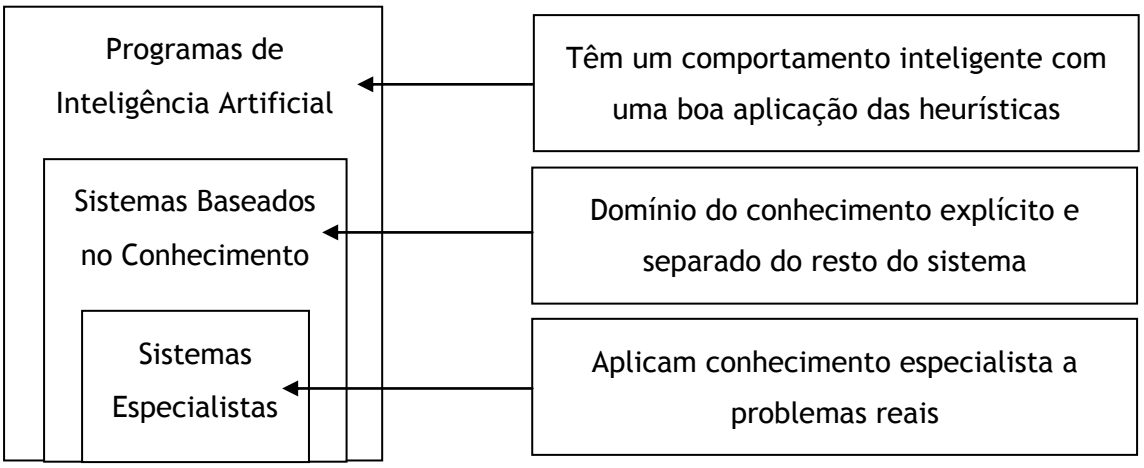

Figura 1: Enquadramento dos sistemas especialistas nos programas de inteligência artificial (Adaptado de Gonzalez e Dankel, 1993).

O sistema especialista que é capaz de projectar equipamento utilizado na indústria química é baseado em heurísticas e conhecimento que só alguns engenheiros e literatura possuem.

1.2 Contributos do Trabalho

Um sistema especialista capaz de decidir e projectar equipamento utilizado na indústria química poupa bastante tempo ao engenheiro. Assim, quando se faz o projecto de um equipamento, além de ser necessário menos tempo também há uma menor probabilidade de cometer erros e a possibilidade de aprender com a aplicação.

O objectivo deste trabalho é esclarecer o que é e como pode ser construído um sistema especialista aplicado à escolha de equipamento. Estudar como é constituído um sistema especialista e as ferramentas necessárias para a sua construção.

1.3 Organização da Tese

Esta tese está dividida em seis capítulos.

No Capítulo 1 apresenta-se o projecto e o seu enquadramento. Enumeram-se algumas limitações dos sistemas especialistas e as formas actualmente existentes que permitem atenuar essas limitações.

No Capítulo 2 referem-se vários sistemas já existentes em diversas áreas.

No Capítulo 3 descreve-se com maior detalhe o desenvolvimento dos sistemas especialistas.

O Capítulo 4 refere-se ao desenvolvimento de sistemas especialistas aplicados à escolha de equipamento utilizado em processos químicos.

No Capítulo 5 enquadra-se o desenvolvimento do projecto e os resultados obtidos com os objectivos iniciais do trabalho.

Por fim, no Capítulo 6 apresenta-se uma análise auto-crítica e faz-se algumas sugestões para trabalho futuro.

2 Estado da Arte

Os sistemas especialistas têm a capacidade de executar algumas tarefas quando têm de resolver determinados problemas como monitorização ou planeamento. Independentemente da área de aplicação, dado o tipo de problema, o especialista armazena e interpreta a informação de uma forma semelhante. Os sistemas especialistas são igualmente criados para realizar tarefas genéricas com base no tipo de problemas dos quais se dá seguidamente alguns exemplos.

Os sistemas de **interpretação**, descrevem a situação de factos observáveis. Esta informação consiste em dados provenientes de fontes como sensores, instrumentos, resultados de testes, etc. Nesta categoria inclui-se a vigilância, análise de imagens, estudo das estruturas químicas, interpretação de sinais, e muitos tipos de análise de inteligência. O sistema FXAA (AI WEEK 1988) presta assistência na negociação de divisas do "Chemical Bank" sediado em Manhattan que realizada centenas de transacções diárias. O FXAA foi criado para identificar transacções irregulares, trabalho que antigamente era feito manualmente por auditores.

Os sistemas de **previsão** são capazes de deduzir consequências prováveis de determinada situação. Estes sistemas tentam prever eventos futuros com base em informação disponível e um modelo do problema. São utilizados na meteorologia e previsões de trânsito. Os modelos têm de conseguir descobrir como é que uma determinada acção influencia futuros eventos, como por exemplo prever os estragos numa plantação por uma praga de insectos como é o caso do PLANT (Boulanger, 1983). Este sistema prevê os estragos numa plantação de milho provocados por minhocas. O sistema recebe dados relacionados com o estado actual da plantação, incluindo informação sobre a quantidade de ervas daninhas, condição da terra e a qualidade do milho que está plantada.

Os sistemas **controlo** verificam o comportamento do sistema atendendo às especificações do problema. Para isso o sistema tem de estar constantemente a interpretar a situação, prever o futuro, diagnosticar causas de problemas prováveis, estabelecer um plano de recuperação e controlar a sua execução. O sistema VM desenvolvido por Fagan monitoriza o estado dos pacientes nos cuidados intensivos hospitalares e controla o tratamento do paciente. O sistema mede as batidas cardíacas, pressão sanguínea e o estado de operação de um ventilador ligado ao paciente. Juntando esta informação com o histórico médico do paciente o sistema consegue ajustar o funcionamento do ventilador (Fagan, 1978).

Os sistemas de **projecto** são capazes de configurar objectos sujeitos a determinadas condições seguindo uma série de passos, cada um com a sua condição específica. São usados

no esboço de circuitos, em projectos de construção e para planear orçamentos. O sistema especialista PEACE (Dincbas, 1980) foi desenvolvido para auxiliar os engenheiros no projecto de circuitos electrónicos. Este sistema consegue sintetizar o circuito em etapas definidas, o que permite completar as especificações do design obedecendo às restrições do problema.

Há sistemas que planeiam acções para atingir determinado objectivo de acordo com as restrições do problema. Alguns sistemas de **planeamento** têm de ter flexibilidade para alterar o conjunto de tarefas planeadas quando têm novas informações sobre o problema. Para isso, têm de ser capazes de recuar e rejeitar a linha de raciocínio actual para poderem explorar uma solução melhor. O PLANPOWER (Stansfield e Greenfield, 1987) fornece uma vasta gama de planos financeiros. O sistema cobre uma gama de produtos financeiros incluindo seguros, valores mobiliários, imobiliários e incentivos fiscais. O sistema PLANPOWER gera planos financeiros individuais, que incluem recomendações específicas de acordo com a situação financeira do cliente por um período de 5 anos.

Os sistemas de **monitorização** comparam a informação observável do comportamento de um sistema com os estados do sistema que são considerados fundamentais para o seu bom funcionamento. Normalmente interpretam sinais de sensores e comparam a informação com os estados conhecidos. Quando um estado fundamental é detectado, começa a executar uma série de tarefas pré-estabelecidas. Estes sistemas são normalmente usados em centrais nucleares, tráfego aéreo, doenças, etc. O sistema NAVEX (Marsh, 1984) monitoriza os dados de radar e estima a velocidade e a posição da nave espacial. O sistema detecta qualquer erro e prevê se pode ocorrer algum problema. Se for detectado algum erro, o sistema dá recomendações para solucionar o problema.

Os sistemas de **diagnóstico** são capazes de tirar conclusões sobre o mau funcionamento de um sistema através de observações. A maior parte dos sistemas especialistas de diagnóstico têm informações sobre possíveis condições de falha de modo a poder descobri-la com base em observações. São usados no diagnóstico médico, electrónico, mecânico e de software. Alguns destes sistemas dão ainda uma solução para o problema como por exemplo, qual o melhor remédio para o paciente. NEAT (MIS Week, 1989) é um sistema especialista que os funcionários não técnicos de um apoio técnico a equipamentos de rede e telecomunicações utilizam. Os utilizadores que têm problemas com os terminais ou aplicações telefonam para o apoio técnico e o NEAT isola, diagnostica e resolve as situações. Começa por pedir ao técnico do apoio pelo número de identificação o terminal avariado. Depois de consultar a base de informação, o sistema determina a configuração do terminal em questão. O sistema continua a fazer várias perguntas ao utilizador e depois sugere alguns diagnósticos simples. O funcionário do *help desk* introduz os resultados no sistema especialista que imediatamente propõe uma reparação.

Os sistemas de **instrução** guiam a aquisição de conhecimentos de um utilizador em determinado tema. Estes sistemas tratam o utilizador como um sistema que necessita de ser diagnosticado e reparado. Começam por interagir com o utilizador de modo a perceber que conhecimentos o estudante já tem sobre o tópico. Depois compara o modelo que formou com o modelo do estudante "ideal" para descobrir onde o estudante está a falhar. De seguida dá informações ao estudante sobre as questões que falhou. O GUIDON (Clancey, 1979) é um sistema especialista utilizado por estudantes de medicina no tratamento de pacientes com infecções bacterianas. O sistema ensina os alunos a seleccionar correctamente a terapia anti-microbiana. Começa com um caso já resolvido e apresenta-o ao estudante para ele o solucionar. O sistema analisa as soluções do estudante e as respostas às perguntas durante a sessão. Depois o sistema compara a resolução do estudante com a sua e as diferenças são utilizadas para ensinar o estudante.

Há sistemas que recomendam soluções para um dado problema do sistema. Este tipo designa-se por sistemas de **resolução** e primeiro faz um diagnóstico para determinar a natureza do problema. Os sistemas mais avançados têm ainda técnicas de planeamento e previsão para desenvolver soluções à medida. O sistema BLUE BOX (Mulsant e Schreiber, 1984) recomenda uma terapia apropriada para pacientes que sofrem de depressão. O sistema utiliza informações como os sintomas e a história do paciente para diagnosticar o tipo e a gravidade da doença. Com esta informação o sistema recomenda uma terapia para controlar a depressão.

3 Sistemas Especialistas

3.1 Introdução aos Sistemas Especialistas

Um sistema especialista resolve problemas que são normalmente desenvolvidos por indivíduos especialistas em determinado assunto. Um sistema especialista é uma aplicação informática que reúne o conhecimento e a experiência de um ou mais especialistas humanos de modo a ter essa informação disponível.

Um sistema especialista é uma aplicação informática desenvolvida para simular a capacidade de resolução de problemas de um especialista humano.

Para Durkin (1994) o sistema especialista acaba por substituir o especialista ou então auxiliar o perito em problemas mais complexos.

Pode ser definido como uma aplicação informática que apresenta, dentro de um domínio específico, um grau de experiência na resolução de problemas que é comparável com o do perito humano. Os sistemas especialistas precisam de explorar um ou mais mecanismos de raciocínio de modo a conseguirem aplicar o conhecimento que retêm nos problemas a resolver. Precisam ainda de um mecanismo que explique os passos que seguiram de modo a que os utilizadores percebam o que foi feito (Chen, 2005).

Na Tabela 1 apresenta-se uma comparação entre um indivíduo especialista e um sistema especialista.

Tabela 1- Comparação entre um perito humano e um sistema especialista

Factor	Indivíduo especialista	Sistema especialista
Tempo de trabalho	Horário laboral	Todas as horas
Localização geográfica	Localização específica	Em qualquer lugar
Segurança	Insubstituível	Substituível
Perecível	Sim	Não
Desempenho	Variável	Contínua
Velocidade	Variável	Contínua
Custos	Elevados	Acessível

O indivíduo especialista tem um horário laboral predefinido e o seu ritmo de trabalho é variável enquanto o sistema especialista pode estar sempre a funcionar, em vários locais, trabalha sempre do mesmo modo e a qualquer momento pode ser substituído por outro mecanismo se este se avariar. Um indivíduo especialista, como ser humano tem limitações

físicas, quando morre o conhecimento desaparece com ele e tem um custo de trabalho mais elevado do que o sistema especialista (Durkin, 1994).

3.2 Características dos sistemas especialistas

3.2.1 Estrutura dos sistemas especialistas

Os sistemas especialistas são divididos em componentes principais possuindo uma interactividade com seres humanos a diversos níveis como se ilustra na Figura 2.

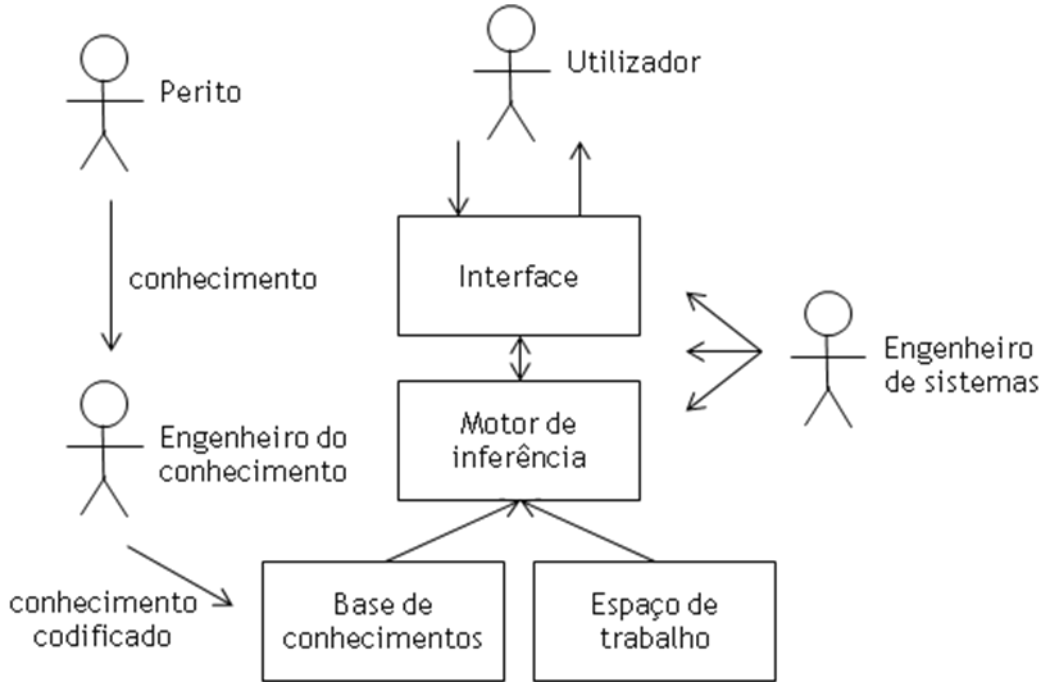

Figura 2 – Esquema de organização dos componentes de um sistema especialista e interactividade com as pessoas (Adaptado de Merrit, 1989).

Os dois componentes principais são a base de conhecimentos e o motor de inferência para que seja possível atingir os principais objectivos do sistema especialista: ter conhecimento especializado e raciocínio simulando o trabalho do perito.

No entanto, o sistema especialista não terá utilidade se não tiver uma boa interface com o utilizador e espaço para armazenar os dados do problema.

- Base de conhecimentos: Contém conhecimentos especializados na área do problema adquiridos através do perito humano. Inclui factos sobre o problema, regras, conceitos e relações. Por exemplo, pode ter o conhecimento de um mecânico para diagnosticar problemas em automóveis. É uma representação declarativa do conhecimento, normalmente feita com regras "se" (*if*) e "então" (*then*). Por exemplo, podem-se usar as seguintes regras para representar o conhecimento para um problema de diagnóstico automóvel:

REGRA 1

SE: O carro não pega.

ENTÃO: O problema pode estar no sistema eléctrico

REGRA 2

SE: O problema for no sistema eléctrico

E: A bateria estiver abaixo dos 10 volts

ENTÃO: A bateria está fraca.

- Espaço de trabalho: Armazena os dados que são específicos ao problema em questão. É a parte do sistema que contém os factos do problema que são revelados durante a sessão pelo utilizador. O sistema combina esta informação com a que vai consultar à sua base de conhecimento para deduzir novos factos. Estes novos factos são então introduzidos no espaço de trabalho e o processo de combinar informação continua. Eventualmente o sistema acaba por chegar a alguma conclusão que é também inserida no espaço de trabalho. Este espaço contém toda a informação sobre o problema, quer seja a introduzida pelo utilizador quer seja a que o sistema já contém. É possível ainda que esta informação seja introduzida no espaço de trabalho vindas de fontes exteriores de armazenamento de informação, como bases de dados, folhas de cálculo ou sensores.

- Motor de inferência: Moldado com base no raciocínio do perito, é o processador do conhecimento. O agente funciona com a informação do problema, juntamente com o conhecimento disponível na base. Deste modo consegue chegar-se a conclusões ou recomendações. O agente procura as regras para combinar entre as suas premissas e a informação disponível no espaço de trabalho. Quando encontra uma correspondência adiciona a conclusão tirada da regra ao espaço de trabalho e continua o processo de procura.

- Interface: O código que controla o diálogo entre o utilizador e o sistema informático. Um dos requisitos básicos da interface é que faça perguntas. Para que a informação recolhida do utilizador seja o mais precisa possível, é preciso ter um cuidado especial no modo como as questões são feitas. Assim convém que o programa funcione através de menus ou gráficos. Pode ainda haver a possibilidade de o utilizador querer visualizar ou alterar a informação que está no espaço de trabalho, o que se torna útil quando o utilizador quer alterar uma resposta anterior.

É necessário compreender a influência que os indivíduos têm no funcionamento do programa para se perceber o que é o sistema especialista.

- Perito na área: o indivíduo que é actualmente o especialista na matéria do problema capaz de resolver os problemas que é suposto o sistema especialista resolver.
- Engenheiro de conhecimento: o trabalho do engenheiro é recolher o conhecimento do perito e inseri-lo em forma de código na base de conhecimento do sistema.
- Utilizador: o indivíduo que utiliza o programa para encontrar soluções para o seu problema, que podiam ter sido dadas pelo perito.
- Engenheiro de sistemas: o indivíduo que desenvolve a interface, projecta o formato da declaração da informação na base de conhecimento e implementa o agente de inferência.

O engenheiro de sistemas pode, em determinadas situações, ser também o engenheiro de conhecimento. Na construção de um sistema específico, o projecto do formato da base de conhecimento e a programação do domínio da informação são semelhantes. O formato tem efeitos significativos na codificação do conhecimento (Merrit, 1989).

3.2.2 Características dos sistemas especialistas

Uma das vantagens de se utilizar os sistemas especialistas é a sua capacidade de explicar o seu raciocínio. Assim, o programa pode explicar o porquê de fazer determinada pergunta e como chegou a determinada conclusão. Esta é uma vantagem tanto para o utilizador como para o criador do sistema pois assim este consegue descobrir erros no sistema e corrigi-los.

- Explicar o "como": Tanto os peritos humanos como os sistemas especialistas conseguem explicar como chegaram a determinada conclusão. Esta habilidade é extremamente importante num sistema especialista pois ao contrário dos programas convencionais, que trabalham um problema bem definido, os sistemas especialistas lidam normalmente com problemas mais complexos. Esta situação leva, muitas vezes, a que se questione a validade das respostas do sistema, o que requer uma justificação a apoiar os resultados. Utilizando o exemplo anterior do automóvel, um perito, para explicar o resultado da bateria estar fraca, poderia dar uma explicação como: Uma vez que o carro não pega, assumi que era um problema no sistema eléctrico. Quando descobri que a bateria estava abaixo dos 10 volts, pude confirmar que o problema era mesmo a bateria estar fraca. Esta explicação demonstra o raciocínio do perito. O sistema especialista demonstra de uma forma semelhante pois apresenta as regras que o fizeram chegar àquela conclusão. Assim o utilizador percebe melhor o caminho seguido e acredita mais facilmente no resultado pois consegue ver o raciocínio feito.

- Explicar o "porque": Quando um indivíduo consulta um perito, a conversa é muito interactiva. Ou seja, a qualquer momento pode pedir para explicar o porque de estar a seguir aquela linha de raciocínio. Seguindo o mesmo exemplo do automóvel, quando o perito pergunta se o carro não pega, pode dizer que fez aquela questão pois se souber que o carro não pega, normalmente assume que o problema é do sistema eléctrico. Da mesma forma, quando questionado, o sistema especialista responderia mostrando a regra "if" e "then".

  Segundo Durkin (1994) um sistema especialista tem de ter um bom desempenho, ao nível de um perito. Mas, além de ter de descobrir as soluções dos problemas, tem de o fazer rapidamente, como os especialistas humanos. No entanto, fora da área de domínio, já não conseguem ser eficazes como por exemplo, se for um sistema especialista para diagnosticar problemas num automóvel, não é capaz de resolver problemas na área de diagnóstico médico.

Os sistemas especialistas mais eficazes são os que são mais específicos, ou seja, em vez de tentarem abranger uma área grande de assuntos, focam-se num só. Segundo Ham (1984), os sistemas especialistas desenvolvidos para actuarem sobre um vasto campo de conhecimentos não tiveram grande sucesso

Recorrendo novamente ao exemplo do automóvel, deve-se tentar dividir o problema. Faz-se dois programas distintos, um para o sistema eléctrico e outro para o combustível. Ainda assim, estas duas áreas devem ser divididas em problemas mais pequenos até ser possível trabalhar como está apresentado na Figura 3, para o exemplo do automóvel.

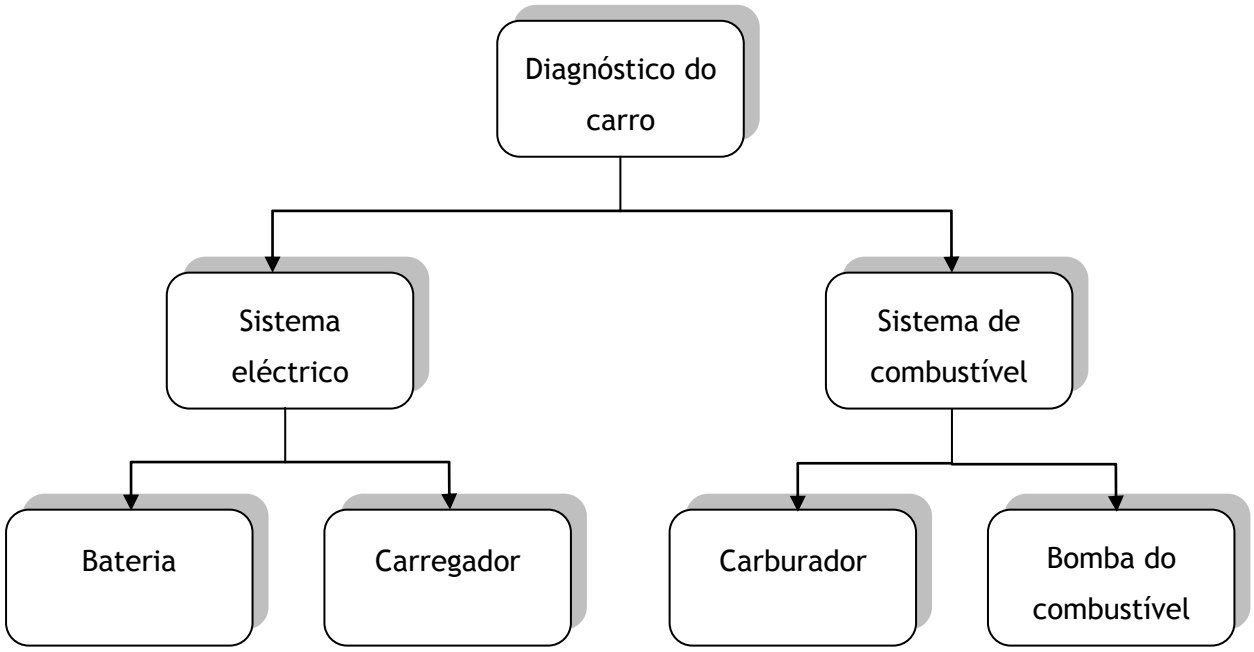

Figura 3 – Estrutura de um sistema para fazer o diagnóstico do carro (Adaptado de Durkin, 1994).

Como demonstrado na estrutura do sistema especialista, a base de conhecimento e o agente de inferência estão em módulos separados. Esta é uma vantagem dos sistemas especialistas e uma característica que os distingue dos programas convencionais.

Os programas convencionais juntam o conhecimento do programa com o controlo dele. Ou seja, o código está feito em conjunto e a sua alteração tem efeitos tanto no controlo como na informação. Além disso, torna-se mais difícil entender que informação existe e como está a ser utilizada, assim como rever o código e encontrar possíveis erros.

A base fundamental dos sistemas especialistas está no modo como o conhecimento está organizado e representado. O conhecimento é a informação que o programa informático necessita antes de poder funcionar como um especialista. Esta informação tem a forma de factos e regras, como se demonstra a seguir:

Facto: O tanque número 3 contém ácido sulfúrico.
Regra: Se o teste ao ião sulfato for positivo, o material derramado é ácido sulfúrico.

Muitas destas regras são heurísticas, ou seja, regras de algibeira (*rules of* thumb) ou simplificações (Waterman, 1983).

Uma vez que os sistemas especialistas executam tarefas complicadas e difíceis de entender, como por exemplo a resolução de um processo judicial, torna-se mais fácil se utilizarem heurísticas. Para Waterman (1983) as heurísticas têm tendência a impedir análises matemáticas rigorosas e soluções algorítmicas.

A vantagem dos sistemas especialistas relativamente aos sistemas convencionais é que os primeiros conseguem separar o conhecimento do controlo uma vez que a base de conhecimento e o agente de inferência são módulos distintos.

Os sistemas especialistas baseiam-se no conhecimento. Estes sistemas apresentam vantagens, assim como desvantagens, relativamente a outras soluções como a utilização de aplicações informáticas convencionais ou especialistas humanos.

Uma das vantagens é a larga distribuição de especialização escassa: Os sistemas baseados no conhecimento reproduzem o conhecimento e as capacidades que possuem os especialistas humanos (indivíduos que são considerados especialistas porque há muito poucos a saberem o seu conhecimento tão específico). Esta capacidade de reproduzir o conhecimento do especialista permite que a informação chegue a mais locais, por um preço razoável. Por exemplo, uma empresa pode ter um especialista nas questões legais da legislação fiscal. Se a empresa tiver vários grupos a necessitar do serviço deste especialista, poderá ser benéfico desenvolver um sistema especialista sobre as leis fiscais com base no conhecimento e nas capacidades do perito fiscal da empresa, providenciando a todos os grupos acesso ilimitado às aptidões do especialista. Enquanto o desenvolvimento deste sistema pode não ser trivial, será provavelmente mais barato do que contratar mais especialistas em legislação fiscal para fazer frente às necessidades da empresa.

O facto de se conseguir separar o conhecimento do mecanismo de raciocionio facilita o processo de modificação do conhecimento. Esta é uma característica importante na programação heurística onde podem acontecer alterações frequentemente. O exemplo anterior do perito fiscal ilustra este aspecto pois as leis fiscais são alteradas todos os anos. A facilidade em fazer alterações é uma característica extremamente vantajosa nestas situações.

Outra vantagem dos sistemas especialistas é que têm sempre consistência nas respostas. Os peritos humanos diferem, várias vezes, na resposta dada ao mesmo problema. Até o mesmo especialista humano pode, em diferentes ocasiões, diferir um pouco na resposta dada. Em alguns casos estas diferenças são pequenas inconsistências com poucas ou nenhumas consequências. Noutros casos são grandes falhas que podem fazer com que o especialista não seja tão confiante nas suas capacidades podendo afectar a sua saúde, estado emocional e cansaço. Por outro lado, os sistemas especialistas são sempre consistentes no seu modo de resolver problemas, dando sempre respostas idênticas. Não são influenciáveis por factores emocionais ou de saúde que podem afectar o seu desempenho.

Os sistemas especialistas permitem (quase sempre) uma acessibilidade completa. Trabalham 24 horas por dia, aos fins-de-semana e feriados e não ficam doentes nem vão de férias, ao contrário de qualquer ser humano.

Em situações em que o custo associado à formação de novos peritos numa determinada área é elevado, onde o especialista está doente ou em vias de se reformar, a experiência e competência do perito podem ser preservadas para a posteridade num sistema especialista.

Os sistemas especialistas, em parte por causa da sua natureza heurística, são capazes de resolver problemas onde não existem dados completos ou exactos. Esta é uma característica importante pois a informação completa e exacta raramente está disponível num problema no mundo real.

Em parte devido à sua natureza heurística, os sistemas especialistas rastreiam o conhecimento usado para chegar às soluções dos problemas. Deste modo os utilizadores curiosos ou com dúvidas podem consultar o sistema para terem explicações sobre como se chegou a determinada conclusão. Estas explicações ajudam o utilizador esclarecendo e justificando os resultados obtidos e, adicionalmente, instruindo o utilizador permitindo que fique mais competente.

No entanto, nem todos consideram um sistema especialista como a melhor ferramenta para resolver determinado problema. Têm algumas falhas nas quais os utilizadores devem atentar.

As respostas dos sistemas especialistas podem nem sempre estar correctas. Por vezes os peritos cometem erros, por isso, é natural que os sistemas especialistas também tenham falhas. No entanto estes erros podem ter consequências graves com custos elevados como é o caso do sistema especialista em legislações fiscais ou um sistema que monitorize um reactor nuclear.

Os sistemas baseados no conhecimento tentam sempre chegar a uma solução de um determinado problema, independentemente de o problema se encontrar ou não no domínio do sistema. Os sistemas informáticos não sabem as limitações do seu conhecimento, nem em que alturas o adquiriram. Assim, o sistema poderá dar respostas incorrectas e o utilizador pode não dar conta do erro. Pelo contrário, os especialistas humanos sabem as limitações do seu conhecimento e conseguem identificar se o problema está fora da sua área de experiência ou não.

Torna-se complicado representar num sistema baseado no conhecimento uma análise crítica ou um senso comum. Pode-se tentar incluir no sistema alguma capacidade crítica mas tem de ser feito explicitamente. Por exemplo, se no cálculo de uma coluna de destilação se obtiver uma altura de 5 km o sistema especialista não consegue identificar que não pode existir, a não ser que esteja explicito um limite máximo para alturas de colunas de destilação. Um exemplo mais geral será que em condições normais, não se pode ter água

líquida a -10 ºC. É do conhecimento geral que a água passa ao estado sólido abaixo de 0ºC. O sistema especialista só poderia saber isso se tivesse sido introduzido esse facto na sua base de dados. Pode-se compensar este problema introduzindo no sistema que a água só está no estado líquido a temperaturas positivas. No entanto, isto torna-se impraticável pois é necessário introduzir muitos dados para não ocorrerem estes tipos de erros (Gonzalez e Dankel, 1993).

3.3 Ferramentas dos Sistemas Especialistas

Há diferentes técnicas de representar o conhecimento que podem ser utilizadas por si só ou em conjunto umas com as outras para se construir o sistema. Cada técnica atribui à aplicação determinada particularidade diferente como ser mais eficiente, maior facilidade de o perceber, ou o facto de ser mais fácil de o alterar. As três técnicas mais utilizadas na construção dos sistemas especialistas são as regras (técnica utilizada neste trabalho), as redes semânticas e as estruturas.

4 Descrição Técnica e Discussão dos Resultados

4.1 Aquisição de conhecimentos em Visual Prolog

Prolog é uma linguagem computacional baseada na lógica (PROgramação em LOGica). Ao contrário de outras linguagens de programação o Prolog é declarativo, ou seja, a lógica do programa é expressa em termos de relações através de factos e regras.

Estas regras e factos (*Horn clauses*) são escritas dentro de *predicados*. Dentro dos *predicados* declaram-se as funções criadas, os seus argumentos e os tipos de dados que cada argumento pode receber. Estes *predicados* são definidos nas *clauses* ou seja, definem o que cada função realmente faz.

4.2 Exemplo desenvolvido em Visual Prolog: Escolha do sistema de separação apropriado para determinado processo

Como forma de utilizar a aplicação Visual Prolog, utilizou-se um exemplo que permite escolher o método de separação mais adequado de acordo com a informação apresentada na Tabela 2 (Tabela retirada dos apontamentos da unidade curricular Projecto de Engenharia do MIEQ, 2009). Apresenta-se a tabela em inglês uma vez que a aplicação também foi desenvolvida em inglês.

Tabela 2 – Métodos de separação e suas características.

Phase of the feed	Separation agent	Developed or added phase	Separation principle	Separation method
L and/or V	Pressure reduction or heat transfer	V and L	Difference in volatility	**Equilibrium flash**
L and/or V	Heat transfer or shaft work	V and L	Difference in volatility	**Distillation**
V	Liquid absorbent	L	Difference in volatility	**Gas absorption**
L	Vapor stripping agent	V	Difference in volatility	**Stripping**
L and/or V	Liquid solvent and heat transfer	V and L	Difference in volatility	**Extractive distillation**
L and/or V	Liquid entrainer and heat transfer	V and L	Difference in volatility	**Azeotropic distillation**
L	Liquid solvent	Second liquid	Difference in solubility	**Liquid-liquid extraction**
L	Heat transfer	Solid	Difference in solubility	**Crystalization**
C	Solid adsorbent	Solid	Difference in adsorbability	**Gas adsorption**
L	Solid adsorbent	Solid	Difference in adsorbability	**Liquid adsorption**
L or V	Membrane	Membrane	Difference in permeability and/or solubility	**Membranes**
S and L	Heat transfer	V	Difference in volatility	**Drying**
S	Liquid solvent			**Leaching**
L or V	Supercritical solvent	Supercritical fluid	Difference in solubility	**Supercritical extraction**

A aplicação Visual Prolog pode apresentar-se em três formas: abrir normalmente a aplicação, através do ficheiro projecto, ou através de um executável.

Se abrir o ficheiro executável, o Visual Prolog corre de seguida a aplicação desenvolvida.

Na situação de projecto visualiza-se o IDE (*Integrated Development Environment*). O IDE apresenta as componentes normais de uma aplicação do Windows como menus e janelas, funcionando também como um auxiliar de desenvolvimento da própria aplicação. O Visual Prolog considera que o que se está a desenvolver é um projecto (como no Word é um documento ou em Excel uma folha de cálculo). No fundo, o projecto é a própria aplicação desenvolvida.

Quando se abre Visual Prolog visualiza-se a árvore do projecto (*Project* Tree). Com esta árvore esquemática tem-se uma visão geral de como estão organizados os ficheiros no projecto. Como se pode ver na Figura 4, sabem-se em que directórios estão os ficheiros e os módulos do projecto.

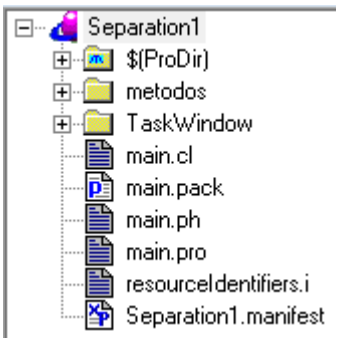

Figura 4 - "Project Tree" do Projecto "Separation1" em Visual Prolog.

No caso do exemplo do sistema de separação, o projecto chama-se "Separation1". A criação de um novo projecto implica que o IDE crie automaticamente o directório "$(ProDir)", o "TaskWindow" e o ficheiro "main.pack" que contém os elementos básicos do projecto e o "Separation1.manifest". Este último é criado com o nome do projecto e contém informações sobre a aplicação.

Para criar a aplicação para o exemplo da escolha do sistema de separação, tem de se utilizar o gerador do IDE que cria o código de Prolog necessário baseado nas declarações que tiverem sido introduzidas. Apesar de neste ponto ainda só se ter declarado o nome do projecto, o IDE considera que é uma aplicação do sistema Windows e por isso gera um código semelhante ao de uma aplicação de Windows com uma janela principal e os menus Ficheiro, Editar, etc, como noutras aplicações de Windows.

Quando é gerado o código (opção *Build*) são acrescentados os restantes ficheiros apresentados na Figura 4 na árvore do projecto. O ficheiro "main.pro" constitui o elemento central do projecto.

A realização dos procedimentos anteriores permite a criação de um ficheiro executável, que corresponde a uma aplicação com os menus Ficheiro, Editar, idêntica a outras aplicações informáticas (Ms-Excel, Ms-Word). Esta é uma das vantagens de utilizar o Visual Prolog. A aplicação praticamente constrói-se por si, sendo que o utilizador necessita de introduzir as especificações que deseja e acrescentar código ou modificá-lo se necessário. Na Figura 5 apresentam-se os ficheiros criados dentro da pasta "métodos".

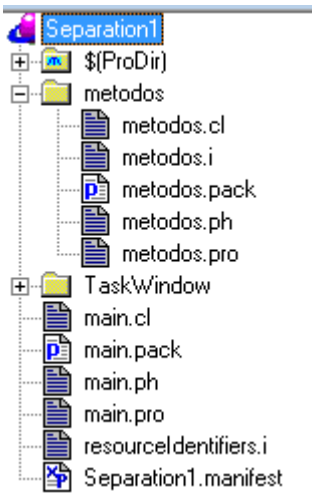

Figura 5 - "Project Tree" com expansão da pasta métodos.

No ficheiro "metodos.pro" declaram-se os dados da Tabela 2, sob a forma de factos. Escrevem-se como se estivessem numa tabela. No entanto é preciso contemplar todas as hipóteses, nomeadamente quando várias combinações possíveis levam ao mesmo resultado e por isso é necessário declarar um facto diferente por cada combinação diferente, como se pode ver no exemplo a seguir para o método destilação:

```
assert(method("Distillation", ["L"], ["Heat transfer"],["L","V"], ["Difference in volatility"])),
assert(method("Distillation", ["V"], ["Heat transfer"],["L","V"], ["Difference in volatility"])),
assert(method("Distillation", ["L","V"], ["Heat transfer"],["L","V"], ["Difference in volatility"])),
assert(method("Distillation", ["L"], ["Shaft work"],["L","V"], ["Difference in volatility"])),
assert(method("Distillation", ["V"], ["Shaft work"],["L","V"], ["Difference in volatility"])),
assert(method("Distillation", ["L","V"], ["Shaft work"],["L","V"], ["Difference in volatility"])),
```

A parte do conhecimento existente é colocada neste ficheiro. Ou seja, este ficheiro é a base de conhecimentos.

Toda a parte de programação e lógica é feita nos ficheiros dentro do directório "TaskWindow" apresentados na Figura 6:

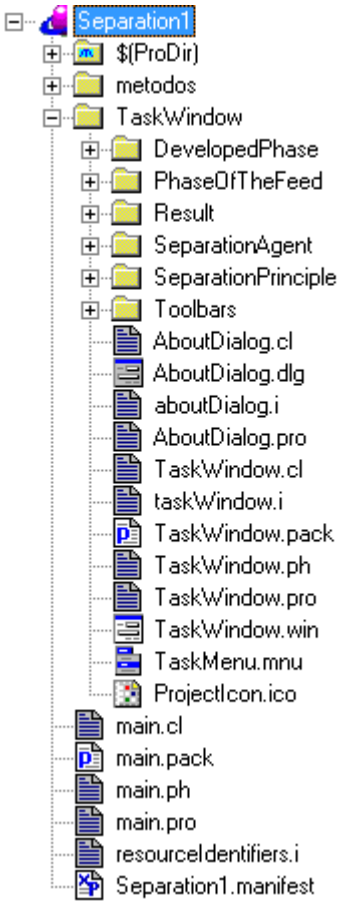

Figura 6 - "Project Tree" com expansão da pasta TaskWindow.

Dentro deste directório foi criada uma pasta para cada característica do sistema de separação (ou seja, para cada coluna da tabela). Cada pasta corresponde a uma janela de interacção com o utilizador. Dentro de cada pasta existem os mesmos ficheiros como se vê na Figura 7:

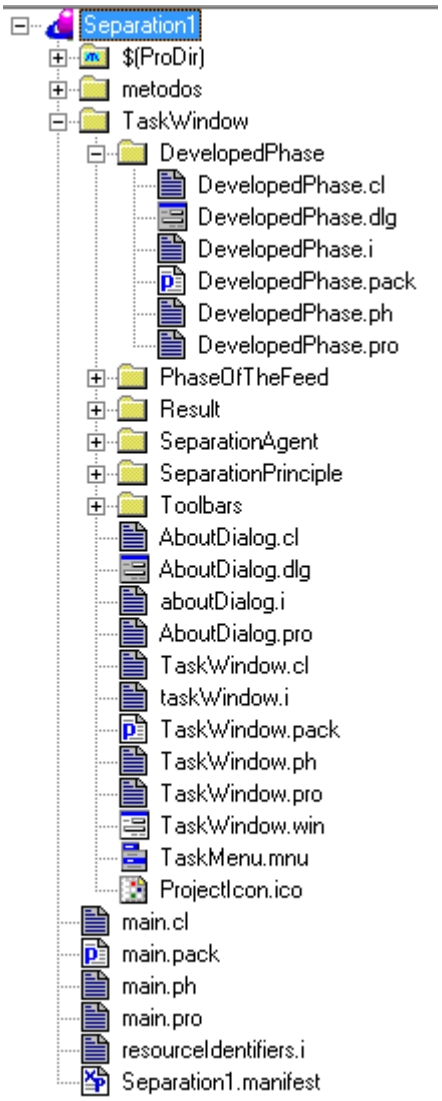

Figura 7 - "Project Tree" com expansão da pasta DevelopedPhase.

Cada pasta tem os ficheiros aplicados ao seu critério ou seja, tem o mesmo conteúdo.

No ficheiro "DevelopedPhase.dlg" define-se a interface gráfica da janela. O IDE do Visual Prolog contém um editor gráfico que facilita a criação de interfaces gráficas através da escolha de botões pré-definidos, opções de controlo, caixas de texto, tamanhos e tipos de letra, ícones, etc. Todas estas propriedades são ajustáveis. As janelas de construção da interface gráfica apresentam-se na Figura 8:

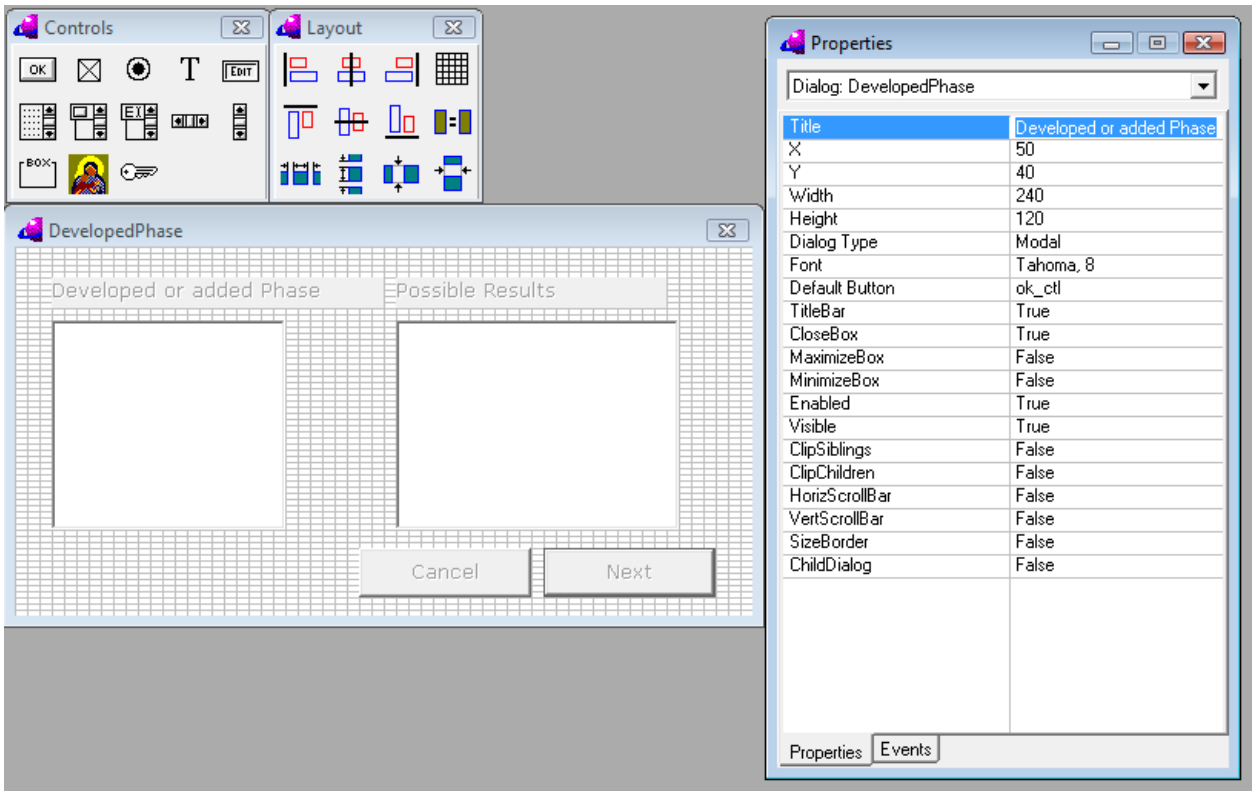

Figura 8 – Editor gráfico do Visual Prolog.

Estes passos tiveram de ser feitos para cada uma das opções da tabela.

No ficheiro "DevelopedPhase.pro" define-se a parte lógica da aplicação correspondente à pergunta em questão. Cada janela de cada pergunta (Figura 9) começa por inicializar a parte visual e preencher o campo do lado esquerdo com os valores possíveis.

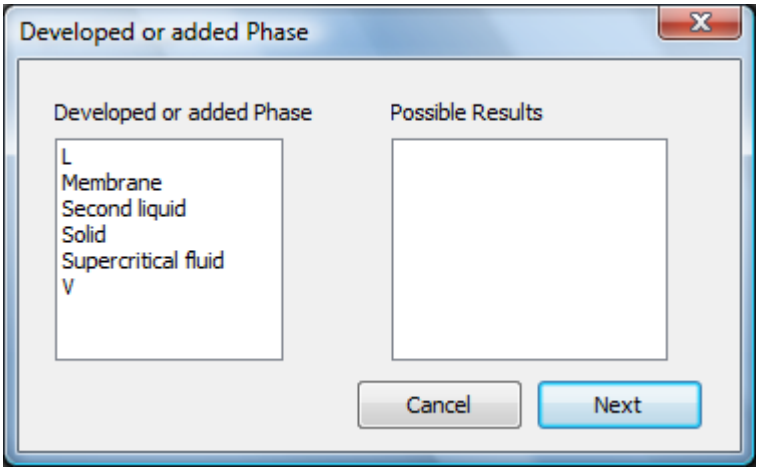

Figura 9 – Janela de questão da "Developed or added phase".

Para o exemplo do critério *Developed or added phase*, escreve-se nas clauses:

```
predicates
   inicializeDevelopedPhase : ().
clauses
   inicializeDevelopedPhase():-
      metodos::populateField("DevelopedPhase",List),
      listbox_ctl:addList(List).
```

A função *populateField* retorna na variável *List* todas as opções disponíveis de resposta definidas na parte de conhecimento do ficheiro *metodos.pro*.

```
class facts
   populate : (string Field, string Id).

clauses
   classInfo(className, classVersion).

   clauses
      fillDatabase() :-
         assert(populate("PhaseOfTheFeed", "L")),
         assert(populate("PhaseOfTheFeed", "V")),
         assert(populate("PhaseOfTheFeed", "S")),
         assert(populate("DevelopedPhase", "V")),
         assert(populate("DevelopedPhase", "L")),
         assert(populate("DevelopedPhase", "Second liquid")),
         assert(populate("DevelopedPhase", "Solid")),
         assert(populate("DevelopedPhase", "Membrane")),
         assert(populate("DevelopedPhase", "Supercritical fluid")),
         assert(populate("SeparationAgents", "Pressure reduction")),
         assert(populate("SeparationAgents","Heat transfer")),
         assert(populate("SeparationAgents","Shaft work")),
         assert(populate("SeparationAgents","Liquid absorbent")),
         assert(populate("SeparationAgents","Vapor stripping agent")),
         assert(populate("SeparationAgents","Liquid solvent")),
         assert(populate("SeparationAgents","Liquid entrainer")),
         assert(populate("SeparationAgents","Solid adsorbent")),
         assert(populate("SeparationAgents","Membrane")),
         assert(populate("SeparationAgents","Supercritical solvent")),
         assert(populate("SeparationPrinciples", "Difference in volatility")),
         assert(populate("SeparationPrinciples","Difference in solubility")),
         assert(populate("SeparationPrinciples","Difference in adsorbability")),
         assert(populate("SeparationPrinciples","Difference in permeability")).
```

Para que ao clicar numa opção de resposta no campo do lado esquerdo, aparecessem os resultados possíveis do lado direito (o método para qual se usa aquele critério), criou-se uma função *onListboxSelectionChanged*:

```
predicates
   onListboxSelectionChanged : listControl::selectionChangedListener.
clauses
   onListboxSelectionChanged(Source):-
      listbox_ctl:getAllSelected(PhaseList,PhaseIndex),
      metodos::perguntaDevAddPhase(PhaseList,MList),
      MetodosList = removeDuplicates(MList),
      listbox1_ctl:clearAll(),
      listbox1_ctl:addList(MetodosList).
```

Sempre que o utilizador selecciona ou não um valor do lado esquerdo, o campo do lado direito é actualizado com os resultados possíveis.

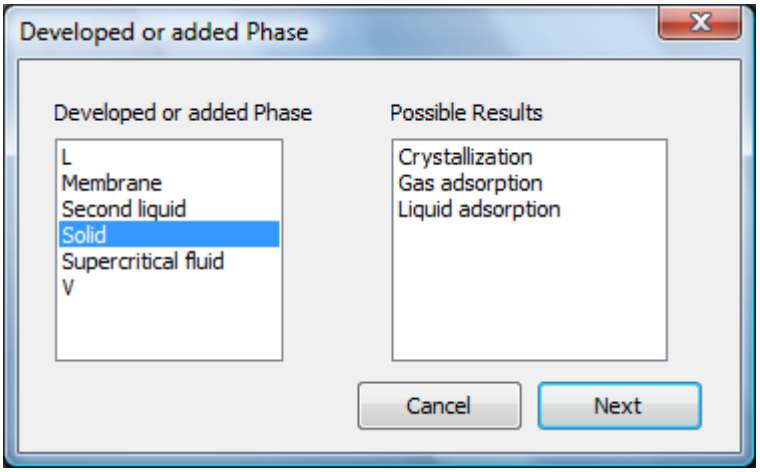

Figura 10 - Janela da questão da "Developed or added phase" com opções seleccionadas e possíveis resultados.

Quando se clica no botão *Next* é accionada a função *onOkClick*:

```
predicates
    onOkClick : button::clickResponder.
clauses
    onOkClick(_Source) = button::defaultAction :-
        listbox_ctl:getAllSelected(PhaseList,_),
        _ = separationPrinciple::display(getParent(), feedPhase, sepAgent, PhaseList).
```

Esta função armazena os valores seleccionados pelo utilizador e avança para o critério seguinte. No caso da janela do *Developed or added phase* avança para o critério *Separation Principle*.

A última janela a aparecer é a do resultado de todas as escolhas do utilizador. Existe uma pasta *Result* dentro da *TaskWindow* que contém os mesmos ficheiros das outras pastas de critérios, mas no entanto não têm o mesmo conteúdo.

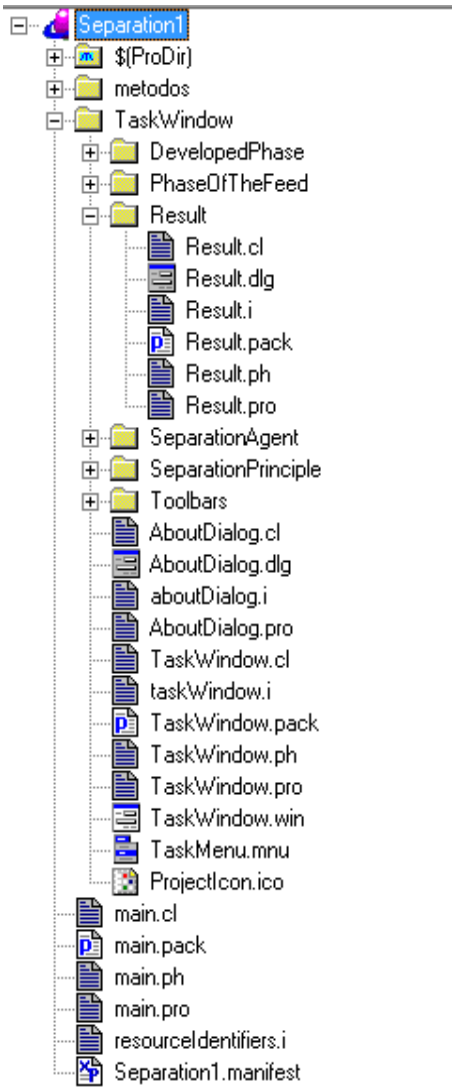

Figura 11 - "Project Tree" com expansão da pasta Results.

Nesta fase, a aplicação faz uma pesquisa às respostas dadas pelo utilizador armazenadas e apresenta-as na janela de resultados. Ao mesmo tempo procura um método comum a todas as opções seleccionadas. Se houver um método comum, existe uma solução. Caso contrário, isto é, se não houver um método comum a todos os critérios aparece uma mensagem com a indicação que nenhum método foi encontrado.

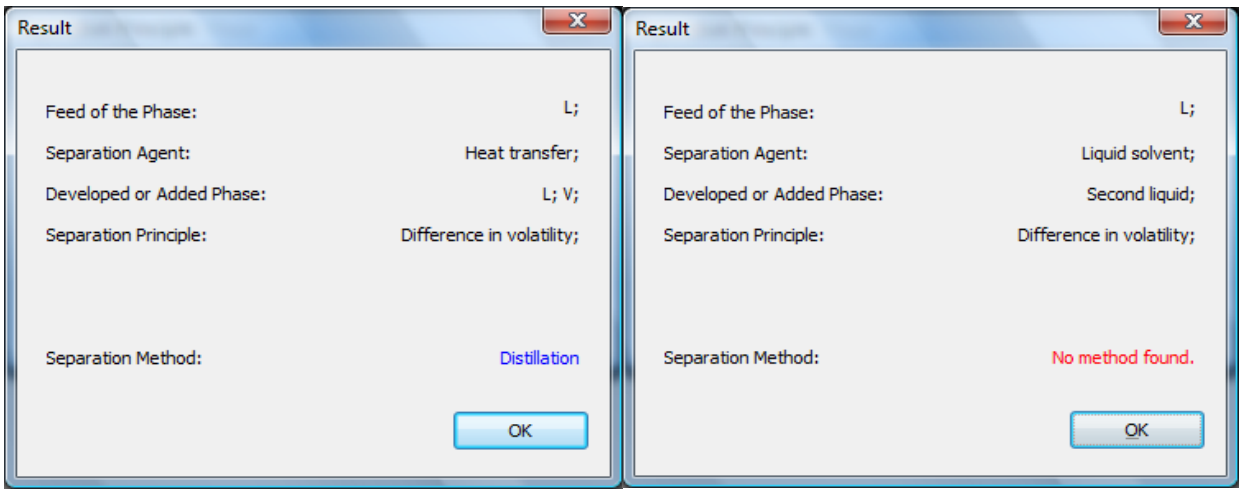

Figura 12 - Janelas de resultados. Direita: Encontrou um resultado; Esquerda - Não encontrou nenhum resultado que satisfaça os critérios seleccionados.

4.3 Aquisição de conhecimentos em Microsoft Access

Trabalhar em Visual Prolog foi bastante complicado. Um dos motivos para esta dificuldade foi facto de a linguagem utilizada não ser conhecida. Não foi possível, com o tempo disponível, adquirir conhecimentos informáticos suficientes para construir uma aplicação apresentável, com diversos exemplos, possibilidade de inserir novos dados, botões de ajuda, etc. Por estes motivos começaram-se a explorar outras opções, como por exemplo o Ms-Access.

Como descrito por Simpson et. al. (2007):

> O Microsoft Access (Ms-Access) é um programa utilizado para fazer gestão de bases de dados, que oferece muitas opções de gestão de dados (informação). O Ms-Access é usado normalmente para gerir *mailing lists*¸*memberships*, dados científicos e estatísticos, pequenos negócios, e praticamente qualquer outra coisa que envolva o armazenamento e gestão de grandes quantidades de informação.

Os ficheiros criados em Ms-Access são guardados com o formato de base de dados (*.accdb*). As bases de dados do Ms-Access são feitas de *objectos*. Os *objectos* podem ser criados, editados e eliminados, cada um com o seu nome único e definições próprias. O Ms-Access trabalha com vários tipos de objectos incluindo objectos para armazenar, visualizar e imprimir dados, assim como objectos feitos de programas escritos pelo utilizador.

Para armazenar dados usam-se *tabelas*, para editar dados no computador usam-se *formulários*, para imprimir usam-se *relatórios*, e para seleccionar e cruzar dados usam-se *queries*. Podem ainda criar-se *macros* e *módulos* que são rotinas escritas pelo utilizador em linguagem de desenvolvimento Visual Basic for Applications.

As *tabelas* são os objectos onde se armazenam os dados. Uma tabela é um objecto do Ms-Access constituído por vários registos. Cada registo contém informação no mesmo formato. Por exemplo, numa lista de endereços cada registo contém informação sobre o nome da pessoa, a morada, o telefone, etc. Cada fragmento individual de informação (por exemplo o nome próprio ou a morada) é um *campo*. A Figura 13 apresenta uma tabela usada neste trabalho. Cada linha corresponde a um *registo* e cada coluna a um *campo* (atributo).

Separation method	Phase of the feed	Separation agent	Developed or added phase	Separation principle
Equilibrium flash	L and/or V	Pressure reduction or heat transfer	V and L	Difference in volatility
Distillation	L and/or V	Heat transfer or shaft work	V and L	Difference in volatility
Gas absorption	V	Liquid absorbent	L	Difference in volatility
Stripping	L	Vapor stripping agent	V	Difference in volatility
Extractive distillation	L and/or V	Liquid solvent and heat transfer	V and L	Difference in volatility
Azeotropic distillation	L and/or V	Liquid entrainer and heat transfer	V and L	Difference in volatility

Figura 13 – Parte da tabela que contém os métodos de separação e os seus critérios.

As *queries* são operações que seleccionam informação conforme critérios específicos escolhidos pelo utilizador. O tipo de *query* mais utilizado serve para seleccionar dados de uma tabela, como por exemplo seleccionar quais os *registos* que se quer incluir num *relatório*. Pode-se criar uma *query* que seleccione todos os métodos de separação que tenham alimentação líquida ou todos os que têm como princípio de separação a diferença de volatilidades. Para criar este tipo de *query*, introduz-se um *critério* que especifica os valores que se querem encontrar em determinados campos nas tabelas. Pode ainda utilizar-se as *queries* para combinar informação de várias tabelas e também criar *campos* calculados como somas, contagens e médias.

Na Figura 14 apresenta-se uma *query* criada para questionar o utilizador sobre qual a fase adicionada e conforme a sua resposta, vai seleccionar-se os dados correspondentes da tabela.

Field:	Separation method	Phase of the feed	Separation agent	Developed or added	Separation principle
Table:	tbl_separation_methc	tbl_separation_methc	tbl_separation_methc	tbl_separation_methc	tbl_separation_methc
Sort:					
Show:	☑	☑	☑	☑	☑
Criteria:				[Enter developed or a	
or:					

Figura 14 – Query de selecção do método com a fase adicionada escolhida pelo utilizador.

O modo mais fácil de introduzir dados é criando um *formulário*. O *formulário* é um documento estandardizado da base de dados que apresenta informação de um ou mais

tabelas. Através dos *formulários* é possível editar dados ou introduzir novos registos, especificar a ordem pela qual aparecem os itens, usar botões e outros tipos de controlo, etc.

A Figura 15 apresenta um formulário criado para introdução de valores, que após posteriores cálculos devolve os valores calculados. Este formulário tem caixas de texto, *radio buttons*, botões de ajuda, etc.

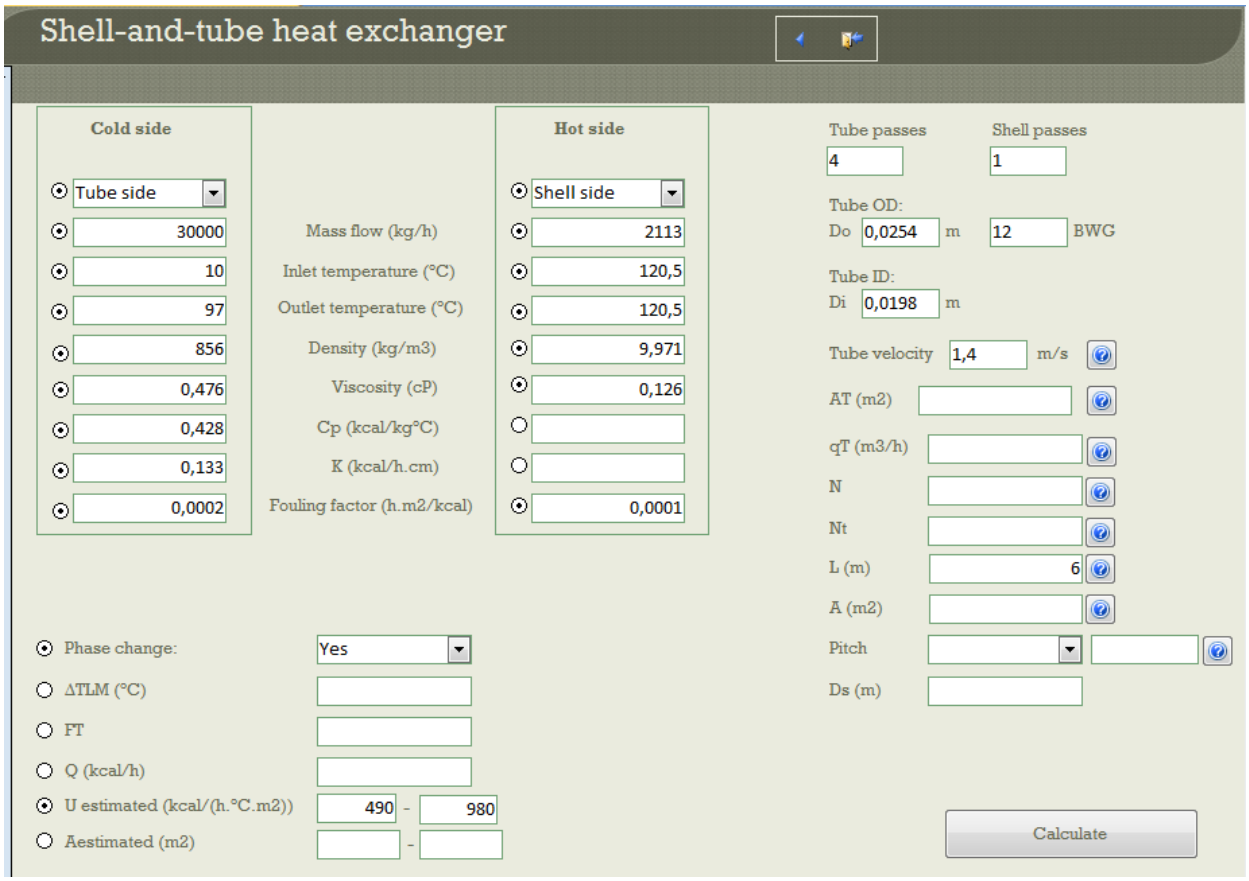

Figura 15 - Formulário onde se inserem os dados de um problema sobre permutadores.

O último *objecto* utilizado para a elaboração deste trabalho é o *módulo*. *Módulo* é outro termo utilizado no ambiente de desenvolvimento *Visual Basic*. O *Visual Basic for Applications* é uma linguagem de programação baseada na linguagem *BASIC*. É orientada especificamente para desenvolver rotinas nas aplicações do Office.

O código de *VBA* é construído por procedimentos. Cada procedimento contém várias linhas de código, cada uma chamada de *statement*. Cada *statement* indica ao VBA a acção que deve fazer. O procedimento que está descrito no *módulo* só executa a acção quando um evento, por exemplo o clicar num botão, chama o procedimento.

4.4 Exemplos desenvolvidos em Ms-Access

Em Ms-Access foram desenvolvidos três exemplos, há medida que se foi adquirindo conhecimento de algumas potencialidades da aplicação.

O primeiro é o mesmo desenvolvido em Visual Prolog sobre métodos de separação. Mantém-se o objectivo de obter qual o melhor método de separação para um processo com determinadas características.

O segundo é sobre tanques de armazenamento. De acordo com Walas et al. (1990), ao consultar as heurísticas do seu livro é possível verificar que conforme a capacidade do tanque, este deve ter determinada posição. Assim, o objectivo da aplicação neste caso é ao ser inserida a capacidade do tanque pelo utilizador, a aplicação devolve qual a posição ideal em que o tanque deve ser colocado.

O último visa sobre permutadores de calor. Este já com um objectivo diferente, pretende que ao inserir determinados dados sobre o processo e após proceder a cálculos, a aplicação devolve as dimensões ideais do permutador de calor.

Quando se abre a aplicação surge o painel principal como apresentado na Figura 16. Nesse painel existe a opção de adicionar dados à aplicação, correr o programa ou sair da aplicação.

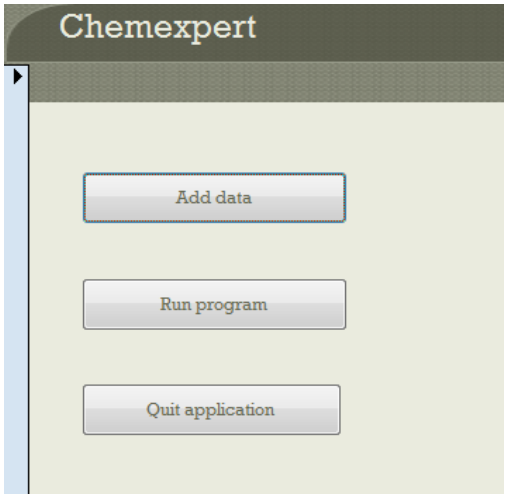

Figura 16 – Painel principal do programa

A escolha das opções “Add data” e “Run Program” permitem avançar para um menu com várias opções (Figura 17).

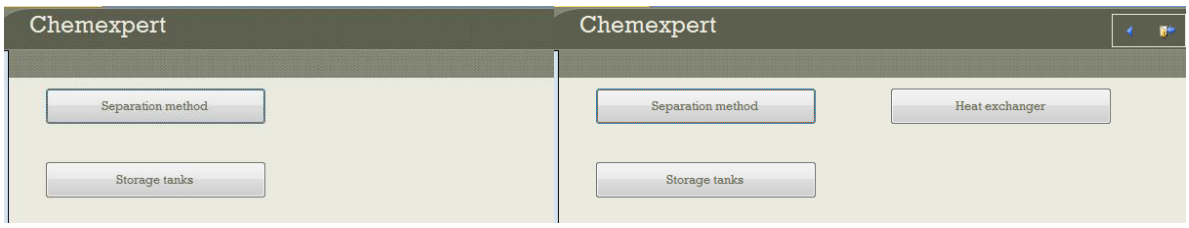

Figura 17 - Menu "Add data" e "Run program".

Nestes menus pretende-se que o utilizador seja encaminhado para a matéria na qual se quer obter informações, ou efectuar um cálculo.

Na opção "Add data" o utilizador pode adicionar novos dados, alterar ou remover informações. Estas informações estão organizadas em tabelas. Uma das vantagens de utilizar o Ms-Access é a maior facilidade que se tem em separar o conhecimento da linguagem de programação em si. Assim, organizando a informação em tabelas, fica mais fácil para o utilizador de as modificar, sendo praticamente intuitivo o modo como o fazer como se pode verificar pela Figura 18.

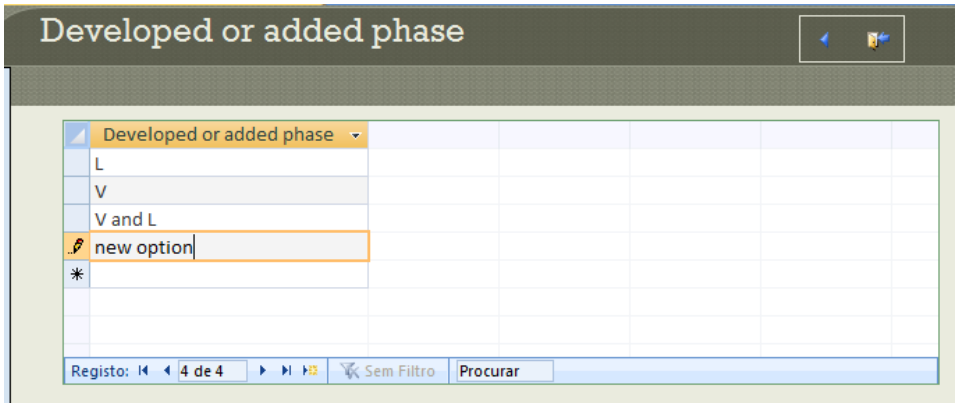

Figura 18 - Adição de dados na tabela da fase adicionada.

A construção de cada exemplo (*Separation method*, *Storage tanks* e *Heat exchangers*) varia de método para método, uma vez que foram sendo inseridas em diferentes fases de compreensão do MS-Access. Este facto também permitiu perceber que se podem fazer diferentes exemplos nesta aplicação à base de perguntas, cálculos, etc.

Quando se selecciona o tema *Separation method* surge uma janela com quatro critérios e uma opção de os adicionar a todos como é apresentado na Figura 19.

Figura 19 – Menus disponíveis para determinar o método de separação.

Quando se escolhe determinado critério, o utilizador visualiza uma janela com as hipóteses possíveis de resposta. Por exemplo para o critério *Developed or added phase* o utilizador fica com a informação que as opções possíveis são *L*, *V* ou *V and L*.

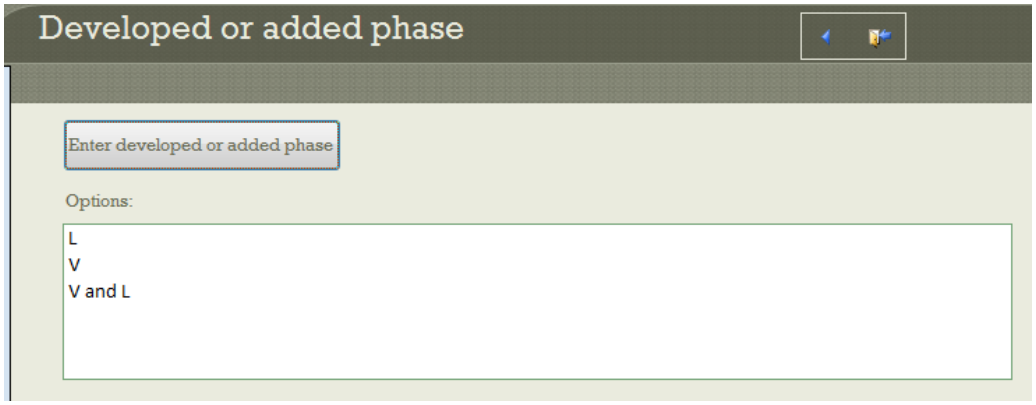

Figura 20 – Opções de fases a adicionar já existentes na base de conhecimento.

Ao clicar na opção *Enter developed or added phase*, surge uma caixa em que o utilizador deverá escrever a opção pretendida.

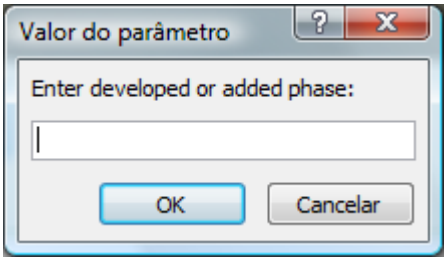

Figura 21 – Janela para inserir a fase adicionada escolhida pelo utilizador.

Em resposta à opção introduzida surge um painel com os métodos possíveis para a escolha daquele critério. No caso de se escolher a *Developed or added phase* como *V and L* surgem diversos métodos como se pode ver na figura 22.

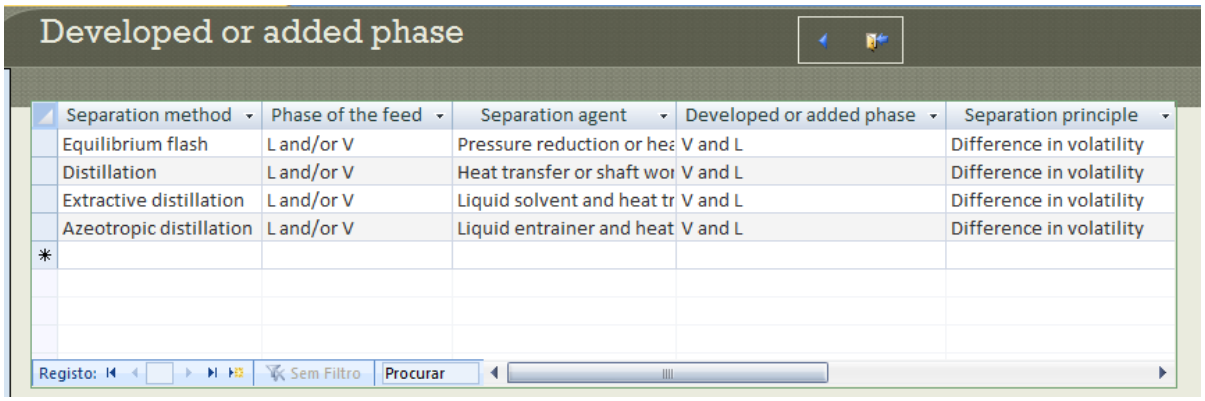

Figura 22 - Métodos possíveis para que a fase adicionada seja V e L.

Neste exemplo, o ideal seria poder escolher a opção directamente da lista em vez de a escrever na caixa de texto uma vez que reduz os possíveis erros de escrita. No entanto, esta funcionalidade ainda não está disponível.

O exemplo *Storage tanks* está construído de um modo semelhante ao do *Separation Method*.

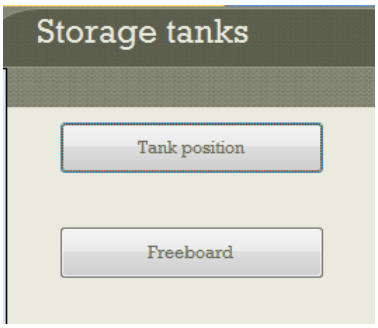

Figura 23 - Menu dos tanques de armazenamento.

Se escolher o critério *Tank position* surge uma janela para o utilizador inserir a capacidade necessária do tanque (Figura 24).

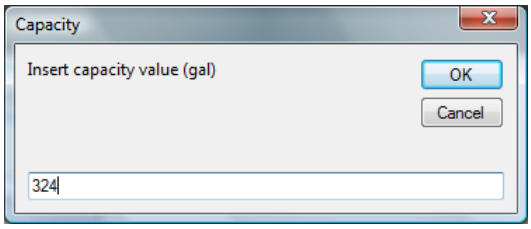

Figura 24 - Janela para inserir a capacidade do tanque de armazenamento.

Após inserir o valor surge uma caixa de texto com a posição em que o tanque deveria estar, conforme a capacidade seleccionada.

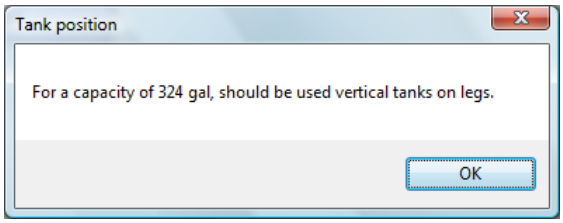

Figura 25 - Resposta da aplicação à capacidade de 324 gal.

A aplicação tem a informação armazenada em tabelas. A adição de nova informação sobre a posição dos tanques deve ser de acordo com os campos inseridos na tabela. A aplicação possui esta funcionalidade seguindo as opções sequenciais *Add data*, *Storage tanks* e *Tank position*. Nessa tabela poderá inserir o limite mínimo e máximo de capacidade para determinada posição do tanque (Figura 26).

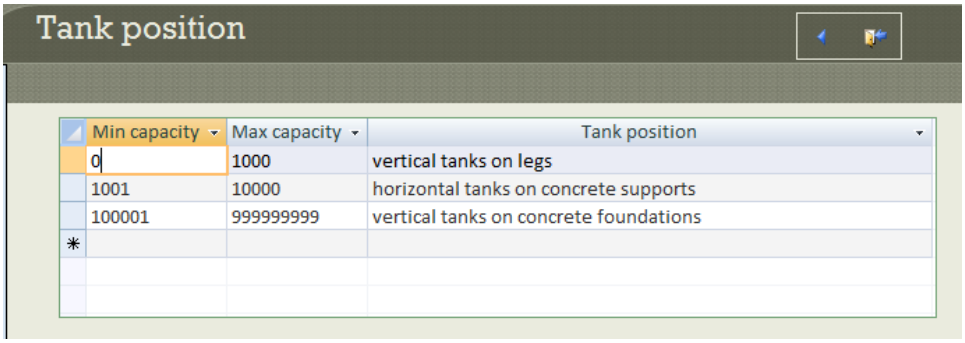

Figura 26 - Tabela que contém os limites de capacidade e respectiva posição ideal dos tanques de armazenamento.

O último exemplo *Heat exchangers* foi construído de um modo completamente diferente, já que o objectivo era saber se era possível calcular um permutador de calor. No entanto tem de ser muito mais desenvolvido, através da introdução de vários exemplos de cálculo, uma vez que um permutador pode ser obtido através de diferentes modos.

No exemplo de estudo considerado pretende-se projectar um permutador de carcaça e tubos para pré-aquecer uma corrente de 30 000 kg/h que contém etilbenzeno e estireno de 10 a 97ºC.

Dados adicionais: Densidade – 856 kg/m3

Viscosidade – 0,4765 cP

Calor específico – 0,428 kcal/(kg.ºC)

Condutividade térmica – 0,133 kcal/(h.m.ºC)

Calor médio fornecido – Vapor saturado a 1 barg

Notas: Para esta aplicação, o fluido do processo é alimentado aos tubos

O ΔP máximo é 0,8 bar.

Fouling – processo: 0,0002; vapor: h.m2/kcal

O objectivo na janela *Heat exchangers* é inserir os dados que o utilizador tem e o programa, conforme os dados inseridos, calcular o permutador ou pedir dados adicionais se necessário.

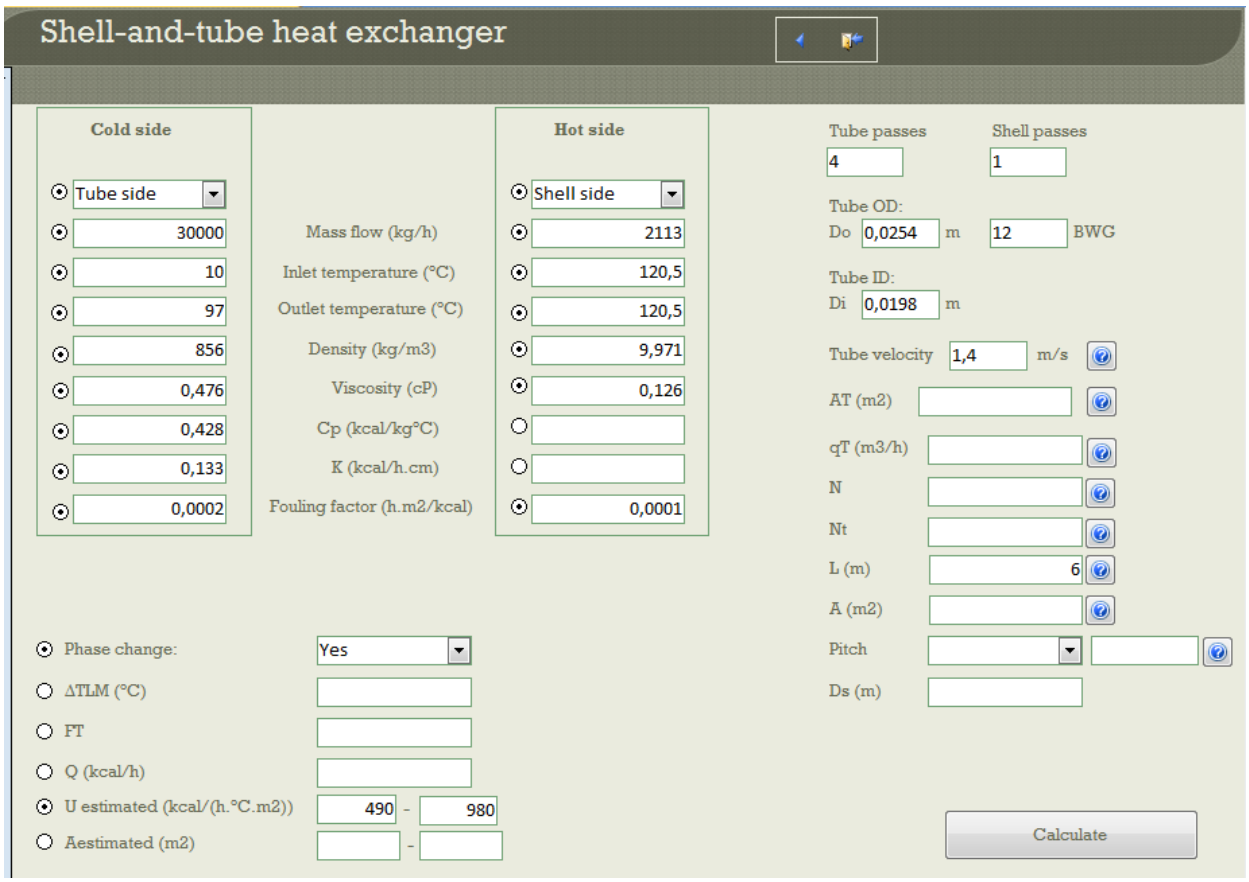

Figura 27 – Formulário para inserir os dados do problema dos permutadores.

Outro dos objectivos da aplicação é proporcionar o máximo de informação ao utilizador, de modo a obter a solução mais adequada. Assim, neste exemplo surgem os botões de ajuda ao lado de cada campo. Se o utilizador não souber para que serve aquele campo, deve fazer um clique no botão de ajuda e surge uma caixa de texto com informações pertinentes.

Tanto os cálculos como os botões de ajuda foram programados em VBA. É neste ponto essencialmente que este exemplo varia dos dois anteriores.

Assim sendo, quando o utilizador insere os dados do problema e pressiona o botão *Calculate*, a aplicação executa as rotinas VBA, apresentando como resultado os valores de projecto do permutador.

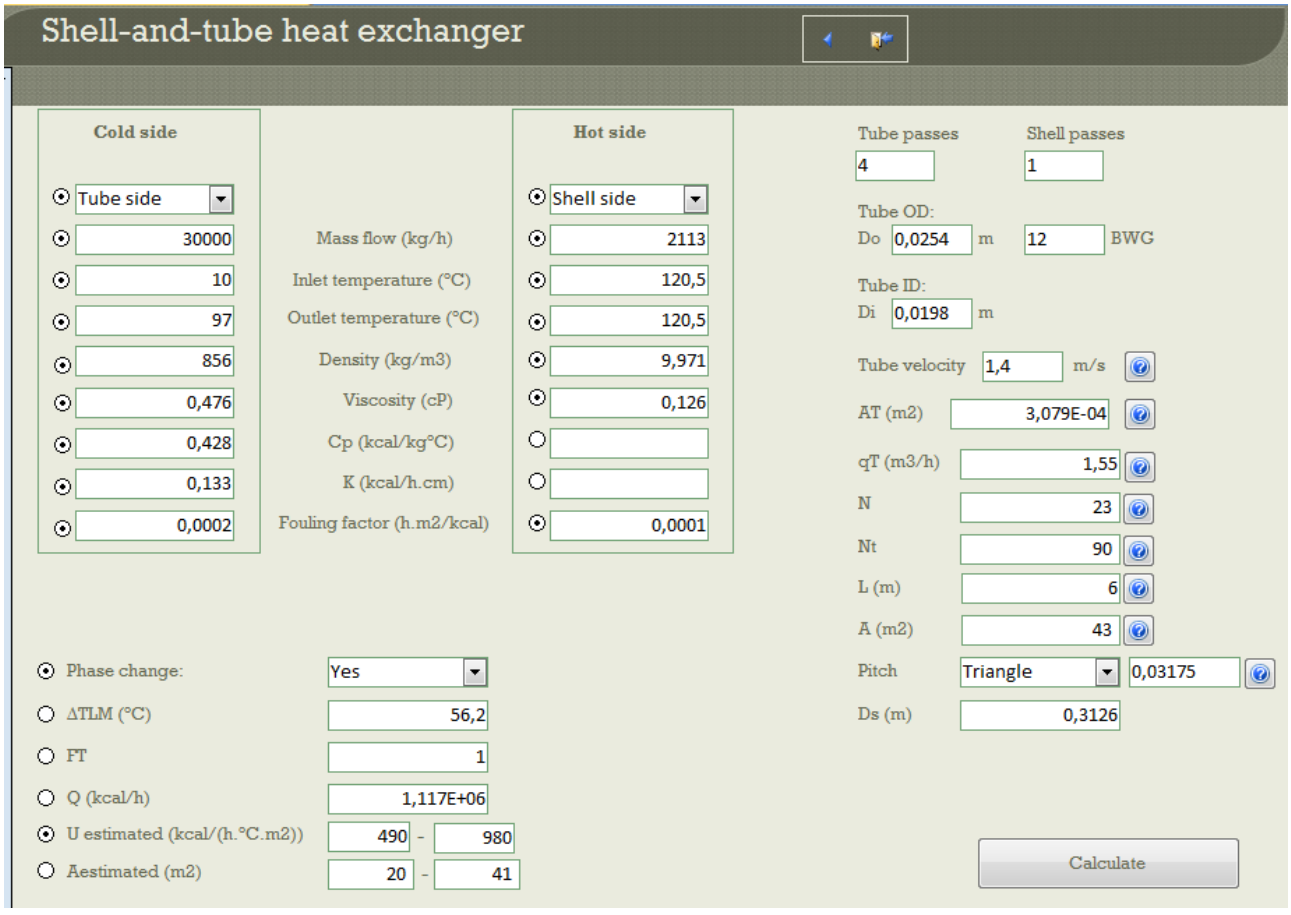

Figura 28 - Formulário após fazer os cálculos do permutador.

4.5 Discussão/comparação entre os dois métodos

O Visual Prolog é o programa mais adequado para o desenvolvimento deste tipo de aplicações. O layout to da aplicação final é mais apelativo e reconhecido pelo utilizador uma vez que é semelhante ao de uma aplicação do Windows.

A linguagem Prolog não é tão intuitiva para o utilizador como o Ms-Access e a bibliografia disponível não é de fácil compreensão para quem começa a programar. Também não é fácil separar a parte de conhecimento da parte de programação pois para inserir informação tem de se saber programar.

O Ms-Access é um programa muito mais intuitivo em que se trabalha de forma semelhante a outros programas do Windows. As tabelas facilitam a introdução e organização dos dados e permitem a separação do conhecimento.

No entanto para cada evento é preciso criar uma caixa de texto, um botão, um menu, etc. Ao clicar no evento corre o procedimento que também é preciso escrever. A linguagem usada é VBA o que é uma vantagem relativamente ao Prolog.

5 Conclusões

Foi estudado como deveria ser construído um sistema especialista para auxiliar o engenheiro na escolha do equipamento.

Iniciou-se a construção dos vários exemplos em dois programas diferentes: Visual Prolog e Ms-Access.

Entre os dois programas utilizados para desenvolver o sistema especialista o mais adequado é o Visual Prolog. Utilizando esta aplicação o sistema especialista fica com um visual mais apelativo, organizado e a forma de programação é mais correcta. No entanto é necessário conhecer muito bem a linguagem de programação e por isso estuda-la exaustivamente. Ou então colaborar com alguém, um engenheiro informático por exemplo, que já conheça a linguagem de modo a que o desenvolvimento seja mais eficiente já que o desenvolvimento adequado deste tipo de aplicações deve envolver várias entidades como apresentado na Figura 2.

Por outro lado o Ms-Access torna-se numa ferramenta mais fácil de usar quando já se conhece a linguagem Visual Basic for Applications. O programa é intuitivo e fácil de utilizar pois já tem alguns objectos pré-programados além da linguagem VBA ser mais simples. No entanto, no final o resultado é uma aplicação não tão organizada como em Prolog assim como a interface gráfica pode não ser tão apelativa.

Assim, verificou-se que é possível desenvolver sistemas especialistas nestes dois programas para apoiar o projecto de equipamento na indústria química. Facilita o trabalho do Engenharia e torna menos improvável a ocorrência de erros.

6 Avaliação do trabalho realizado

6.1 Objectivos Realizados

O objectivo deste trabalho era conhecer o que é um sistema especialista e como deve ser construído.

Após leitura de informação variada sobre este assunto adquiriu-se conhecimento sobre as diversas componentes dos sistemas especialistas.

Conclui-se igualmente que um sistema especialista pode ser construído de diferentes modos, usando diferentes plataformas. Cada exemplo apresenta uma especificidade elevada e por isso mesmo dentro da mesma aplicação há diferentes maneiras de desenvolver o sistema.

6.2 Limitações e Trabalho Futuro

A maior limitação na elaboração deste trabalho foi o tempo. O tempo curto estipulado e diversos contratempos não permitiram que o trabalho fosse mais desenvolvido.

No futuro deve-se continuar a explorar outras aplicações e plataformas para desenvolver os sistemas especialistas.

É necessário também fazer uma pesquisa mais exaustiva sobre sistemas especialistas já desenvolvidos na área de Engenharia Química para se conhecer melhor o trabalho já feito e que provavelmente pode ajudar a desenvolver este trabalho.

Ao nível da organização da informação, deve ser encontrado um método mais eficiente de armazenar a informação de modo a esta ficar mais acessivél ao utilizador. O utilizador pode ter necessidade de visualizar ou editar a informação existente ou até mesmo adicionar dados. Para isso deve-se desenvolver um processo de fácil percepção para o utilizador.

Nos exemplos desenvolvidos é necessário explorar com maior detalhe como é possível acrescentar mais funcionalidades. Como exemplo, na escolha do método de separação deve desenvolver-se uma forma que permita seleccionar de uma lista, em vez de escrever a opção.

Por fim, é aconselhável escrever um manual de utilização da aplicação. Uma vez que se pretende que a aplicação seja intuitiva e fácil de usar, deverá ser um manual bastante básico e compacto.

6.3 Apreciação final

A criação de uma ferramenta para apoiar o projecto de equipamento em indústrias de processo químico revela-se muito importante já que torna o trabalho mais eficiente, previne erros e ajuda o Engenheiro Químico na tomada de decisões.

Os programas estudados são adequados para a construção de um sistema especialista, no entanto este deve ser projectado por alguém com elevados conhecimentos em informática em colaboração com o especialista no assunto.

Na opinião do autor os objectivos foram cumpridos tendo em conta todas as limitações que surgiram durante o trabalho.

Referências

- AI Week, Chemical Bank Develops Foreign Trading ES, 8-9, 1988.
- Barr, A., Feigenbaum, E.A., *The Handbook of Artificial Intelligence*, Volume 1, William Kaufman, 1981.
- Beynon-Davies, P., *Expert Database Systems: a Gentle Introduction*, McGraw-Hill Book, London, 1991.
- Branan, C., *Rules of thumb for Chemical Engineers*, 4ª edição, Elsevier Inc., Oxford, 2005.
- Boulanger, A.G, *The Expert System PLANT/CD: A Case Study in Applying the General Purpose Inference System ADVISE to Predicting Black Cutworm Damage in Corn*, M.S. Thesis, Computer Science Dept., University of Illinois, 1983.
- Chan, F.T.S, Chan, H.K., *Design of a PBC plant with expert system and simulation approach*, Expert Systems with Applications 28, 409–423, 2005.
- Clancey, W.J., *Tutoring Rules for Guiding a Case Method Dialog*, International Journal of Man-Machine Studies, volume 11, 25-49, 1979.
- Dincbas, M., *A Knowledge-Based Expert System for Automatic Analysis and Synthesis in CAD*, Information Processing 80, IFIPS Proceedings, 705-710, 1980.
- Durkin, J, *Expert Systems*¸ Macmillan , New York, 1994.
- Fagan, L., *Knowledge engineering for dynamic clinical settings. Giving advice in the intensive care unit*, Doctoral Dissertation, Stanford University, 1979.
- Gonzalez, A.J., Dankel, D.D., *The Engineering of Knowledge-based Systems*, Prentice Hall, New Jersey, 1993.
- Hayes-Roth, F., Waterman, D. A., Lenat, D. B, *Building Expert Systems*, Addison-Wesley Publishing Company, 1983.
- Marsh, A.K., *Pace of Artificial Intelligence Research Shows Acceleration*, Aviation Week & Space Technology, Dezembro 10, 1984.
- Merritt, D, *Building Expert Systems in Prolog*¸ Springer-Verlag, New York, 1989.
- Miller, R. K, *Artificial Inteligence – Applications for manufacturing*, SEAI Technical Publications, Madison, 1985.
- MIS Week, Agosto 28, 1989.

- Mulsant, B., Servan-Schreiber, D., *Knowledge Engineering: A Daily Activity on a Hospital Ward*, Computers and Biomedical Research, volume 17, 71-91, 1984.
- Ranky, P. G., *Manufacturing Database Management and Knowledge Based Expert Systems*, CIMware Limited, Guilford, 1990.
- Seider, W. D., Seader, J. D., Lewin, D. R., *Product and process design principles: synthesis, analysis, and evaluation*, Wiley, New York, 2004.
- Simpson, A., Young, M. L., Barrows, A., Wells, A., McCarter, J., *Microsoft Office Acces 2007 All-in-One Desk Reference For Dummies*, Wiley Publishing, Indianapolis, 2007.
- Stansfield, J.L., Greenfield, N.R., *PlanPower: A Comprehensive Financial Planner*, IEEE Expert, volume 2, número 3, 51-60, Outono 1987.
- Stockman, J. C., Simpson, A., *Access 2007 VBA Programming For Dummies*, Wiley Publishing, Indianapolis, 2007.
- Walas, S. M., *Chemical process equipment: selection and design*, Butterworth Heinemann, Boston, 1990.
- Waterman, Donald A., *A Guide to Expert Systems*, Addison-Wesley Publishing Company, 1986.

Anexo 1 Executar a aplicação construída em Visual Prolog

Para correr a aplicação construída em Visual Prolog é preciso extrair o ficheiro com o formato zip e nome *Separation1*. Depois abrir a pasta Separation1, Exe, e correr o ficheiro Separation1.

Após abrir a aplicação o programa começa a colocar questões após um clique no menu *File*, *Separation Method*.

Começa por perguntar qual a fase de alimentação do processo. Para seleccionar uma fase clicar numa das opções do lado direito. Se desejar mais do que uma fase clicar nas duas pretendidas. Para deixar de estar alguma fase seleccionada basta clicar de novo. Do lado direito vão aparecendo os possíveis resultados.

Clicar Next para ir para o próximo critério e proceder da mesma forma. Após responder às quatro questões aparece a janela de resultados.

Para se poder alterar os conteúdos da aplicação é necessário ter o programa Visual Prolog instalado. Em http://www.visual-prolog.com, encontra-se diversa informação sobre o programa assim como a possibilidade de fazer download.