SIGESTÃO

Módulo SIG integrado no sistema de informação de gestão agro-pecuário AGRO.GESTAO ®

Miguel Mirador Fernandes

Trabalho de Projecto apresentado como requisito parcial para obtenção do grau de Mestre em Ciência e Sistemas de Informação Geográfica

Instituto Superior de Estatística e Gestão de Informação
da Universidade Nova de Lisboa

SIGESTÃO

Módulo SIG integrado no sistema de informação de gestão agro-pecuário AGRO.GESTAO ®

Trabalho de projecto orientado por

Professor Doutor Miguel Castro Neto

Setembro de 2008

AGRADECIMENTOS

A todos que directa e indirectamente deram contributo para a realização deste trabalho. De forma particular:

Ao Professor Doutor Miguel Castro Neto pela sábia orientação e total disponibilidade.

Á FZ.AGROGESTAO na pessoa do Eng. José Pedro Sarmento pelo interesse e ajuda.

Aos companheiros e amigos André Barriguinha e Paulo Ribeiro, por toda a ajuda nestes 2 anos divertidos e árduos.

Aos amigos David, Luís, Ivan, Ricardo e Artur pela distracção que proporcionaram e que clareou várias vezes as ideias, permitindo ganhar fôlego para continuar o trabalho.

Ao Eng. José Pedro Abreu Barreira e família pelo suporte, amizade e o tempo dentro das horas de serviço, sem o qual seria impossível concluir este trabalho neste espaço de tempo.

À minha família por todo o apoio, sem o qual não teria sido possível ter chegado até aqui.

Á minha esposa Joana pelo seu amor, grato pelas leituras atentas a este trabalho.

SIGESTÃO

Módulo SIG integrado no sistema de informação de gestão agro-pecuário AGRO.GESTÃO ®

RESUMO

Cada vez mais as empresas agropecuárias que desejam ser competitivas recorrem a diversos programas informáticos comerciais para optimizar todos os seus recursos como a gestão de stocks e actividades. O sistema de informação de gestão AGRO.GESTAO ® tem como objectivo englobar todas essas necessidades informáticas das empresas agropecuárias. O recurso Terra é o elo de ligação de muitas dessas funcionalidades presentes no AGRO.GESTAO ®. É este recurso que condiciona as actividades de curto e longo prazo, é por isso determinante estar bem identificado e claramente caracterizado.

O módulo SIG (SIGestão) concebido neste projecto, vai adicionar vantagens na identificação e na caracterização do recurso Terra aliando uma componente visual nessa identificação e caracterização, que no AGRO.GESTAO ® é apenas alfanumérica.

Concebido em ambiente Visual Basic ® com recurso à biblioteca de objectos MapObjects, proporciona ao utilizador um espaço onde é visualizado camadas de informação espacial, tanto raster como vectorial. Além das ferramentas de manipulação normais a um visualizador SIG, o SIGestão possui ainda nas suas funcionalidades capacidade de conexão a bases de dados dbase e MS Access. Esta última é a que permite de forma automática a conexão com o sistema de informação AGRO.GESTAO ®, tornado possível a construção de mapas temáticos a partir dos dados contidos no AGRO.GESTAO ®.

SIGESTÃO

GIS module integrated into agriculture information management system AGRO.GESTÃO ®

ABSTRACT

Increasingly, agro-companies wishing to be competitive use various commercial softwares to optimize all their resources as the management of stocks and activities. The information management system AGRO.GESTAO ® aims to encompass all such information needs of agricultural businesses. The resource Land is the link of many of these features present in AGRO.GESTAO ®. It is this feature that determines the activities of short and long term, is so crucial to be clearly identified and deeply characterized.

The GIS module (SIGestão) designed in this project, add advantages in the identification and characterization of the Land feature combining a visual component on the identification and characterization, which is in AGRO.GESTAO ® only alphanumeric.

Designed in Visual Basic ® environment using the library of objects MapObjects, offers the user an area where is visible layers of spatial information, both raster and vector. Besides the normal tools of manipulation to a GIS viewer, the SIGestão also has features in its ability to connect to databases dbase and MS Access. The latter is the form that allows for an automatic connection to the information management system AGRO.GESTAO ®, allowing the construction of thematic maps with the data contained in AGRO.GESTAO ®.

PALAVRAS-CHAVE

Agricultura

AGRO.GESTAO ®

Desenvolvimento SIG

MapObjects

Visual Basic ®

KEYWORDS

Agriculture

AGRO.GESTAO ®

GIS development

MapObjects

Visual Basic ®

ÍNDICE DO TEXTO

ÍNDICE DE FIGURAS

1. Introdução

1.1. Enquadramento

O AGRO.GESTÃO ® é uma aplicação em MS Access de contabilidade de gestão que permite organizar toda a informação das empresas agrícolas clientes.

O facto de este *software* se encontrar numa área de interesse tanto pessoal como profissional levou à hipótese de construção e introdução de um módulo SIG que pudesse ser utilizado junto com o AGRO.GESTAO ®.

O sistema de informação de gestão AGRO.GESTÃO ®, até ao momento, lida apenas com informação alfanumérica, uma vez que a dimensão espacial é uma variável fundamental no processo de tomada de decisão no contexto agro-pecuário, o desenvolvimento de um módulo SIG vai acrescentar novas e fundamentais funcionalidades à aplicação comercial da FZ.AGROGESTÃO.

1.2. Objectivos

Com este projecto pretende-se criar um modelo de um módulo SIG construído em ambiente Visual Basic com recurso aos componentes do MapObjects 2.4 da ESRI, adaptado às necessidades da empresa FZ.AGROGESTÃO valorizando o sistema de informação de gestão técnica e económica para empresas agro-pecuárias que esta empresa comercializa, o AGRO.GESTÃO ®.

O módulo a construir deverá cumprir uma série de requisitos mínimos, descritos em seguida:

- Suportar e visualizar camadas de informação geográfica em formato vectorial e *raster*, como parcelários, ortofotomapas, levantamentos topográficos, cartas militares desde que devidamente georeferenciados;
- funções de visualização, como *pan*, *zoom*, *full extent*;
- visualização da informação tabular associada aos elementos vectoriais;
- conexão da informação existente em tabela com a base de dados existente;
- construção de mapas temáticos a partir de dados da informação associada aos elementos vectoriais bem como da conectada posteriormente a tabela de base de dados.

O módulo SIG deverá permitir ao sistema de informação de gestão agro-pecuária AGRO.GESTÃO ® oferecer aos utilizadores uma nova forma de manipulação e visualização do enquadramento e características dos elementos georreferenciados disponíveis actualmente para os quais já dispõe de informação alfanumérica de natureza técnico-económica;

1.3. Premissas

As empresas que utilizam o sistema de informação de gestão AGRO.GESTÃO ® são empresas agro-pecuárias que possuem em geral quadros técnicos superiores que usam a informação registada no software como auxílio no planeamento e tomada de decisão, no âmbito das actividades agro-pecuárias que desenvolvem na empresa. Os sistemas de informação geográfica (SIG) funcionam muito bem em sistemas de suporte à decisão (Kraak 1996), esta sinergia inclui a visualização de dados espaciais beneficiando de vários modelos de gestão de informação. Como metodologias de gestão e planeamento de propriedades (Karkanis *et al.* 1997).

O sistema de informação de gestão AGRO.GESTÃO ® abrange diversos sectores da empresa agro-pecuária, desde o económico ao técnico, em que são registados todos os *inputs* e *outputs* de determinada parcela ou conjunto de parcelas (zona). No final de cada ano podem ser construídas contas de cultura/actividade, monitorizar a produção de uma determinada cultura numa determinada parcela ou zona, entre muitas outras operações.

A componente espacial introduz dados adicionais originados da contextualização visual que os SIG proporcionam e pode por isso aumentar a eficiência, eficácia, rapidez e qualidade da tomada de decisão. Esta vantagem pode levar a uma valorização deste produto no mercado, beneficiando fundamentalmente os clientes e utilizadores do AGROGESTAO ® dotando-os de mais uma ferramenta na tomada de decisões mais fundamentadas.

Um SIG pode ser definido, entre outras coisas, como um inventário mecanizado de elementos e características distribuídas geograficamente(Longley *et al.* 2005). Estes variam desde simples visualizadores de mapas, até complexos sistemas de análise e integração de base de dados espaciais complexas aumentando a eficiência de outras ferramentas na agricultura moderna (Committee on Assessing Crop Yield: Site-Specific Farming 1997).

A grande maioria dos *softwares* SIG utilizados actualmente integra muitas funcionalidades que são inúteis para as efectivas necessidades dos utilizadores, onde somente 10 a 20% da totalidade de funcionalidades disponíveis são efectivamente utilizadas pela maioria dos utilizadores(Painho *et al.* 1999). Existem duas maneiras de colmatar esta deficiência (Oliveira *et al.* 1997).

- A adaptação e treino dos utilizadores ao *software* SIG;
- A personalização do *software* SIG ao perfil do utilizador e dos requerimentos pretendidos para a aplicação.

A primeira opção parece ser inaceitável do ponto de vista da interacção operador-computador(Oliveira *et al.* 1997). Por outro lado A construção de uma aplicação de raiz para determinadas funções eliminaria o anterior facto, bem como diminuiria a complexidade de utilização que o *software* SIG possui também nos dias de hoje permitindo a utilizadores menos especializados realizarem funções que de outra maneira não o fariam (Painho *et al.* 1999). É muito importante que se construam SIG inteiramente funcionais, fáceis de aprender por "não-especialistas", com intuito de transferir definitivamente esta tecnologia para a comunidade agrícola (Committee on Assessing Crop Yield: Site-Specific Farming 1997).

Actualmente, já é possível construir aplicações personalizadas e de utilização bastante simples a partir de bibliotecas de componentes ou ferramentas de desenvolvimento rápido de aplicações.

É precisamente esse o objectivo deste projecto: a construção de um *software* SIG a partir de objectos programáveis em ambientes de programação visual padronizados utilizando o Visual Basic ®.

1.4. Metodologia

O projecto foi desenvolvido realizando as seguintes fases de forma sequencial:

- Revisão bibliográfica e aprofundamento de conhecimentos relativos ao tema e aos *softwares* em causa nomeadamente Visual Basic 6, Visual Studio, MapObjects 2.4.
- Especificação dos requisitos do módulo SIG, definidos em sintonia com o representante da empresa FZ.AGRO.GESTÃO, Eng. José Pedro Salema e o orientador de projecto, o Prof. Doutor Miguel de Castro Neto;
- Construção da aplicação.

- Redacção de um Relatório Final de Projecto e disponibilização do módulo SIG adiante designado SIGestão.

2. Ambientes de desenvolvimento SIG e linguagens de programação

2.1. Objectivos do capítulo

Este capítulo tem como objectivo fazer uma introdução ao desenvolvimento de *software* SIG e às linguagens de programação e *frameworks* mais usados. Sendo também esclarecida a questão do porquê da escolha do MapObjects e do Visual Basic para a construção do SIGestão. Vai ser ainda tema deste capítulo uma introdução ao sistema de informação de gestão "AGRO.GESTÃO ®", quais as suas funcionalidades e objectivos, bem como qual o seu público-alvo.

2.2. Ambientes de desenvolvimento de *software* SIG

A interoperabilidade é um dos aspectos mais frustrantes com que os utilizadores SIG se podem confrontar. A transversalidade do mundo SIG e a necessidade de usar dados de diferentes origens, com diversos formatos e características, dificulta um pouco o trabalho que se quer desenvolver e pode impedir de atingir os objectivos desejados. Existe deste modo a necessidade de personalizar aplicações à medida das necessidades do utilizador para que este consiga atingir os objectivos com maior facilidade.

A personalização em ambiente SIG é o processo de modificação de *software* SIG para, por exemplo, adicionar funcionalidades novas a aplicações ou embutir funções SIG noutras aplicações (Longley *et al.* 2005).

Os ambientes de desenvolvimento de *software* SIG são ambientes de desenvolvimento de carácter geral como o Visual Basic ®, Visual C++®, Delphi®, entre outros, que usam bibliotecas de objectos programáveis sobre

mapas, que permitem o desenvolvimento de aplicações com mapas (Painho *et al.* 1999), aplicações estas que podem por sua vez ser personalizadas à medida das necessidades.

Um dos mais populares ambientes de desenvolvimento/personalização de *software* SIG de uso local utilizado é o Microsoft Visual Basic.

Em todos estes ambientes é preciso usar conjuntos de objectos visuais (bibliotecas sobre mapas), dos quais são exemplo o OpenMap e o MapObjects.

O OpenMap é uma biblioteca em Java, *freeware*, que permite apenas visualização de dados espaciais com muito poucas funcionalidades de análise (GisLounge 2008). Suporta muitos tipos de dados espaciais, existindo uma aplicação exemplo que pode servir como ponto de partida, sendo possível personalizá-la à medida das necessidades (OpenMap 2008)

O MapObjects é um controlo "*active X*" comercial, que permite construir aplicações de visualização, de consulta e de análise a variados dados espaciais. Este permite que se possa incluir numa aplicação não SIG, algumas capacidades espaciais. É muito usado para dotar sítios *web* de capacidades SIG. Pode ser programado em várias linguagens: Visual Basic, Visual C++, Java e Delphi, o que o torna muito acessível a vários tipos de utilizador(ESRI 2008b).

2.3. Desenvolvimento de aplicações SIG

Podem ser encontradas aplicações de SIG virtualmente em quase todo o tipo de plataformas e escritas em quase todas as linguagens de programação disponíveis, *Arc Macro Language, Avenue*, Programação C e C++, Java.

O Visual Basic é uma linguagem acessível, orientada a objectos, que permite a criação de qualquer tipo de aplicativo, eficazes e independentes (Brown 1999).

Esta é talvez a linguagem e o ambiente mais utilizada no mundo quando se quer criar uma determinada aplicação (Brown 1999).

A utilização de programação por objectos traz vantagens, como (Painho *et al.* 1999):

- código modular e reutilizável;
- capacidade de adaptação à mudança;
- desenvolvimento mais rápido;
- custos de manutenção reduzidos.

O uso deste modelo de programação cumpre dois objectivos importantes (Painho *et al.* 1999):

- implementação que corresponda por completo aos requisitos do sistema informático desenvolvido;
- garantir que muitos dos módulos aqui desenvolvidos possam ser "reutilizáveis" em futuras implementações.

Desta maneira, a aplicação vai ficando cada vez mais completa e eficiente, levando a uma maior reutilização de código já construído e diminuição do tempo de implementação dos projectos (Painho *et al.* 1999).

2.4. Exemplos de aplicações SIG

O mundo SIG é tão vasto e transversal a tantas áreas que é natural que existam inúmeras aplicações, umas de âmbito mais geral, e outras mais específicas, como é o caso da descrita neste trabalho, o SIGestão.

A ESRI – Environmental Systems Research Incorporated é reconhecidamente o líder em software SIG. O principal pacote distribuído por esta empresa é o ArcGIS Desktop.

O Idrisi32 é também das aplicações mais populares, desenvolvida numa base não lucrativa pelos Clark Labs (Huber 2000).

Outra grande empresa que desenvolve várias aplicações SIG é a Intergraph. São exemplos das principais aplicações, o GeoMedia Professional e o Modular GIS Environment.

Entre outros exemplos de aplicações, também com alguma notoriedade podem ser referidos o AGISMap, o Autodesk MapGuide 5.0 e o Xmap.

3. Módulo SIG para o sistema de informação de gestão "AGRO.GESTÃO ®"

3.1. Objectivos do capítulo

Este capítulo pressupõe a descrição integral do desenvolvimento da aplicação em Linguagem Visual Basic 6, com recurso à biblioteca MapObjects *2.4* bem como de todas as suas funcionalidades, capacidades e recursos utilizados. Foi incluído um manual de utilização, para melhor entendimento da aplicação. O código integral da aplicação construída neste projecto pode ser consultado em anexo.

3.2. Descrição do sistema de informação de gestão "AGRO.GESTÃO ®"

O AGRO.GESTÃO ® é uma aplicação em MS Access de contabilidade de gestão que permite organizar toda a informação da empresa. É o principal produto da empresa FZ.AGROGESTÃO, Lda., uma empresa de serviços apostada no desenvolvimento de soluções informáticas que reforcem a capacidade de gestão das empresas do meio rural. Esta empresa possui ainda várias aplicações personalizadas de que são exemplo:

- PLANIGESTÃO - Orçamentação e planeamento de curto prazo
- AGROPDA - Recolha de informação de campo c/ PC de Bolso
- AGROGESTÃO.Comercial - Facturação e gestão documental
- AGROGESTÃO.Salários - Processamento de vencimentos
- ZOOGESTÃO ® - Gestão de efectivos pecuários
- ZOOPDA - Solução de recolha de informação de campo c/ PC de Bolso

- ZOOCHIP.ID - Solução de gestão de identificação electrónica de animais
- ZOOGESTÃO ® - Gestão de equinos
- ZOOGESTÃO ® - Registo Zootécnico

Os objectivos principais do AGRO.GESTÃO ® são:

- servir de apoio ao planeamento e controlo da gestão de dia-a-dia;
- gerar informação e relatórios internos para a gestão empresarial, permitindo servir de base à tomada de decisão não rotineira;
- definição de estratégias e de grandes planos, assim como de políticas a seguir para a empresa;
- permitir a elaboração de documentos para serem utilizados no exterior.

Os criadores do serviço pretendem que o sistema possa fornecer as respostas que a agricultura moderna exige, permitindo a boa utilização deste sistema de controlo de gestão:

- fundamentar as decisões com informação detalhada e pertinente;
- melhorar a eficácia de todo o sistema produtivo;
- identificar estrangulamentos e gastos desnecessários;
- elaborar relatórios para cada actividade.

O AGRO.GESTÃO ® é um *software* de gestão para empresas do meio rural. Por utilizar uma linguagem de fácil entendimento e por ser desenvolvido num interface gráfico amigável é uma ferramenta acessível a todos os potenciais utilizadores como: agricultores, empresários agrícolas em nome individual e gerentes de empresas agro-pecuárias. Esta aplicação possui mais de 200 utilizadores neste momento entre os quais diversas associações de agricultores (FZ.AGROGESTAO 2008).

O objectivo central do AGRO.GESTÃO ® é o apuramento e discriminação dos custos e das receitas das diversas actividades desenvolvidas numa empresa

agrícola, o que permite, por um lado, o melhor planeamento da actividade agrícola, e por outro, uma progressiva racionalização da utilização de recursos com os consequentes benefícios ao nível dos rendimentos do agricultor (Salema 2004).

Um desses recursos principais é o recurso terra, capital fundiário. No AGRO.GESTÃO ® este recurso é devidamente caracterizado (figura 1) com informação alfanumérica como a designação da parcela, a sua área, o nº da parcela (código) e ainda existe um campo para “Observações”, onde se pode descrever um pouco mais cada parcela que possa ajudar a identificá-la no seu contexto real.

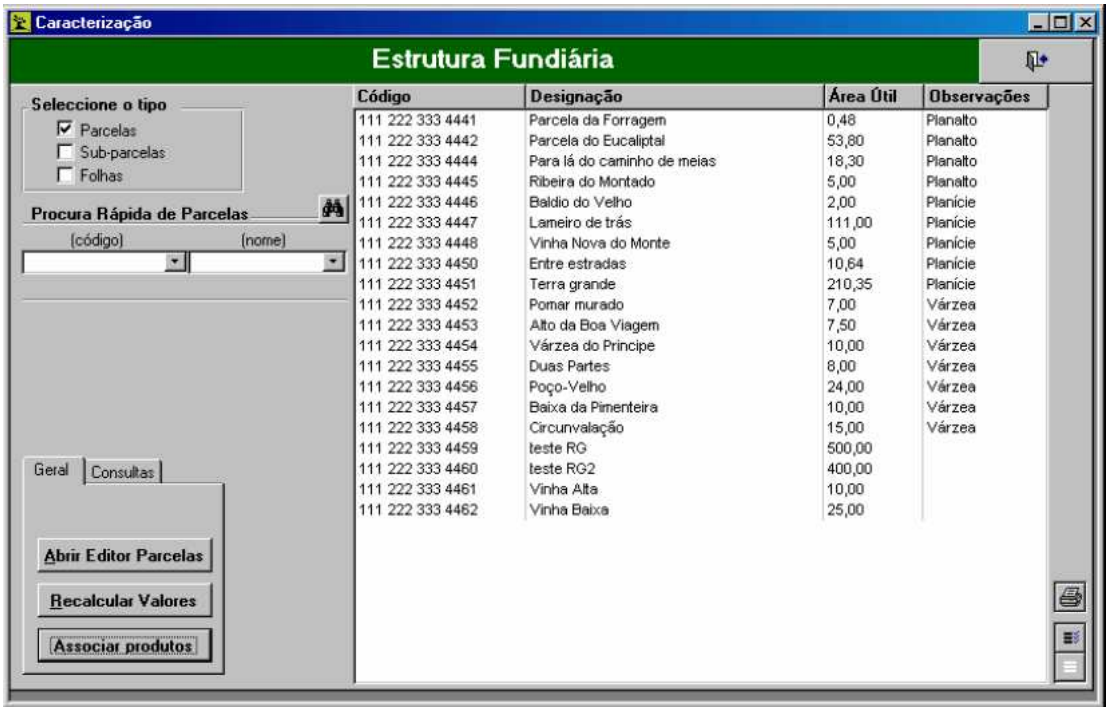

Figura 1 – Caracterização da estrutura fundiária (FZ.AGROGESTAO 2006)

Além destas informações, há espaço ainda no AGRO.GESTAO ® onde se podem incluir todas as informações complementares oficiais (figura 2) presentes no Parcelário Agrícola.

Nº Definitivo Parcelário	Nº Provisório Parcelário	Nº Contrib.	Nº IINGA	Secção Finanças	Artigo	Nome da parcela	Rel.	Tit.	Localização dcf	Localização Freguesia	Cult. Bloco	Cult. Parc.	Oliv.	Fig/Ar
111 222 333 4441		555 555 555	188881	K	34	Parcela da Forragem	2	P	1111117	São Nunca	PP	PP	0	0
111 222 333 4442		555 555 555	188881	K	56	Parcela do Eucaliptal	2	P	1111117	São Nunca	FL	FL	0	0
111 222 333 4444		555 555 555	188881	K	33	Para lá do caminho de meias	2	P	1111117	São Nunca	CA	CA	0	0
111 222 333 4445		555 555 555	188881	K	33	Ribeira do Montado	1	P	1111117	São Nunca	FR			
111 222 333 4446		555 555 555	188881	K	33	Baldio do Velho	1	P	1111117	São Nunca				
111 222 333 4447		502 797 229	1111	K	11	Lameiro de trás	4	R	1111118	Sta Maria	CA	CA	0	0

Record: 1 of 20

Figura 2 – Informação complementar das parcelas (FZ.AGROGESTAO 2006)

3.3. Desenvolvimento

A aplicação, designada “SIGestão”, foi desenvolvida com recurso ao Microsoft Visual Studio *6.0*, em linguagem Visual Basic 6.0 com a adição dos seguintes componentes, além dos presentes por defeito:

- Esri MapObjects 2.4;
- Esri MapObjects Legend Control;
- Microsoft Common Dialog Control 6.0 (SP3);
- Microsoft FlexGrid Control 6.0;
- Microsoft Tabbed Dialog Control 6.0;
- Microsoft Windows Common Control 6.0 (SP6).

Todas as funcionalidades foram apoiadas ou adaptadas em (ESRI 1996a), (ESRI 1996b), (Ralston 2002) e (ESRI 2008a).

3.4. Funcionalidades e Interface

O SIGestão foi desenvolvido tendo em consideração um conjunto de requisitos definidos pelos responsáveis do AGRO.GESTAO ®. Esses requisitos mínimos formam a capacidade de visualização de informação espacial em formato vectorial e em formato *raster*, parcelários agrícolas e ortofotomapas, respectivamente. A esta função deveriam estar associados um conjunto de ferramentas como o *zoom in*, *zoom out*, *full extent* e *pan map*.

Adicionalmente, o controlo das propriedades da informação espacial em formato vectorial, nomeadamente a cor e o tamanho das linhas; aceder a informação intrínseca de parcela(s) seleccionada(s), como por exemplo a área, o perímetro, o tipo de ocupação do solo, e outros que caracterizam de forma mais ou menos perpétua cada parcela; algumas ferramentas de análise espacial como o "*merge*", "*intersect*" e o "*clip*"; realizar mapas temáticos tanto com campos de formato alfanumérico como campos com formato numérico a partir de informação intrínseca e de informação externa; a presença de legenda das camadas presentes e da capacidade de rotular objectos nas camadas; ferramenta para procura de itens presentes no mapa pelo nome ou pelo seu valor; salvar todas as alterações executadas durante a sessão.

Por último, foi estabelecido que para que o SIGestão pudesse ser um módulo que distribuído com o AGRO.GESTÃO ® daria a este uma diferenciação e um valor acrescentado significativo, o SIGestão deveria conseguir fazer de interface entre a informação espacial nos formatos vectoriais com a informação alfanumérica contida no AGRO.GESTÃO ®. Para isso o SIGestão teria que poder ligar-se ao AGRO.GESTÃO ® por meio de campo comum.

3.4.1 Funcionalidades presentes

De seguida faz-se uma breve descrição de todas as funcionalidades presentes nesta versão do SIGestão, que pode ser complementado com subcapítulo seguinte, Manual.

3.4.1.1 Carregar informação espacial em formato vectorial/*raster*

O SIGestão possui a capacidade de visualizar informação vectorial ou *raster*. A informação vectorial que previsivelmente será mais usada é o parcelário agrícola e no caso de dados em formato *raster* serão os ortofotomapas. A utilização de ambas em camadas permite rentabilizar a visualização da informação (figura 3).

Figura 3 – Excerto da visualização de ortofotomapa (*raster*) e de parcelário agrícola (vectorial) usado em simultâneo, no SIGestão

Os dados pode sem carregados por três vias. Duas delas em *design mode* (fase de construção da aplicação, possível alterar o código) e uma outra em *run mode* (fase de utilização da aplicação, impossível alterar o código) Em *design mode* pode-se optar por usar, de forma mais intuitiva, as propriedades do objecto “MapLayer” pertencente ao MapObjects 2.4 ou escrever directamente linhas de código que carregam uma determinada camada de dados (vectorial ou *raster*) de determinado endereço. Assumindo que o património Terra dos clientes do AGRO.GESTÃO ® é muito constante, esta seria a melhor forma de

carregar no início da aplicação os parcelários e os ortofotomapas. Para dados ou informação, tanto vectorial ou *raster*, que seja menos geral, como os levantamentos topográficos (cercas, pivots, florestações, …), carta militar, entre outros, o SIGestão possui a capacidade de carregar dados vectoriais e/ou *raster* em *run time*, e poder controlar em termos da sua ordem esses dados.

3.4.1.2 Zoom in/out, Panmap e Full extent

No SIGestão existem 4 botões para controlo do mapa como um todo, o *zoom in*, o *zoom out*, o *panmap* e o *full extent* assinalados pela mesma ordem com círculo vermelho na figura 4.

Figura 4 – Botões de controlo do SIGestão

É possível fazer ampliação de uma área determinada, pelo desenho de um rectângulo com o cursor *(zoom in*), e através de cliques simples proceder ao inverso (*zoom out*), o *panmap* facilita a movimentação do mapa na área destinada à visualização do mapa. É ainda possível com apenas um clique de rato na tecla do *full extent* visualizar na totalidade todos os elementos que constituem o mapa.

3.4.1.3 Controlo das propriedades de exibição de dados vectoriais

O duplo clique no nome da camada vectorial (não é possível em formatos *raster*) permite que se possam controlar as propriedades visuais dessa camada, nomeadamente o tipo e a cor do preenchimento dos polígonos e a ausência ou presença de limites.

3.4.1.4 Realização de mapas temáticos

No mesmo menu de controlo das propriedades visuais existem outros 2 separadores que permitem, através de informação intrínseca do polígono ou de informação conectada, construir mapas temáticos. O menu SIMPLES, mais adequado para mapas temáticos de campos alfanuméricos, por exemplo o nº de parcela, o solo dominante da parcela, entre outros e o menu CLASSES mais adequado na construção de mapas temáticos de campos numéricos (agrupados em classes de valores), por exemplo a área da parcela, produção média de uma cultura em determinada parcela, etc.

3.4.1.5 Aceder a informação intrínseca dos dados vectoriais

O botão que representa livros permite aceder à informação intrínseca (figura 5) associada à informação vectorial, essa informação é descrita textualmente de uma só parcela ou de um conjunto de parcelas indicadas pelo utilizador. Essa informação encontra-se num ficheiro *dbase* que acompanha a camada vectorial, ambos com a mesma designação. Esta conexão é feita quando os dados vectoriais são carregados de forma automática.

Figura 5 – Exemplo da apresentação, no SIGestão, da informação intrínseca de uma camada vectorial

3.4.1.6 Legenda e rótulos

A legenda das camadas de dados *raster* ou vectorial carregadas na área de mapa do SIGestão é apresentada na lateral da moldura de visualização permitindo uma melhor compreensão dos dados presentes no mapa (figura 6).

Figura 6 – Exemplo de legenda no SIGestão

É possível rotular cada um dos polígonos das camadas vectoriais (não é possível rotular *raster*), através do último separador, ROTULOS, do controlo das propriedades das camadas vectoriais, fazendo o duplo clique no nome da camada em causa.

3.4.1.7 Ferramenta PROCURAR ITEMS

Esta ferramenta tem como objectivo identificar no mapa um ou mais polígonos de que à partida conhecemos o valor de um ou mais campos. É uma ferramenta muito útil para grandes quantidades de informação em uso.

3.4.2 Interface com dados externos

Nesta versão do SIGestão, podem ser feitas ligações a dois tipos de ficheiros, ligações a ficheiros DBASE (.dbf) e a ficheiros MS ACCESS (.mdb).

A ligação realizada pelo SIGestão entre os dados espaciais (mapa do SIGestão) e os dados alfanuméricos externos ao SIGestão (AGRO.GESTÃO ®) é feita através duma conexão OLEDB (*Object Linking and Embedding, Database* enquanto que conexão DAO (Data Access Object) foi utilizado para

conectar aos ficheiros dbase. Tanto uma ligação como a outra podem ser executadas em *run time* (nesta versão a ligação a dbase) ou em *design time* (nesta versão a ligação a MS Access), podendo assim escolher qual a melhor procedimento que se adapta a cada utilizador. Em princípio a ligação ao MS ACCESS, especificamente ao AGROGESTAO, deve ser inscrita no código e pode até ser interessante que esta ligação seja feita em simultâneo com o início da aplicação, dependendo do caso. No caso de ligações a ficheiros dbase, por outro lado, trará mais vantagens que se mantenha de navegação em run time, visto que as origens destas fontes de informação serão mais variáveis.

3.5. Manual de utilização

O manual tem como objectivo aumentar o conhecimento das funcionalidades que esta versão do SIGestão possui. A janela principal possui duas áreas principais separadas pela barra de ferramentas. Assim cada camada de dados carregada causará dois acontecimentos: o primeiro será a visualização dos dados na área mais à direita da janela; e o segundo o aparecimento na área mais à esquerda da legenda dessas camadas. No caso de ser camada do tipo vectorial a cor de representação dos polígonos é aleatória.

Figura 7 – Principal janela do SIGestão

A barra de ferramentas que divide essas duas áreas é constituída por 9 botões. De cima para baixo:

- o sinal de mais, verde, que permite a abertura de uma outra janela para carregar camadas de dados espaciais em *run time*;
- lupa com sinal de mais no interior (*zoom in*);
- lupa com sinal de menos no interior (*zoom out*);
- mão branca (*pan map*);
- dois livros, este botão permite aceder a informação intrínseca de um ou mais polígonos seleccionados pelo utilizador;
- globo azul (*full extent*);
- Binóculos pretos que permite usarem uma outra janela para procurar itens presentes no mapa;
- Ícone do AGRO.GESTÃO ® permite aceder a menu para ligação a base de dados externa;
- Disquete (salvar como), ainda em construção.

Cada uma destas funcionalidades vai a seguir ser descrita com recurso a exemplos.

3.5.1 Carregar dados espaciais

Os dados espaciais capazes de serem carregados nesta versão do SIGestão são os vectoriais e *raster*.

No caso dos dados de formato vectorial devem ser só utilizados do tipo *shapefile* (shp). A maioria dos dados vectoriais é do tipo *shapefile*, como por exemplo os Parcelários agrícolas e levantamentos topográficos com recurso a instrumentos ligados com GPS.

No caso da informação *raster* estes são do tipo imagem (tiff, gif, …) e serão sobretudo ortofotomapas e digitalizações de cartas militares.

Toda a informação espacial utilizada deve estar georeferenciada e no mesmo sistema de coordenadas, de forma a evitar imprecisões que podem induzir em erro o utilizador.

Para proceder ao carregamento de camadas no mapa (com ou sem camadas carregadas), basta clicar uma vez no 1º botão da barra (sinal de MAIS verde) que abrirá uma janela mais reduzida, designada “controlo de camadas” (figura 8). Esta janela possui como constituintes uma caixa de listagem e quatro botões, botão “adicionar camada”, botão “remover camada” e ainda 2 botões com setas brancas em fundo verde (quando activos).

Figura 8 – Janela secundária “controlo de camadas”

O botão “carregar camada” origina o aparecimento de uma nova janela de diálogo do Windows com a instrução “Seleccionar ficheiro para nova camada” na barra de identificação (figura 9).

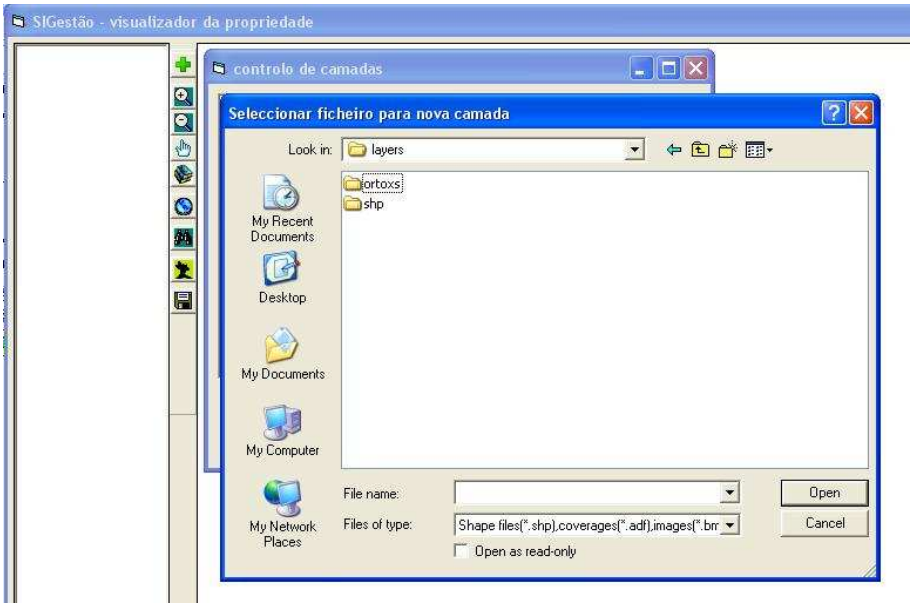

Figura 9 – Janela de diálogo do Windows

Usando esta janela podemos navegar por todas as pastas presentes no computador do utilizador e visualizar todos os ficheiros permitidos pelo filtro programado em *design mode*, que nesta versão do SIGestão permite *shapefiles*, *coverages* e ficheiros de imagem.

Depois de adicionadas todas as camadas pretendidas, , podemos ordenar as camadas pela indexação pretendida, através dos botões com setas que estarão disponíveis no caso de haver mais que uma camada na caixa de listagem. Esta ordenação é bastante importante no caso de haver camadas em sobreposição, permitindo posicionar as camadas da forma mais conveniente (ver sequência nas figuras seguinte: fig. 10, fig. 11 e fig. 12)

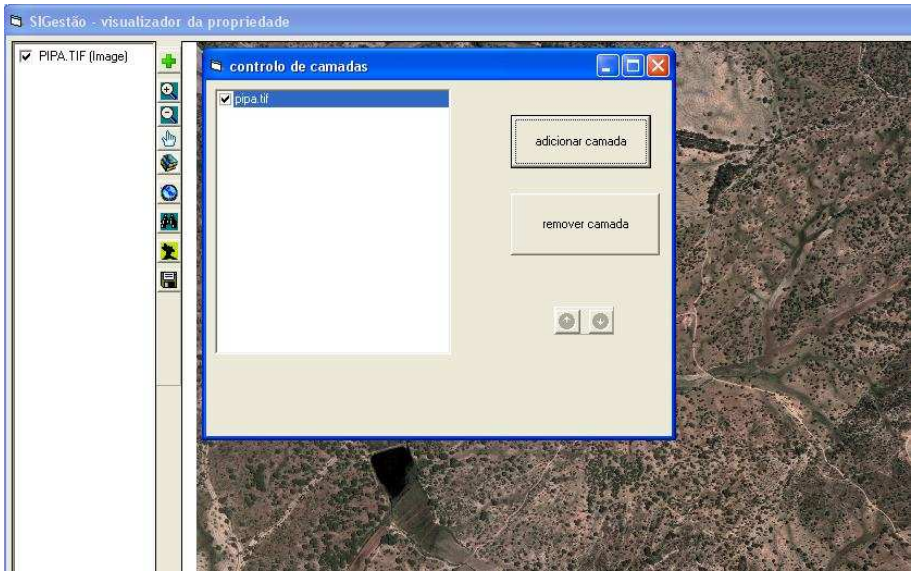

Figura 10 – Controlo de camadas do SIGestão com apenas uma camada carregada

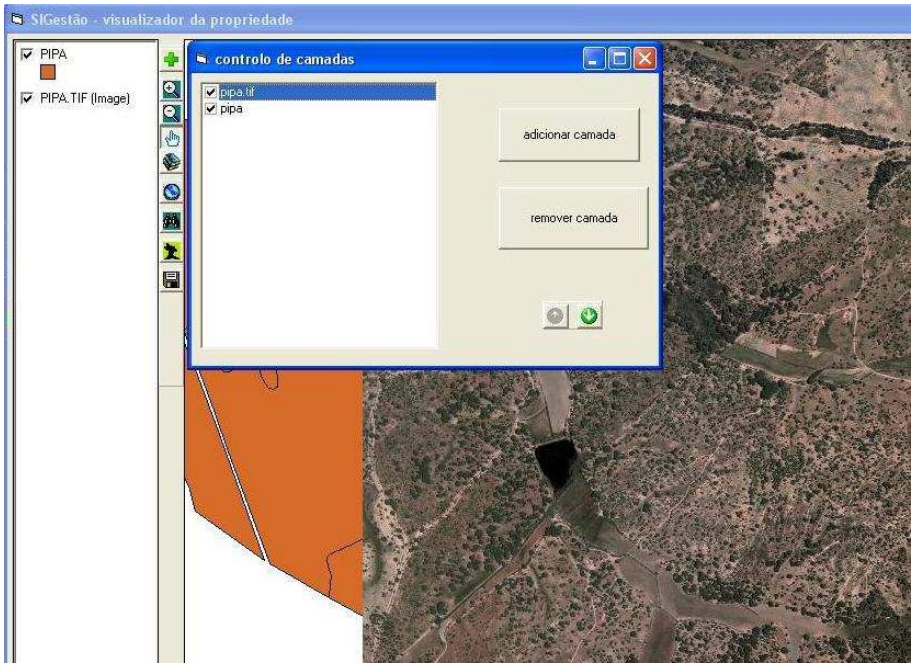

Figura 11 - Controlo de camadas do SIGestão com duas camada carregadas, ortofotomapa (pipa.tif) no topo e parcelário (pipa) por trás

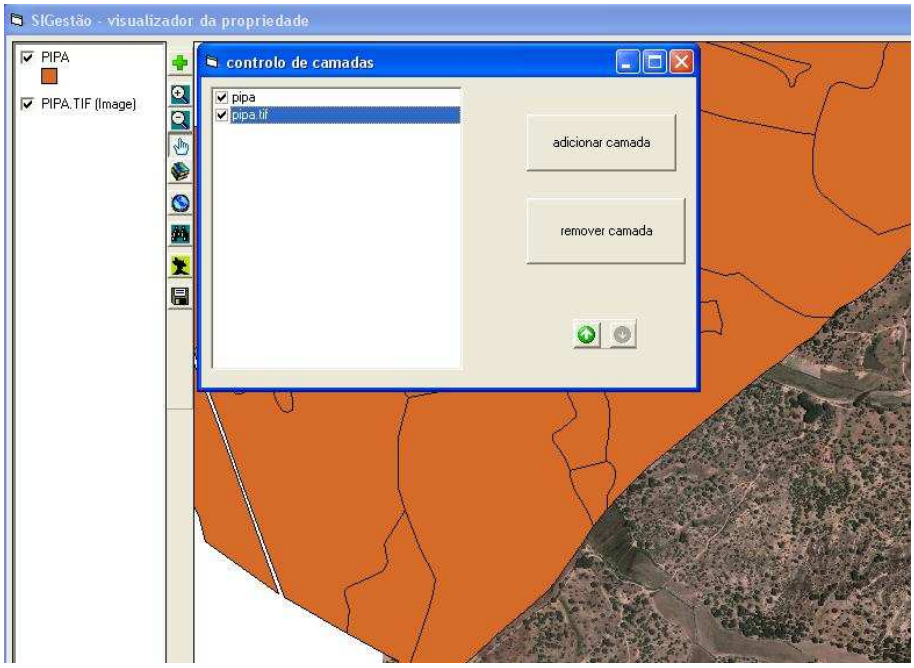

Figura 12 - Controlo de camadas do SIGestão com as mesmas duas camada carregadas, com ordem invertida em relação à figura anterior

Caso exista alguma camada que se deseje retirar, basta seleccioná-la e carregar no botão “remover camada” (figura 13).

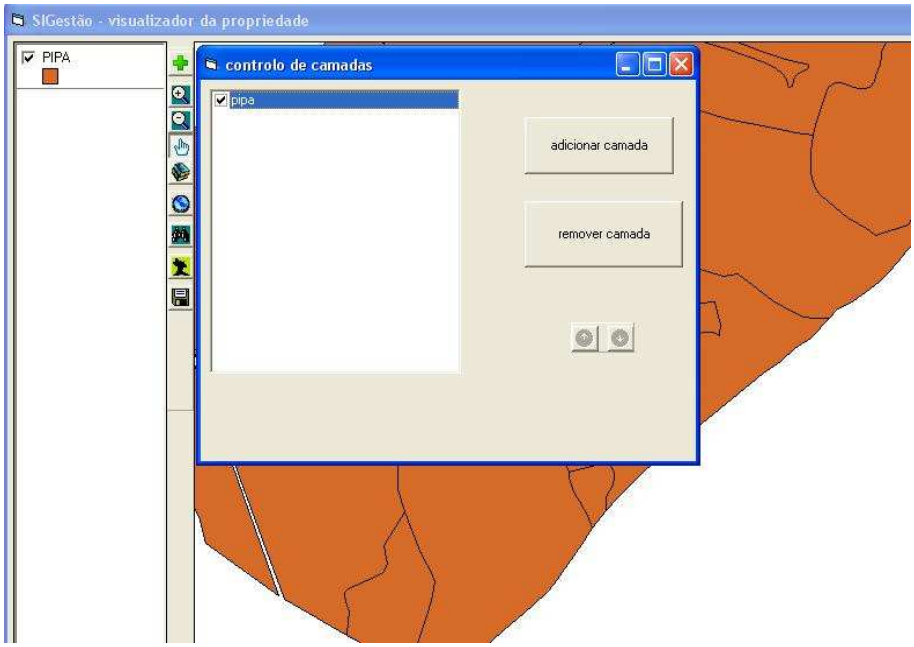

Figura 13 - Controlo de camadas do SIGestão com apenas 1 camada carregada, removida a camada “pipa.tif”

Estando todas as camadas pretendidas para visualizar na caixa de listagem, pela ordem mais conveniente, é só recorrer ao botão de encerrar a janela para que essas camadas sejam desenhadas no mapa. Este menu pode ser acedido diversas vezes durante o trabalho, para adicionar mais camadas, para remover as existentes, ou ainda para alterar a ordem de visualização das mesmas.

3.5.2 Usar as ferramentas de visualização

As ferramentas de visualização descritas anteriormente são muito intuitivas e possuem o comportamento tradicional neste tipo de aplicações. No caso do *zoom in*, *zoom out* e *pan map*, é necessário carregar o botão, correspondente a cada uma das ferramentas, que ficará no estado de pressionado, dando capacidade ao cursor de interagir com o mapa.

O *zoom in,* por intermédio do desenho de um rectângulo no mapa (figura 14) que seleccionará a área do mapa que ocupará de imediato, por ampliação, a totalidade da área do mapa (figura 15). Na figura 16, temos uma grande ampliação onde se evidencia a capacidade da aplicação para visualização, sem qualquer problema de imagens de boa qualidade

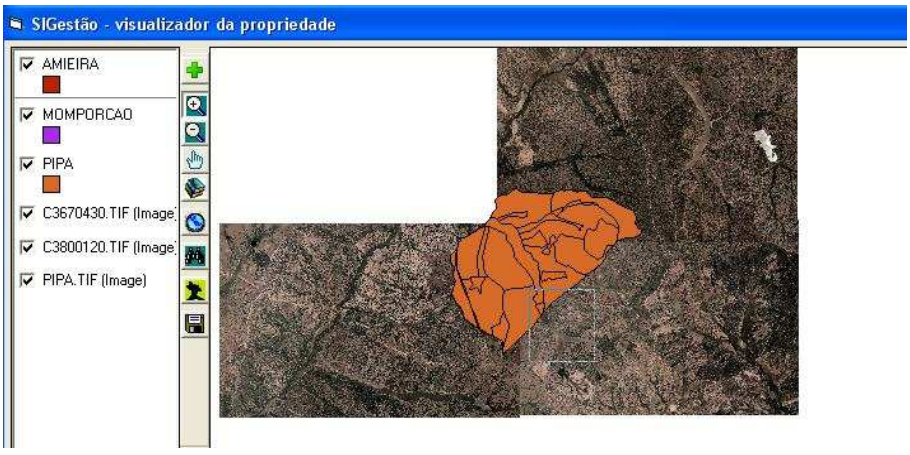

Figura 14 – Seleccionar área do mapa a ampliar através de rectângulo

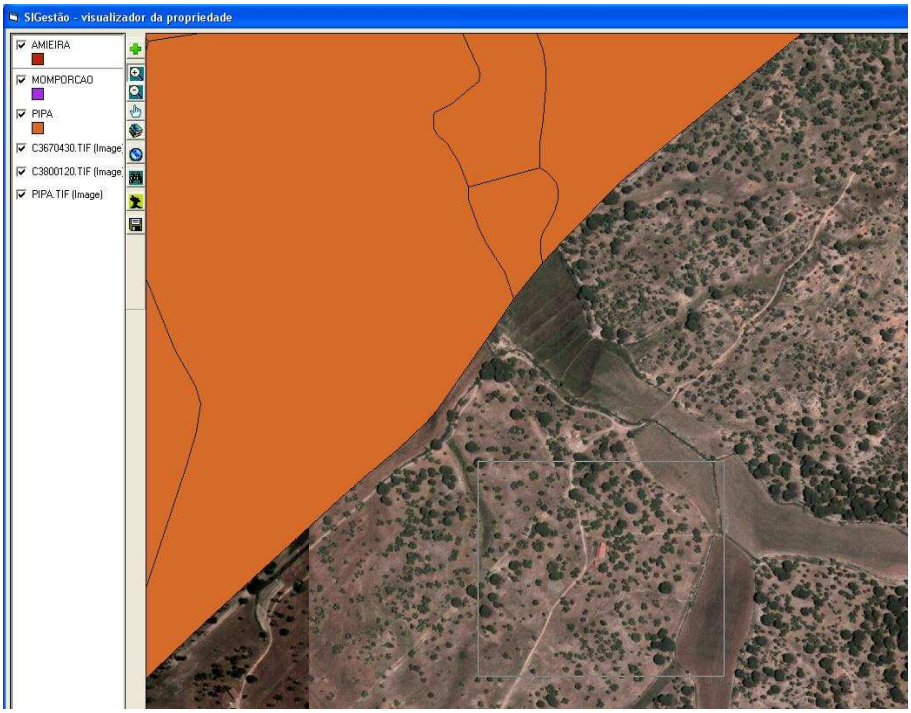

Figura 15 – Área ampliada através do rectângulo desenhado na figura 14 e desenho de novo rectângulo para maior ampliação

Figura 16 – Grande ampliação obtida do rectângulo desenhado na figura 15

O *zoom out* com cliques sucessivos do cursor no mapa, fará exactamente o inverso (figura 17).

Figura 17 – Um clique no mapa provoca o inverso (*zoom out*)

O *pan map* permite deslocar o mapa para todas as direcções por intermédio de arrastamento. Com estas 3 ferramentas é possível colocar no local pretendido a área do mapa.

O botão *full extent* permite colocar todas as camadas presentes no mapa na área de visualização (figura 18).

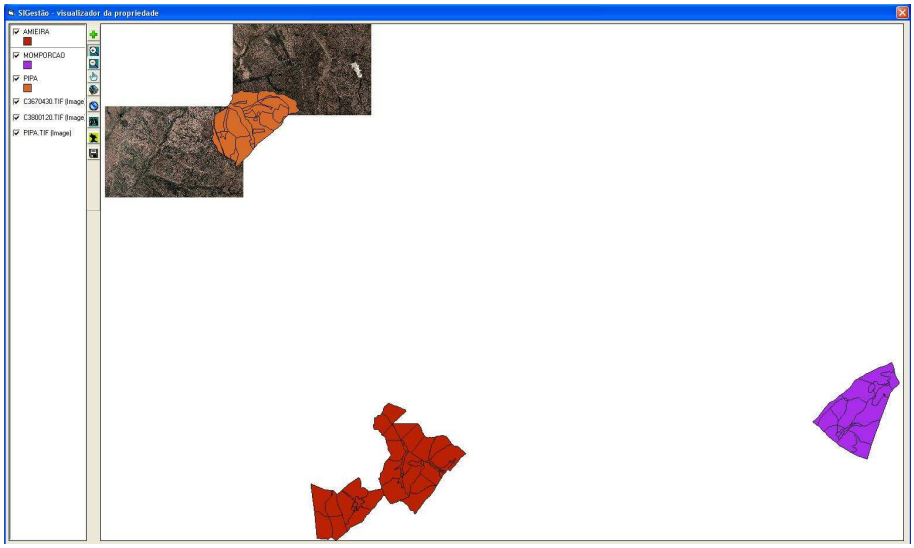

Figura 18 – Aspecto do resultado da ferramenta *full extent*

3.5.3 Modificar as propriedades visuais dos dados vectoriais

É possível modificar as propriedades de visualização de todas as camadas em formato vectorial. O duplo clique no nome da camada a alterar, permite o aparecimento de uma janela de diálogo “Propriedades da camada [nome da camada em causa]” (figura 19). Esta janela tem quatro separadores. Neste caso utilizaremos apenas o 1º, designado GERAL. Nele é possível alterar a cor do preenchimento e do limite dos polígonos dessa camada, bem como a espessura desse limite (figura 20).

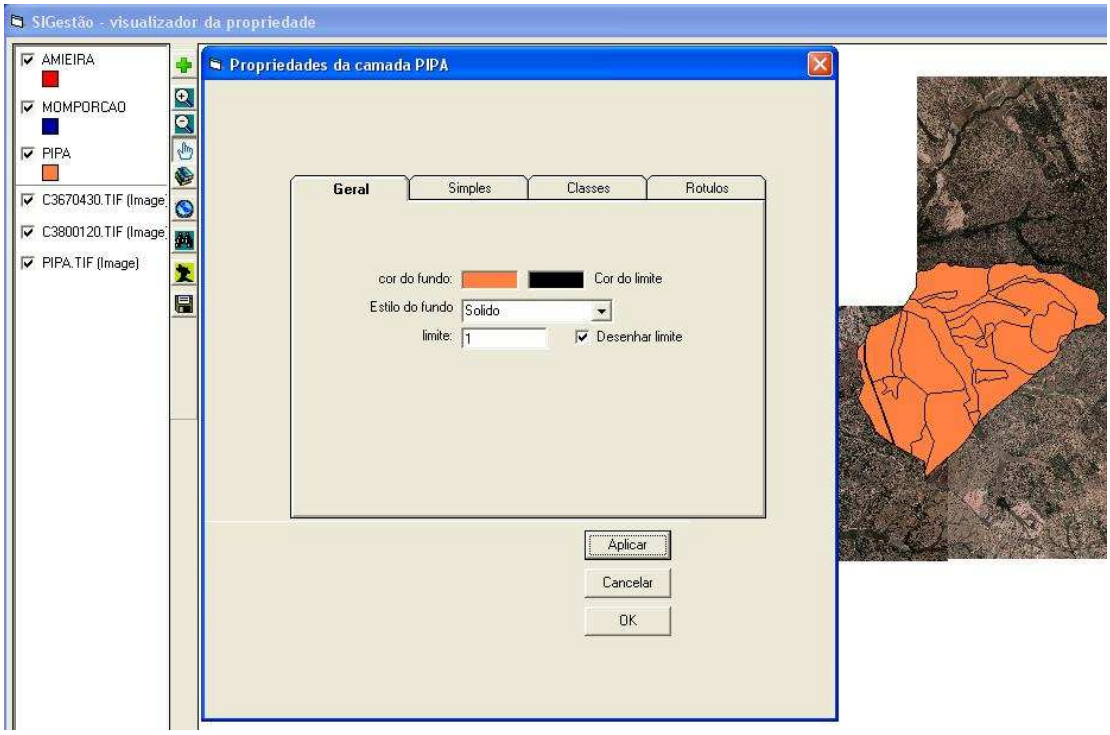

Figura 19 - Janela “Propriedades da camada [nome da camada em causa]” neste caso da camada PIPA

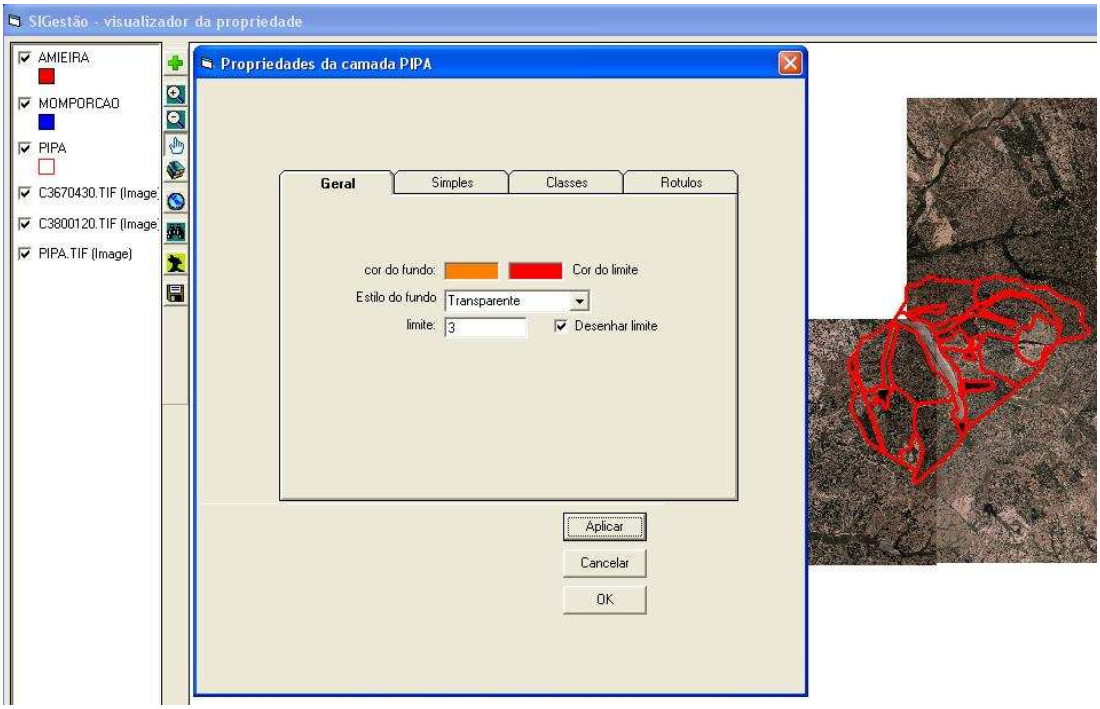

Figura 20 – Modificação das propriedades da camada PIPA, preenchimento, cor e espessura do limite

O preenchimento com diversos padrões (figura 21 e 22) ou simplesmente o transparente adiciona possibilidades à caracterização da camada e permite ajustar a melhor visualização a cada caso. Depois de definida a melhor composição para a camada basta clicar no botão “Aplicar” e no ”OK” para sair.

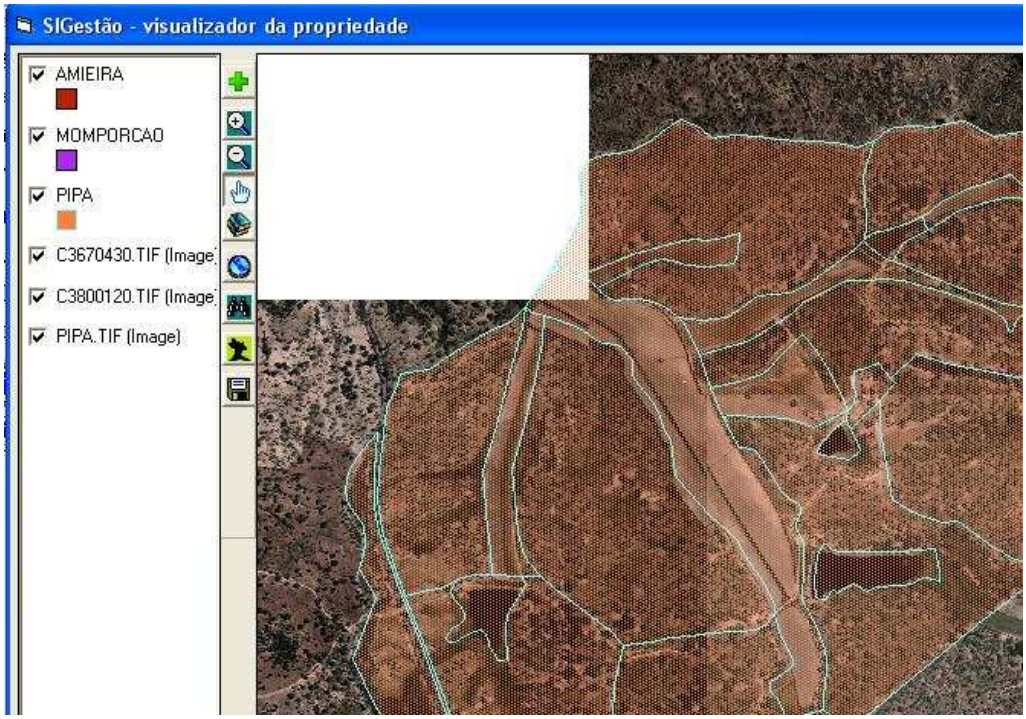

Figura 21 – Aspecto das possibilidades de personalizar o aspecto das camadas

3.5.4 Construção de mapas temáticos e rótulos

Os mapas temáticos podem ser de dois tipos:

1. Os que usam dados alfanuméricos de forma absoluta, que atribuem uma cor a cada um dos valores do campo escolhido;

2. E os que usam dados exclusivamente numéricos e que são organizados por intervalos de valores em numero personalizável.

Para a construção de mapas temáticos do 1º tipo deve-se usar o separador SIMPLES da janela “propriedades da camada [nome da camada em causa]”. Escolher o campo desejado na caixa de selecção para a construção do mapa, redesenhar a legenda e carregar no botão “Aplicar” (figura 22).

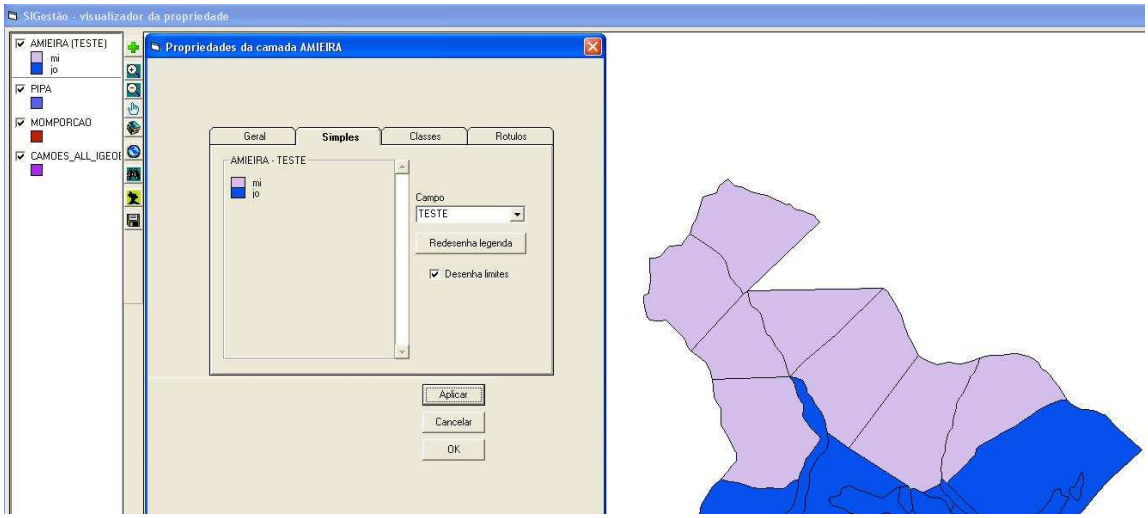

Figura 22 – Mapas temáticos com dados alfanuméricos

No caso de o campo pretendido ter valores numéricos, pode ser mais conveniente usar o terceiro separador (CLASSE), em que é possível seleccionar além do campo, o nº de classes pretendido, redesenhar a legenda e aplicar (figura 23 e 24).

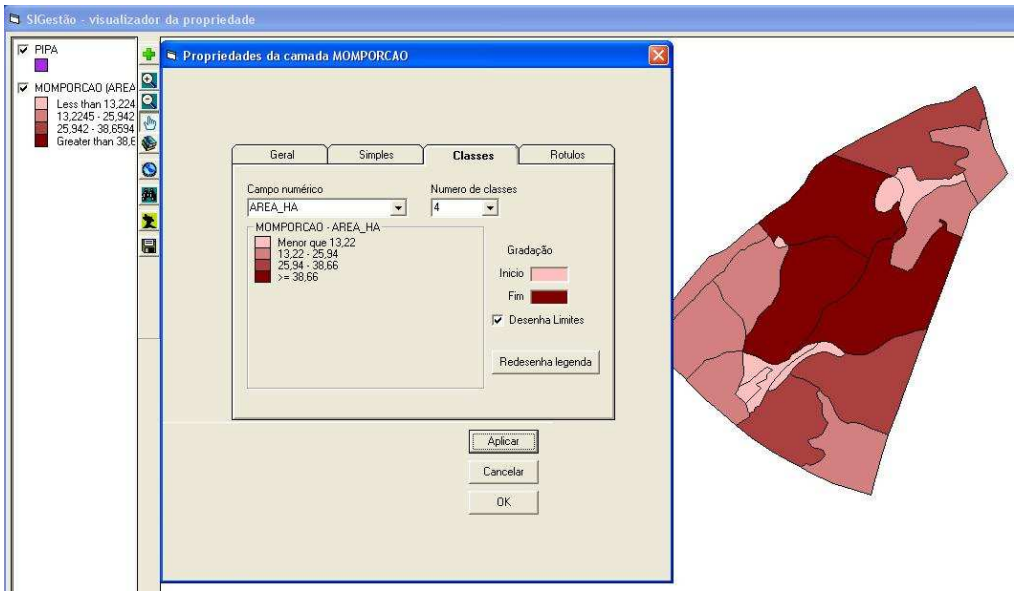

Figura 23 – Mapas temáticos exclusivamente a partir de dados numéricos (4 classes de valores)

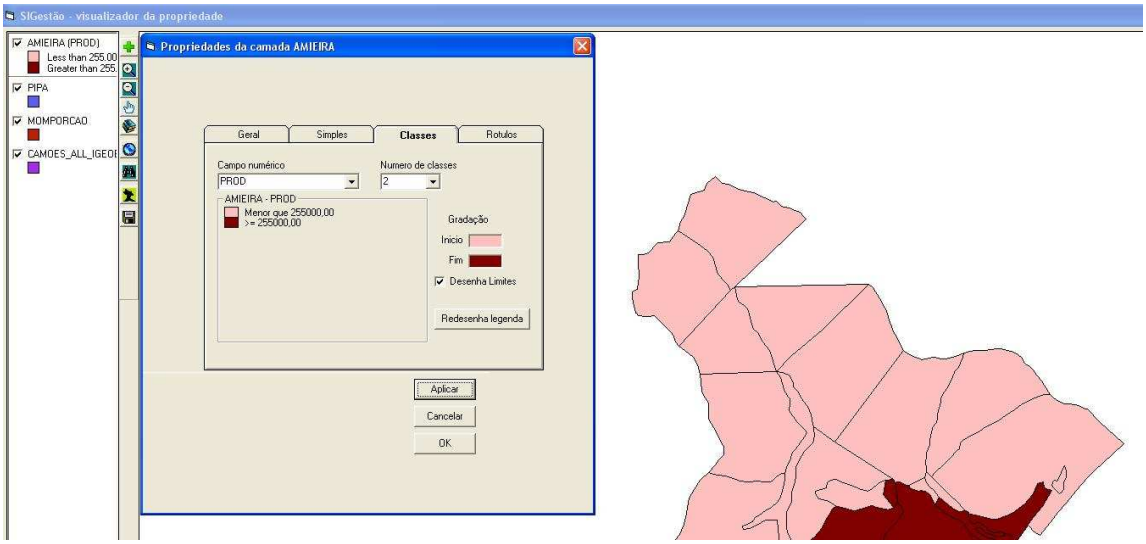

Figura 24 - Mapas temáticos exclusivamente a partir de dados numéricos (2 classes de valores)

O quarto e último separador, ROTULOS, permite escrever dentro de cada polígono o valor do campo alfanumérico desejado. Assim como controlar uma série de propriedades desses rótulos, como a fonte e o tamanho da letra e a posição dentro do polígono (figura 25).

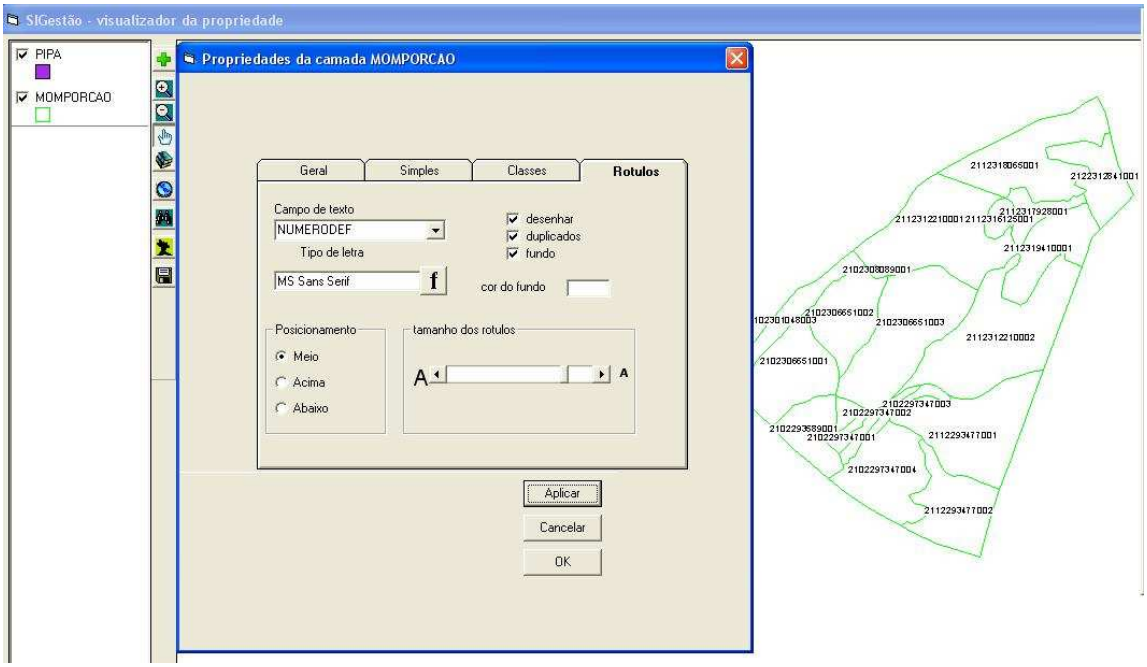

Figura 25 – Separador que permite rotular cada um dos polígonos

3.5.5 Informação intrínseca dos dados espaciais

Os dados espaciais possuem normalmente informação própria. Tomemos como exemplo o parcelário agrícola. A cada parcela agrícola (polígono) tem associado num ficheiro *dbase* informação como o concelho, freguesia, a área, número da parcela, o número do bloco, o proprietário. No fundo, qualquer informação da parcela, que é mais ou menos constante, pode ser inscrita neste ficheiro *dbase*. Para consultar essa informação basta seleccionar a camada que se pretende consultar. Em seguida com o botão "informação intrínseca" (ícone livros) pressionada, podemos clicar num polígono para aceder à sua informação (figura 26).

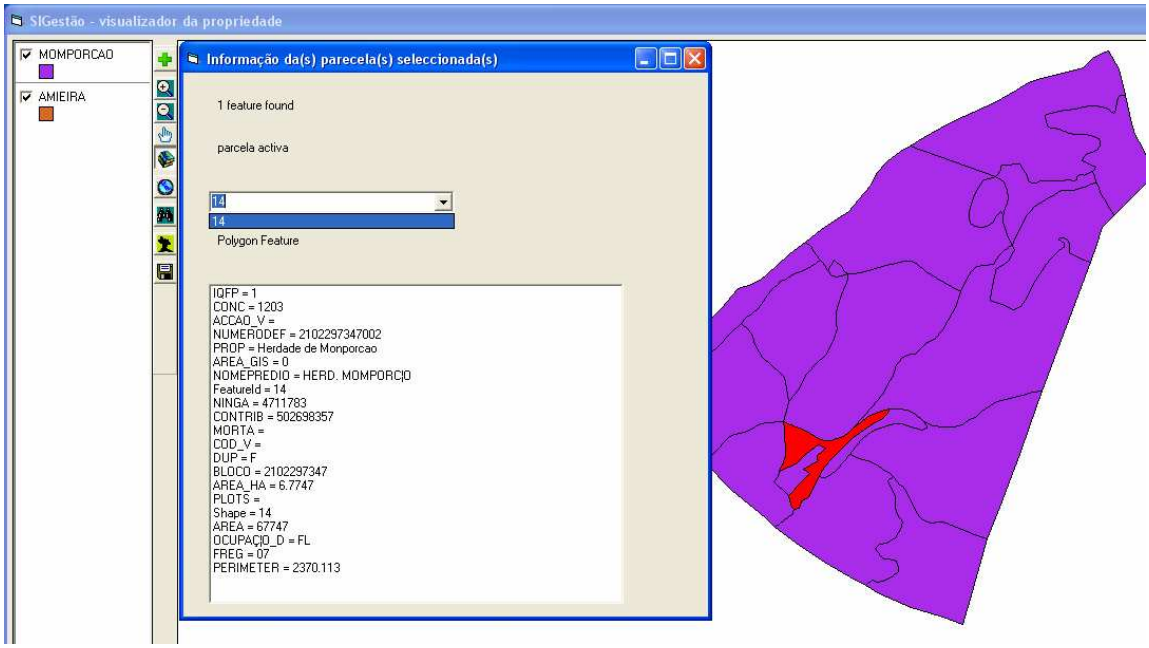

Figura 26 – Janela secundária "Informação da(s) parcela(s) seleccionada(s)", neste caso apenas uma parcela seleccionada

Podemos seleccionar mais que um polígono ao mesmo tempo, através do desenho de um rectângulo em cima da camada que está activa (figura 27). Todos os polígonos que intersectarem com esse rectângulo entrarão na janela "Informação da(s) parcela(s) seleccionada(s). Depois disso, na caixa de

selecção podemos escolher dentro do grupo de polígonos seleccionados, o polígono do qual queremos informação (figura 28).

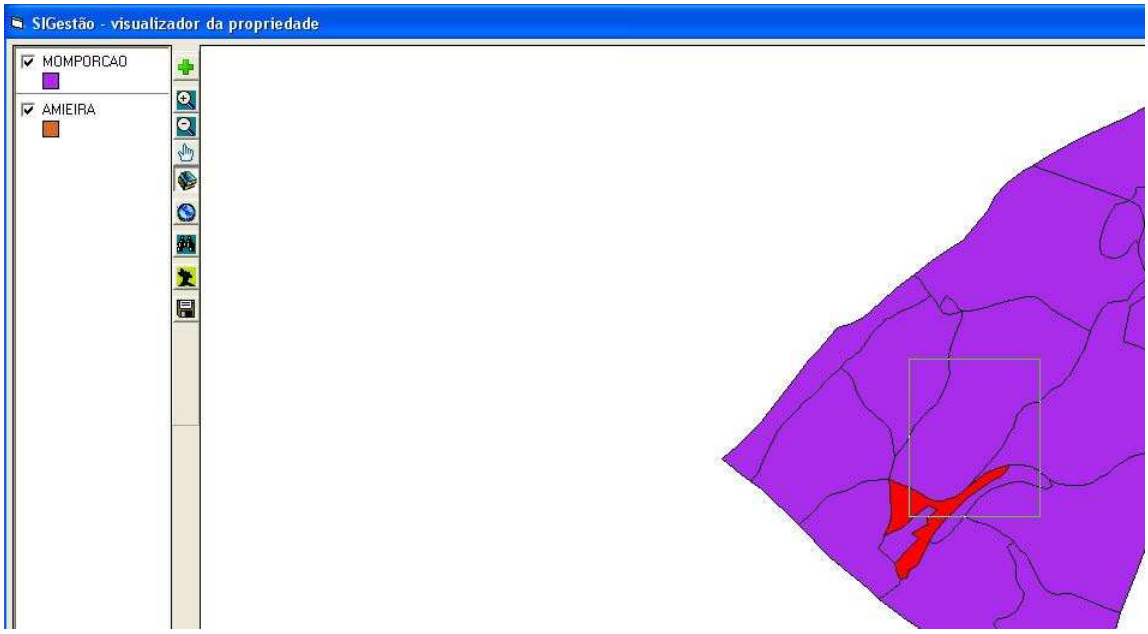

Figura 27 – Desenho do rectângulo sobre as parcelas pretendidas

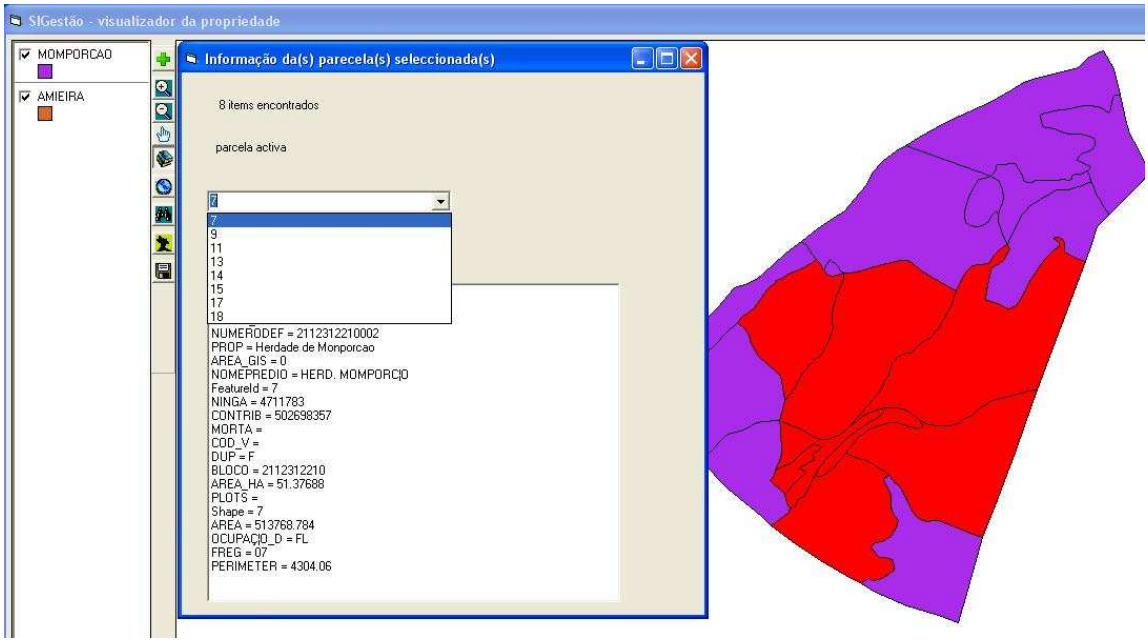

Figura 28 - Janela secundária “Informação da(s) parcela(s) seleccionada(s)”, neste caso mais que uma parcela seleccionada

3.5.6 Ferramenta Procurar e botão Salvar Projecto

Estas duas funcionalidades estão ainda em fase de construção, não estando ainda completamente desenvolvidas até à data de entrega deste trabalho. Como não fazem parte dos requisitos mínimos, nem são essenciais ao bom funcionamento do SIGestão, não impedem que este cumpra o objectivo traçado no início. Mas, apesar de ainda estarem a ser desenvolvidas, não há qualquer impedimento que se faça referência a elas. A ferramenta procurar é a que está mais desenvolvida das duas, faltando pouco para ficar concluída, servirá para que qualquer elemento do mapa seja identificado e localizado por um valor ou campo presente na sua informação intrínseca (figura 29). O "salvar projecto" está também na fase de construção e o seu objectivo é simples: que se possa guardar as alterações efectuadas ao projecto.

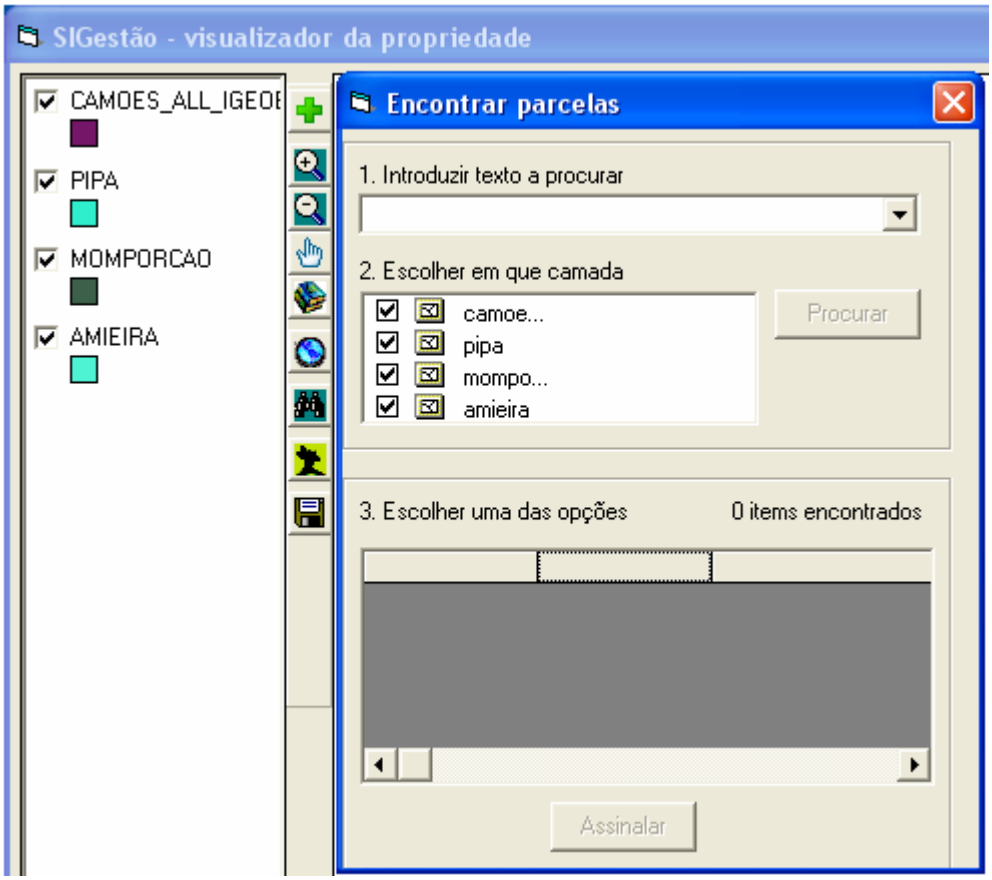

Figura 29 – Janela secundária para procura de elementos no mapa (em construção)

3.5.7 Ligação externa a base de dados

Esta é a funcionalidade essencial do SIGestão. A mais valia do SIGestão assenta sobre esta funcionalidade. Isto deve-se à capacidade que este tem em ligar dados espaciais representados no mapa com informação externa armazenada no AGRO.GESTAO ®.

A ligação pode ser feita a qualquer ficheiro *dbase*, tanto produzido pelo sistema de gestão de informação AGRO.GESTAO ® (este *software* possui essa capacidade) como um outro ficheiro dbase produzido por outro meio. Ou pode ser feita uma ligação a um ficheiro com a extensão .mdb (caso do AGRO.GESTAO ® propriamente dito) que corresponde ao formato do Microsoft Access.

3.5.7.1 Ligar a ficheiro dbase (.dbf)

O utilizador precisa de identificar o campo de ligação e navegar até à localização do ficheiro *dbase* com que se vai executar a conexão. Deve-se executar o seguinte procedimento:

1- Clicar no botão com ícone do AGRO.GESTAO ®, abrirá uma janela com a listagem de todos os campos que fazem parte da informação intrínseca da camada.

2- Escolher o que vai funcionar como ponte, que normalmente é o número da parcela (“numerodef”, no caso em exemplo)

3- Clicar no botão “Ligar” (figura 30), que por sua vez faz abrir uma janela de diálogo do Windows que servirá para navegar até ao local do ficheiro *dbase* pretendido e executar a ligação (figura 31).

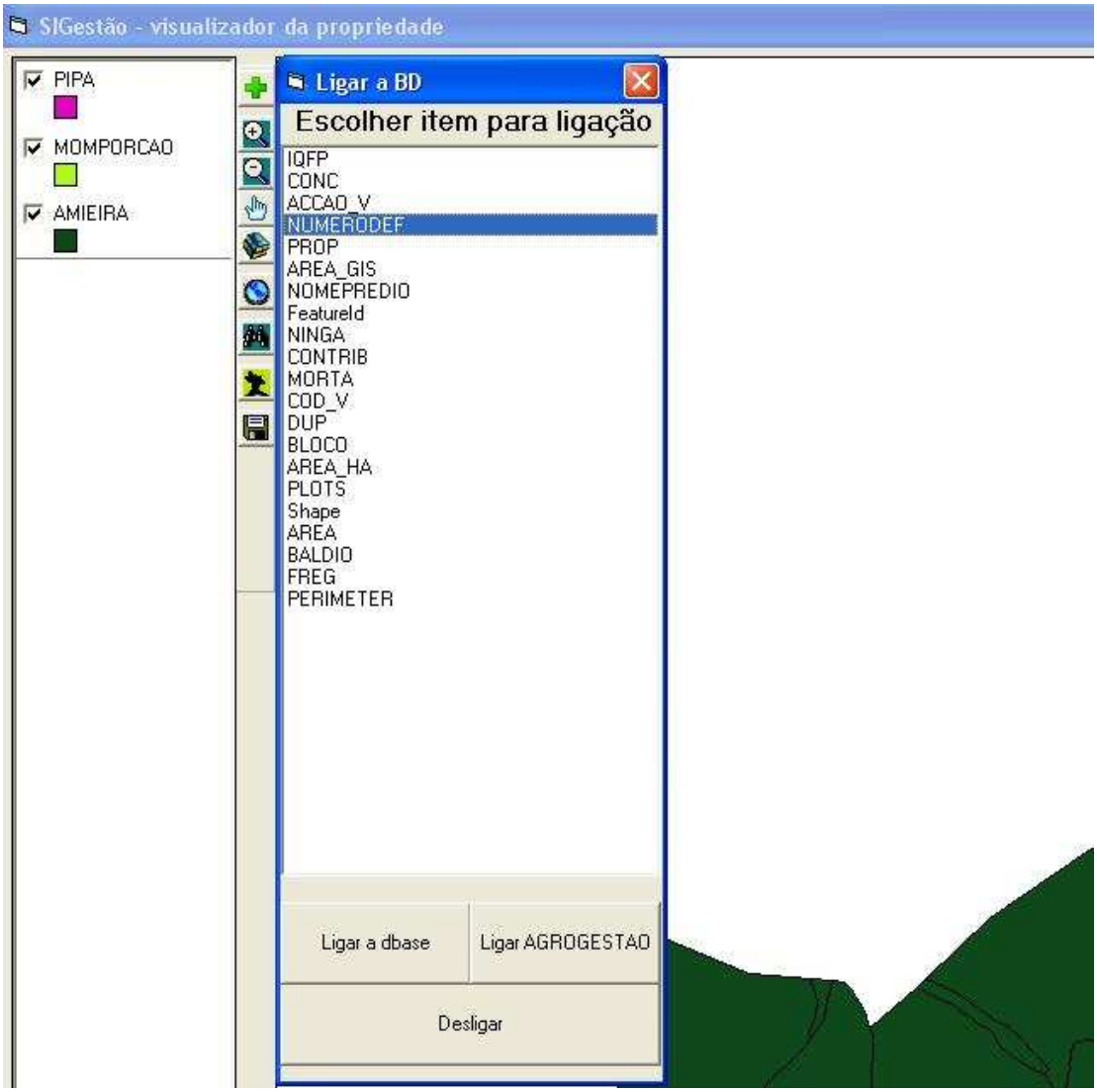

Figura 30 – Janela secundária “Ligar ao AGROGESTÂO” com campos da informação intrínseca da camada em causa

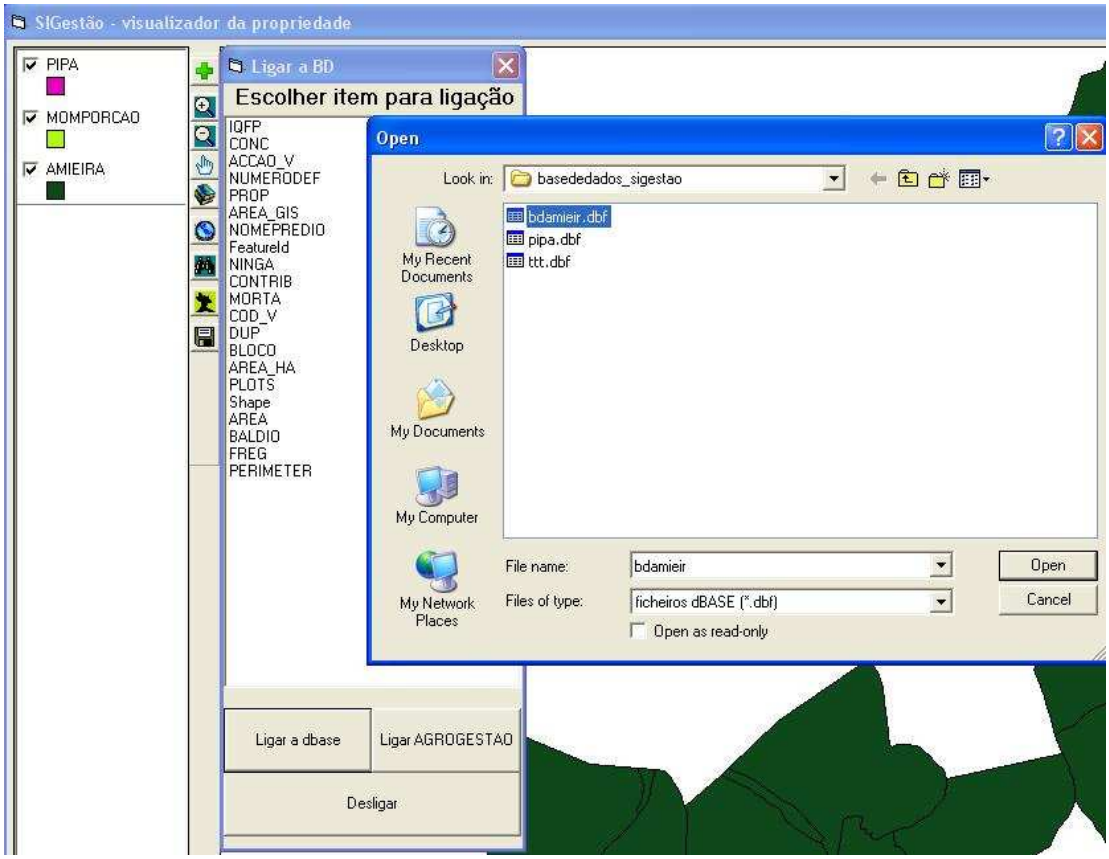

Figura 31 – Janela de diálogo para definir ficheiro dbase a ligar

Se essa ligação tiver sucesso, aparecerá na caixa de listagem, todos os campos presentes no ficheiro *dbase* (figura 32). A partir daqui, podem ser construídos mapas temáticos com a informação externa do mesmo modo como se fez para a informação local (figura 33).

Figura 32 - Janela secundária “Ligar ao AGROGESTÃO” já com os campos originados do ficheiro externo

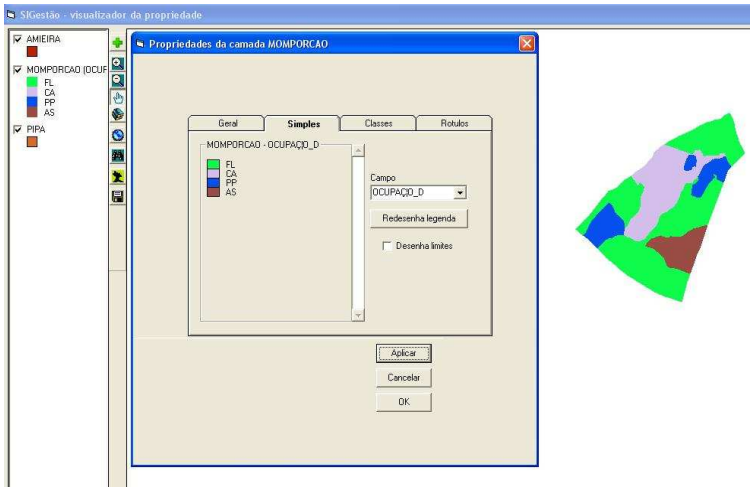

Figura 33 – Mapa temático com dados externos

Para remover a ligação basta voltar à janela “Ligação ao AGRO.GESTAO” e clicar no botão “Desligar”.

3.5.7.2 Ligar ao AGRO.GESTAO

Como no procedimento anterior, o utilizador necessita de escolher o campo de ligação e clicar no botão “ligar AGROGESTAO”. O endereço do ficheiro .mdb do AGRO.GESTAO, bem como da tabela alvo, a ligar já se encontra inscrito no código do formulário. Automaticamente todos os campos dessa tabela do AGRO.GESTAO serão adicionados à lista juntamente com os dados intrínsecos à camada em causa.

Da mesma maneira, esses campos exteriores, podem ser usados para a construção de mapas temáticos (figura 33).

4. Considerações Finais

4.1. Objectivos do capítulo

Este capítulo tratará de discutir algumas questões relacionadas com a avaliação da aplicação desenvolvida ao longo deste projecto. Como todas as aplicações de *software*, o SIGestão não é estático e apesar de estar perto de estar em condições para ser distribuído ao utilizador final, necessita de alguns melhoramentos para que isso aconteça. Apesar disso, é bastante satisfatório perceber que os objectivos iniciais foram atingidos, e que esta versão pode ser o pontapé de saída para uma versão comercial totalmente integrada no AGRO.GESTÃO ® e de grande utilidade para os seus utilizadores.

4.2. Vantagens e limitações da aplicação

O desenvolvimento de uma aplicação SIG, mesmo a partir de biblioteca de objectos SIG específicos, continua a ser um desafio exigente com bastante consumo de tempo e com grandes riscos de não cumprir os requisitos previstos inicialmente (Kosters *et al.* 1996)

A aliança entre o SIGestão e o AGRO.GESTÃO ® produz diversas vantagens para os utilizadores. A ideia de que uma imagem vale mais que mil palavras, aplica-se aqui extraordinariamente bem. A identificação da parcela ou da zona de terreno, no AGRO.GESTÃO ®, como em todos os Sistemas de gestão concorrentes que não contem esta componente SIG, é feita através do número da parcela. Estas parcelas são identificadas por um número de 13 algarismos, ao qual se pode acoplar um nome para a melhor identificação do local. Não se pode dizer que este modo de identificar um local é insuficiente (ou que pode originar erros ou equívocos) mas é natural que essa identificação seja muito

mais facilitada com a visualização de um mapa do local (figura 34), sobre informação *raster* como um ortofotomapa. Além disso, algumas características intrínsecas das parcelas terão um acesso mais fácil, o que pode ajudar significativamente a tomada de decisão.

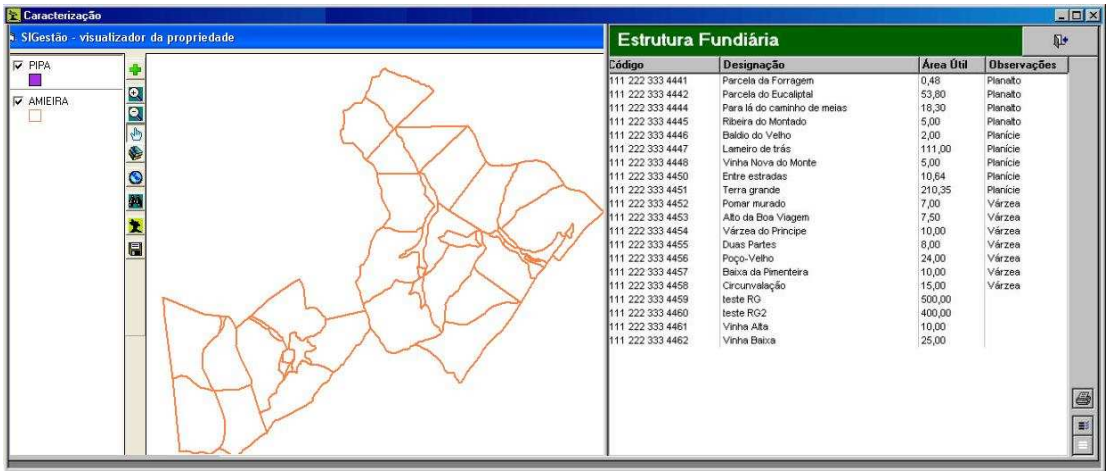

Figura 34- Fotomontagem de previsível aspecto do AGRO.GESTAO ® com a inclusão de módulo de visualização (SIGestão)

As limitações que o SIGestão pode apresentar neste momento, além das inerentes a uma aplicação jovem e pouco testada, alguma complexidade na personalização dos caminhos para o AGRO.GESTAO ®, o que pode significar a necessidade da existência de técnicos com algum grau de conhecimento. Esta pode ser considerada nula uma vez que a FZ.AGROGESTAO possui quadros com essa capacidade. A deslocação desses técnicos ao local onde se trabalha com o AGRO.GESTAO ® para proceder ao devido endereçamento da aplicação pode ser minimizada com recurso assistência remota.

4.3. Desafios e desenvolvimentos futuros

A versão do SIGestão aqui desenvolvida, apesar de respeitar todos os requisitos mínimos, precisa de mais algum investimento para que se possa considerar a sua incorporação no AGRO.GESTAO ®. Estas funcionalidades e outras que oportunamente podem ser desenvolvidas e adicionadas, podem ser a ponte para a continuação do desenvolvimento desta aplicação.

Um dos aspectos que se deve pensar futuramente é o carregamento das camadas espaciais no início da aplicação. Sendo que o património Terra é relativamente constante ao longo do tempo, pode ser melhor que todos os dados espaciais de base, parcelários e ortofotomapas, de cada um dos utilizadores, estivessem carregados no início da aplicação. Desta maneira só podem introduzidos de duas maneiras, sempre em *design mode*, escritos no próprio código da aplicação ou nas características do objecto “map” do MapObjects.

Da mesma forma a ligação ao AGROGESTAO ® pode ser carregada logo no inicio da aplicação, simplificando os processos que o utilizador, caso contrário, teria de fazer. No entanto, no caso de a tabela da base de dados a ligar possuir muitos campos, pode não haver vantagem nisso. Uma tabela do MS Access com muitos campos implica a presença de, também, muitos campos nas caixas de selecção, podendo tornar confuso o seu funcionamento.

A incorporação das sugestões referidas levaria a uma personalização do SIGestão que pode ser considerada excessiva. O SIGestão de um utilizador seria diferente do SIGestão de outro, na medida em que o dado espacial carregado de base e o caminho para o AGRO.GESTAO® também já estaria pré-determinado. Há que clarificar que é possível no entanto tornar o SIGestão mais generalista, assim sendo todos os utilizadores usariam a mesma versão. Esta personalização pode ser benéfica, visto que reduz bastante os processos que o utilizador pode executar, tornando mais fácil a aprendizagem, alargando

também o leque de potenciais utilizadores. A validação desta opção pode ser feita com um pequeno estudo de mercado ou mesmo com um teste a uma amostra de utilizadores.

A evolução de uma aplicação local, como a que foi construída, para uma com acesso remoto, como um *WebGIS*, será de considerar no caso de haver interesse em:

- aceder à informação em pontos diversos;
- que parte do serviço de carregamento e tratamento dos dados, nomeadamente operações mais complexas ou delicadas seja feito, remotamente, por técnicos mais especializados.

A proximidade entre o utilizador/cliente e a empresa fornecedora do produto e do serviço seria maior, podendo muito provavelmente aumentar a satisfação por parte do cliente e a sua eficiência empresarial. Mais uma vez um estudo de mercado ou um teste com amostragem de utilizadores seria necessário para determinar se há receptividade dos clientes a estas ideias.

O *MapServer* e o *GeoServer* são *softwares* de desenvolvimento *OpenSource* que possibilitam a construção de WebGIS e que podem ser usados para atingir o que foi referido anteriormente.

Apesar da aplicação resultante deste projecto ser uma aplicação *freeware*, o MapObjects com que foi construída é um *software* comercial. O desenvolvimento futuro da aplicação pode ficar comprometida, sem a aquisição da licença do MapObjects e do Visual Basic. No entanto existem recursos *freeware* que podem funcionar de igual maneira e que devem ser considerados. Claro que isso implica um investimento de tempo em novas aprendizagens, mas que no final pode compensar.

REFERÊNCIAS BIBLIOGRÁFICAS

Brown, S., 1999, *Visual Basic 6, Bíblia do Programador* (São Paulo: Editora Berkeley).

Committee on Assessing Crop Yield: Site-Specific Farming, I.S., and Research Opportunities, National Research Council, 1997, *Precision Agriculture in the 21st Century: Geospatial and Information Technologies in Crop Management* (Washington, D.C.: The National Academies Press).

ESRI, 1996a, *Building Applications with MapObjects* (Redlands: Environmental Systems Research Institute, INC).

ESRI, 1996b, *MapObjects Programmer's Reference* (Redlands: Environmental Systems Research Institute, INC).

ESRI, 2008a, MapObjects - Windows Edition. (URL: http://www.esri.com/software/mapobjects/index.html consulta em 25-08-2008 2008).

ESRI, 2008b, Página da ESRI. (URL: http://www.esri.com/ (consulta em 25-08-2008).

FZ.AGROGESTAO, 2006, Agrogestao Guia do utilizador. (URL: http://agrogestao.com/ficheiros/agrogestao_manual.pdf consulta em 25-08-2008).

FZ.AGROGESTAO, 2008, (URL: http://agrogestao.com/ consulta em 25-08-2008).

GisLounge, 2008, GisLounge. (URL: http://gislounge.com/ consulta em 25-08-2008).

Huber, B., 2000, A Review of IDRISI32. (URL: http://www.directionsmag.com/features.php?feature_id=40 consulta em 25-08-2008).

Karkanis, S.A. and Bonanos, G.S., 1997, A GIS based Decision Support Tool for the Management of Industrial Risk. In *Proceedings of the SRA '97 Annual Conference*, Society for Risk Analysis (Ed.) (Stockholm, Sweden), pp. 652-658.

Kosters, G. and Six, H.-w., 1996, GeoOOA: Object-oriented analysis for geographic information systems. In *Proceedings of the 2nd IEEE International Conference on Requirements Engineering*, IEEE (Ed.) (Colorado Springs, Colorado, USA., pp. 245-253.

Kraak, M.J., 1996, Integrating Multimédia in *Proceedings of the* Geographical Informations Systems. *IEEE MultiMedia*, 3, pp. 59-65.

Longley, P.A., Goodchild, M.F., Maguire, D.J. and Rhind, D.W., 2005, *Geographic Information Systems ans Science* (Barcelona: John Wiley & Sons Ldt).

Oliveira, J.L.D., Medeiros, C.B. and Cilia, M.A., 1997, Active Customization of GIS User Interfaces. In *Proceedings of the Intl Conference on Data Engineering (ICDE)*, IEEE (Ed.) (Birmingham, U.K.), pp. 487-496.

OpenMap, 2008, BBN Technologies OpenMap. (URL: http://openmap.bbn.com/2008 consulta em 25-08-2008)

Painho, M., Sena, R. and Cabral, P., 1999, Metodologias de desenvolvimento para aplicações de sistemas de informação geográfica. [CDROM] In *Actas do III Encontro de Utilizadores de Informação Geográfica - ESIG 1999*, USIG (Ed.) (TagusPark, Oeiras) 14 p.

Ralston, B.A., 2002, *Developing GIS Solutions with MapObjects and Visual Basic* (Albany: OnWord Press).

Salema, J.P., 2004, Software de Gestão + Formação + Consultoria = Conhecimento. [CDROM] In *Actas 1º Congresso Luso-Brasileiro de Tecnologias de Informação e Comunicação na Agro-Pecuária*, APDTICA (Ed.) (CNEMA, Santarém) 73 p.

ANEXOS

1. Código integral do projecto

```
bd - 1
Option Explicit
Private Sub Command1_Click()
'Identificar BD para relacionar
'
Dim fName As String, dName As String
CommonDialog1.Filter = "ficheiros dBASE (*.dbf)|*.dbf"
CommonDialog1.ShowOpen
'
'Conectar usando DAO
'
If CommonDialog1.filename <> "" And List1.ListIndex > -1 Then
Dim pTable As New MapObjects2.Table
pTable.Database = "dBase IV;DATABASE=" & Left(CommonDialog1.filename,
Len(CommonDialog1.filename) - Le
n(CommonDialog1.FileTitle) - 1)
pTable.Name = Left(CommonDialog1.FileTitle,
Len(CommonDialog1.FileTitle) - 4)
Dim pLayer As MapObjects2.MapLayer, pFName As String, i As Integer
Set pLayer = Form1.Map1.Layers(0)
pFName = List1.List(List1.ListIndex)
'
' Checar item se pertence a tabela
'
For i = 0 To pTable.Records.TableDesc.FieldCount - 1
If pTable.Records.TableDesc.FieldName(i) = pFName Then
'
' Fazer ligação de acordo com o item escolhido
'
If pLayer.AddRelate(pFName, pTable, pFName, True) Then
ListLayers
Else
MsgBox "Relacionamento falhou"
End If
End If
Next i
Else
MsgBox "Escolha um item da lista, e um ficheiro e tente de novo"
End If
End Sub
Private Sub Command2_Click()
'
'Remove ligações
'
Form1.Map1.Layers(0).RemoveRelates
ListLayers
End Sub
Private Sub ListLayers()
'
' Listar items pertencentes ao mapa
'
List1.Clear
Dim pFld As MapObjects2.Field
For Each pFld In Form1.Map1.Layers(0).Records.Fields
```

```
List1.AddItem pFld.Name
Next pFld
End Sub
Private Sub Form_Load()
ListLayers
Command1.Caption = "Ligar a dbase"
Command2.Caption = "Desligar"
Command3.Caption = "Ligar AGROGESTAO"
' Command4.Caption = "Desligar AGROGESTAO"
bd - 2
Label1.Alignment = 1
End Sub
'*********************
'*********************
'*********************
'*********************
'*********************
'*********************
'*********************
Private Sub Command3_Click()
Dim fName As String, dName As String, path As String
Dim pTable As New MapObjects2.Table
pTable.Database = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" &
App.path & "\bd.mdb" '& ";Jet OLED
B:Database Password=nest"
pTable.Name = "bd1"
Dim pLayer As MapObjects2.MapLayer, pFName As String, i As Integer
Set pLayer = Form1.Map1.Layers(0)
pFName = List1.List(List1.ListIndex)
'
' Checar item se pertence a tabela
'
For i = 0 To pTable.Records.TableDesc.FieldCount - 1
If pTable.Records.TableDesc.FieldName(i) = pFName Then
'
' Fazer ligação de acordo com o item escolhido
'
If pLayer.AddRelate(pFName, pTable, pFName, True) Then
ListLayers
Else
MsgBox "Ligação falhou"
End If
End If
Next i
End Sub
'Private Sub Command4_Click()
'
' Remove all existing Relates.
'
' Form1.Map1.Layers(0).RemoveRelates
' ListLayers
' ListView1.ListItems.Clear ' clear list view
'ListView1.ColumnHeaders.Clear
'End Sub
Private Sub Label1_Click()
End Sub
```

```
Form1 - 1
Public g_activelayer As Object
Public recs As MapObjects2.Recordset
Option Explicit
Private Sub Form_Load()
'**fazer o refrehs do combo
'RefreshCombo1
'*
'**sincroniza a legenda com o mapa---Link legend to the Map control
legmapdisp.setMapSource Map1
legmapdisp.LoadLegend True
legmapdisp.ShowAllLegend
legmapdisp.Active(0) = True
'*
End Sub
'
'**********************************************************************
*******************************
************************************
' MAP LEGEND
'**********************************************************************
*******************************
************************************
'
Private Sub legMapDisp_LayerDblClick(Index As Integer)
'Duplo click na camda para abrir o editor de camada
Set g_activelayer = Map1.Layers(Index)
If g_activelayer.LayerType = moImageLayer Then
MsgBox "Não é possivel definir as propriedades de uma imagem" & _
vbCrLf
Exit Sub
End If
frmLayerSymbol.Show vbModal
End Sub
'**********************************************************************
*******************************
************************************
'**********************************************************************
***************
'**Tamanho e Posição da Janela de trabalho
Private Sub Form_Resize()
Dim maptop As Integer
Dim mapleft As Integer
maptop = Label1.Top + Label1.Height + 75
mapleft = legmapdisp.Left + legmapdisp.Width + Toolbar1.Width
Map1.Move mapleft, 75, ScaleWidth - mapleft - 150, ScaleHeight - 150
legmapdisp.Move 75, 75, 2000, ScaleHeight - 150
Toolbar1.Move legmapdisp.Left + legmapdisp.Width, 150,
Toolbar1.ButtonWidth, ScaleHeight
Me.Top = 0
Me.Left = 0
Me.Width = Screen.Width - 2000
Me.Height = Screen.Height - 2000
End Sub
'**Caso se queira por o maximizar janela enable então é necessário
rever estas
```

```
'instruções
'*****************************************************************
***************
Private Sub legmapdisp_BeforeSetLayerVisible(Index As Integer, Cancel
As Boolean)
Map1.Refresh
End Sub
Form1 - 2
Private Sub legmapdisp_MouseDown(Index As Integer, Button As Integer,
Shift As Integer, X As Single, Y
As Single)
'Determina a camada activa
If Map1.Layers.Count > 0 And legmapdisp.getActiveLayer > -1 Then
Set g_activelayer = Map1.Layers(legmapdisp.getActiveLayer)
Else
Set g_activelayer = Nothing
End If
Toolbar1.Buttons(6).Enabled = True
End Sub
Private Sub Map1_AfterLayerDraw(ByVal Index As Integer, ByVal canceled
As Boolean, ByVal hDC As stdole
.OLE_HANDLE)
Dim sym As New MapObjects2.Symbol
If gSelection Is Nothing Then
Exit Sub
End If
If Index > 0 Then
Exit Sub
End If
sym.Color = moRed
gSelection.MoveFirst
Do While Not gSelection.EOF
Map1.DrawShape gSelection("Shape").Value, sym
gSelection.MoveNext
Loop
End Sub
Private Sub Map1_DropFiles(ByVal fileNames As Object, ByVal X As
Single, ByVal Y As Single)
Dim dcx As New MapObjects2.DataConnection
Dim shpfile As Variant
Dim i As Integer
Dim ml As MapObjects2.MapLayer
shpfile = (Dir(fileNames.Item(0), vbDirectory))
shpfile = CStr(Left(shpfile, Len(shpfile) - 4))
dcx.Database = Left(fileNames.Item(0), Len(fileNames.Item(0)) -
Len(shpfile) - 5)
If dcx.Connect Then
For i = 0 To fileNames.Count - 1
Set ml = New MapObjects2.MapLayer
shpfile = Dir(fileNames.Item(i), vbDirectory)
shpfile = CStr(Left(shpfile, Len(shpfile) - 4))
Set ml.GeoDataset = dcx.FindGeoDataset(shpfile)
Map1.Layers.Add ml
legmapdisp.LoadLegend
Next i
Dim ptcoll As New Collection
```

```
Dim linecoll As New Collection
Dim polycoll As New Collection
Dim imagecoll As New Collection
For i = 0 To Map1.Layers.Count - 1
If Map1.Layers(i).LayerType = moImageLayer Then
imagecoll.Add Map1.Layers(i)
ElseIf Map1.Layers(i).LayerType = moMapLayer Then
Select Case Map1.Layers(i).shapeType
Case moShapeTypePoint
ptcoll.Add Map1.Layers(i)
Form1 - 3
Case moShapeTypeLine
linecoll.Add Map1.Layers(i)
Case moShapeTypePolygon
polycoll.Add Map1.Layers(i)
End Select
End If
Next i
Map1.Layers.Clear
Dim p As MapObjects2.MapLayer
For Each p In polycoll
Map1.Layers.Add p
Next p
Dim l As MapObjects2.MapLayer
For Each l In linecoll
Map1.Layers.Add l
Next l
For Each p In ptcoll
Map1.Layers.Add p
Next p
Dim im As MapObjects2.ImageLayer
For Each im In imagecoll
Map1.Layers.Add im
Next im
End If
Map1.Extent = Map1.FullExtent
Map1.Refresh
End Sub
Private Sub Map1_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
Dim curRectangle As New MapObjects2.Rectangle
Dim pt As MapObjects2.Point
If Toolbar1.Buttons(3).Value = 1 Then
Map1.Extent = Map1.TrackRectangle
Map1.Refresh
ElseIf Toolbar1.Buttons(4).Value = 1 Then
Set curRectangle = Map1.Extent
curRectangle.ScaleRectangle (2#)
Set Map1.Extent = curRectangle
Dim Loc As New MapObjects2.Point
Set Loc = Map1.ToMapPoint(X, Y)
Map1.CenterAt Loc.X, Loc.Y
Map1.Refresh
ElseIf Toolbar1.Buttons(5).Value = 1 Then
Map1.Pan
Map1.Refresh
```

```
ElseIf Toolbar1.Buttons(6).Value = 1 Then
Dim r As Rectangle
Dim recs As MapObjects2.Recordset
Dim tol As Double
Set r = Map1.TrackRectangle
Set recs = g_activelayer.SearchShape(r, moAreaIntersect, "")
If recs.EOF Then
tol = Map1.ToMapDistance(100)
Set pt = Map1.ToMapPoint(X, Y)
Set recs = g_activelayer.SearchByDistance(pt, tol, "")
End If
If recs.Count > 0 Then
Set gSelection = recs
Else
Set gSelection = g_activelayer.SearchExpression("featureId = -1")
End If
Map1.Refresh
Form1 - 4
If selectedform.Visible = True Then
LoadSelectedForm
Else
If recs.Count > 0 Then
LoadSelectedForm
selectedform.Show
End If
End If
Else
Dim ln As MapObjects2.Line
Set ln = Map1.TrackLine
MsgBox "Unidades de mapa: " & ln.Length & vbCr & _
"Unidades de controlo: " & Map1.FromMapDistance(ln.Length)
End If
End Sub
Public Sub LoadSelectedForm()
Dim curfield As MapObjects2.Field
SelectedFormUp = True
With selectedform
Select Case g_activelayer.shapeType
Case moPoint
.ftypelabel.Caption = "Point Feature"
Case moLine
.ftypelabel.Caption = "Line Feature"
Case moPolygon
.ftypelabel.Caption = "Polygon Feature"
End Select
If .cboIDs.listCount > 0 Then
.cboIDs.Clear
End If
If .fieldlist.listCount > 0 Then
.fieldlist.Clear
End If
If gSelection.Count <= 0 Then
.numfound.Caption = "No features found"
Else
If gSelection.Count = 1 Then
.numfound.Caption = "1 feature found"
```

```
Else
.numfound.Caption = Str(gSelection.Count) + " items encontrados"
End If
gSelection.MoveFirst
Do While Not gSelection.EOF
.cboIDs.AddItem gSelection("FeatureID").ValueAsString
gSelection.MoveNext
Loop
.cboIDs.ListIndex = 0
gSelection.MoveFirst
For Each curfield In gSelection.Fields
If curfield.Type = moString Then
.fieldlist.AddItem curfield.Name + " = " + curfield.Value
Else
.fieldlist.AddItem curfield.Name + " = " + curfield.ValueAsString
End If
Next curfield
Map1.FlashShape gSelection("shape").Value, 3
End If
End With
SelectedFormUp = False
End Sub
Form1 - 5
Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
Select Case Button.Index
Case 1
Form2.Show
Case 8
Map1.Extent = Map1.FullExtent
Map1.Refresh
Case 10 'find
frmFind.Show
frmFind.ZOrder 0
Case 12 'adicionarbd
bd.Show
'Form3.Show
End Select
End Sub
Form2 - 1
Option Explicit
Dim FormUp As Boolean
Private Sub cmdRemove_Click()
If lstLayers.ListIndex < 0 Then
Exit Sub
End If
Form1.Map1.Layers.Remove (lstLayers.ListIndex)
lstLayers.Clear
Form_Load
refreshButtons
End Sub
Private Sub Form_Load()
Dim i As Integer
FormUp = True
For i = 0 To Form1.Map1.Layers.Count - 1
lstLayers.AddItem Form1.Map1.Layers(i).Name
lstLayers.Selected(i) = Form1.Map1.Layers(i).Visible
```

```
Next i
refreshButtons
FormUp = False
End Sub
Private Sub Form_Unload(Cancel As Integer)
Form1.legmapdisp.LoadLegend
End Sub
Private Sub layerTools_ButtonClick(ByVal Button As MSComctlLib.Button)
Dim curIndex As Integer
curIndex = lstLayers.ListIndex
Select Case Button.Index
Case 1
Form1.Map1.Layers.MoveTo curIndex, curIndex - 1
lstLayers.Clear
Form_Load
lstLayers.ListIndex = curIndex - 1
Case 3
Form1.Map1.Layers.MoveTo curIndex, curIndex + 1
lstLayers.Clear
Form_Load
lstLayers.ListIndex = curIndex + 1
End Select
refreshButtons
Form1.Map1.Refresh
End Sub
Private Sub lstLayers_Click()
refreshButtons
End Sub
Private Sub lstlayers_itemcheck(i As Integer)
If lstLayers.listCount = 0 Then
Exit Sub
End If
If Not FormUp Then
Form1.Map1.Layers(i).Visible = lstLayers.Selected(i)
Form1.Map1.Refresh
End If
End Sub
Private Sub cmdAdd_Click()
addFile
Form2 - 2
Form1.legmapdisp.LoadLegend
End Sub
Private Sub addFile()
'Esta rotina define o common dialog e devolve a shapefile ou imagem
para ser preocessada como camada
Dim fullFile As String, path As String, tempChar As String, ext As
String
Dim Test As Boolean
Dim textPos As Long, periodPos As Long
Dim curPath As String
'Executa o common dialog para seleccionar o ficheiro para abrir.
Dim strShape As String, strImage As String, strOtherImage As String
Dim strCov As String, strAll As String
strShape = "Ficheiros Shape (*.shp)|*.shp"
strImage = "Imagens (*.bmp; *.tif)|*.bmp;*.tif"
```

```
strAll = "Shape
files(*.shp),coverages(*.adf),images(*.bmp,*.tif)|*.shp;*.bmp;*.tif;aa
t.adf;pat.adf;
nat.adf;txt.adf;*.tat;*.pat;*.rat"
CommonDialog1.Filter = strAll & "|" & strShape & "|" & strCov & "|" &
strImage & "|" & strOtherImage
CommonDialog1.DialogTitle = "Seleccionar ficheiro para nova camada"
CommonDialog1.ShowOpen
If CommonDialog1.filename = "" Then Exit Sub
fullFile = Trim$(CommonDialog1.filename)
textPos = Len(fullFile)
Test = False
Do While Test = False
textPos = textPos - 1
tempChar = Mid$(fullFile, textPos, 1)
If tempChar = "." Then
periodPos = textPos
ElseIf tempChar = "\" Or textPos = 0 Then
Test = True
End If
Loop
curPath = Left$(fullFile, textPos - 1)
Dim filename As String
filename = CommonDialog1.FileTitle
ext = LCase(Mid$(fullFile, periodPos + 1, 3))
If ext = "shp" Then
Call addShapeFile(curPath, filename)
Else
Call addImage(fullFile)
End If
lstLayers.Clear
Form_Load
refreshButtons
End Sub
Private Sub addImage(imageFile As String)
Dim iLayer As New ImageLayer
iLayer.File = imageFile
If Form1.Map1.Layers.Add(iLayer) Then
Form1.Map1.Layers.MoveToBottom 0
Else
MsgBox "Este ficheiro, " & imageFile & ", não é um ficheiro de imagem
válido"
End If
End Sub
Form2 - 3
Private Sub addShapeFile(basepath As String, shpfile As String)
Dim dCon As New DataConnection
Dim gSet As GeoDataset
dCon.Database = basepath
If dCon.Connect Then
shpfile = GetFirstToken(shpfile, ".")
Set gSet = dCon.FindGeoDataset(shpfile)
If gSet Is Nothing Then
MsgBox "Erro a abrir " & shpfile
Exit Sub
Else
```

```
Dim newLayer As New MapLayer
newLayer.GeoDataset = gSet
newLayer.Name = shpfile
Form1.Map1.Layers.Add newLayer
End If
Else
MsgBox ConnectErrorMsg(dCon.ConnectError), vbCritical, "Connection
error"
End If
End Sub
Private Sub refreshButtons()
Dim listCount As Integer
Dim curItem As Integer
listCount = Form1.Map1.Layers.Count
curItem = lstLayers.ListIndex
If listCount = 0 Then
cmdremove.Enabled = False
layerTools.Buttons(1).Enabled = False
layerTools.Buttons(3).Enabled = False
End If
If curItem = -1 Then
cmdremove.Enabled = False
layerTools.Buttons(1).Enabled = False
layerTools.Buttons(3).Enabled = False
ElseIf listCount = 1 Then
cmdremove.Enabled = True
layerTools.Buttons(1).Enabled = False
layerTools.Buttons(3).Enabled = False
ElseIf curItem = 0 Then
cmdremove.Enabled = True
layerTools.Buttons(1).Enabled = False
layerTools.Buttons(3).Enabled = True
ElseIf curItem = listCount - 1 Then
cmdremove.Enabled = True
layerTools.Buttons(1).Enabled = True
layerTools.Buttons(3).Enabled = False
Else
cmdremove.Enabled = True
layerTools.Buttons(1).Enabled = True
layerTools.Buttons(3).Enabled = True
End If
End Sub
frmFind - 1
Option Explicit
Dim Recs2() As MapObjects2.Recordset
Dim layerName() As String
Dim layerNum() As Integer
Dim f_Action As String
Dim LayerStatus() As Integer
Private Sub rebuildListView()
lvwLayerList.ListItems.Clear
Dim numLayers As Integer
numLayers = Form1.Map1.Layers.Count
Dim curLayer As Object
Dim curLayerIndex As Integer
Dim curLayerName As String
```

```
Dim curLayerType As Integer
Dim curShapeType As Integer
Dim curListItem As ListItem
Dim iconIndex As Integer
Dim i As Integer
For i = 0 To numLayers - 1
Set curLayer = Form1.Map1.Layers(i)
curLayerName = curLayer.Name
curLayerType = curLayer.LayerType
If curLayerType = moMapLayer Then
curShapeType = curLayer.shapeType
End If
If curLayerType = moImageLayer Then
iconIndex = 4
ElseIf curLayerType = moMapLayer Then
If curShapeType = moShapeTypePoint Then iconIndex = 1
If curShapeType = moShapeTypeLine Then iconIndex = 2
If curShapeType = moShapeTypePolygon Then iconIndex = 3
End If
If LayerStatus(i) = 0 Then iconIndex = iconIndex + 4
If iconIndex <> 4 And iconIndex <> 8 Then
Set curListItem = lvwLayerList.ListItems.Add(, , curLayerName, ,
iconIndex)
End If
Next i
End Sub
Private Sub check_cmdFindButton()
frmFind.cmdFindButton.Enabled = False
If Not IsNull(frmFind.cboSearchList.Text) Then
Dim i As Integer
For i = 0 To Form1.Map1.Layers.Count - 1
If LayerStatus(i) = 1 And frmFind.cboSearchList.Text <> "" Then
frmFind.cmdFindButton.Enabled = True
Exit For
End If
Next i
End If
End Sub
Private Sub toggleCheckbox()
Dim selItem As Integer
frmFind - 2
selItem = lvwLayerList.SelectedItem.Index - 1
If LayerStatus(selItem) = 1 Then
LayerStatus(selItem) = 0
Else
LayerStatus(selItem) = 1
End If
Call rebuildListView
End Sub
Private Sub FindFeatures()
Dim Exp As String, searchExp As String
Dim layerCnt As Integer, layerCount As Integer
layerCnt = Form1.Map1.Layers.Count
ReDim layerName(layerCnt)
ReDim Recs2(layerCnt)
Dim aValue As String, aName As String
```

```
Dim layer_Name As String, featCount As Integer
Dim i As Integer, curCount As Integer
Dim recs As MapObjects2.Recordset
Dim Test As Boolean
Dim aField As Object
Form1.Map1.TrackingLayer.ClearEvents
Screen.MousePointer = vbHourglass
featCount = -1: layerCount = -1
searchExp = " like '%" + frmFind.cboSearchList + "%'"
frmFind.grdFeatList.FixedRows = 0
frmFind.grdFeatList.Rows = 1
For i = 0 To Form1.Map1.Layers.Count - 1
If LayerStatus(i) = 1 Then
layer_Name = Form1.Map1.Layers(i).Name
Exp = ""
aName = ""
For Each aField In Form1.Map1.Layers(i).Records.Fields
If aField.Type = moString Then
If Exp = "" Then
Exp = aField.Name + searchExp
Else
Exp = Exp + " ou " + aField.Name + " como '%" + frmFind.cboSearchList
+ "%'"
End If
If aName = "" Then
aName = aField.Name
End If
End If
Next aField
'Execute the query.
If Exp = "" Then
Set recs = Nothing
Else
Set recs = Form1.Map1.Layers(i).SearchExpression(Exp)
End If
'Loop through selected features and store pointers.
layerCount = layerCount + 1
layerName(layerCount) = layer_Name
Set Recs2(layerCount) = recs
curCount = -1
If Not recs Is Nothing Then
While Not recs.EOF
curCount = curCount + 1
For Each aField In recs.Fields
If aField.Type = moString Then
aValue = recs(aField.Name)
Dim theString As String
frmFind - 3
theString = "*" + frmFind.cboSearchList + "*"
If aValue Like theString Then
featCount = featCount + 1
If featCount = 0 Then
frmFind.grdFeatList.Row = 0: frmFind.grdFeatList.Col = 0
frmFind.grdFeatList.Text = "Camada"
frmFind.grdFeatList.Col = 1: frmFind.grdFeatList.Text = "Campo"
frmFind.grdFeatList.Col = 2: frmFind.grdFeatList.Text = "Valor"
```

```
End If
ReDim Preserve layerNum(2, featCount)
layerNum(1, featCount) = layerCount: layerNum(2, featCount) = curCount
frmFind.grdFeatList.AddItem layer_Name & Chr(9) & aField.Name & Chr(9)
& aValue
End If
End If
Next aField
recs.MoveNext
Wend
End If
End If
Next i
If featCount >= 0 Then
frmFind.grdFeatList.FixedRows = 1
End If
frmFind.lblNumFeats.Caption = Str(featCount + 1) + " items
encontrados"
Test = True
For i = 0 To frmFind.cboSearchList.listCount
If frmFind.cboSearchList.List(i) = frmFind.cboSearchList Then
Test = False
End If
Next i
If Test Then
frmFind.cboSearchList.AddItem frmFind.cboSearchList, 0
End If
Screen.MousePointer = 0
End Sub
Private Sub cboSearchList_Change()
Call check_cmdFindButton
End Sub
Private Sub cboSearchList_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
If frmFind.cmdFindButton.Enabled Then
Call FindFeatures
End If
End If
End Sub
Private Sub Form_Terminate()
Form1.Map1.TrackingLayer.ClearEvents
End Sub
Private Sub grdFeatList_SelChange()
grdFeatList.Col = 0
grdFeatList.ColSel = 2
If grdFeatList.Row > 0 Then
'cmdHighlight.Enabled = True
cmdInsertPin.Enabled = True
' cmdPanto.Enabled = True
'cmdZoomto.Enabled = True
Else
' cmdHighlight.Enabled = False
cmdPanto.Enabled = False
' cmdZoomto.Enabled = False
frmFind - 4
' cmdInsertPin.Enabled = False
```

```
End If
End Sub
Private Sub cmdFindButton_Click()
Call FindFeatures
End Sub
Private Sub Form_Load()
frmFind.grdFeatList.Rows = 1
frmFind.grdFeatList.Cols = 3
frmFind.grdFeatList.ColWidth(0) = 1300
frmFind.grdFeatList.ColWidth(1) = 1300
frmFind.grdFeatList.ColWidth(2) = 2200
frmFind.cboSearchList.Clear
lvwLayerList.View = lvwReport
ReDim LayerStatus(Form1.Map1.Layers.Count)
Dim i As Integer
For i = 0 To Form1.Map1.Layers.Count - 1
If Form1.Map1.Layers(i).LayerType = moImageLayer Then
LayerStatus(i) = 0
Else
LayerStatus(i) = 1
End If
Next i
Call rebuildListView
Dim fnt As New StdFont
fnt.Name = "Wingdings"
fnt.Bold = False
Dim lSymbols As Long
lSymbols = Form1.Map1.TrackingLayer.SymbolCount
Form1.Map1.TrackingLayer.SymbolCount = lSymbols + 1
Form1.Map1.TrackingLayer.Symbol(lSymbols).Color = moBlue
Form1.Map1.TrackingLayer.Symbol(lSymbols).Style = moTrueTypeMarker
Form1.Map1.TrackingLayer.Symbol(lSymbols).Font = fnt
Form1.Map1.TrackingLayer.Symbol(lSymbols).Size = 16
Form1.Map1.TrackingLayer.Symbol(lSymbols).CharacterIndex = 88
Form1.Map1.TrackingLayer.ClearEvents
End Sub
Private Sub Find_Actions(f_Action As String)
Dim curRec As MapObjects2.Recordset
Dim Rect As Rectangle, Rect2 As Rectangle
Dim curIndex As Integer, aIndex As Integer
Dim aRec As Integer, i As Integer
Dim aName As String
Dim shapeX As Double, shapeY As Double
Dim deltax As Double, deltay As Double
Dim theShape As Object, pinPoint As MapObjects2.Point
curIndex = frmFind.grdFeatList.Row - 1
If IsNull(curIndex) Or curIndex < -1 Then
Exit Sub
End If
aIndex = layerNum(1, curIndex)
aRec = layerNum(2, curIndex)
aName = layerName(aIndex)
frmFind - 5
Set curRec = Recs2(aIndex)
curRec.MoveFirst
If aRec > 0 Then
```

```
For i = 1 To aRec
curRec.MoveNext
Next i 'vvvvvvvvvvvvvvvverrrr
End If
Select Case f_Action
Case "cmdHighlight"
Form1.Map1.FlashShape curRec("shape").Value, 3
Case "insert_pin"
Set pinPoint = Nothing
Select Case curRec("shape").Type
Case moPoint
Set pinPoint = curRec("shape").Value
Case moLine
MsgBox "Cannot insert a pin for a line feature"
Case moPolygon
Set pinPoint = curRec("shape").Value.Centroid
End Select
If Not pinPoint Is Nothing Then
Form1.Map1.TrackingLayer.AddEvent pinPoint,
Form1.Map1.TrackingLayer.SymbolCount - 1
End If
Case "cmdPanto"
Set Rect2 = Form1.Map1.Extent
Set theShape = curRec("shape").Value
If curRec("shape").Type = moPoint Then
shapeX = curRec("shape").Value.X
shapeY = curRec("shape").Value.Y
Else
Set Rect = curRec("shape").Value.Extent
shapeX = Rect.Center.X
shapeY = Rect.Center.Y
End If
deltax = shapeX - Rect2.Center.X
deltay = shapeY - Rect2.Center.Y
Rect2.Offset deltax, deltay
Form1.Map1.Extent = Rect2
Form1.Map1.Refresh
Form1.Map1.FlashShape theShape, 3
Case "cmdZoomto"
Set theShape = curRec("shape").Value
If curRec("shape").Type = moPoint Then
Set Rect2 = Form1.Map1.Extent
shapeX = curRec("shape").Value.X
shapeY = curRec("shape").Value.Y
deltax = shapeX - Rect2.Center.X
deltay = shapeY - Rect2.Center.Y
Rect2.Offset deltax, deltay
Rect2.ScaleRectangle 0.1
Form1.Map1.Extent = Rect2
Else
Set Rect = curRec("shape").Value.Extent
Rect.ScaleRectangle 1.1
Form1.Map1.Extent = Rect
End If
Form1.Map1.Refresh
Form1.Map1.FlashShape theShape, 3
```

```
End Select
End Sub
Private Sub lvwLayerList_ItemClick(ByVal Item As MSComctlLib.ListItem)
Call toggleCheckbox
Call check_cmdFindButton
End Sub
frmFind - 6
Private Sub cmdHighlight_Click()
Call Find_Actions("cmdHighlight")
End Sub
Private Sub cmdInsertPin_Click()
Call Find_Actions("insert_pin")
End Sub
Private Sub cmdPanto_Click()
Call Find_Actions("cmdPanto")
End Sub
Private Sub cmdZoomto_Click()
Call Find_Actions("cmdZoomto")
End Sub
frmLayerSymbol - 1
Option Explicit
Dim strMarkerStyle(4) As String
Dim strLineStyle(4) As String
Dim strFillStyle(10) As String
Dim strPanelDesc(5) As String
Dim lyr As MapObjects2.MapLayer
Dim recs As MapObjects2.Recordset
Dim tDesc As MapObjects2.TableDesc
Dim flds As MapObjects2.Fields
Dim a As Integer
Dim vmr As MapObjects2.ValueMapRenderer
Dim cbr As MapObjects2.ClassBreaksRenderer
Dim lr As MapObjects2.LabelRenderer
Dim lp As MapObjects2.LabelPlacer
Dim colorMask, colorText As Long
Dim justOpened As Boolean
Dim lFactor As Double
Private Sub Form_Load()
Me.Top = 0
Me.Left = Screen.Width - Me.Width
colorText = moBlack
colorMask = moWhite
Set lyr = Form1.g_activelayer
Set recs = lyr.Records
Set tDesc = recs.TableDesc
Set flds = recs.Fields
frmLayerSymbol.Caption = "Propriedades da camada " & UCase(lyr.Name)
strPanelDesc(0) = ""
strPanelDesc(1) = ""
strPanelDesc(2) = ""
strPanelDesc(3) = ""
strFillStyle(0) = "Solido"
strFillStyle(1) = "Transparente"
strFillStyle(2) = "Linhas Horizontais"
strFillStyle(3) = "Linhas Verticais"
strFillStyle(4) = "Diagonal crescente"
```

```
strFillStyle(5) = "Diagonal descendente"
strFillStyle(6) = "Cruzado"
strFillStyle(7) = "Diagonal cruzado"
strFillStyle(8) = "Sombreado Claro"
strFillStyle(9) = "Sombreado"
strFillStyle(10) = "Sombreado Escuro"
justOpened = True
'Lê os tipos de conteudo da camada activa e corre o LOAD apropriado
para preencher o TAB com toda a
info necessária
Select Case True
Case lyr.Renderer Is Nothing
sstLayerProp.Tab = 0
Call LoadSingleSymbol
Case TypeOf lyr.Renderer Is MapObjects2.ValueMapRenderer
sstLayerProp.Tab = 1
Call LoadUniqueValues
Case TypeOf lyr.Renderer Is MapObjects2.ClassBreaksRenderer
sstLayerProp.Tab = 2
Call LoadClassBreaks
Case TypeOf lyr.Renderer Is MapObjects2.LabelPlacer
sstLayerProp.Tab = 3
colorMask = lyr.Renderer.MaskColor
frmLayerSymbol - 2
Call LoadNoOverlapLabels
Case Else
sstLayerProp.Tab = 0
Call LoadSingleSymbol
End Select
lFactor = 0.9
End Sub
'Carregado no botão aplicar vai correr o form adequado
Private Sub cmdApply_Click()
Select Case sstLayerProp.Tab
Case 0: Call ApplySingleSymbol
Case 1: Call ApplyUniqueValues
Case 2: Call ApplyClassBreaks
Case 3: Call ApplyNoOverlapLabels
End Select
'Rescreve a Legenda
Form1.legmapdisp.LoadLegend
'Redesenha o Mapa
Form1.Map1.Refresh
End Sub
Private Sub cmdCancel_Click()
Unload frmLayerSymbol
End Sub
Private Sub cmdOK_Click()
'Apply e depois unload
Call cmdApply_Click
Unload frmLayerSymbol
End Sub
Private Sub cmdNOL_Click()
'Tipo de letra dos rotulos
cdlgLayerProp.Color = colorText
cdlgLayerProp.Flags = cdlCFEffects Or cdlCFBoth
```

```
cdlgLayerProp.ShowFont
txtNOL.ForeColor = cdlgLayerProp.Color
txtNOL.Text = cdlgLayerProp.FontName
colorText = cdlgLayerProp.Color
End Sub
Private Sub cmdCB_Click()
'Redesenha a legenda no TAB classes
Call PopulateNewCBlegend(cboCB(0).Text)
cmdApply.Enabled = True
cmdOK.Enabled = True
End Sub
Private Sub cmdUV_Click()
'Redesenha a legenda no TAB unica
Call PopulateNewUVlegend(cboUV.Text)
cmdApply.Enabled = True
cmdOK.Enabled = True
frmLayerSymbol - 3
End Sub
Private Sub cboSSP_Click(Index As Integer)
Dim fnt As New StdFont
Select Case Index
Case 0 'escolhe o tipo de preenchimento
If Index = 0 Then
If cboSSP(0).Text = "TrueType marker" Then
Dim i As Integer
cboSSP(1).Enabled = True
cboSSP(2).Enabled = True
lblSSP(3).Enabled = True
lblSSP(4).Enabled = True
lblSSP(5).Enabled = True
lblSSP(7).Enabled = True
'hsbSSP.Enabled = True
For i = 0 To Screen.FontCount - 1
cboSSP(1).AddItem Screen.Fonts(i)
Next i
cboSSP(1).ListIndex = 0
fnt.Name = cboSSP(1).Text
Set cboSSP(2).Font = fnt
cboSSP(2).Clear
For i = 0 To 255
cboSSP(2).AddItem Chr(i)
Next
Else 'if not TT font, then disable controls specific to TT fonts
'cboSSP(1).Clear
'cboSSP(2).Clear
'cboSSP(1).Enabled = False
'cboSSP(2).Enabled = False
'lblSSP(3).Enabled = False
'lblSSP(4).Enabled = False
'lblSSP(5).Enabled = False
'lblSSP(7).Enabled = False
'hsbSSP.Enabled = False
End If
End If
Case 1 'combo com os tipos de letras escolhidas
cboSSP(2).Clear
```

```
fnt.Name = cboSSP(1).Text
Set cboSSP(2).Font = fnt
For i = 0 To 255
cboSSP(2).AddItem Chr(i)
Next
End Select
End Sub
'Private Sub hsbSSP_Change()
'Sets the rotation on a single symbol point marker
'that is using a TT font
'lblSSP(7).Caption = hsbSSP.Value
'End Sub
'Private Sub hsbSL_Scroll()
'Sets the rotation on standard label text
'lblSL(7).Caption = hsbSL.Value
frmLayerSymbol - 4
'End Sub
'Private Sub hsbSL_Change()
'Sets the rotation on standard label text
'lblSL(7).Caption = hsbSL.Value
'End Sub
Private Sub picCBramp_Click(Index As Integer)
'começo e o fim da rampa de cores na TAB classes
cdlgLayerProp.ShowColor
picCBramp(Index).BackColor = cdlgLayerProp.Color
End Sub
Private Sub picNOL_Click()
'Muda a cor de fundo dos rotulos
If colorMask <> moWhite Then
cdlgLayerProp.Color = colorMask
End If
cdlgLayerProp.ShowColor
picNOL.BackColor = cdlgLayerProp.Color
colorMask = cdlgLayerProp.Color
'liga a cor de fundo check box
chkNOL(2).Value = 1
End Sub
Private Sub picSSP_Click(Index As Integer)
'define a cor para o TAB Unico
cdlgLayerProp.ShowColor
picSSP(Index).BackColor = cdlgLayerProp.Color
End Sub
Private Sub vsbUV_Change()
'redimendionamento do local da previsão da legenda
fraUVinner.Top = 200 - (vsbUV.Value * 200)
End Sub
Private Sub sstLayerProp_Click(PreviousTab As Integer)
'Quando se carrega em cada um dos TABs carrega as propriedades já
existentes da camada em causa par
a o TAB aberto
Dim lyrRend As Object
lblPanelDesc.Caption = strPanelDesc(sstLayerProp.Tab)
If lyr.Renderer Is Nothing Then
Set lyrRend = New MapObjects2.Point
Else
Set lyrRend = lyr.Renderer
```

```
End If
If PreviousTab = 2 Then
If TypeOf lyrRend Is MapObjects2.ValueMapRenderer Then
Exit Sub
End If
End If
Select Case sstLayerProp.Tab
Case 0
If TypeOf lyrRend Is MapObjects2.Point Then
frmLayerSymbol - 5
Call LoadSingleSymbol
Else
Call InitSingleSymbol
End If
Case 1
If TypeOf lyrRend Is MapObjects2.ValueMapRenderer Then
Call LoadUniqueValues
Else
Call InitUniqueValues
End If
Case 2
If TypeOf lyrRend Is MapObjects2.ClassBreaksRenderer Then
Call LoadClassBreaks
Else
Call InitClassBreaks
End If
Case 3
If TypeOf lyrRend Is MapObjects2.LabelPlacer Then
Call LoadNoOverlapLabels
Else
Call InitNoOverlapLabels
End If
End Select
End Sub
'''
'THE SIX PROCEDURES THAT FOLLOW, THAT BEGIN WITH THE WORD "INIT..."
'ARE THOSE THAT RUN WHEN A RENDERER IS CHOSEN WHICH DOES NOT
'COINCIDE WITH THE ACTIVE LAYER'S CURRENT RENDERER. THE OPTION
'CONTROLS ON THAT TAB ARE LOADED WITH DEFAULT VALUES THAT THE USER
'CAN CHANGE.
' InitSingleSymbol()
' InitUniqueValues()
' InitClassBreaks()
' InitStandardLabels()
' InitNoOverlapLabels()
' InitZRenderer()
'''
Private Sub InitSingleSymbol()
Dim i As Integer
Dim fnt As New StdFont
cboSSP(0).Clear
Select Case lyr.shapeType
Case moShapeTypePoint
'set control visibility
cboSSP(1).Visible = True
cboSSP(2).Visible = True
```

```
chkSSP.Visible = False
'hsbSSP.Visible = True
picSSP(1).Visible = False
lblSSP(3).Visible = True
lblSSP(4).Visible = True
lblSSP(5).Visible = True
lblSSP(6).Visible = False
lblSSP(7).Visible = True
'retrieve and display current values
txtSSP(0).Text = 5
lblSSP(0).Caption = "Marker Color:"
lblSSP(2).Caption = "Size:"
For i = 0 To 4
cboSSP(0).AddItem strMarkerStyle(i)
Next
picSSP(0).BackColor = moGreen
cboSSP(0).Text = strMarkerStyle(moSquareMarker)
frmLayerSymbol - 6
cboSSP(0).ListIndex = 1
'hsbSSP.Value = 0
lblSSP(7).Caption = "0"
cboSSP(1).Enabled = False
cboSSP(2).Enabled = False
lblSSP(3).Enabled = False
lblSSP(4).Enabled = False
lblSSP(5).Enabled = False
lblSSP(7).Enabled = False
'hsbSSP.Enabled = False
Case moShapeTypeMultipoint
'set control visibility
cboSSP(1).Visible = True
cboSSP(2).Visible = True
chkSSP.Visible = False
'hsbSSP.Visible = True
picSSP(1).Visible = False
lblSSP(3).Visible = True
lblSSP(4).Visible = True
lblSSP(5).Visible = True
lblSSP(6).Visible = False
lblSSP(7).Visible = True
'retrieve and display current values
txtSSP(0).Text = 5
lblSSP(0).Caption = "Marker Color:"
lblSSP(2).Caption = "Size:"
For i = 0 To 4
cboSSP(0).AddItem strMarkerStyle(i)
Next
picSSP(0).BackColor = moGreen
cboSSP(0).Text = strMarkerStyle(moSquareMarker)
cboSSP(0).ListIndex = 1
'hsbSSP.Value = 0
lblSSP(7).Caption = "0"
cboSSP(1).Enabled = False
cboSSP(2).Enabled = False
lblSSP(3).Enabled = False
lblSSP(4).Enabled = False
```

```
lblSSP(5).Enabled = False
lblSSP(7).Enabled = False
'hsbSSP.Enabled = False
Case moLine
'set visibility
cboSSP(1).Visible = False
cboSSP(2).Visible = False
chkSSP.Visible = False
'hsbSSP.Visible = False
picSSP(1).Visible = False
lblSSP(3).Visible = False
lblSSP(4).Visible = False
lblSSP(5).Visible = False
lblSSP(6).Visible = False
lblSSP(7).Visible = False
'retrieve and display current values
txtSSP(0).Text = 1
lblSSP(0).Caption = "Line Color:"
lblSSP(2).Caption = "Line width:"
For i = 0 To 4
cboSSP(0).AddItem strLineStyle(i)
Next
picSSP(0).BackColor = moBlue
cboSSP(0).Text = strLineStyle(0)
cboSSP(0).ListIndex = 0
Case moPolygon
'set visibility
'cboSSP(1).Visible = False
' cboSSP(2).Visible = False
chkSSP.Visible = True
'hsbSSP.Visible = False
frmLayerSymbol - 7
picSSP(1).Visible = True
lblSSP(3).Visible = False
'lblSSP(4).Visible = False
'lblSSP(5).Visible = False
'lblSSP(6).Visible = True
'lblSSP(7).Visible = False
'retrieve and display current values
txtSSP(0).Text = "1"
lblSSP(0).Caption = "cor de preenchimento:"
lblSSP(2).Caption = "limite:"
For i = 0 To 10
cboSSP(0).AddItem strFillStyle(i)
Next
picSSP(0).BackColor = moLightGray
picSSP(1).BackColor = moBlack
cboSSP(0).Text = strFillStyle(0)
cboSSP(0).ListIndex = 0
chkSSP.Value = 1
End Select
cmdApply.Enabled = True
cmdOK.Enabled = True
End Sub
Private Sub InitUniqueValues()
Dim i As Integer
```

```
'Load ComboBox with layer field names
cboUV.Clear
For i = 0 To tDesc.FieldCount - 1
cboUV.AddItem tDesc.FieldName(i)
Next
cboUV.ListIndex = 0
fraUVouter.Caption = "Legend Preview"
'If a legend already exists, unload it
If picUV.Count > 1 Then
For i = (picUV.Count - 1) To 1 Step -1
Unload picUV(i)
Unload lblUV(i)
Next
End If
picUV(0).Visible = False
lblUV(0).Visible = False
cmdApply.Enabled = False
cmdOK.Enabled = False
chkUV.Visible = (lyr.shapeType = moShapeTypePolygon)
End Sub
Private Sub InitClassBreaks()
Dim i As Integer
Dim fld As MapObjects2.Field
'Clear and reload ComboBoxes
cboCB(0).Clear
cboCB(0).AddItem "FeatureID"
cboCB(1).ListIndex = 3
For i = 0 To tDesc.FieldCount - 1
Set fld = flds(tDesc.FieldName(i))
If fld.Type = moDouble Or fld.Type = moLong Then
cboCB(0).AddItem fld.Name
End If
Next
cboCB(0).ListIndex = 0
'If a legend already exists, unload it
frmLayerSymbol - 8
If picCBlegend.Count > 1 Then
For i = (picCBlegend.Count - 1) To 1 Step -1
Unload picCBlegend(i)
Unload lblCBlegend(i)
Next
End If
picCBlegend(0).Visible = False
lblCBlegend(0).Visible = False
cmdApply.Enabled = False
cmdOK.Enabled = False
chkCB.Visible = (lyr.shapeType = moShapeTypePolygon)
End Sub
'Private Sub InitStandardLabels()
'Dim i As Integer
'Dim sFirstStringFld As Integer, bFoundString As Boolean
'bFoundString = False
''' For i = 0 To tDesc.FieldCount - 1
'cboSL(0).AddItem tDesc.FieldName(i)
'If tDesc.FieldType(i) = moLong Or _
' tDesc.FieldType(i) = moDouble Then
```

```
'cboSL(3).AddItem tDesc.FieldName(i)
'cboSL(4).AddItem tDesc.FieldName(i)
'ElseIf (Not bFoundString) And (tDesc.FieldType(i) = moString) Then
' sFirstStringFld = i
' bFoundString = True
' End If
' Next
' cboSL(0).ListIndex = sFirstStringFld
' cboSL(1).ListIndex = 1
' cboSL(2).ListIndex = 1
' cdlgLayerProp.FontName = "MS Sans Serif"
' cdlgLayerProp.FontSize = 10
' cdlgLayerProp.FontBold = False
' cdlgLayerProp.FontItalic = False
' cdlgLayerProp.FontStrikethru = False
' cdlgLayerProp.FontUnderline = False
' cdlgLayerProp.Color = moBlack
' cmdApply.Enabled = True
' cmdOK.Enabled = True
' This function may be called if the layer currently has a
' LabelPlacer set.
' If Not lyr.Renderer Is Nothing Then
' If TypeOf lyr.Renderer Is MapObjects2.LabelPlacer Then
' Work out scale based on layer's extent. This value will be used
' to help convert the slider height value into a label size in
' map units.
' Dim scaleHeightUnit As Double
' scaleHeightUnit = lyr.Extent.Width / 10000
' Use size from existing LabelPlacer.
' Dim currSizeMapUnits As Double
' currSizeMapUnits = lyr.Renderer.DefaultSymbol.Height
' Convert this Map Units size to a size in Points size, and set
' the Font size value appropriately.
' Dim sglTemp As Single
' sglTemp = Form1.Map1.FromMapDistance(currSizeMapUnits) * (1 /
lFactor)
' Dim currSizePoints As Double
' currSizePoints = ScaleY(sglTemp, vbTwips, vbPoints)
' cdlgLayerProp.FontSize = currSizePoints
' Copy other values if appropriate.
' cboSL(0).Text = lyr.Renderer.Field
frmLayerSymbol - 9
' End If
'End If
'End Sub
Private Sub InitNoOverlapLabels()
Dim i As Integer
cboNOL.Clear
Dim sFirstStringFld As Integer, bFoundString As Boolean
bFoundString = False
For i = 0 To tDesc.FieldCount - 1
cboNOL.AddItem tDesc.FieldName(i)
If (Not bFoundString) And (tDesc.FieldType(i) = moString) Then
sFirstStringFld = i
bFoundString = True
End If
```

```
Next
cboNOL.ListIndex = sFirstStringFld
cdlgLayerProp.FontName = "MS Sans Serif"
cdlgLayerProp.FontSize = 10
cdlgLayerProp.FontBold = False
cdlgLayerProp.FontItalic = False
cdlgLayerProp.FontStrikethru = False
cdlgLayerProp.FontUnderline = False
cdlgLayerProp.Color = moBlack
cmdApply.Enabled = True
cmdOK.Enabled = True
'bug: o franol so fica enabled quando se volta a ir as propriedades
VERVERVERVER
fraNOL(0).Enabled = (lyr.shapeType <> moShapeTypePolygon)
For i = 0 To optNOL.Count - 1
optNOL(i).Enabled = (lyr.shapeType <> moShapeTypePolygon)
Next
' This function may be called if the layer currently has a
' LabelRenderer set. Approximate the size of the existing label
' size (Font points) into Map units.
If Not lyr.Renderer Is Nothing Then
If TypeOf lyr.Renderer Is MapObjects2.LabelRenderer Then
' Work out scale based on layer's extent. This value will be used
' to help convert the slider height value into a label size in
' map units.
Dim scaleHeightUnit As Double
scaleHeightUnit = lyr.Extent.Width / 10000
' Use size from existing LabelRenderer - need to work out an
' equivalent size in Map Units from this font size.
Dim currSizeFontPoints As Double
currSizeFontPoints = CInt(lyr.Renderer.Symbol(0).Font.Size)
' Have a current LabelRenderer Symbol Font Size - in points.
' convert this points size to a MapUnits size, and set the slider
' value appropriately.
Dim sglTemp As Single
sglTemp = ScaleX(CSng(currSizeFontPoints), vbPoints, vbTwips)
Dim currSizeMapUnits As Double
currSizeMapUnits = Form1.Map1.ToMapDistance(sglTemp) * lFactor
hsbNOL.Value = 1000 - (currSizeMapUnits / scaleHeightUnit)
' Copy other values if appropriate.
cboNOL.Text = lyr.Renderer.Field
End If
End If
End Sub
'Public Sub InitZRenderer()
frmLayerSymbol - 10
'Dim i As Integer
'If picZRlegend.Count > 1 Then
' For i = (picZRlegend.Count - 1) To 1 Step -1
' Unload picZRlegend(i)
' Unload lblZRlegend(i)
' Next
' End If
'picZRlegend(0).Visible = False
' lblZRlegend(0).Visible = False
' cmdApply.Enabled = False
```

```
' cmdOK.Enabled = False
'End Sub
'''
'THE SIX PROCEDURES THAT FOLLOW, THAT BEGIN WITH THE WORD "LOAD..."
'ARE THOSE THAT RUN WHEN A RENDERER IS CHOSEN WHICH COINCIDES
'WITH THE ACTIVE LAYER'S CURRENT RENDERER. THE PROPERTIES OF
'THAT RENDERER ARE LOADED INTO THE OPTION CONTROLS ON THAT
'RENDERER'S FORM. THE USER CAN CHANGE THEM AT THAT POINT.
' LoadSingleSymbol()
' LoadUniqueValues()
' LoadClassBreaks()
' LoadStandardLabels()
' LoadNoOverlapLabels()
' LoadZRenderer()
'''
Private Sub LoadSingleSymbol()
Dim i As Integer
Dim fnt As New StdFont
cboSSP(0).Clear
Select Case lyr.shapeType
Case moShapeTypePoint
'set control visibility
cboSSP(1).Visible = True
cboSSP(2).Visible = True
chkSSP.Visible = False
'hsbSSP.Visible = True
picSSP(1).Visible = False
lblSSP(3).Visible = True
lblSSP(4).Visible = True
lblSSP(5).Visible = True
lblSSP(6).Visible = False
lblSSP(7).Visible = True
'retrieve and display current values
txtSSP(0).Text = lyr.Symbol.Size
lblSSP(0).Caption = "Marker Color:"
lblSSP(2).Caption = "Size:"
For i = 0 To 4
cboSSP(0).AddItem strMarkerStyle(i)
Next
picSSP(0).BackColor = lyr.Symbol.Color
cboSSP(0).Text = strMarkerStyle(lyr.Symbol.Style)
cboSSP(0).ListIndex = lyr.Symbol.Style
'hsbSSP.Value = lyr.Symbol.Rotation
lblSSP(7).Caption = lyr.Symbol.Rotation
If lyr.Symbol.Style = moTrueTypeMarker Then
cboSSP(1).Enabled = True
cboSSP(2).Enabled = True
lblSSP(3).Enabled = True
lblSSP(4).Enabled = True
lblSSP(5).Enabled = True
lblSSP(7).Enabled = True
frmLayerSymbol - 11
'hsbSSP.Enabled = True
For i = 0 To Screen.FontCount - 1
cboSSP(1).AddItem Screen.Fonts(i)
Next i
```

```
cboSSP(1).Text = lyr.Symbol.Font.Name
For i = 0 To cboSSP(1).listCount - 1
If cboSSP(1).List(cboSSP(1).ListIndex) = cboSSP(1).Text Then
cboSSP(1).ListIndex = 1
Exit For
End If
Next
fnt.Name = cboSSP(1).Text
Set cboSSP(2).Font = fnt
cboSSP(2).Clear
For i = 0 To 255
cboSSP(2).AddItem Chr(i)
Next
cboSSP(2).Text = lyr.Symbol.CharacterIndex
cboSSP(2).ListIndex = lyr.Symbol.CharacterIndex
End If
Case moShapeTypeMultipoint
'set control visibility
cboSSP(1).Visible = True
cboSSP(2).Visible = True
chkSSP.Visible = False
'hsbSSP.Visible = True
picSSP(1).Visible = False
lblSSP(3).Visible = True
lblSSP(4).Visible = True
lblSSP(5).Visible = True
lblSSP(6).Visible = False
lblSSP(7).Visible = True
'retrieve and display current values
txtSSP(0).Text = lyr.Symbol.Size
lblSSP(0).Caption = "Marker Color:"
lblSSP(2).Caption = "Size:"
For i = 0 To 4
cboSSP(0).AddItem strMarkerStyle(i)
Next
picSSP(0).BackColor = lyr.Symbol.Color
cboSSP(0).Text = strMarkerStyle(lyr.Symbol.Style)
cboSSP(0).ListIndex = lyr.Symbol.Style
'hsbSSP.Value = lyr.Symbol.Rotation
lblSSP(7).Caption = lyr.Symbol.Rotation
If lyr.Symbol.Style = moTrueTypeMarker Then
cboSSP(1).Enabled = True
cboSSP(2).Enabled = True
lblSSP(3).Enabled = True
lblSSP(4).Enabled = True
lblSSP(5).Enabled = True
lblSSP(7).Enabled = True
'hsbSSP.Enabled = True
For i = 0 To Screen.FontCount - 1
cboSSP(1).AddItem Screen.Fonts(i)
Next i
cboSSP(1).Text = lyr.Symbol.Font.Name
For i = 0 To cboSSP(1).listCount - 1
If cboSSP(1).List(cboSSP(1).ListIndex) = cboSSP(1).Text Then
cboSSP(1).ListIndex = 1
Exit For
```

```
End If
Next
fnt.Name = cboSSP(1).Text
Set cboSSP(2).Font = fnt
cboSSP(2).Clear
For i = 0 To 255
cboSSP(2).AddItem Chr(i)
Next
cboSSP(2).Text = lyr.Symbol.CharacterIndex
cboSSP(2).ListIndex = lyr.Symbol.CharacterIndex
frmLayerSymbol - 12
End If
Case moShapeTypeLine
'set visibility
cboSSP(1).Visible = False
cboSSP(2).Visible = False
chkSSP.Visible = False
'hsbSSP.Visible = False
picSSP(1).Visible = False
lblSSP(3).Visible = False
lblSSP(4).Visible = False
lblSSP(5).Visible = False
lblSSP(6).Visible = False
lblSSP(7).Visible = False
'retrieve and display current values
txtSSP(0).Text = lyr.Symbol.Size
lblSSP(0).Caption = "Line Color:"
lblSSP(2).Caption = "Line width:"
For i = 0 To 4
cboSSP(0).AddItem strLineStyle(i)
Next
picSSP(0).BackColor = lyr.Symbol.Color
cboSSP(0).Text = strLineStyle(lyr.Symbol.Style)
cboSSP(0).ListIndex = lyr.Symbol.Style
Case moShapeTypePolygon
'set visibility
'cboSSP(1).Visible = False
'cboSSP(2).Visible = False
chkSSP.Visible = True
'hsbSSP.Visible = False
picSSP(1).Visible = True
'lblSSP(3).Visible = False
'lblSSP(4).Visible = False
'lblSSP(5).Visible = False
'lblSSP(6).Visible = True
'lblSSP(7).Visible = False
'retrieve and display current values
If lyr.Symbol.Size = 0 Then
txtSSP(0).Text = 1
Else
txtSSP(0).Text = lyr.Symbol.Size
End If
lblSSP(0).Caption = "cor do fundo:"
lblSSP(2).Caption = "limite:"
For i = 0 To 10
cboSSP(0).AddItem strFillStyle(i)
```

```
Next
picSSP(0).BackColor = lyr.Symbol.Color
picSSP(1).BackColor = lyr.Symbol.OutlineColor
cboSSP(0).Text = strFillStyle(lyr.Symbol.Style)
cboSSP(0).ListIndex = lyr.Symbol.Style
Select Case lyr.Symbol.Outline
Case True: chkSSP.Value = 1
Case False: chkSSP.Value = 0
End Select
End Select
End Sub
Private Sub LoadUniqueValues()
Dim i As Integer
Set vmr = lyr.Renderer
cboUV.Clear
cboUV.Text = vmr.Field
For i = 0 To tDesc.FieldCount - 1
cboUV.AddItem tDesc.FieldName(i)
If tDesc.FieldName(i) = vmr.Field Then
cboUV.ListIndex = i
frmLayerSymbol - 13
End If
Next
Select Case vmr.Symbol(0).Outline
Case True: chkUV.Value = 1
Case False: chkUV.Value = 0
End Select
chkUV.Visible = (lyr.shapeType = moShapeTypePolygon)
Call PopulateExistingUVlegend
End Sub
Private Sub LoadClassBreaks()
Dim i, j As Integer
Set cbr = lyr.Renderer
j = -1
cboCB(0).Clear
cboCB(0).Text = cbr.Field
For i = 0 To tDesc.FieldCount - 1
If tDesc.FieldType(i) = moDouble Or _
tDesc.FieldType(i) = moLong Then
j = j + 1
cboCB(0).AddItem tDesc.FieldName(i)
If tDesc.FieldName(i) = cbr.Field Then
cboCB(0).ListIndex = j
End If
End If
Next
cboCB(1).Text = cbr.BreakCount + 1
cboCB(1).ListIndex = cbr.BreakCount - 1
Select Case cbr.Symbol(0).Outline
Case True
chkCB.Value = 1
Case False
chkCB.Value = 0
End Select
picCBramp(0).BackColor = cbr.Symbol(0).Color
picCBramp(1).BackColor = cbr.Symbol(cbr.BreakCount).Color
```

```
chkCB.Visible = (lyr.shapeType = moShapeTypePolygon)
Call PopulateExistingCBlegend
End Sub
'Private Sub LoadStandardLabels()
'Dim i As Integer
'Dim strFN As String
'Set lr = lyr.Renderer
'For i = 0 To tDesc.FieldCount - 1
'strFN = tDesc.FieldName(i)
'cboSL(0).AddItem strFN
'If lr.Field = strFN Then
'cboSL(0).ListIndex = i
'End If
'If tDesc.FieldType(i) = moLong Or _
'tDesc.FieldType(i) = moDouble Then
'cboSL(3).AddItem strFN
'If lr.XOffsetField = strFN Then
'cboSL(3).ListIndex = i
'End If
'cboSL(4).AddItem strFN
'If lr.YOffsetField = strFN Then
'cboSL(4).ListIndex = i
'End If
'End If
'Next
'Select Case lr.Symbol(0).HorizontalAlignment
frmLayerSymbol - 14
'Case moAlignLeft
'cboSL(1).ListIndex = 0
'Case moAlignCenter
'cboSL(1).ListIndex = 1
'Case moAlignRight
'cboSL(1).ListIndex = 2
'End Select
'Select Case lr.Symbol(0).VerticalAlignment
'Case moAlignTop
'cboSL(2).ListIndex = 0
'Case moAlignCenter
'cboSL(2).ListIndex = 1
'Case moAlignBottom
'cboSL(2).ListIndex = 2
'End Select
'hsbSL.Value = lr.Symbol(0).Rotation
'lblSL(7).Caption = lr.Symbol(0).Rotation
'For i = 0 To 3
'chkSL(i).Value = 0
'Next
'If lr.DrawBackground Then
' chkSL(0).Value = 1
'End If
'If lr.AllowDuplicates Then
' chkSL(1).Value = 1
'End If
'If lr.SplinedText Then
' chkSL(2).Value = 1
'End If
```

```
'If lr.Flip Then
' chkSL(3).Value = 1
'End If
'txtSL.Text = lr.Symbol(0).Font.Name
'txtSL.ForeColor = lr.Symbol(0).Color
'cdlgLayerProp.FontName = lr.Symbol(0).Font.Name
'cdlgLayerProp.FontSize = lr.Symbol(0).Font.Size
'cdlgLayerProp.FontBold = lr.Symbol(0).Font.Bold
'cdlgLayerProp.FontItalic = lr.Symbol(0).Font.Italic
'cdlgLayerProp.FontStrikethru = lr.Symbol(0).Font.Strikethrough
'cdlgLayerProp.FontUnderline = lr.Symbol(0).Font.Underline
'cdlgLayerProp.Color = lr.Symbol(0).Color
'End Sub
Private Sub LoadNoOverlapLabels()
Dim i As Integer
Dim scaleHeightUnit As Double
Dim strFN As String
Set lp = lyr.Renderer
colorMask = lp.MaskColor
cboNOL.Clear
For i = 0 To tDesc.FieldCount - 1
strFN = tDesc.FieldName(i)
cboNOL.AddItem strFN
If lp.Field = strFN Then
cboNOL.ListIndex = i
End If
Next
cdlgLayerProp.FontName = lp.DefaultSymbol.Font.Name
cdlgLayerProp.FontSize = lp.DefaultSymbol.Font.Size
cdlgLayerProp.FontBold = lp.DefaultSymbol.Font.Bold
cdlgLayerProp.FontItalic = lp.DefaultSymbol.Font.Italic
cdlgLayerProp.FontStrikethru = lp.DefaultSymbol.Font.Strikethrough
frmLayerSymbol - 15
cdlgLayerProp.FontUnderline = lp.DefaultSymbol.Font.Underline
colorText = lp.DefaultSymbol.Color
txtNOL.Text = lp.DefaultSymbol.Font.Name
txtNOL.ForeColor = colorText
optNOL(0).Value = lp.PlaceOn
optNOL(1).Value = lp.PlaceAbove
optNOL(2).Value = lp.PlaceBelow
Select Case lp.DrawBackground
Case False: chkNOL(0).Value = 0
Case True: chkNOL(0).Value = 1
End Select
Select Case lp.AllowDuplicates
Case False: chkNOL(1).Value = 0
Case True: chkNOL(1).Value = 1
End Select
Select Case lp.MaskLabels
Case False: chkNOL(2).Value = 0
Case True
chkNOL(2).Value = 1
picNOL.BackColor = lp.MaskColor
End Select
scaleHeightUnit = lyr.Extent.Width / 10000
'scaleHeightUnit = frmMain.mapDisp.FullExtent.Width / 10000
```

```
hsbNOL.Value = 1000 - (lp.DefaultSymbol.Height / scaleHeightUnit)
'hsbNOL.Value = 700 - (lp.DefaultSymbol.Height / scaleHeightUnit)
End Sub
'Private Sub LoadZRenderer()
'Dim i, j As Integer
'Set zRend = lyr.Renderer
'j = -1
'cboZRclasses.Clear
'cboZRclasses.Text = zRend.BreakCount + 1
'cboZRType.ListIndex = zRend.ValueCalculation
'picZRramp(0).BackColor = zRend.Symbol(0).Color
'picZRramp(1).BackColor = zRend.Symbol(zRend.BreakCount).Color
'Call PopulateExistingZRlegend
'End Sub
'''
'THE SIX PROCEDURES THAT FOLLOW, THAT BEGIN WITH THE WORD "APPLY..."
'ARE THOSE THAT RUN WHEN THE APPLY OR OK BUTTONS ARE CLICKED.
'THE CURRENT VALUES OF THE OPTION CONTROLS ARE READ, WRITTEN
'INTO A NEW RENDERER OBJECT. THEN THAT RENDERER OBJECT IS
'USED TO DRAW THE ACTIVE LAYER.
' ApplySingleSymbol()
' ApplyUniqueValues()
' ApplyClassBreaks()
' ApplyStandardLabels()
' ApplyNoOverlapLabels()
' ApplyZRenderer()
''
Private Sub ApplySingleSymbol()
Dim sym As MapObjects2.Symbol
Set sym = lyr.Symbol
'lyr.Name = txtLayerName.Text
Set lyr.Renderer = Nothing
Select Case lyr.shapeType
frmLayerSymbol - 16
Case moShapeTypePoint
sym.Color = picSSP(0).BackColor
sym.Style = cboSSP(0).ListIndex
'If hsbSSP.Value = 0 Then
'sym.Rotation = 0
'Else
'Clockwise instead of the default counter-clockwise
'sym.Rotation = 360 - hsbSSP.Value
'End If
If IsNumeric(txtSSP(0).Text) Then
sym.Size = txtSSP(0).Text
Else
sym.Size = 5
End If
If sym.Style = moTrueTypeMarker Then
Dim fnt As New StdFont
fnt.Name = cboSSP(1).Text
Set sym.Font = fnt
a = Asc(cboSSP(2).Text)
sym.CharacterIndex = Asc(cboSSP(2).Text)
End If
Case moShapeTypeMultipoint
```

```
sym.Color = picSSP(0).BackColor
sym.Style = cboSSP(0).ListIndex
'If hsbSSP.Value = 0 Then
'sym.Rotation = 0
'Else
'Clockwise instead of the default counter-clockwise
'sym.Rotation = 360 - hsbSSP.Value
'End If
If IsNumeric(txtSSP(0).Text) Then
sym.Size = txtSSP(0).Text
Else
sym.Size = 5
End If
If sym.Style = moTrueTypeMarker Then
Dim fnt2 As New StdFont
fnt2.Name = cboSSP(1).Text
Set sym.Font = fnt2
a = Asc(cboSSP(2).Text)
sym.CharacterIndex = Asc(cboSSP(2).Text)
End If
Case moLine
sym.Color = picSSP(0).BackColor
sym.Style = cboSSP(0).ListIndex
If IsNumeric(txtSSP(0).Text) Then
sym.Size = txtSSP(0).Text
Else
sym.Size = 1
End If
Case moPolygon
sym.Color = picSSP(0).BackColor
sym.OutlineColor = picSSP(1).BackColor
sym.Style = cboSSP(0).ListIndex
If IsNumeric(txtSSP(0).Text) Then
sym.Size = txtSSP(0).Text
Else
sym.Size = 1
End If
Select Case chkSSP.Value
Case 1
lyr.Symbol.Outline = True
Case 0
lyr.Symbol.OutlineColor = lyr.Symbol.Color
lyr.Symbol.Outline = False
End Select
End Select
End Sub
frmLayerSymbol - 17
Private Sub ApplyUniqueValues()
Dim i As Integer
Dim symInt As Integer
If lyr.shapeType = moShapeTypeMultipoint Then
symInt = 0
Else
symInt = lyr.shapeType - 21
End If
vmr.SymbolType = symInt
```

```
If vmr.SymbolType = moFillSymbol Then
For i = 0 To vmr.ValueCount - 1
Select Case chkUV.Value
Case 0
vmr.Symbol(i).Outline = False
vmr.Symbol(i).OutlineColor = vmr.Symbol(i).Color
Case 1
vmr.Symbol(i).Outline = True
vmr.Symbol(i).OutlineColor = moBlack
End Select
Next
End If
Set lyr.Renderer = vmr
End Sub
Private Sub ApplyClassBreaks()
Dim i As Integer
Dim symInt As Integer
If lyr.shapeType = moShapeTypeMultipoint Then
symInt = 0
Else
symInt = lyr.shapeType - 21
End If
cbr.SymbolType = symInt
If cbr.SymbolType = moFillSymbol Then
For i = 0 To cbr.BreakCount
Select Case chkCB.Value
Case 0
cbr.Symbol(i).OutlineColor = cbr.Symbol(i).Color
cbr.Symbol(i).Outline = False
Case 1
cbr.Symbol(i).OutlineColor = moBlack
cbr.Symbol(i).Outline = True
End Select
Next
End If
Set lyr.Renderer = cbr
End Sub
'Private Sub ApplyStandardLabels()
'Dim fnt As New stdole.StdFont
'fnt.Name = cdlgLayerProp.FontName
'fnt.Size = cdlgLayerProp.FontSize
'fnt.Bold = cdlgLayerProp.FontBold
'fnt.Italic = cdlgLayerProp.FontItalic
'fnt.Strikethrough = cdlgLayerProp.FontStrikethru
'fnt.Underline = cdlgLayerProp.FontUnderline
'Set lr = New MapObjects2.LabelRenderer
'lr.Field = cboSL(0).Text
frmLayerSymbol - 18
'With lr.Symbol(0)
'Select Case cboSL(1).Text
'Case "Left"
'.HorizontalAlignment = moAlignLeft
'Case "Center"
'.HorizontalAlignment = moAlignCenter
'Case "Right"
'.HorizontalAlignment = moAlignRight
```

```
'End Select
'Select Case cboSL(2).Text
'Case "Top"
'.VerticalAlignment = moAlignTop
'Case "Center"
'.VerticalAlignment = moAlignCenter
'Case "Bottom"
'.VerticalAlignment = moAlignBottom
'End Select
'Set .Font = fnt
'.Color = cdlgLayerProp.Color
'Rotate clockwise instead of counter-clockwise
'If lblSL(7).Caption = 0 Then
'.Rotation = 0
'Else
'.Rotation = 360 - (7).Caption
'End If
'End With
'With lr
'.XOffsetField = cboSL(3).Text
'.YOffsetField = cboSL(4).Text
'.DrawBackground = (chkSL(0).Value = 1)
'.AllowDuplicates = (chkSL(1).Value = 1)
'.SplinedText = (chkSL(2).Value = 1)
'.Flip = (chkSL(3).Value = 1)
'End With
'Set lyr.Renderer = lr
'End Sub
Private Sub ApplyNoOverlapLabels()
Dim fnt As New stdole.StdFont
Set lp = New MapObjects2.LabelPlacer
lp.Field = cboNOL.Text
fnt.Name = cdlgLayerProp.FontName
fnt.Bold = cdlgLayerProp.FontBold
fnt.Italic = cdlgLayerProp.FontItalic
fnt.Strikethrough = cdlgLayerProp.FontStrikethru
fnt.Underline = cdlgLayerProp.FontUnderline
Dim scaleHeightUnit As Double
scaleHeightUnit = lyr.Extent.Width / 10000
With lp.DefaultSymbol
.Height = scaleHeightUnit * (1001 - hsbNOL.Value)
.Color = colorText
Set .Font = fnt
End With
lp.PlaceAbove = optNOL(1)
lp.PlaceBelow = optNOL(2)
lp.PlaceOn = optNOL(0)
Select Case chkNOL(0).Value
Case 0: lp.DrawBackground = False
Case 1: lp.DrawBackground = True
End Select
frmLayerSymbol - 19
Select Case chkNOL(1).Value
Case 0: lp.AllowDuplicates = False
Case 1: lp.AllowDuplicates = True
End Select
```

```
Select Case chkNOL(2).Value
Case 0: lp.MaskLabels = False
Case 1
lp.MaskLabels = True
lp.MaskColor = picNOL.BackColor
End Select
Set lyr.Renderer = lp
End Sub
'Private Sub ApplyZRenderer()
'Dim symInt As Integer
'If lyr.shapeType = moShapeTypeMultipoint Then
'symInt = 0
'Else
'symInt = lyr.shapeType - 21
'End If
'zRend.SymbolType = symInt
'Set lyr.Renderer = zRend
'End Sub
'''
'THE NEXT SIX PROCEDURES THAT START WITH THE WORD "POPULATE..."
'ARE THOSE THAT RUN WHEN ONE OF THE "RESET LEGEND" BUTTONS
'ARE PRESSED. THESE PROCEDURES CALCULATE AND LOAD A LEGEND
'PREVIEW ON THE LAYERSYMBOL FORM THAT THE USER CAN EXAMINE
'BEFORE APPLYING TO THE MAP.
' PopulateNewUVLegend, PopulateExistingUVLegend (unique values)
' PopulateNewCBLegend, PopulateExistingCBLegend (class breaks)
' PopulateNewZRLegend, PopulateExistingZRLegend (elevation Z values)
''
Private Sub PopulateNewUVlegend(rendField As String)
Dim strsUniqueValues As New MapObjects2.Strings
Dim fld As MapObjects2.Field
Dim i As Integer
Set fld = flds(rendField)
If fld Is Nothing Then
Exit Sub
End If
Screen.MousePointer = vbHourglass
recs.MoveFirst
Do While Not recs.EOF
strsUniqueValues.Add fld.Value
recs.MoveNext
Loop
Screen.MousePointer = vbDefault
If strsUniqueValues.Count > 100 Then
Dim yn As Integer
yn = MsgBox("Number of unique values is greater than 100. Would you
like to continue?", _
vbYesNo, "Unique values")
If yn = 7 Then
Exit Sub
End If
End If
frmLayerSymbol - 20
If picUV.Count > 1 Then
For i = (picUV.Count - 1) To 1 Step -1
Unload picUV(i)
```

```
Unload lblUV(i)
Next
End If
Set vmr = New MapObjects2.ValueMapRenderer
vmr.Field = rendField
vmr.ValueCount = strsUniqueValues.Count
For i = 0 To strsUniqueValues.Count - 1
vmr.Value(i) = strsUniqueValues(i)
Next
fraUVouter.Caption = UCase(lyr.Name) & " - " & cboUV.Text
picUV(0).Visible = True
lblUV(0).Visible = True
picUV(0).BackColor = vmr.Symbol(0).Color
lblUV(0).Caption = vmr.Value(0)
For i = 1 To vmr.ValueCount - 1
Load picUV(i)
With picUV(i)
.Left = picUV(0).Left
.Width = picUV(0).Width
.Height = picUV(0).Height
.Top = picUV(i - 1).Top + 180
.BackColor = vmr.Symbol(i).Color
.Visible = True
End With
Load lblUV(i)
With lblUV(i)
.Left = lblUV(0).Left
.Width = lblUV(0).Width
.Height = lblUV(0).Height
.Top = lblUV(i - 1).Top + 180
.Caption = vmr.Value(i)
.Visible = True
End With
Next
fraUVinner.Height = (vmr.ValueCount * 180) + 250
If strsUniqueValues.Count > 18 Then
vsbUV.Enabled = True
vsbUV.Min = 0
vsbUV.Max = vmr.ValueCount - 20
vsbUV.SmallChange = 1
vsbUV.LargeChange = 10
Else
vsbUV.Enabled = False
End If
End Sub
Private Sub PopulateExistingUVlegend()
If Not justOpened Then
Exit Sub
End If
Dim recs As MapObjects2.Recordset
Dim fld As MapObjects2.Field
Dim i As Integer
Set vmr = lyr.Renderer
If picUV.Count > 1 Then
For i = (picUV.Count - 1) To 1 Step -1
Unload picUV(i)
```

```
frmLayerSymbol - 21
Unload lblUV(i)
Next
End If
fraUVouter.Caption = UCase(lyr.Name) & " - " & lyr.Renderer.Field
picUV(0).Visible = True
lblUV(0).Visible = True
picUV(0).BackColor = vmr.Symbol(0).Color
lblUV(0).Caption = vmr.Value(0)
For i = 1 To vmr.ValueCount - 1
Load picUV(i)
With picUV(i)
.Left = picUV(0).Left
.Width = picUV(0).Width
.Height = picUV(0).Height
.Top = picUV(i - 1).Top + 180
.BackColor = vmr.Symbol(i).Color
.Visible = True
End With
Load lblUV(i)
With lblUV(i)
.Left = lblUV(0).Left
.Width = lblUV(0).Width
.Height = lblUV(0).Height
.Top = lblUV(i - 1).Top + 180
.Caption = vmr.Value(i)
.Visible = True
End With
Next
fraUVinner.Height = (vmr.ValueCount * 180) + 300
If vmr.ValueCount > 18 Then
vsbUV.Enabled = True
vsbUV.Min = 0
vsbUV.Max = vmr.ValueCount - 20
vsbUV.SmallChange = 1
vsbUV.LargeChange = 10
Else
vsbUV.Enabled = False
End If
justOpened = False
End Sub
Private Sub PopulateNewCBlegend(rendField As String)
Dim stats As MapObjects2.Statistics
Dim range As Double
Dim i, numClasses, numBreaks As Integer
If Trim(cboCB(0).Text) = vbNullString Then
cboCB(0).ListIndex = 0
End If
Set stats = recs.CalculateStatistics(cboCB(0).Text)
numClasses = cboCB(1).Text
numBreaks = numClasses - 1
If picCBlegend.Count > 1 Then
For i = (picCBlegend.Count - 1) To 1 Step -1
Unload picCBlegend(i)
Unload lblCBlegend(i)
Next
```

```
End If
Set cbr = New MapObjects2.ClassBreaksRenderer
cbr.Field = cboCB(0).Text
frmLayerSymbol - 22
cbr.BreakCount = numBreaks
range = stats.Max - stats.Min
For i = 0 To numBreaks - 1
cbr.Break(i) = stats.Min + ((range / numClasses) * (i + 1))
Next
cbr.RampColors picCBramp(0).BackColor, picCBramp(1).BackColor
fraCB.Caption = UCase(lyr.Name) & " - " & cboCB(0).Text
picCBlegend(0).Visible = True
lblCBlegend(0).Visible = True
picCBlegend(0).BackColor = cbr.Symbol(0).Color
lblCBlegend(0).Caption = "Menor que " & Format(cbr.Break(0), "#0.00")
For i = 1 To cbr.BreakCount
Load picCBlegend(i)
With picCBlegend(i)
.Left = picCBlegend(0).Left
.Width = picCBlegend(0).Width
.Height = picCBlegend(0).Height
.Top = picCBlegend(i - 1).Top + 180
.BackColor = cbr.Symbol(i).Color
.Visible = True
End With
Load lblCBlegend(i)
With lblCBlegend(i)
.Left = lblCBlegend(0).Left
.Width = lblCBlegend(0).Width
.Height = lblCBlegend(0).Height
.Top = lblCBlegend(i - 1).Top + 180
.Visible = True
Select Case i
Case cbr.BreakCount
.Caption = ">= " & Format(cbr.Break(cbr.BreakCount - 1), "#0.00")
Case Else
.Caption = Format(cbr.Break(i - 1), "#0.00") & " - " &
Format(cbr.Break(i), "#0.00")
End Select
End With
Next
End Sub
Public Sub PopulateExistingCBlegend()
If Not justOpened Then
Exit Sub
End If
Dim stats As MapObjects2.Statistics
Dim i, numClasses, numBreaks As Integer
Set stats = recs.CalculateStatistics(lyr.Renderer.Field)
numBreaks = lyr.Renderer.BreakCount
If picCBlegend.Count > 1 Then
For i = (picCBlegend.Count - 1) To 1 Step -1
Unload picCBlegend(i)
Unload lblCBlegend(i)
Next
End If
```

```
Set cbr = lyr.Renderer
cbr.Field = lyr.Renderer.Field
fraCB.Caption = UCase(lyr.Name) & " - " & cboCB(0).Text
picCBlegend(0).Visible = True
lblCBlegend(0).Visible = True
picCBlegend(0).BackColor = cbr.Symbol(0).Color
lblCBlegend(0).Caption = "Less than " & Format(cbr.Break(0), "#0.00")
frmLayerSymbol - 23
For i = 1 To cbr.BreakCount
Load picCBlegend(i)
With picCBlegend(i)
.Left = picCBlegend(0).Left
.Width = picCBlegend(0).Width
.Height = picCBlegend(0).Height
.Top = picCBlegend(i - 1).Top + 180
.BackColor = cbr.Symbol(i).Color
.Visible = True
End With
Load lblCBlegend(i)
With lblCBlegend(i)
.Left = lblCBlegend(0).Left
.Width = lblCBlegend(0).Width
.Height = lblCBlegend(0).Height
.Top = lblCBlegend(i - 1).Top + 180
.Visible = True
Select Case i
Case cbr.BreakCount
.Caption = ">= " & Format(stats.Max, "#0.00")
Case Else
.Caption = Format(cbr.Break(i - 1), "#0.00") & " - " &
Format(cbr.Break(i), "#0.00")
End Select
End With
Next
justOpened = False
End Sub
'Private Sub PopulateNewZRLegend()
'Dim n As Integer
'Dim range As Double
'Dim MinZ As Double, MaxZ As Double
'MinZ = lyr.Extent.Floor
'MaxZ = lyr.Extent.Ceiling
'MaxZ = 1400
'range = MaxZ - MinZ
' clear existing legend
'If picZRlegend.Count > 1 Then
'For n = (picZRlegend.Count - 1) To 1 Step -1
'Unload picZRlegend(n)
'Unload lblZRlegend(n)
'Next
'End If
' set new breakcount
'Set zRend = New MapObjects2.ZRenderer
'zRend.BreakCount = cboZRclasses.List(cboZRclasses.ListIndex) - 1
' Set the breaks using simple equal interval ranges...
'For n = 1 To zRend.BreakCount
```

```
'zRend.Break(n - 1) = MinZ + ((range / zRend.BreakCount + 1) * (n))
'Next n
'Build symbol array by ramping start and end colors
'zRend.RampColors picZRramp(0).BackColor, picZRramp(1).BackColor
' handle the base items in the object arrays
'picZRlegend(0).BackColor = picZRramp(0).BackColor
'lblZRlegend(0).Caption = "Less Than " & Format(zRend.Break(0),
"#0.00")
'picZRlegend(0).Visible = True
'lblZRlegend(0).Visible = True
'For n = 1 To zRend.BreakCount
' Set up the color boxes
frmLayerSymbol - 24
'Load picZRlegend(n)
'With picZRlegend(n)
'.Top = picZRlegend(n - 1).Top + 180
'.BackColor = zRend.Symbol(n).Color
'.Visible = True
'End With
' Set up the labels
'Load lblZRlegend(n)
'With lblZRlegend(n)
'.Top = lblZRlegend(n - 1).Top + 180
'.Visible = True
'Select Case n
'Case zRend.BreakCount
'.Caption = "Greater Than " & Format(zRend.Break(n - 1), "#0.00")
'Case Else
'.Caption = Format(zRend.Break(n - 1), "#0.00") & " - " &
Format(zRend.Break(n), "#0.00")
'End Select
'End With
'Next n
'End Sub
'Private Sub PopulateExistingZRlegend()
'If Not justOpened Then
'Exit Sub
'End If
'Dim stats As MapObjects2.Statistics
'Dim i, numClasses, numBreaks As Integer
'Set zRend = lyr.Renderer
'If picZRlegend.Count > 1 Then
'For i = (picZRlegend.Count - 1) To 1 Step -1
'Unload picZRlegend(i)
'Unload lblZRlegend(i)
'Next
'End If
'picZRlegend(0).Visible = True
'lblZRlegend(0).Visible = True
'picZRlegend(0).BackColor = zRend.Symbol(0).Color
'lblZRlegend(0).Caption = "Less than " & Format(zRend.Break(0),
"#0.00")
'For i = 1 To zRend.BreakCount
'Load picZRlegend(i)
'With picZRlegend(i)
'.Left = picZRlegend(0).Left
```

```
'.Width = picZRlegend(0).Width
'.Height = picZRlegend(0).Height
'.Top = picZRlegend(i - 1).Top + 180
'.BackColor = zRend.Symbol(i).Color
'.Visible = True
'End With
'Load lblZRlegend(i)
'With lblZRlegend(i)
'.Left = lblZRlegend(0).Left
'.Width = lblZRlegend(0).Width
'.Height = lblZRlegend(0).Height
'.Top = lblZRlegend(i - 1).Top + 180
'.Visible = True
'Select Case i
'Case zRend.BreakCount
'.Caption = ">= " & Format(stats.Max, "#0.00")
'Case Else
'.Caption = Format(zRend.Break(i - 1), "#0.00") & " - " &
Format(zRend.Break(i), "#0.00")
'End Select
'End With
frmLayerSymbol - 25
'Next
'justOpened = False
'End Sub
selectedform - 1
Option Explicit
Public g_activelayer As Object
Private Sub cboIDs_Click()
Dim i As Integer
Dim curfield As MapObjects2.Field
If SelectedFormUp Then
Exit Sub
End If
gSelection.MoveFirst
fieldlist.Clear
For i = 1 To cboIDs.ListIndex
gSelection.MoveNext
Next i
For Each curfield In gSelection.Fields
If curfield.Type = moString Then
fieldlist.AddItem curfield.Name + " = " + curfield.Value
Else
fieldlist.AddItem curfield.Name + " = " + curfield.ValueAsString
End If
Next curfield
Form1.Map1.FlashShape gSelection("shape").Value, 3
End Sub
Private Sub Form_Unload(Cancel As Integer)
Form1.Map1.Refresh
Unload selectedform
End Sub
modStringHandler - 1
Option Explicit
Function GetFirstToken(StrIn As String, Delim As String) As String
' Gets the portion of String "S", up to the
```

```
' first occurrance of delimiter "D"
' Returns String token "T"
Dim Split As Long
Dim Tok As String
StrIn = Trim$(StrIn)
Split = InStr(1, StrIn, Delim)
If (Split <= 0) Then
' No delimiter in the string. Return the whole thing.
Tok = StrIn
Else
' Get everything up to the first delimiter.
Tok = (Trim$(Left$(StrIn, Split - 1)))
End If
GetFirstToken = Tok
End Function
modUtility - 1
Option Explicit
Public gSelection As MapObjects2.Recordset
Public ActiveLayer As MapObjects2.MapLayer
Public SelectedFormUp As Boolean
Public drawLayer As MapObjects2.MapLayer
'ConnectErrorMsg - Defines an appropriate error message for
' the Data Connection object
Public Function ConnectErrorMsg(errNum As Integer) As String
Select Case errNum
Case moNoError: ConnectErrorMsg = "No Error"
Case moUnknownError: ConnectErrorMsg = "Unknown Error"
Case moAccessDenied: ConnectErrorMsg = "Access Denied"
Case moInvalidUser: ConnectErrorMsg = "Invalid User"
Case moNetworkTimeout: ConnectErrorMsg = "Network Timeout"
Case moInvalidDatabase: ConnectErrorMsg = "Invalid Database"
Case moTasksExceeded: ConnectErrorMsg = "Tasks Exceeded"
Case moFileNotFound: ConnectErrorMsg = "File Not Found"
Case moInvalidDirectory: ConnectErrorMsg = "Invalid Directory"
Case moHostUnknown: ConnectErrorMsg = "Unknown Host"
Case Else: ConnectErrorMsg = "Unrecognized Error Code"
End Select

End Function
```

2. CD com SIGestão.exe e projecto Visual Basic