Universidade do Porto
Faculdade de Engenharia
FEUP

Henrique Miguel Gouveia da Silva

UTILIZAÇÃO DE SISTEMAS MÓVEIS PARA ACTUALIZAÇÃO DE DADOS GEOREFERENCIADOS

Porto, 2001

Faculdade de Engenharia da Universidade do Porto

Utilização de Sistemas Móveis para Actualização de Dados Georreferenciados

Henrique Miguel Gouveia da Silva

Licenciado em Informática de Gestão
pelo Instituto Superior da Maia

Dissertação submetida para satisfação parcial dos
requisitos do grau de Mestre em
Gestão de Informação

Dissertação realizada sob a supervisão do
Professor Auxiliar João António Correia Lopes,
do Departamento de Engenharia Electrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto
e do Professor Associado Alexandre Valente Sousa
do Instituto Superior da Maia

Porto, Dezembro de 2001

Abstract

In recent years the World Wide Web has become a popular vehicle for information distribution and geographic information systems (GIS) are rapidly evolving and adapting to this new environment.

The main constraint for building true interoperable distributed geographic information systems is the lack of any standard exchange mechanism between those GISes. Recent efforts by the Open GIS Consortium have resulted in several specifications to solve these problems. The Geographic Markup Language (GML) is one example of a proposed standard.

GML is an XML (eXtensible Markup Language) encoding for the transport over the Internet and the storage of geographic information, including both geometry and properties of geographic features. The parsing techniques have performance issues since the size of GML documents may be potentially huge.

Mobile Computing devices (PDA) have evolved very fast over the last years. They are no longer seen as task suppporting devices, such as electronic personal information managers, but as powerful new tools to manage all kinds of data.

This evolution forced a shift on system development paradigms, requiring new ways to reconcile replicas stored on different devices, with a common database. This syncronization poses several challenges.

The main focus of our study is to evaluate how to use Mobile Computing devices on Geographic Information Systems, mainly for the purpose of updating geo referenced data.

This work focuses on the development of an application for mobile devices, which allows the updating of georreferenced data, a Palm device was chosen to demonstrate this. The GML standard is evaluated and used thoroughly on this.

This standard was found useful and should be used to transform data among different applications. We concluded that mobile devices can be used (as long as code is developed taking into account the specific constraints) to develop applications existing up to now only on larger systems.

Resumo

Actualmente a Internet é um veículo popular para distribuição de informação e os sistemas de informação geográficos (SIG) estão-se rapidamente a desenvolver e adaptar a este novo ambiente.

A grande restrição à criação de sistemas verdadeiramente abertos que permitam a distribuição de dados, é a falta de um mecanismo padrão de intercâmbio entre os diversos SIGs. Esforços recentes levados a cabo pelo *Open GIS Consortium* (OGC) resultaram num conjunto de especificações para resolver este problema, sendo o GML (Geography Markup Language – Linguagem de Marcação Geográfica) um desses padrões.

GML é uma codificação XML (eXtensible Markup Language — Linguagem de Marcação eXtensível) para o transporte pela Internet e para o armazenamento de informação geográfica, incluindo tanto a geometria como as propriedades das características geográficas. As técnicas de análise dos documentos GML têm, porém, que levar em conta considerações de desempenho visto que os documentos GML podem ser potencialmente enormes.

Os dispositivos de Computação Móvel (vulgo PDA) têm sofrido uma grande evolução durante os últimos tempos. Já não são apenas vistos como instrumentos de apoio à gestão de tarefas (ou calendários electrónicos) mas como ferramentas potentes de manutenção de dados.

Esta evolução provocou alterações nos paradigmas de desenvolvimento de sistemas, obrigando à avaliação de novas formas de reconciliar réplicas de dados modificados em diferentes dispositivos, com uma base comum. Vários desafios se colocam na hora de efectuar esta sincronização.

O principal objectivo deste estudo é avaliar a possibilidade de utilização de dispositivos de Computação Móvel nos Sistemas de Informação Geográficos, nomeadamente para efeito de actualização de dados georreferenciados.

Neste trabalho desenvolve-se uma aplicação para dispositivos móveis que permite efectuar a manutenção de dados georreferenciados num destes dispositivos (neste caso num Palm), sendo analisada e utilizada a norma GML para a transformação destes dados entre formatos distintos.

Verificou-se que a norma apresentada é útil e pode ser utilizada na sua plenitude para efectuar transformação entre formatos. Concluiu-se que os dispositivos móveis podem ser utilizados (mediante código desenvolvido de raíz e que leve em linha de conta as restrições específicas a estes dispositivos) para aplicações até agora desenvolvidas somente para grandes sistemas.

Agradecimentos

Importa referir aqui todos os que auxiliaram, de alguma forma, na conclusão deste trabalho.

À minha esposa, pelo apoio dado durante as noites passadas em branco.

Aos meus pais e sogros, por sempre se terem mostrado disponíveis para me substituir nas minhas ausências.

Aos meus orientadores, pelo apoio prestado e tempo dispensado. Sem o apoio de ambos este trabalho não teria sido terminado.

Ao Duarte Gomes, pela amizade e pelo auxílio na realização dos trabalhos da componente lectiva do Mestrado.

Ao Domingos Cruz e Carlos Carvalho, pelas ideias apresentadas e por sempre terem auxiliado, principalmente nas alturas de maior desânimo.

Ao Carlos Moreira, Sá Couto, Carla Sousa e demais membros do CIAF, pela disponibilidade e compreensão.

A todos os meus colegas do ISMAI, e à sua direcção, pelo apoio prestado sempre que necessário.

Por fim, a todos quantos me apoiaram durante todo o tempo de duração deste trabalho, sempre com palavras de incentivo.

Dedico este trabalho ao meu futuro filho e ao meu irmão, Paulo.

Índice

Lista de Figuras

Lista de Tabelas

Siglas e Acrónimos

API Application Programming Interface

CGIS Canada Geographical Information System

DTD Document Type Definition

FGDC Federal Geographic Data Committee

Flex Fast Lexical Analyser Generator

GIS Geographic Information System

GML Geographic Markup Language

GPS Global Positioning System

GUI Graphical User Interface

HTML Hyper Text Markup Language

IDE Integrated Development Environment

MNCRS Mobile Network Computing Reference Specification (Data Synchronization Working Group)

NCGIA National Centre for Geographical Information and Analysis

OGC Open Gis Consortium

PDA Personal Digital Assistant

PDM Plano Director Municipal

RDF Resource Description Framework

SIG Sistemas de Informação Geográfica

SRS Spatial Reference System

XML eXtensible Markup Language

XSD XML Schema Definition

Capítulo 1

Introdução

No mundo dos Sistemas de Informação Geográficos (SIG) grandes mudanças têm vindo a ser operadas. O que até há bem pouco tempo eram aplicações para um mundo restrito de profissionais, tem-se vindo a disseminar e a entrar em organizações sem especial apetência para a geografia. Cada vez mais os dados só fazem sentido quando relacionados com aspectos espaciais e temporais.

O aparecimento de aplicações com funcionalidades básicas de SIG, algumas das quais no formato código aberto (*open source*), como por exemplo OpenMap [BBN Technologies 2001], GeoTools [Center for Computational Geography 2001] e *CGAL* [cgal.org 2000], ou de utilização livre (*freeware*) como o TNT Lite [MicroImages, Inc. 1997], e a disponibilidade na Internet de *applets* e *plugins* para visualização de mapas, levou a que o mundo dos SIG tenha evoluído bastante nos últimos anos.

Serviços de localização são disponibilizados diariamente, tendo a maior parte deles obtido um sucesso considerável. Porém, tentar efectuar cruzamento entre os milhares de dados existentes em formatos distintos torna-se uma tarefa desmotivante. A existência de dezenas de formatos diferentes, apenas apoiados pela existência de alguns formatos aceites pela comunidade[1], só que vocacionados para a apresentação de imagens, não disponibilizam todas as funcionalidades suportadas por um SIG.

1.1 Motivação

Estão neste momento a ser levadas a cabo iniciativas para tentar encontrar uma base comum, aceite pelos principais fabricantes de SIG, que permita a fácil transformação entre formatos, não perdendo porém as vantagens da utilização dos SIG.

Também a inclusão de meta-dados, por forma a obter informação válida e correcta relativamente a fenómenos georreferenciados[2] permitirá representar informação até agora não

[1] Tal como o formato DXF que não sendo uma norma é um padrão de facto.

[2] E a possível inclusão de uma coordenada temporal, como previsto para a próxima versão do *Abstract*

possível de idealizar.

Para obter dados passíveis de inserir num SIG, torna-se necessário realizar diversas visitas ao terreno, tirar medidas, analizar altitudes, etc. Este trabalho tem vindo a ser realizado através de anotações efectuadas sobre mapas, que são posteriormente inseridos na base de dados. Seria mais fácil se esses dados fossem adquiridos directamente (ou quase) para a base de dados geográfica, de forma a evitar novas visitas ao terreno.

Graças às recentes evoluções tecnológicas, nomeadamente graças aos avanços nas tecnologias de computação móvel, essa tarefa está cada vez mais perto de ser possível.

Os sistemas móveis de computação têm-se afirmado cada vez mais como grandes auxiliares para o trabalho informático. Se até algum tempo atrás estes sistemas eram vistos apenas nas mãos de um grupo restrito de pessoas, em que a sua função implicava a utilização vulgar de meios informáticos, hoje em dia já se encontram disseminados por utilizadores de perfis muito variados.

Até há bem pouco tempo, estes "brinquedos" eram vistos apenas como dispositivos capazes de melhorar a produtividade individual. Eram apenas uma forma conveniente de armazenar dados pessoais. Limitados em memória e de difícil programação, estes dispositivos raramente possuíam uma grande capacidade de armazenamento e processamento. Para além destas restrições, apresentam-se com um ecrã de tamanho reduzido, o que obriga a desenhar as aplicações de uma forma simples e compacta.

Em 1996 a *Palm, Inc.*, que pertencia à 3Com[3], lançou no mercado o primeiro produto a integrar a família PalmPilot. Este dispositivo, de acordo com Puma Technologies [1999], suportava três grandes vantagens relativamente aos seus antecessores:

1. extremamente fácil de utilizar;
2. conectividade integrada, o que posibilitava a sincronização de dados entre o dispositivo e o PC;
3. disponibilidade de excelentes ferramentas para criação de aplicações PalmPilot, incluindo componentes para estender a conectividade, acrescentando possibilidades de sincronização a diferentes aplicações.

Porém, nem tudo são rosas nos sistemas móveis. Problemas como a sincronização e reconciliação de dados, aliados à existência de múltiplas cópias de uma base comum, levam à necessidade de estudar e implementar novos algoritmos sobre esses dados.

Também a existência de diversas plataformas, que impedem que, na maior parte das vezes, as aplicações possam comunicar de uma forma transparente, tem levado a que os programadores se vejam frequentemente com código semelhante, porém incompatível.

Feature Model — Modelo Abstracto de Características — que implicará a redefinição das tecnologias inerentes como seja o GML.

[3]Agora é uma empresa independente.

Do mesmo modo, a existência de formatos nativos, proprietários e incompatíveis, relativamente aos dados de cada aplicação impede, na maior parte das vezes, que o utilizador tenha a possibilidade de seleccionar a aplicação de entre todas as existentes, pois incorreria na possível perda dos dados existentes num outro sistema.

Por definição, os utilizadores móveis não estão sempre ligados a uma rede e aos dados armazenados no sistema anfitrião. Os utilizadores obtêm os dados do sistema anfitrião via uma ligação local ou remota e armazenam uma cópia (réplica) no dispositivo móvel, onde efectuam modificações sobre esses dados. Periodicamente estes utilizadores ligam-se ao sistema anfitrião e enviam as alterações efectuadas ao(s) repositório(s) principal(is). Nessa altura recebem também as modificações efectuadas a esses dados durante o tempo em que não estavam ligados. Ocasionalmente, é necessário resolver conflitos entre as diferentes modificações efectuadas. Esta operação de reconciliação (em que as modificações são transmitidas e os conflitos são resolvidos) é conhecida como sincronização de dados. Assim, sincronizar é o processo de transformar dois conjuntos de dados, que deveriam ser idênticos, em dois conjuntos idênticos. Para um dispositivo móvel esta sincronização aplica-se aos dados que o dispositivo tem armazenados localmente.

Esta dissertação pretende demonstrar, através de um caso prático, que já é possível efectuar a actualização e manipulação no local de dados georreferenciados através da utilização de um dos dispositivos de computação móvel existentes.

1.2 Tema

Como se depreende da secção anterior, este trabalho pretende avaliar as potencialidades dos sistemas de computação móvel para efectuar a manutenção, em tempo real, dos dados armazenados num SIG.

Pretende-se igualmente dar a conhecer uma das iniciativas[4] para criar um formato que possibilite a transformação entre formatos geográficos e verificar se este formato poderá vir a ter o sucesso que se pretende.

1.3 Objectivos do Trabalho

Este trabalho, para ser bem sucedido, deverá cumprir um mínimo de objectivos. Estes objectivos são:

- avaliar a norma GML proposta para transformação entre formatos distintos na área dos SIG;
- identificar os problemas inerentes aos sistemas de computação móvel;

[4] A norma GML, apresentada pelo *Open GIS Consortium*.

- estudar as possibilidade de ligação entre os sistemas de computação móvel e os Sistemas de Informação Geográfica;
- criar uma aplicação que permita efectuar a manutenção de dados georreferenciados num sistema de computação móvel;
- testar a aplicação resultante em sistemas reais;
- avaliar os resultados obtidos.

1.4 Ambiente de Programação Literária

Para realizar o trabalho proposto é necessário desenvolver software que envolve componentes para serem executados numa plataforma servidora e componentes para serem executados numa plataforma móvel. A criação desses componentes na prática envolve tecnologias e ambientes de desenvolvimento distintos e não integrados, pelo que se decidiu, para melhorar a compreensão e facilitar a manutenção do sistema, integrar os ficheiros criados para os vários sistemas com a escrita desta dissertação utilizando um sistema de programação literária [Knuth 1992].

Na programação literária combina-se código e documentação no *mesmo* ficheiro, as ferramentas de programação literária encarregam-se de extrair o código e criar a documentação. Um programa é estruturado como uma rede de fragmentos interligados. Fragmentos de documentação são escritos numa linguagem de formatação (no caso presente LaTeX) que suporte a inclusão de gráficos, tabelas e fórmulas matemáticas. Fragmentos de código são escritos na linguagem de programação pretendida, qualquer que ela seja. É comum serem colocados no mesmo ficheiro fragmentos escritos em linguagens de programação diferentes, por exemplo o código de um applet cliente em Java, o código do servidor em C, a Makefile para compilar o código, e os *scripts* de instalação para interpretadores de comandos UNIX e Windows.

O conceito de programação literária leva a que o código seja escrito através de refinamentos sucessivos que representam a sequência do pensamento de quem está a criar o código (não necessariamente *top-down* ou *bottom-up*) e não obedecem necessariamente à estrutura do código (i.e. não poderiam ser editados num editor dirigido pela sintaxe). Não precisamos de obedecer à estrutura porque a ferramenta de extracção do código vai rearranjar este do formato apropriado para ser lido por uma pessoa para o formato apropriado para ser processado por uma máquina (compilador ou interpretador).

A documentação é escrita ao lado do código, no *mesmo* ficheiro. Isto parece ser a chave para se possuir a documentação actualizada e a manutenção do código facilitada. Obviamente que é necessário algum esforço para actualizar a descrição caso o código seja alterado, mas é necessária menos disciplina do que se a documentação estivesse num ficheiro à parte.

Por forma a utilizar este sistema, recorreu-se à ferramenta dotNoweb [Sousa 2001]. A presente Dissertação é um documento dotNoweb.

1.5 Estrutura da Dissertação

Esta tese encontra-se dividida em capítulos, cada um destes em secções e, quando necessário, em sub-secções. Os capítulos que se podem encontrar nas páginas seguintes são:

Meta-dados em SIG que introduz os conceitos fundamentais relacionados com Sistemas de Informação Geográficos, a organização que elabora normas no mundo dos SIG e a norma apresentada por este organismo para transformação de dados entre formatos;

Uma Aplicação para Sistemas Móveis em que se apresentam os requisitos fundamentais da aplicação a ser desenvolvida, as tecnologias necessárias e o processo que irá ser seguido para o desenvolvimento da aplicação;

Obtenção de Dados GML onde se mostra como foi possível gerar, com base nos documentos GML, as bases de dados para visualização e actualização na plataforma móvel dos dados georreferenciados;

Manipulação de Dados na Plataforma Móvel que apresenta a aplicação desenvolvida, para Palm, no âmbito da dissertação. Esta aplicação permite efectuar a manutenção de dados georreferenciados sobre um mapa;

Sincronização dos Dados que mostra como, após as alterações efectuadas no dispositivo móvel, se pode novamente obter um documento GML para utilizar em qualquer Sistema de Informação Geográfico;

Avaliação dos Resultados onde são apresentados os resultados alcançados e quais os testes que foram efectuados para garantir a validade do sistema apresentado;

Conclusões onde se revêem os objectivos propostos e se propõem continuações ao presente trabalho.

Capítulo 2

Meta-dados em SIG

Neste Capítulo são apresentados os principais tópicos da presente dissertação. Explicam-se os conceitos fundamentais dos Sistemas de Informação Geográfica. Apresenta-se uma das principais iniciativas que pretende a integração de dados georreferenciados entre diferentes aplicações, os novos ambientes de registo de Meta-Dados e sua aplicação aos dados georreferenciados.

2.1 Introdução

Os Sistemas de Informação Geográfica revelaram-se úteis no passado como auxiliares das outras ciências. Porém, desde há algum tempo que se revelaram imprescindíveis na percepção dos fenómenos. Tudo ocorre no tempo e no espaço. Somente a cartografia poderá dar resposta a novas necessidades de georreferenciar fenómenos.

Dado que a evolução destes sistemas foi, durante bastante tempo, efectuada *ad-hoc*, sem que houvesse um controle sobre os métodos utilizados, isso provocou a que, por diversas vezes, o investimento não fosse rentabilizado com obtenção de informação relevante.

Surgiram nos Estados Unidos as primeiras iniciativas para regular e normalizar a obtenção de dados georreferenciados. Apesar de estas iniciativas se terem revelado infrutíferas (principalmente porque cada Departamento utilizava a **sua** norma), revelaram a necessidade da existência de um organismo regulador.

Este organismo, composto pelos principais organismos de referência no mundo dos SIG, conseguiu há bem pouco tempo disponibilizar o seu modelo de dados (o Modelo Abstracto de Características [Open GIS Consortium 1998a]). Esse modelo começa agora a ser integrado nas aplicações que trabalham com estes dados.

Era necessário avançar mais. Somente a existência deste modelo não conseguia alcançar a totalidade das aplicações propostas pela organização.

De modo a evitar que fosse necessário desenvolver conversores *de* e *para* todos os pares de formatos existentes, normalizou-se um formato que permitisse a transformação dos da-

dos existentes nos diferentes sistemas para esse formato e vice-versa. Assim reduziu-se um problema que era de $O(n^2)$ a um problema de $O(n)$.

Essa norma, baseada em XML, é GML (Linguagem de Marcação Geográfica). Com o aparecimento desta norma, novas possibilidades foram disponibilizadas a toda uma comunidade que necessitava de se ligar. Com esta norma é possível que aplicações distintas partilhem os seus dados e espera-se que estes possam brevemente ser inclusivé visualizados em navegadores Internet.

Embora com atrasos, esta tecnologia está praticamente concluída. Espera-se apenas que comecem a aparecer as primeiras aplicações a aproveitar as novas funcionalidades disponibilizadas.

A norma, mais do que fornecer uma forma de comunicação de dados de diferentes formatos, permite também a criação de Esquemas (definição das regras semânticas de um documento XML), que podem ser partilhados, dependendo da aplicação desejada. É possível, assim, que os mesmos dados tenham significados e formatos distintos, dependendo da aplicação. Torna-se assim mais fácil efectuar a análise dos dados.

GML abre também a possibilidade de incorporar funcionalidades de manutenção e manipulação de dados geográficos em sistemas vulgarmente mais limitados (em termos de recursos), como sejam os Assistentes Pessoais Digitais — como por exemplo o PalmPilot).

Até agora estes sistemas têm sido apenas utilizados para permitir a visualização de mapas[1]. Mas podemos ir mais além. Devido à sua mobilidade, estes sistemas poderão (e deverão) ser utilizados para, no próprio local, efectuar a manutenção dos dados, com referência ao local geográfico que se referem.

2.2 Sistemas de Informação Geográfica

> "As Ciências de Informação Geográfica têm-se vindo a evidenciar como um domínio científico com autonomia. Já não são vistas apenas como um instrumento, como uma junção ocasional de conhecimentos de outras áreas científicas."
>
> [Matos 2001]

Dada a percepção de a generalidade dos fenómenos ser georreferenciável, cada vez mais há uma preocupação em fazer esta ciência evoluir autonomamente, embora mantendo o apoio às outras ciências. Visto que a humanidade está a evoluir segundo o conceito da "aldeia global", onde todos têm lugar, então verifica-se a necessidade de uma ciência estável, concisa e coerente, que permita estudar os fenómenos de acordo com as características espaço-temporais onde os mesmos ocorrem.

[1] Com bastante sucesso, principalmente desde que se tornou possível a integração com sistemas de geo-posicionamento.

A conceptualização do espaço em que nos inserimos é uma ciência tão velha quanto a própria humanidade.

Matos [2001] refere que:

> "(...) o mais antigo vestígio de um mapa data de 3800 A. C., uma placa de argila mesopotâmica representando montanhas, cursos de água e outros objectos passíveis de representação cartográfica, mas a ideia será seguramente mais antiga."

Desde essa data até aos dias de hoje, a ciência cartográfica tem evoluído, pode-se dizer, muito lentamente. Ao contrário de outros sistemas de informação que aproveitaram desde os primórdios a existência do computador como auxílio, no caso da ciência cartográfica este instrumento foi, até há bem pouco tempo, olhado com desconfiança pelos seus elementos.

Embora o primeiro sistema percursor dos SIG (o CGIS) tenha sido iniciado na década de 60, estes sistemas tinham, até finais da década de 80 (mais concretamente 1988, com a criação do NCGIA), apenas uma função de armazenamento de dados georreferenciados, não sendo efectuadas operações de análise e cruzamento de dados, como hoje em dia se começa a verificar.

Matos [2001] identifica 5 fases no desenvolvimento dos Sistemas de Informação Geográficos (Tabela 2.1):

Tabela 2.1: Fases no Desenvolvimento dos Sistemas de Informação Geográficos

Pioneiros	1950-1970
Consolidação	1970-1980
Desenvolvimento/Divulgação	1980-1990
Reconversão/Aquisição de Dados	1990-1995
Vulgarização da Aplicação/Ciência	1995-2000

1. **Pioneirismo**, que correspondeu ao aparecimento das possibilidades tecnológicas e ao despertar para os problemas associados à modelação geográfica;
2. **Consolidação**, em que decorreu uma tomada de consciência das características da modelação geográfica comuns a inúmeros domínios de aplicação e à existência de pontos de contacto;
3. **Desenvolvimento e Divulgação**, onde predominou o trabalho de promoção e venda de tecnologia[2];
4. **Reconversão e Aquisição de Dados**, principalmente realizados por grandes organismos públicos e universidades; e

[2]Por vezes esse trabalho foi lúcido e com a correcta intuição do futuro, outras vezes foi oportunista e gerador de falsas expectativas.

5. **Vulgarização e Constituição de uma ciência com um corpo de conhecimento próprio**, com a divulgação da tecnologia para empresas de menor dimensão, sustentada pela redução e melhoramento do *hardware*, e para o cidadão comum, com a generalização dos serviços baseados na Internet e nos sistemas de geo-posicionamento (GPS).

O trabalho descrito nesta dissertação pode ser visto na sequência dos anteriores trabalhos, num prisma de complementaridade, relativamente às novas possibilidades abertas pela inclusão de meta-dados nos sistemas geo-espaciais.

2.2.1 Dados Georreferenciados

Mapas são abstracções do espaço. Significa isto que não é pretendida uma percepção dos dados no seu todo, mas apenas da sua representação num sistema restrito e identificado. Ao obter estes dados, necessitamos de tomar decisões relativamente à forma de os representar graficamente. Agrupamos os dados, decidimos o sistema de projecção a utilizar, etc. Regra geral, os dados são recolhidos e numa fase posterior são incorporados numa base de dados cartográfica para uso no SIG. Cada vez mais, porém, os dados podem ser inseridos directamente no SIG, sendo criada uma base de dados geográfica directamente das observações efectuadas. Também, com o advento do GPS , o registo de dados é efectuado sobre "mapas inexistentes", criados no momento.

É necessário, assim, seleccionar um método para efectuar a modelação necessária e integrar os dados observados num SIG, para posterior visualização (ou edição).

Por forma a efectuar esta modelação, vários sistemas foram desenvolvidos. Diversos autores, entre os quais Demers [2000] e Matos [2001], consideram a existência de dois modelos fundamentais de representação de informação georreferenciada.

O primeiro modelo **quantifica** (divide) o espaço num conjunto de pacotes, cada um representando uma quantidade limitada da superfície terrestre. Este modelo denomina-se geralmente por ***raster*** (ou **Matricial**). O modelo matricial permite definir estes pacotes sobre variadas formas geométricas (quadrados, hexágonos, triângulos), desde que as mesmas possam ser ligadas (tenham uma aresta comum). Os dados obtidos por satélite, por exemplo, são rasterizados, sendo armazenadas as imagens neste formato.

O segundo modelo permite indicar explicitamente localizações espaciais, assumindo um espaço contínuo e não quantificado como grelhas mais pequenas. Esse método é designado por ***Vectorial***. Neste caso, são identificados pontos, linhas e polígonos que permitem representar o mapa. As posições dos vértices são armazenadas sobre um eixo de referência, pela sua distância ao início (ponto (0, 0)) dos eixos.

Devido à utilização generalizada dos dados obtidos por satélite, a maior parte dos sistemas permite a representação de dados no formato matricial, sob a forma de pixeis, embora o método Vectorial tenha vindo a obter cada vez mais adeptos[3].

[3]Para efectuar a conversão entre formatos, nomeadamente de *raster* para vectorial veja-se Winter [1998],

Numa imagem vectorial, os pontos e linhas estão associados geométrica e matematicamente. Geralmente estes pontos são representados por coordenadas (a duas ou três dimensões).

Visto que o método Vectorial permite uma grande compactação dos dados representados, ele oferece inúmeras vantagens. Porém, no que se refere a pesquisas sobre esses mesmos dados, surgem diversos problemas que começam a ser resolvidos pela utilização de algoritmos matemáticos (veja-se o caso das árvores R e R*, que surgem como uma aproximação das árvores B+ aos sistemas georreferenciados, apresentadas no trabalho de Gutman [1984], e posteriormente refinadas nos trabalhos de Beckmann et al. [1990] e Brinkhoff et al. [1993]).

Winter [1998] apresenta um modelo que incorpora as duas representações, com utilidade para a interoperabilidade de dados existentes nos dois sistemas. Este autor introduz o conceito de modelo Topológico, que pode ser verificado com mais pormenor em Neto [1998].

Devido à existência destes diferentes modelos e dos problemas criados com a transformação de dados entre aplicações distintas, um grupo criado especialmente para estudar a organização de informação georreferenciada tentou criar uma norma que contemplasse as vantagens de cada método. Surgiu, pelo OGC (Open GIS Consortium), o Modelo Abstracto de Características, um formato que pretende ser aplicável à estrutura interna de cada fabricante de Sistemas de Informação Geográfica [Open GIS Consortium 2000].

2.3 Consórcio *Open GIS*

O OGC é uma organização não lucrativa dedicada ao geoprocessamento em sistemas abertos.

O OGC pretende possibilitar uma integração completa entre recursos e dados geo-espaciais e o uso alargado de processamento interoperacional entre produtos e aplicações cujo âmbito integre geo-dados, pela criação de uma infra-estrutura de informação. Por forma a possibilitar esta integração, propõe especificações abstractas e especificações de implementação (através de uma API) para componentes de software relacionados com Sistemas de Informação Geográfica e geoprocessamento. Segundo Open GIS Consortium [1998a], as abstracções são alcançadas através de um modelo essencial (descrições de como o mundo funciona, ou deveria funcionar) e um modelo de especificação (descrição de como o software deve funcionar).

Em conjunto, o modelo essencial e o modelo de especificação, são denominados de Modelo Abstracto de Características.

Os documentos que constituem as normas definidas pelo OGC podem ser encontrados na Figura 2.1.

Wu [2000b] e Wu [2000a].

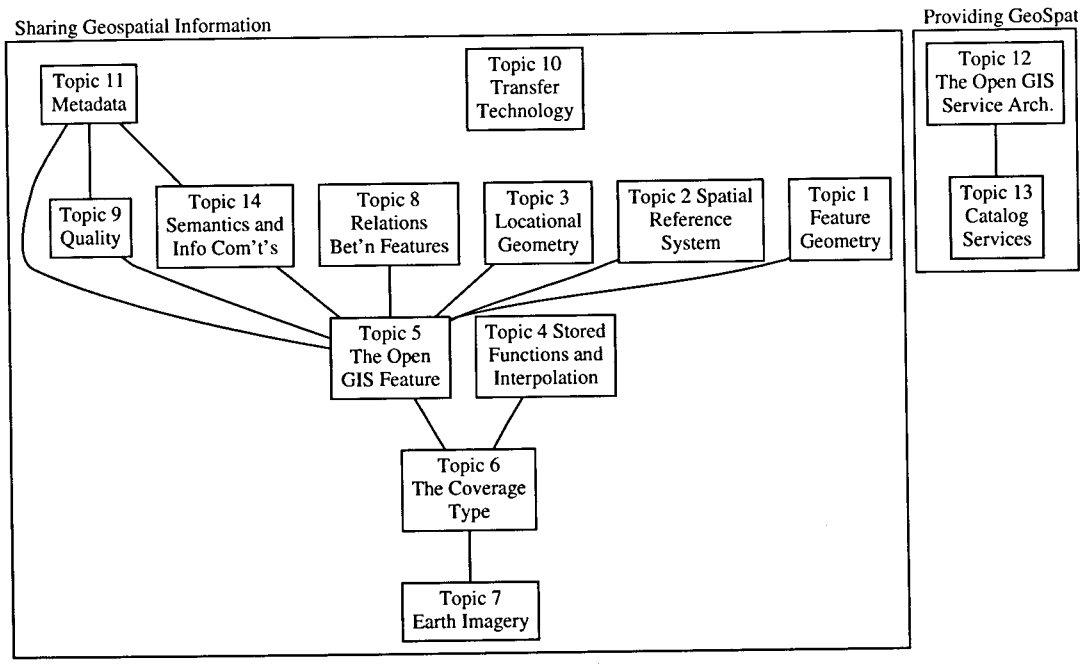

Figura 2.1: Documentos do Modelo de Especificação Abstracto [Open GIS Consortium 1998a]

2.3.1 A Característica

O conceito mais importante do Modelo Abstracto é a Característica (*Feature*)[4]. É o átomo da informação geográfica, ou seja, não é concebível pelo modelo a existência de informação geográfica sem a existência de Características. Cada Característica representa uma abstracção de um fenómeno do mundo real[5].

Uma Característica é constituída (geralmente) por informação que a permite posicionar relativamente a coordenadas da terra, ou relativamente a uma outra Característica. A forma mais comum de representar a forma e a posição de uma Característica é através de uma geometria. Na Tabela 2.2 podemos ver alguns exemplos da aplicação de Características.

Relativamente a Características, o OGC considera a existência de três tipos distintos: a *Open GIS Feature*, a *Coverage* e a *Earth Image*.

Para além dos três tipos referidos, existe também um conceito ainda não aceite plenamente pelo Consórcio: as Colecções de Características. Inclusivé, é ponderado se estas são necessárias, devido à complexidade inerente a cada Característica. Entre outras razões, aponta-se o facto de uma Característica poder ser constituída por outras Características (o que lhe permite ter o estatuto de colecção). Porém, uma Colecção de Características permite

[4]Embora uma *Feature* tenha uma categoria mais abrangente, a melhor tradução encontrada é a de Característica.

[5]O conceito de mundo real, segundo o OGC é bastante complexo. Para melhor percepção deste conceito, recomenda-se a leitura de Open GIS Consortium [1998e]

Tabela 2.2: Exemplos de Características

Um segmento de estrada entre intersecções consecutivas	Uma autoestrada consistindo em diversos segmentos	Uma estrada segmentada dinamicamente
Uma imagem de satélite georreferenciada	Um pixel da imagem mencionada à esquerda	Uma rede de drenagem
Um *overlay* de temperaturas num mapa meteorológico	Uma rede triangulada irregular	Um conjunto de eventos sísmicos e contornos de magnitude

efectuar um paralelismo mais estreito com o mundo real, como por exemplo pela organização de projectos distintos numa Colecção[6] e implica um conjunto de operações como por exemplo iterar.

Uma Característica[7] é constituída por [Open GIS Consortium 1998d, pág. 20]:

Tipo Uma Característica é principalmente de um tipo. Pode pertencer a mais que um tipo, desde que a definição do tipo integre esses sub-tipos, por exemplo através de herança (i.e. terá que ter um tipo principal — **simples** no entender do OGC — dos identificados pelo Modelo Abstracto de Características, a partir do qual se compõem outros tipos — **complexos** segundo o OGC).

Atributos Cada atributo é constituído por um nome e um valor, de acordo com o domínio de valores do atributo. Os nomes e os domínios dos atributos são definidos pelo tipo.

Participação em Processos O conjunto de processos válidos em que uma Característica pode participar são definidos (directa ou indirectamente) pelo seu tipo. Este conceito é alargado ao ponto de possibilitar a inclusão de métodos, funções e procedimentos (isto significa que alguns processos válidos podem não ser definidos directamente pela definição do tipo).

Identidade Cada Característica tem um identificador que tem de ser único no domínio e, em geral, independente do valor de qualquer dos seus atributos. Devido à unicidade ser apenas necessária dentro do âmbito da Característica, permite a criação de identificadores temporários numa determinada aplicação[8].

Persistência Neste momento ainda não existe consenso relativamente a este ponto, embora se saiba que virá a ser disponibilizada uma Especificação de Implementação CORBA[9].

[6] De referir que, de acordo com a discussão no Consórcio sobre este tema, os principais apoiantes do conceito são os que efectuam desenvolvimento para os Catálogos e seus Serviços.

[7] Também pode ser designada por Instância de Característica.

[8] Porém, caso seja necessário estabelecer ligação com outras bases de dados geográficas, será necessária a existência desse identificador, único no conjunto das bases de dados acedidas.

[9] Neste momento, a persistência das Características é deixada a cargo de cada implementador.

2.3.2 A Geometria

A Geometria é a combinação entre uma geometria de coordenadas e um sistema de referência. O OGC aponta dois sistemas de referência fundamentais: o sistema de coordenadas terrestres e o sistema de coordenadas do SIG. É possível que o SIG utilize um sistema de coordenadas geodésicas. Neste caso, ambos os sistemas são iguais. Uma figura geométrica é representada pela união de pontos (embora um único ponto seja também considerado como uma figura geométrica). Relativamente à geometria de coordenadas, esta é constituída por 4 itens:

- uma sequência de pontos do mesmo sistema de referência;
- uma colecção de geometrias, todas do mesmo sistema de referência;
- um sistema de referência espaço-temporal que relaciona a coordenada e a localização, fornecendo uma interpretação de "mundo real" à geometria;
- um algoritmo que "constrói" uma entidade geométrica que indirectamente define a extensão da geometria no espaço e no tempo.

As geometrias podem ser simples, ou complexas. As geometrias complexas são compostas por diversas geometrias simples.

2.4 Linguagem de Marcação Geográfica

Apenas a obtenção de dados geográficos sem associação a fenómenos não apresenta grande interesse para o mundo dos SIG. Cada vez mais a utilidade de um SIG é avaliada pelas operações que se podem efectuar sobre os meta-dados (informação sobre dados), ou seja, a associação de fenómenos a locais referenciados espacialmente é de enorme importância [Cox 2001].

Open GIS Consortium [1998c] descreve os meta-dados de acordo com ao norma ISO/TC 211. Os meta-dados podem ser utilizados para indicar informação:

1. de Identificação
2. sobre a Qualidade dos Dados
3. sobre a Origem dos Dados
4. sobre a Representação de Dados Espaciais
5. sobre a Referência Espacial
6. sobre o Catálogo de Características da Aplicação
7. de Distribuição

8. sobre Referência de Meta-Dados
9. de Citações
10. de Contacto
11. de Morada

Federal Geographic Data Committee [2000]; Open GIS Consortium [1998c] referem que os principais usos de meta-dados são:

1. manter o investimento interno de uma organização em dados geo-espaciais,
2. fornecer informação sobre os dados de uma organização, e
3. fornecer informação necessária para processar e interpretar dados recebidos de uma fonte externa.

Ou seja, os meta-dados permitem ir além da descrição geral existente nos dados, adicionando uma camada que permite obter informação acerca dos dados. Existem três níveis para obter conhecimento (Figura 2.2):

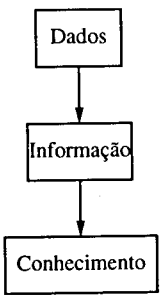

Figura 2.2: Modelo de Conhecimento (Fundamental)

Pretende-se mostrar na figura 2.2 o processo de aquisição de conhecimento. A análise dos dados permite a obtenção de informação válida relativamente a uma determinada área de conhecimento. Através da relação entre as diferentes informações obtidas pela análise de dados, podemos dizer que adquirimos conhecimento sobre um determinado assunto.

A este esquema, poderemos adicionar o nível de meta-dados, como se pode ver na Figura 2.3.

Ou seja, os meta-dados permitem mais facilmente obter informação pela análise dos dados, visto que é incorporada a explicação do sentido a dar e as regras de utilização. Da mesma forma, com o conhecimento obtido podemos aumentar o conhecimento disponibilizado pelos meta-dados.

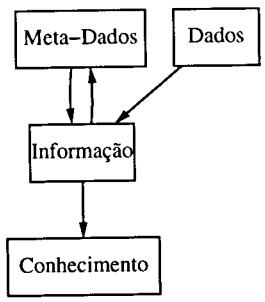

Figura 2.3: Modelo de Conhecimento (Reformulado)

Devido à importância reconhecida ao nível da obtenção de conhecimento através de dados, diversas entidades governamentais, e mesmo empresas particulares, iniciaram a procura do novo "Graal": os meta-dados inteligentes. Empresas como a Microsoft passaram a incluir suporte para meta-dados nos seus Sistemas de Gestão de Bases de Dados, em que o exemplo mais visível é o Microsoft SQL Server (versões 7.0 e 2000) [Microsoft b], conjugado com o Microsoft English Query [Microsoft a]. Estas duas aplicações efectuam um uso extensivo do dicionário de dados (e dos meta-dados aí armazenados) por forma a automatizar operações que, até há bem pouco tempo, careciam de documentação externa. Por exemplo, o Microsoft English Query permite a um utilizador sem qualquer conhecimento de Bases de Dados (nomeadamente da linguagem de programação e interrogação de Bases de Dados, Transact-SQL), efectuar pesquisas complexas sobre os dados, utilizando apenas a linguagem natural (apesar de ser necessária alguma experiência até se conseguir obter bons resultados).

Neste momento, devido à grande utilização da Internet e aos conteúdos aí existentes (até há bem pouco tempo inimagináveis), este conceito tem vindo a ser aplicado principalmente a esta plataforma. O W3C (organização reguladora das normas na Internet) apresentou propostas relativamente a este ponto, definindo o XML [W3C 2000a]. Esta linguagem permite efectuar uma definição do conjunto de meta-dados definindo esquemas que, espera-se, sejam aceites pelas organizações que gerem cada área científica. No caso específico dos SIG, essa entidade é o *Open GIS Consortium* (OGC).

O OGC, no espírito da utilização das tecnologias mais avançadas, decidiu elaborar uma norma para transferência de formatos nativos através de uma camada desenvolvida especificamente para o efeito. Em Maio de 2000 apresentou a norma GML 1.0, sendo substituída em Fevereiro de 2001 pela versão 2.0[10] [Lake 2001].

2.4.1 GML 1.0

GML 1.0 baseia-se numa combinação dos DTDs [Champin 2001; W3C 2000a] com RDF [Bray 2001]. Esta combinação, embora obsoleta, tornou-se útil nos seus primórdios. Os

[10] A versão 3.0 está prevista para finais do corrente ano (2001).

DTDs encontravam-se em franca expansão, embora impedissem (não tinham suporte para) técnicas avançadas, como por exemplo a herança. Para além disso, não possuíam um modelo semântico de suporte e não permitiam o uso de espaços de nomes. Por seu lado, os RDFs eram menos aceites, mas ofereciam suporte para espaços de nomes, integração distribuída de esquemas, hierarquias de tipos de dados (a base das linguagens orientadas a objectos) e um modelo semântico simples. Embora fosse possível combinar estas duas técnicas, era complicado efectuar a sua ligação.

GML 1.0 oferece três perfis (*profiles*) distintos, referidos como GML.1, GML.2, e GML.3. Estes perfis permitem alguma flexibilidade em GML 1.0, através de diferentes abordagens na combinação dos métodos anteriormente referidos (DTDs e RDFs [Bray 2001]).

Um excerto de um documento GML 1.0 (no Perfil 1) pode ser como apresentado a seguir:

25a ⟨*Excerto de Documento GML.1* 25a⟩≡

```
<Feature typeName="Road">
   <description>Georgia Street</description>
   <property typeName="numberLanes" type="integer">4</property>
   <geometricProperty typeName="linearGeometry">
      <LineString srsName="EPSG:4326">
         <coordinates> 0.0,100.0 100.0,0.0 </coordinates>
      </LineString>
   </geometricProperty>
</Feature>
```

De notar que este exemplo não utiliza espaços de nomes (que não eram suportados nos DTDs) e que o tipo da Característica não está definido. O Perfil 1 de GML 1.0 não permitia a inclusão do esquema das Características independentemente da instância de dados.

O Perfil 2 permitia mais suporte a esquemas, através da utilização de DTDs definidos pelo utilizador (criador do esquema); este suporte é continuado na versão 2.0, embora através de um método distinto: os Esquemas XML (*XML Schemas*) [W3C 2000b].

Segue-se um exemplo de um excerto de um documento GML 1.0 (no Perfil 2):

25b ⟨*Excerto de Documento GML.2* 25b⟩≡

```
<Road>
   <description>Georgia Street</description>
   <numberLanes>4</numberLanes>
   <centerLineOf>
      <LineString srsName="EPSG:4326">
         <coordinates>0.0,50.0 0.0,100.0 </coordinates>
      </LineString>
   </centerLineOf>
</Road>
```

Através do Perfil 2, os utilizadores podiam definir os seus tipos de Características, utilizando DTDs. As instâncias eram facilmente identificadas pela presença de tipos como elementos XML, ou marcas (*tags*), como por exemplo `<Road>`. Porém, nota-se a falta de suporte para espaços de nomes, bem como de uma hierarquia estabelecida.

Algumas destas restrições foram levantadas no Perfil 3. As instâncias definidas são muito semelhantes às encontradas em GML 2.0, como veremos a seguir.

Segue-se um exemplo de um excerto de um documento GML 1.0 (Perfil 3):

26 ⟨*Excerto de Documento GML.3* 26⟩≡

```
<os:Road>
   <gml:description> Georgia Street </gml:description>
   <os:numberLanes>4</os:numberLanes>
   <gml:centerLineOf>
      <gml:LineString srsName="EPSG:4326">
         <gml:coordinates>0.0,100.0 100.0,0.0</gml:coordinates>
      </gml:LineString>
   <gml:centerLineOf>
</os:Road>
```

É relevante verificar neste exemplo a inclusão de esquemas definidos pelo utilizador, como se torna óbvio no nome do elemento `<os:Road>`. O uso de prefixos de espaços de nomes (os antes de Road, no exemplo) permite que os criadores de conteúdo estabeleçam os seus vocabulários próprios, baseados na organização, ou na comunidade de interesse. Através desta técnica, é fácil de distinguir entre `<os:Road>`, `<usgs:Road>` ou mesmo `<nrcan:Road>`, pois todos estes nomes são distintos.

2.4.2 GML 2.0

GML 2.0 [Cox et al. 2001], que substitui a anterior versão, é baseado inteiramente nos Esquemas XML [W3C 2000b]. A adopção destes é um enorme avanço. Os Esquemas XML evoluíram bastante nos últimos tempos. Incorporam suporte para herança de tipos, integração distribuída de esquemas e espaços de nomes. Para além disto, há já uma imensidão de ferramentas e parsers que suportam esta nova tecnologia, sendo esperados muitos mais num futuro próximo [Lake 2001].

2.4.2.1 Esquemas XML

Esquema XML (*XML Schema — XSD*) é um Candidato a Recomendação da W3C para dar estrutura a documentos XML [W3C 2000b]. A especificação de XML Schema assume que pelo menos dois documentos XML são utilizados: um documento instância e pelo menos um documento esquema. O documento instância contém a informação propriamente dita e o documento esquema descreve a estrutura e tipo do documento instância. A distinção

entre instância e esquema é semelhante à distinção entre objecto e classe em linguagens de programação orientadas a objeto. Uma classe descreve um objeto assim como um esquema descreve um documento instância [Box et al. 2000].

Em Esquemas XML, o modelo de conteúdo de um elemento pode ser especificado a partir da declaração de um tipo. Um tipo pode ser Simples ou Complexo. Um tipo simples pode ser atribuído a um atributo ou a um elemento simples, que possui somente texto e não possui elementos filhos (*child elements*. Já um tipo complexo é utilizado para dizer quais são os subelementos permitidos para um determinado elemento. Tipos simples são declarados com `simpleType`, e tipos complexos com `complexType`. A declaração `element` liga um tipo a um nome de elemento.

Um `simpleType` pode ser de um dos tipos básicos definidos em Esquema XML, tais como "string", "integer", "date", entre outros. Uma relação completa dos tipos simples definidos por XML Schema pode ser encontrada em W3C [2000c].

Um `complexType` define restrições para o modelo de conteúdo de um determinado elemento, o que é feito através dos atributos para especificação de cardinalidade `minOccurs` e `maxOccurs`, e dos delimitadores de grupos de elementos "sequence", "all" e "choice". O atributo `minOccurs` especifica o número mínimo de vezes que um subelemento pode aparecer, e `maxOccurs`, o número máximo. A declaração `<element name="AUTOR" type="pub:tAutor" minOccurs="1" maxOccurs="unbounded"/>` especifica que o elemento AUTOR pode aparecer, no mínimo, uma vez, e que não há um limite máximo. A declaração dos atributos `minOccurs` e `maxOccurs` não é obrigatória. Quando omitidos, são assumidos os valores 1 (um) para `minOccurs` e 1 (um) para `maxOccurs`, respectivamente. Isso faz com que o elemento em questão seja obrigatório.

O grupo "sequence" indica que os subelementos devem aparecer na instância XML na mesma ordem em que foram declarados no esquema. O grupo "choice" diz que somente um dos elementos declarados no grupo pode aparecer na instância, e o grupo "all" estabelece que todos os elementos do grupo podem aparecer uma ou nenhuma vez e que eles podem aparecer em qualquer ordem.

O exemplo mostrado abaixo exige que o elemento `REFERENCIA` apareça sempre após o elemento `CITACAO`, e que cada um apareça somente uma vez em cada `PARAGRAFO`.

27 ⟨*version/FundTeorica/exemplo.xsd* 27⟩≡

```
<element name="PARAGRAFO" type="pub:tParagrafo"/>
<complexType name="tParagrafo" mixed="true">
   <sequence>
      <element name="CITACAO" type="string"/>
      <element name="REFERENCIA" type="string/>
   <sequence>
</complexType>
```

Definições de tipo podem ser reutilizadas através de uma referência ao seu nome. Além disso, elementos globais podem ser referenciados noutras declarações.

Em Esquemas XML, atributos são declarados utilizando a marca `attribute`. Atributos em XML costumam ser utilizados para representar **meta-dados**.

GML 2.0 fornece um método único de codificação e uma abordagem para a criação de esquemas, como veremos a seguir.

Tal como no Perfil 3 de GML 1.0, é possível utilizar espaços de nomes para criar diferentes vocabulários. Para além disso é possível incorporar herança de tipos e esquemas distribuídos para criar famílias como mostrado na Figura 2.4, sem preocupação com conflitos de nomes.

A Figura 2.4 mostra três vocabulários. Um é o conjunto básico de definições de Características e geometrias (o *Common Geo-spatial Vocabulary*), enquanto os outros dois fornecem tipos específicos de Características para dois domínios distintos (neste caso o florestal — *Forestry* e de ambiente — *Environment*). De salientar que, devido ao uso dos prefixos de espaços de nomes, cada um dos domínios identificados na figura pode definir o mesmo esquema de tipos, sem preocupação de conflito de nomes. Por exemplo, uma `Road` terá noções diferentes, sendo identificadas pelos utilizadores através do prefixo utilizado (ou seja, `<env:Road>` para o domínio ambiental, ou `<for:Road>` para o domínio florestal).

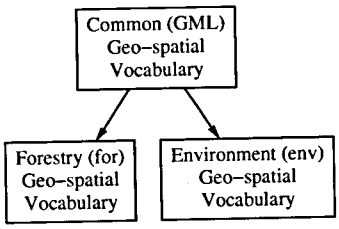

Figura 2.4: Criando Vocabulários de Tipos de Característica [Galdos Systems, Inc. 2001]

GML 2.0 fornece as definições base e os mecanismos para construir uma hierarquia de tipos de Características. Neste esquema residem as fundações da "Web Geo-espacial".

No mundo real existem relacionamentos. Construções na margem de Estradas, Estradas que se interceptam, etc. No passado, alguns SIG permitiam estes relacionamentos de uma forma muito restrita, não havendo real aproveitamento do conceito de relacionamento. A maior parte estava restrita a relacionamentos topológicos. Com GML 2.0 prevê-se uma mudança.

Através da integração de técnicas de *XLink* e *XPointer*, pretende-se abrir a porta para relacionamentos entre entidades geo-espaciais. Isto significa que os relacionamentos podem ser expressos quer entre Características constantes numa mesma base de dados, quer com outras existentes em bases de dados distintas (o objectivo principal é integrar diversas bases de dados disponíveis na Internet). Por forma a estabelecer relacionamentos entre diferentes Características não será necessário, de acordo com o OGC, efectuar modificações nas bases de

dados participantes. Apenas acesso de leitura será necessário. Neste momento ainda é cedo para saber se esta pretensão terá consenso.

GML 2.0 aproveita os mesmos conceitos que tanto sucesso granjearam ao HTML (a possibilidade de exprimir ligações entre páginas Web e de usar localizadores universais), aplicando neste caso a Bases de Dados (mais concretamente, às Características geo-espaciais).

Para permitir que os relacionamentos sejam extensíveis, o OGC considera-os como sendo um tipo de Característica GML. Assim, os relacionamentos podem ter propriedades, para além de identificarem as associações entre Características distintas. Com GML 2.0 é possível exprimir relacionamentos simples (binários) entre duas entidades, bem como relacionamentos mais complexos que envolvam múltiplos (i.e. mais que dois) recursos distribuídos [Open GIS Consortium 1998f].

Da mesma forma como o HTML teve um papel primordial e fulcral no desenvolvimento da Internet como uma imensa colecção de páginas ligadas, o OGC espera que GML 2.0 possibilite o desenvolvimento de uma *Internet Geo-espacial* como uma colecção ligada de Características geo-espaciais [Open GIS Consortium 1998b].

2.4.3 GML 3.0

Com GML 2.0, a linguagem adoptada para dados geo-espaciais atingiu um grau de maturidade que permite a construção de dados espaciais integrados reais, a fácil transferência de informação e a construção de relacionamentos distribuídos. Neste momento esperava-se que GML 2.0 fosse mais utilizado. O seu impacto ainda não foi o esperado, pelo que a definição da norma seguinte (GML 3.0) está atrasada.

Previsto para o Outono deste ano (o que não parece que venha a acontecer), espera-se que GML 3.0 ofereça bastantes melhorias relativamente à versão 2.0, embora mantendo compatibilidade com esta. Algumas das novas funcionalidades são o suporte para topologias, novas classes geométricas (mais complexas), eventos, históricos e *timestamps* nas Características, unidades de medida, meta-dados (normas para a utilização), e coberturas (*coverages*) [Galdos Systems, Inc. 2001].

Como se vê, GML está em movimento, com o objectivo de disponibilizar e dinamizar a *Web Geo-espacial.*

2.5 Sumário

Da discussão anterior, verifica-se rapidamente a utilidade que as normas apresentadas pelo OGC trazem ao mundo dos SIG. Porém, este trabalho está longe de estar concluído. Novas funcionalidades são pedidas, em grande parte devido à rápida evolução tecnológica verificada.

A inclusão de meta-dados, por forma a obter informação válida e correcta relativamente a

fenómenos georreferenciados, e a possível inclusão de uma coordenada temporal, como previsto para a próxima versão do Modelo Abstracto de Características, implicará a redefinição das tecnologias inerentes (como por exemplo GML), e permitirá alcançar informação até agora não possível de idealizar.

Nos próximos anos verificar-se-á um crescimento exponencial de dados georreferenciados- que, apesar de imensos neste momento, deverão ser apenas uma gota no oceano infindável de aplicações possíveis. Desde já as grandes empresas do sector estão envolvidas neste projecto, participando activamente na sua evolução. Entre estas destaca-se a Galdos Inc., que tem sido, até agora, a maior fornecedora de informação relativamente ao assunto, através de publicações disponibilizadas pelo seu director, Ron Lake (www.galdosinc.com). Entre outras iniciativas, esta companhia está a preparar um sítio web (www.gmlworld.com) para discussão e análise das normas emanadas pelo OGC, nomeadamente GML. Outro dos sítios dinamizados por Ron Lake é www.gmlcentral.com onde já pode ser encontrada informação sobre GML.

Apesar de esta tecnologia já ter algumas (embora poucas) provas dadas, não pode ser esquecida a sua recente aplicação. Talvez por este motivo ainda existam tão poucas fontes de informação relativamente a ela. Também por este motivo se torna um pouco frustrante obter informação extra, visto que ainda é muito restrita aos membros do consórcio, o que provoca grandes investimentos, embora sejam frutíferos a médio prazo.

Capítulo 3

Uma Aplicação para Sistemas Móveis

O objectivo desta dissertação é provar que é possível utilizar sistemas móveis de computação para efectuar, no local, a manutenção de dados georreferenciados.

Após termos visto em que consistem os Sistemas de Informação Geográficos e uma das iniciativas para permitir a interoperabilidade entre dados de diferentes aplicações, iremos concentrar os nossos esforços no desenvolvimento dos módulos de software que constituem a aplicação que possibilita efectuar essa manutenção.

Neste capítulo apresentam-se os requisitos necessários para o funcionamento de um sistema móvel para manutenção de dados georreferenciados. Este sistema utiliza tecnologias específicas, devido às características inerentes à plataforma de hardware e às normas escolhidas para o seu funcionamento.

É também apresentado o processo pelo qual se irá efectuar o desenvolvimento da aplicação e o modelo previsto de funcionamento do sistema.

Finalmente é referido o caso específico no qual os testes se irão realizar.

3.1 Identificação dos Requisitos

Esta dissertação tem o objectivo de provar que os sistemas móveis de computação (vulgo PDA) são viáveis para a manutenção de dados georreferenciados. Torna-se assim necessária a implementação de uma aplicação que permita a realização deste propósito.

Algumas restrições relativamente ao sistema a desenvolver estão identificadas no próprio tema da dissertação. Assim, será necessário que a aplicação seja desenvolvida para ser executada em pelo menos um sistema móvel. Por outro lado, e visto que é necessário utilizar dados georreferenciados, torna-se necessário seleccionar uma aplicação que se possa integrar com o sistema a desenvolver, aplicação essa que permita a visualização de mapas. Como se pretende que este seja um sistema aberto, verifica-se a necessidade de utilizar formatos padrão ou, no

caso de tal não ser possível, de permitir a transformação dos formatos seleccionados pelas aplicações desenvolvidas para formatos padrão, que permitam a interacção com outros sistemas já existentes. Devido ao facto de ser necessária a utilização de uma ferramenta externa para visualização dos mapas, a aplicação desenvolvida e módulos auxiliares terão, necessariamente, que estar directamente ligados a formatos nativos, visto que não há conhecimento de aplicações a utilizar os formatos padrão (nomeadamente o GML, referido anteriormente).

Assim, serão considerados como requisitos fundamentais para provar a viabilidade da plataforma móvel:

- utilização de ferramentas de criação de aplicações para sistemas móveis;
- criação de aplicações do lado do servidor que permitam a transformação de dados entre o(s) formato(s) nativo(s) utilizado(s) no Palm e o formato padrão adoptado no sistema anfitrião;
- identificação e resolução de problemas criados pela actualização de dados em sistemas móveis, com a necessária reconciliação de dados entre os sistemas;
- elaboração de uma API (*Application Programming Interface*) que permita que outras aplicações possam comunicar com a aplicação desenvolvida;
- avaliação relativamente à coerência e validação dos documentos em formato padrão gerados após modificações efectuadas nos sistemas móveis;

3.2 Tecnologias Necessárias

3.2.1 Plataformas de Computação Móvel

Relativamente à aplicação a ser implementada, esta deveria ser executada numa plataforma móvel. Existindo diversas plataformas distintas (e infelizmente incompatíveis), foi necessário seleccionar uma delas. Isto levou a que o código não fosse portável o que, não sendo um problema crítico, é uma limitação do sistema desenvolvido.

Relativamente a plataformas móveis possíveis de utilizar, analisaram-se as seguintes:

- Windows CE
- Palm OS

De entre estas preferiu-se a utilização de um sistema Palm, visto que é o mais generalizado destes sistemas.

3.2.2 Ambientes de Desenvolvimento

Seleccionou-se como sistema de desenvolvimento para o Palm o *Metrowerks CodeWarrior 7.0 for Palm OS*. Este sistema está optimizado para a criação de aplicações Palm, contendo um ambiente de desenvolvimento gráfico (GUI/IDE).

Para os módulos da aplicação que serão executados no Servidor (por exemplo a criação dos PDB a partir de um ficheiro GML), utilizou-se o Java (JDK 1.3.1). Desta forma, garante-se a portabilidade para qualquer Sistema Operativo que inclua a máquina virtual Java.

Para os mecanismos de Sincronização, havia diversos ambientes à escolha, desde que fosse possível a criação de uma DLL.

De entre as diversas linguagens conhecidas, optou-se pela utilização do Microsoft Visual C++ 6.0, visto que é possível efectuar o *download* de um *template*[1] que permite a utilização de um Assistente (*Wizard*) para a criação de código genérico de sincronização.

3.2.3 Características das Plataformas

Tendo considerado até agora as particularidades dos SIG, nomeadamente no que diz respeito à representação de meta-dados georeferenciados, será também necessário identificar quais os problemas que podem surgir devido ao facto de a aplicação a implementar dever funcionar num Sistema de Computação Móvel[2].

As plataformas móveis de computação possuem um conjunto de características específicas que obrigam a novas abordagens no momento de idealizar uma aplicação, como veremos a seguir.

Apesar dos dispositivos com o sistema Windows CE permitirem um grande conjunto de funcionalidades quando comparado com os dispositivos Palm, graças ao excelente sistema operativo que incluem bastante semelhante ao Windows, a família de dispositivos Palm tem melhor tempo de resposta e de pesquisa de dados. Graças ao sistema interno de gestão de memória, também se verifica que a bateria tem maior duração (visto que as instruções de refrescamento de ecrã são mais simples).

As principais vantagens dos dispositivos Palm resumem-se a duas: fácil utilização para aplicações móveis (por exemplo PIMs – Gestores de Informação Pessoal) e compatibilidade com um elevado número de aplicações já existentes. Qualquer aplicação que execute num Palm III garantidamente executará num outro dispositivo mais avançado sem necessidade de actualização.

Neste momento assiste-se a uma batalha entre as duas plataformas que, mais que uma batalha entre duas companhias, é uma batalha entre duas ideologias. Por um lado o Palm, que representa a simplicidade (ou poderemos dizer, minimalismo), por outro um sistema que incorpora um conjunto de funcionalidades que na maior parte das vezes não são utilizadas.

[1] Pode ser obtido em http://www.palmos.com/dev/tech/tools/cdk/win/palmcdk402a.exe.
[2] Também conhecido como PDA.

Como não necessitámos das funcionalidades extra (tais como suporte para streaming áudio e vídeo) acrescentadas pelos dispositivos com Windows CE e como se espera que a aplicação aqui desenvolvida seja portável para o maior número possível de utilizadores, a escolha tornou-se simples: o sistema Palm.

A partir do momento em que foi definida a utilização da plataforma Palm, foi necessário descobrir, para além das características inerentes aos Sistemas Móveis em si, quais as características específicas deste sistema. Para tal, efectuaram-se análises de diversas páginas Web comparativas destes sistemas, desde os sítios com mais informação (teoricamente) sobre o assunto (o próprio sítio da Palm www.palmos.com e o sítio dedicado a implementadores de aplicações www.palmos.com/dev/), passando por sítios de terceiros, sem ligação aparente à empresa Palm, Inc.

3.3 Plataformas Móveis de Computação

Satyanarayanan [1996] identificou a existência de quatro restrições nos sistemas móveis:

- pobreza de recursos;
- potencialmente falíveis;
- conectividade é muito variável, quer em desempenho, quer em disponibilidade;
- assentam em fontes de energia finitas.

Estas restrições não dependem da actual tecnologia, mas são inerentes ao próprio conceito de mobilidade. Estes problemas obrigam a que a parte crítica do sistema fique entregue a um servidor.

Por outro lado, é necessário que os próprios sistemas móveis imponham regras aos seus utilizadores. Assim, os sistemas móveis devem ser adaptativos. Satyanarayanan [1996] identifica, através de um esquema, três estratégias de adaptabilidade (Figura 3.1).

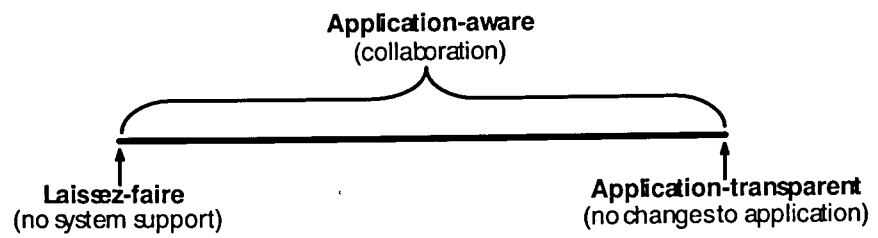

Figura 3.1: Níveis de Estratégias de Adaptação [Satyanarayanan 1996]

Relativamente aos sistemas móveis, há necessidade de identificar e resolver, em cada aplicação, o problema relativo à sincronização de dados.

3.3.1 Sincronização de Dados

Como não é recomendável efectuar a passagem de todos os dados para um sistema móvel (principalmente devido aos fracos recursos) e visto que estes dados podem ser modificados em diversos ambientes distintos (por exemplo, dois utilizadores podem modificar uma mesma cópia de dados), a sincronização de dados torna-se uma das tarefas principais. A reconciliação poderá ser simples (caso apenas haja modificação num único cliente), de média complexidade (caso a modificação seja efectuada em mais que um cliente), ou de grande complexidade (caso a modificação seja efectuada em diversos clientes e no próprio servidor de dados). Relativamente a cada caso, as abordagens de sincronização têm, necessariamente, de ser diferentes.

Diversos autores, entre os quais Agrawal e Sengupta [1989]; Satyanarayanan [1996]; Wu et al. [1993], estudaram as implicações relativas à sincronização, utilizando para tal o sistema CODA [Satyanarayanan et al. 1990].

Aplicações para Redes Móveis de Computadores geralmente geram ou modificam dados partilhados com um servidor e/ou com outros dispositivos móveis. Visto que um dispositivo móvel se encontra desligado da rede durante a maior parte do tempo, conectado através de ligações frequentemente caras (por exemplo, através de uma ligação telefónica, no caso dos celulares), ou lenta para utilizar sempre que os dados são modificados, estas aplicações geralmente mantêm cópias separadas dos dados em dispositivos distintos. Estas cópias necessitam ser sincronizadas de quando em quando com outras cópias, de forma a que todas contenham os mesmos dados.

A sincronização de dados é um processo complexo. Algumas organizações, como o SyncML [SyncML 2001] e o MNCRS, estão a tentar especificar uma API que forneça serviços comuns de sincronização de dados. Estes serviços permitirão que a tarefa de escrever aplicações para dispositivos móveis seja simplificada.

A iniciativa do MNCRS, que se prevê alcance um grande sucesso, principalmente desenvolvida para integração com a plataforma Java, identifica um "armazém de sincronização" (*sync store*) como uma colecção de objectos Java. Cada objecto está associado com um identificador (*sync id*). Dois armazéns podem ser sincronizados, ou reconciliados para um estado comum que reflicta as actualizações efectuadas em cada um separadamente, mesmo que estes se encontrem em redes distintas. Estes armazéns são denominados réplicas. Visto que é possível que um objecto tenha sido modificado de formas diferentes em armazéns distintos, é necessário existir uma tarefa de reconciliação com actualizações potencialmente conflituosas. Compete ao programador de cada aplicação seleccionar e implementar os algoritmos e técnicas específicos a cada aplicação.

O identificador (id) do objecto deve ser determinado na primeira inserção deste no armazém. Este identificador poderá ser criado automaticamente ou definindo um conjunto de identificadores válidos, baseados nos dados da aplicação. Deve ser possível examinar, actualizar, ou eliminar um determinado objecto identificado pelo seu id (ou seja, este id será a chave

primária do objecto, pensando em termos de bases de dados relacionais). As inserções, actualizações e eliminações num armazém local devem ser propagadas a outros armazéns durante a sincronização.

Cada objecto de um armazém é criado através da inserção numa réplica. Visto que o objecto é propagado a outras réplicas através da sincronização e potencialmente actualizado em diferentes réplicas, variações do objecto correspondente a um determinado identificador podem ser encontrados em diferentes réplicas. Para duas variantes do mesmo objecto, existem duas possibilidades:

- uma variante é o resultado de um determinado número de actualizações à outra variante;
- cada variante resultou da aplicação de um conjunto diferente de modificações efectuadas a uma terceira variante.

No primeiro caso, diz-se que a variante que incorpora as actualizações é mais recente que a outra variante. No segundo caso, dizemos que as duas variantes estão em conflito. Quando uma réplica contém uma variante mais recente de outra réplica, ambas as réplicas devem conter o valor da nova variante. Quando duas réplicas contêm variantes em conflito, a sincronização deve criar uma variante que seja a combinação das variantes em conflito. Esta nova variante é considerada mais recente que as variantes que lhe deram origem.

O relacionamento entre duas variantes de um objecto reflecte a história das modificações sobre o objecto não tendo em atenção o conteúdo das variantes. Ou seja, duas variantes distintas poderão ter o mesmo conteúdo. É comum que, durante o processo de reconciliação, duas variantes de um objecto sejam combinadas numa terceira variante que descarte o conteúdo de uma das variantes. Mesmo que a variante resultante tenha o mesmo conteúdo que uma das variantes originais, é mais recente que ambas as variantes originais. Este é um dos casos em que o resultado final foi obtido pela análise do conteúdo de cada uma das variantes originais.

O "calcanhar de Aquiles" da computação móvel é a sincronização de dados. Todos os dispositivos móveis (computadores portáteis, telefones móveis, pagers, PDAs) sincronizam os seus dados com aplicações de rede, calendários pessoais num PC ou distribuídos, ou outras localizações onde a informação está armazenada. A possibilidade de aceder e actualizar esta informação rapidamente, é a chave para a natureza da computação móvel. Porém, cada dispositivo utiliza tecnologias diferentes para efectuar sincronização de dados.

Por forma a detectar diferenças entre documentos baseados em XML, existe trabalho efectuado por diversos autores, sendo os mais relevantes os de Wang et al. [2000] (X-Diff), Ball e Douglis [1996]; Berk [1996]; Douglis et al. [1998] (HtmlDiff) e Chen et al. [2000] (TopBlend, baseado no HtmlDiff).

3.4 Sincronização e Condutas

Relativamente à sincronização em ambientes Palm, Gossett [2000] identifica quatro formatos básicos:

1. **sincronização nos dois sentidos**, utilizada para sincronizar dados que podem ser modificados tanto no dispositivo móvel como no servidor;
2. **transferir do Palm**, usado quando os dados são apenas modificados no dispositivo;
3. **transferir para o Palm**, usado quando os dados são apenas modificados no servidor;
4. **adaptável**, sempre que as opções anteriores não são suficientes.

Por exemplo no sistema operativo Windows, uma conduta (*conduit*) não é mais que uma biblioteca (`DLL`) que permite efectuar a ligação entre um dispositivo Palm e uma determinada aplicação Windows.

Qualquer *Conduit* deve obedecer aos seguinte requisitos impostos pela Palm:

rápida execução devem ser desenhados tendo em atenção a velocidade de processamento (que se pretende máxima) e a quantidade de dados transferidos (que se pretende mínima);

sem perda de dados devem ser tomadas todas as medidas necessárias para prevenir a perda de dados, mesmo que exista perda de ligação durante o período de sincronização;

gestão de conflitos sendo efectuada uma sincronização nos dois sentidos, devem ser detectados e resolvidos potenciais conflitos de actualização. Devem ser acresentadas entradas no `LOG` por forma a notificar o utilizador da situação;

sem intervenção do utilizador desde que o utilizador pressiona o botão de *HotSync*, todo o processo deve decorrer sem mais necessidade de intervenção.

Gossett [2000]

3.5 Arquitectura e Processo de Trabalho

O sistema delineado para possibilitar a manutenção dos dados georreferenciados está apresentado na Figura 3.2.

Como se vê pelo modelo da Figura 3.2, o trabalho inicia-se com um (ou mais) documento(s) GML. Nestes estão armazenados os dados georreferenciados. É possível que os dados referentes à cartografia se encontrem misturados com os meta-dados, ou poderão encontrar-se em ficheiros separados. Também é possível que utilizem esquemas distintos, aplicados a

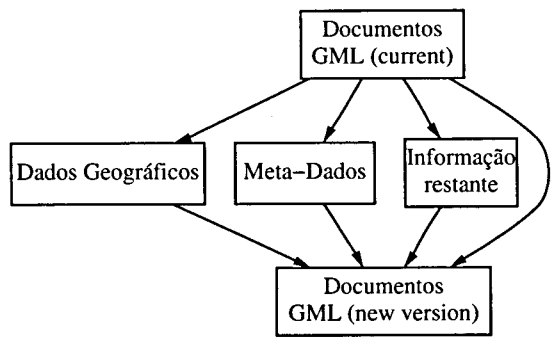

Figura 3.2: Modelo Previsto de Funcionamento da Aplicação

casos específicos. As possibilidades são infindáveis. Os mesmos dados poderão representar informação diferente para utilizadores distintos.

Esses documentos GML são inicialmente analisados e convertidos para bases de dados de um sistema móvel (neste caso foi seleccionado o Palm), conhecidas como PDB.

A partir desse momento, uma aplicação Palm permite a visualização da cartografia, efectuando chamadas à aplicação que permite a manutenção dos meta-dados. Para essa tarefa seleccionou-se a aplicação Mordomo (www.mordomo.com).

Sempre que for necessário efectuar a reconciliação dos dados (através de uma operação de sincronização), uma aplicação[3] obtem as actualizações efectuadas, gerando um novo GML que pode ser inserido noutro dispositivo móvel.

Um dos problemas existentes relaciona-se com a possibilidade de ocorrerem alterações simultâneas em múltiplas réplicas. Porém, no âmbito da presente dissertação, embora seja efectuada uma avaliação das implicações desse facto, não irá ser implementada uma solução, embora seja um problema reconhecido e para o qual é necessária uma solução.

Na Figura 3.3 pode-se encontrar o esquema utilizado, com as funções desempenhadas por cada uma das aplicações desenvolvidas.

Como se pode verificar pelo esquema representado na Figura 3.3, o processo inicia-se pela análise e separação entre elementos de documentos GML. O módulo que efectua esta análise é `GML2PDB`, que separa os elementos de acordo com a sua categoria, nas três bases de dados utilizadas pelo Palm. O processo irá ser descrito com mais detalhe no Capítulo 4.

Estas Bases de Dados são utilizadas pelas aplicações existentes no Palm, nomeadamente pela aplicação Mordomo, que efectuà a visualização de cartografia e de pontos de interesse (o acesso a estas bases de dados é apenas de leitura), e pelo módulo desenvolvido MetaGeo[4] que apresenta e permite modificar dados na Base de Dados `metageo.pdb`. A descrição do funcionamento deste módulo pode ser encontrada no Capítulo 5.

[3]Em terminologia Palm, uma Conduta (*Conduit*).

[4]Visto que este módulo tem o mesmo nome que a aplicação desenvolvida utiliza-se `MetaGeo` para referir o módulo e `MetaGEO` para referir a aplicação.

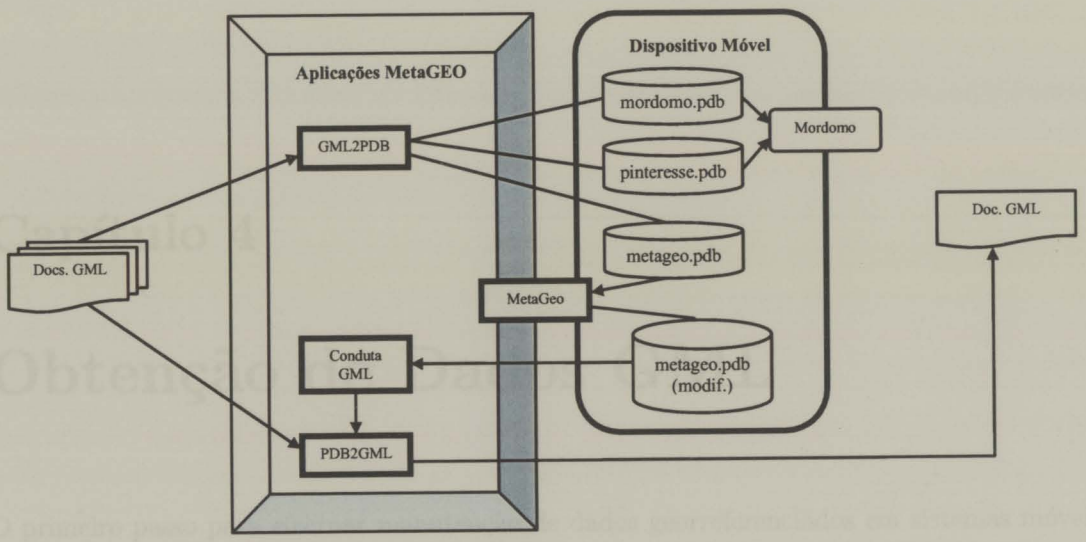

Figura 3.3: Arquitectura proposta desenvolvida para a aplicação

Esta última Base de Dados (a única modificada no Palm) é, sempre que se efectua uma sincronização, convertida numa árvore DOM que é comparada com a árvore DOM resultante da análise do documento GML existente no sistema anfitrião, pelo módulo `PDB2GML`. Este módulo é invocado pela Conduta desenvolvida (`Conduta GML`) que envia os registos modificados no Palm. O módulo `PDB2GML` gera o novo documento GML, como se pode ver no Capítulo 6.

Os capítulos seguintes apresentam, assim, as formas encontradas para a execução de cada uma das tarefas anteriormente referidas.

Capítulo 4

Obtenção de Dados GML

O primeiro passo para efectuar manutenção de dados georreferenciados em sistemas móveis consiste em efectuar a transformação para um formato reconhecível pelo sistema móvel. Aqui reside o primeiro problema com o sistema seleccionado: não existe o conceito de base de dados. Existem, isso sim, ficheiros com uma estrutura definida, onde é possível armazenar registos (formato PDB).

Uma das primeiras tarefas foi idealizar um modelo de dados que permitisse a transformação dos dados GML para o formato PDB. Este modelo de dados, baseado nos conceitos do modelo relacional de Codd [1970], permite a existência de relacionamentos.

A seguir apresentam-se os passos que foi necessário realizar por forma a ser possível a criação dos ficheiros PDB, a ser inseridos no Palm, contendo os dados existentes num ficheiro GML.

Este módulo foi baptizado `GML2PDB`.

4.1 Conversão de GML para formato nativo Palm

O módulo `GML2PDB` tem como função efectuar a divisão de um ficheiro GML nos diferentes componentes que o constituem e gerar as bases de dados no formato Palm [Palm Inc. 2000a] que serão posteriormente inseridas nesta plataforma.

Este módulo foi primeiro realizado para suportar o GML 1.0, tendo posteriormente evoluído para suportar GML 2.0, que surgiu no início do desenvolvimento deste módulo, com funcionalidades importantes como vimos na subsecção 2.4.2. Visto que houve grande diferença entre os formatos, de seguida apresentam-se os métodos utilizados para desenvolver cada uma das versões.

4.1.1 Conversão em GML 1.0

Por forma a validar a pretensão de implementar um módulo para permitir a manipulação de dados num dispositivo móvel através de GML, era primeiro necessário visualizar um mapa neste dispositivo. Para que tal fosse possível, foi necessário efectuar a conversão de um ficheiro compatível com o formato GML 1.0 para o formato nativo da aplicação Mordomo.

Neste momento, havia a possibilidade de definir um formato nativo de armazenamento de dados, o que implicaria o desenvolvimento de um módulo que efectuasse a conversão de GML para esse formato, ou utilizar o próprio GML armazenado como um arquivo no Palm.

Embora a segunda possibilidade fosse atraente, a verdade é que os ficheiros GML (instância específica de XML) tendem a ser muito extensos. Isto implicaria que o ficheiro original tivesse que ser comprimido por forma a ser armazenado no Palm.

Após analisar diversos algoritmos de compressão concluiu-se que, dadas as limitações de memória existentes neste dispositivo, o mais apropriado seria o XMill, desenvolvido por Liefke e Suciu [2000]. Este algoritmo, optimizado para a utilização com ficheiros XML permite uma grande compressão de dados, já que o dicionário de compressão é constituído pelas marcas existentes no ficheiro. Porém, a descompressão obriga a descomprimir todo o arquivo, o que apresenta grandes restrições[1].

Visto que não sabemos o tamanho de cada bloco uma vez descomprimido, qual a memória disponível no dispositivo móvel para efectuar a descompressão e qual o bloco necessário (ou mais que um, já que os dados são georreferenciados, sem nenhuma ordem especial), temos um problema de difícil resolução.

Uma das formas de contornar este problema seria definir fragmentos pequenos para compressão. Isto levaria a que o algoritmo perdesse bastante em termos de eficiência (Liefke e Suciu [2000] apresenta resultados efectuados com tamanhos diferentes no tamanho do *buffer* de compressão).

Outra hipótese seria modificar o algoritmo de compressão por forma a definir um limite no tamanho dos blocos auxiliares de descompressão. Assim, seria possível comprimir apenas blocos de tamanho fixo, ao invés do algoritmo utilizado, em que os blocos eram fixos relativamente ao tamanho depois de comprimido. Isto implicava que, no momento da descompressão soubéssemos qual o bloco a descomprimir, o que é simples utilizando um cabeçalho com um apontador para o início do bloco e o seu tamanho.

Esta solução parecia, assim, viável. Porém, outro problema surgia no momento de guardar as alterações. Era necessário comprimir novamente o bloco na sua totalidade. Este novo bloco poderia, quando descomprimido, ter um tamanho diferente (superior ou inferior) relativamente ao tamanho definido para a compressão. Poderia, mesmo, não haver memória suficiente para armazenar o bloco, para além do facto de ser necessário efectuar operações de redimensionamento no arquivo de armazenamento.

[1] Principalmente no que toca à quantidade de memória necessária.

Outro problema impede a utilização deste algoritmo. Visto que os dados são georeferenciados, é conveniente ter já disponíveis (i.e. descomprimidos em memória) os dados referentes ao pedaço de mapa apresentado no momento. Porém, esses pedaços poderiam encontrar-se armazenados em mais que um bloco, o que implicaria muitas operações de leitura e escrita.

Posto isto, parecia ser pouco apropriada a utilização do ficheiro GML, sobre que forma fosse, directamente no Sistema Móvel. Também, a recomendação do OGC a respeito deste assunto é para utilizar o GML apenas como plataforma de transferência de dados e não para manipulação directa [Cox et al. 2001].

Ainda durante esta análise, pensou-se na hipótese de definir um conjunto de rectângulos a ser comprimidos, em diversos níveis de zoom, utilizando as ideias gerais apresentadas por Yu e DeWitt [1996]. (ver Figura 4.1).

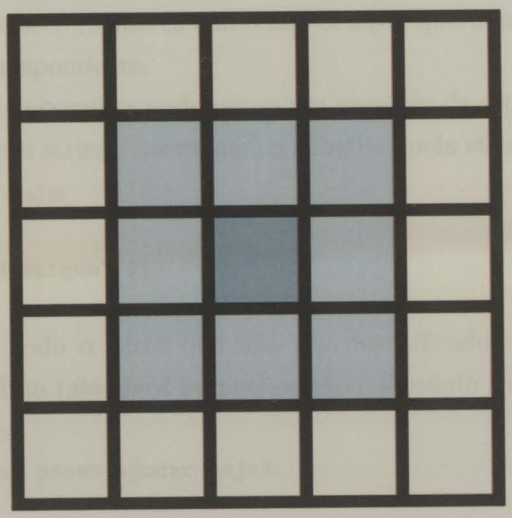

Figura 4.1: Exemplo da Utilização de Rectângulos

Na Figura 4.1 apresenta-se com cor mais escura o fragmento do mapa visualizado no momento (neste caso o quadrado central). Com um cinzento mais claro estão os fragmentos carregados em memória para uma rápida movimentação através do mapa. Os restantes quadrados são carregados conforme sejam necessários (i.e. durante o tempo inactivo do sistema). Em teoria, é possível obter uma representação bastante rápida da cartografia, visto que em memória já se encontram armazenadas os fragmentos correspondentes à sequência do mapa (consequentemente com mais probabilidade de ser visualizada) — este sistema era bastante utilizado nos longínquos tempos do ZX Spectrum. Não foi possível, porém, testar neste trabalho (principalmente devido a restrições de tempo).

Visto que a aplicação Mordomo utilizava um formato definido, foi necessário desenvolver um módulo que transformasse as coordenadas geométricas em GML para esse formato em Palm.

O código de suporte a este módulo foi desenvolvido com recurso às ferramentas FLEX e

BISON, tendo sido desenvolvida uma gramática reduzida para obter as "palavras" relevantes.

4.1.1.1 FLEX

Flex é "uma ferramenta para geração de aplicações que executam pesquisa de padrões sobre texto" [Paxson 1995]. As aplicações geradas por esta ferramenta, são denominadas de analizadores léxicos (*lexical scanners*): programas que reconhecem padrões léxicos sobre texto.

Dada uma sequência de texto[2], o *scanner* processa este texto procurando padrões representados sob a forma de expressões regulares e executando acções definidas sobre estas expressões. A aplicação gera código `C/C++` que é posteriormente compilado por forma a gerar uma aplicação executável.

Quando a aplicação resultante é executada, a entrada (*input*) é analisada, sendo pesquisadas ocorrências das expressões regulares definidas. Sempre que uma expressão é encontrada, é executado o código correspondente.

No fragmento de código seguinte pode ver-se um exemplo da especificação de um *scanner* que, sempre que encontra a string "username" a substitui pela string "Henrique"[3].

43a ⟨*Flex Code Simple Sample* 43a⟩≡

```
%%
username   printf( "Henrique" );
```

Como, por omissão, todo o texto que não seja identificado como expressão regular é enviado para a saída padrão (*standard output*), então o seguinte fragmento de código

43b ⟨*Ficheiro de Exemplo* 43b⟩≡

```
Bom dia username. Como posso ajudar hoje?
```

quando interpretado pelo scanner anterior, daria o resultado seguinte:

43c ⟨*Resultado para o Ficheiro de Exemplo* 43c⟩≡

```
Bom dia Henrique. Como posso ajudar hoje?
```

4.1.1.2 BISON

`Bison` é um "gerador de *parsers* que converte uma gramática `LALR(1)` num programa em `C` que efectua a verificação (*parse*) dessa gramática" [Donnely e Stallman 1995].

A ferramenta utiliza o ficheiro gerado pelo `FLEX`, chamando o código `C` do analizador léxico (i.e. sempre que um *token* seja necessário).

[2] Quer sobre a forma de ficheiros, ou como caracteres lidos sobre a entrada padrão (*standard input*).
[3] Os caracteres `%%` marcam o início das regras.

4.1.1.3 Utilização de FLEX & BISON

Uma primeira utilização destas ferramentas e a análise da norma GML 1.0 permitiu identificar um conjunto de expressões, nomeadamente <CList>, <BoundingBox> e <Feature> de onde eram obtidas as coordenadas geográficas. Estas coordenadas eram armazenadas internamente, sendo posteriormente enviadas para a base de dados em formato Palm OS do Mordomo[4].

Os fragmentos seguintes apresentam a gramática inicialmente identificada que tem relação directa com a representação de cartografia (i.e. o mapa a ser visualizado no Palm).

44a ⟨*Marcas GML 1 de Início* 44a⟩≡ (44c)

```
"<BoundingBox>"          { PushToken(INIBOUNDS); }
"<Feature>"              { PushToken(INIPOLYLINE); BEGIN featureON; }
"<Feature "[^>]*">"      { PushString(INIPOLYLINESTR); BEGIN featureON; }
"<CList>"                { }
<boundBoxON>"<CList>"    { PushToken(INIBOUNDS);   }
<featureON>"<CList>"     { PushToken(INIPOLYLINE); }
```

44b ⟨*Marcas GML 1 de Fim* 44b⟩≡ (44c)

```
"</BoundingBox>"         { PushToken(ENDBOUNDS); }
"</Feature>"             { PushToken(ENDPOLYLINE); BEGIN 0; }
"</CList>"               { }
{duplo}","{duplo}        { PushPonto(PontoFull); }

<boundBoxON>"</CList>"   { PushToken(ENDBOUNDS);   }
<featureON>"</CList>"    { PushToken(ENDPOLYLINE); }
```

44c ⟨*Marcas GML 1* 44c⟩≡ (46b)

⟨*Marcas GML 1 de Início* 44a⟩
⟨*Marcas GML 1 de Fim* 44b⟩

```
<featureON>{duplo}       { BEGIN gotX; PushString(PontoXDbl); }
<gotX>,{duplo}           { PushString(PontoYDbl); BEGIN 0; }
{palavra}                { discard; }
```

Um documento em GML 1.0 consiste num conjunto de pontos que definem o rectângulo que delimita o mapa (*bounding box*) e um conjunto de *features* (Características).

44d ⟨*Estrutura do Documento* 44d⟩≡ (46b)

```
program: file { ImprimeMapa(); } ;
file: TheBounds TheFeatures
  | TheFeatures
  ;
```

[4]Nesta altura a única preocupação eram as coordenadas geográficas, por forma a obter um primeiro mapa visualizável no Palm, sem preocupação com os meta-dados.

O rectângulo que delimita o mapa corresponde aos limites representados no mapa.

45 ⟨*Estrutura da Bounding Box* 45⟩≡ (46b)

```
TheBounds: INIBOUNDS ConjuntoBounds ENDBOUNDS
  | INIBOUNDS ENDBOUNDS
  ;
ConjuntoBounds: Pontos
  | ConjuntoBounds Pontos
    { $$ = stringConcatenate($1, $2); }
  ;
Pontos: PontoFull
  { $$ = $1; PoePBound($1); }
      | PontoX PontoY
  { $$ = stringConcatenate($1, $2);
    PoeBounds($1, $2); free($1); free($2);
  }    | PontoX PontoYDbl
  { $$ = stringConcatenate($1, $2);
    PoeBounds($1, $2); free($1); free($2);
  }    | PontoXDbl PontoY
  { $$ = stringConcatenate($1, $2);
    PoeBounds($1, $2); free($1); free($2);
  }    | PontoXDbl PontoYDbl
  { $$ = stringConcatenate($1, $2);
    PoeBounds($1, $2); free($1); free($2);
  }
  ;
```

Na Figura 4.2 pode ver-se um fragmento do mapa do Porto, tal como é disponibilizado no Palm pela aplicação Mordomo.

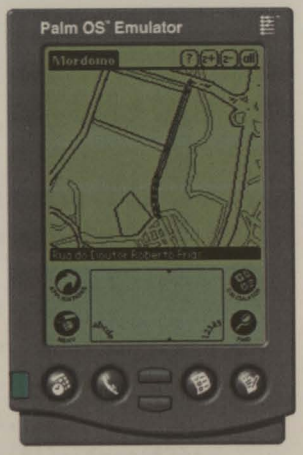

Figura 4.2: Fragmento de Mapa do Porto da aplicação Mordomo

Cada uma das Características é composta, entre outras marcas, de um conjunto de pontos.

46a ⟨*Estrutura das Features* 46a⟩≡ (46b)

```
TheFeatures: ListaDePoligonos
  ;
ListaDePoligonos: PoligonoX
  | ListaDePoligonos PoligonoX
  ;
PoligonoX: PoliStr ConjuntoDePontos ENDPOLYLINE
  ;
PoliStr: INIPOLYLINE
  { InserePoligono("fid=0"); }
      | INIPOLYLINESTR
  { InserePoligono($1); }
  ;
ConjuntoDePontos: PontosF
  | ConjuntoDePontos PontosF
    { $$ = stringConcatenate($1, $2); }
  ;
```

Os pontos podem estar representados pelas marcas <Coord> ou pela marca <Coordinates>. No primeiro caso é necessário identificar o valor de cada ponto (<X> e <Y>), enquanto que no segundo os pontos encontram-se em sequência no formato X0,Y0 X1,Y1 ... Xn,Yn.

46b ⟨*Gramática GML* 46b⟩≡

```
⟨Marcas GML 1 44c⟩
⟨Estrutura do Documento 44d⟩
⟨Estrutura da Bounding Box 45⟩
⟨Estrutura das Features 46a⟩
PontosF: PontoFull
  { $$ = $1; PoePonto($1); }
      | PontoX PontoY
  { $$ = stringConcatenate($1, $2);
    InserePonto($1, $2); free($1); free($2);
  }    | PontoX PontoYDbl
  { $$ = stringConcatenate($1, $2);
    InserePonto($1, $2); free($1); free($2);
  }    | PontoXDbl PontoY
  { $$ = stringConcatenate($1, $2);
    InserePonto($1, $2); free($1); free($2);
  }    | PontoXDbl PontoYDbl
  { $$ = stringConcatenate($1, $2);
    InserePonto($1, $2); free($1); free($2);
  }
  ;
```

Isto permitiu definir em memória uma estrutura contendo os pontos da cartografia. A função `ImprimeMapa()`, chamada no fim do documento, enviava a estrutura para o formato necessário para a aplicação Mordomo, através da utilização de funções criadas pela ParadigmaXis. Obtivemos assim um primeiro mapa GML visualizado no Palm.

Quando foi disponibilizada a público a versão 2.0 da norma GML, e após uma análise cuidada, verificou-se que haveria vantagem na transformação do código existente na altura para a nova versão. Porém, devido a esta versão ter características mais próximas do XML, e existindo uma grande variedade de *parsers* para esta linguagem, em vez de se implementar um específico, decidiu-se aproveitar um dos que já existiam[5].

4.1.2 Conversão em GML 2.0

Após algumas pesquisas efectuadas a modelos de *parsing* sobre documentos XML, foram identificadas e avaliadas duas metodologias distintas: *Simple API for XML* (SAX[6]) e *Document Object Model* (DOM[7]). Para uma discussão sobre a diferença entre as duas metodologias, veja-se Idris [1999].

Diversas ferramentas foram encontradas, embora duas destas tivessem desde logo chamado a atenção. Essas ferramentas são Xerces e `GeoTools` e serão analisadas nas próximas secções.

4.1.2.1 Xerces

A ferramenta `Xerces` é um *parser* de XML com funções de validação [The Apache Software Foundation 2001]. Existem implementações em `C++` e `Java`. Utilizando as classes definidas pela ferramenta, é fácil para qualquer programa ler e escrever dados em XML. No caso da ferramenta desenvolvida para `C++`, utiliza-se uma biblioteca de funções para efectuar *parsing*, geração, manipulação e validação de documentos XML.

As implementações mais recentes do Xerces (versão 1.5.1 para `C++` e versão 1.4.2 para `Java`) permitem a validação de documentos compatíveis com a recomendação XML 1.0[8] e normas associadas (DOM 1.0[9], DOM 2.0[10], SAX 1.0[11], SAX 2.0[12], Namespaces[13]). Também permite uma implementação (embora limitada), de um subconjunto das recomendações de esquemas (Schema[14]).

O *parser*, aparentemente, consegue aliar a velocidade à modularidade e parece ser facilmente expansível através de herança de código (tanto o `C++` como o `Java` são linguagens

[5]Ou seja, optou-se po abandonar Flex e Bison, para ser utilizada uma nova ferramenta:
[6]http://www.megginson.com/SAX/index.html
[7]http://www.w3.org/TR/REC-DOM-Level-1/
[8]http://www.w3.org/TR/REC-xml
[9]http://www.w3.org/TR/REC-DOM-Level-1/
[10]http://www.w3.org/TR/DOM-Level-2-Core/
[11]http://www.megginson.com/SAX/SAX1/index.html
[12]http://www.megginson.com/SAX/index.html
[13]http://www.w3.org/TR/REC-xml-names/
[14]http://www.w3.org/TR/xmlschema-0/

Orientadas a Objectos). Em grande parte devido à existência de código fonte em formato *Open Source*, com a existência de diversos exemplos de utilização e uma documentação em formato HTML, é fácil para um novo utilizador familiarizar-se rapidamente com a sintaxe.

Por forma a permitir a fácil reutilização em diversos Sistemas Operativos, a ferramenta em `C++` (existem binários compilados para Win32, Linux RedHat 6.1, Solaris 2.6, AIX 4.3 e HP-UX 11, sendo possível compilar para outros Sistemas), utiliza o mínimo de funcionalidades não padrão, como sejam *templates*, RTTI, *namespaces*, e o máximo de funcionalidades de pré-processamento (através da cláusula `#ifdef`).

Graças às suas funcionalidades de geração e validação, este *parser* pode ser utilizado para [The Apache Software Foundation 2001]:

1. criar Servidores Web "conhecedores" de XML;
2. criar Aplicações com dados no formato XML;
3. validar, no momento, documentos XML (para criação de editores de XML);
4. garantir a integridade de dados de *e-business* expressos em XML;
5. criar aplicações XML localizáveis.

Como se verifica do anteriormente referido, e visto que a norma GML se baseia em documentos XML, este *parser* será uma mais valia para confirmar a validade dos testes efectuados. Por outro lado, uma ferramenta equivalente poderá ser utilizada para o sistema de transformação entre os formatos (nativo da aplicação desenvolvida para Palm e GML a armazenar no Servidor para utilização com outras ferramentas).

4.1.2.2 GeoTools

As GeoTools são uma biblioteca de classes Java (desenvolvidas pelo Centro para Geometria Computacional – Center for Computational Geography – da Escola de Geografia da Universidade de Leeds) com o objectivo de desenvolver *applets* e aplicações interactivas com funcionalidades geográficas [Center for Computational Geography 2001].

Neste momento a aplicação permite utilizar os seguintes formatos (ou está a ser ponderada a sua utilização):

ESRI Shapefiles Permite leitura e escrita. Para construir estes ficheiros utiliza-se o objecto `ShapefileReader`. Embora o formato permita diversos tipos de características (pontos, linhas, polígonos, etc), um ficheiro apenas pode conter um tipo de característica.

Arcview Permite a geração de ficheiros neste formato e no formato `GRID`. Suporte para o formato `E00` ainda não está implementado.

Map/Info Ainda está em início de desenvolvimento o suporte para o formato MIF 300 (*Mapinfo Interchange File*)[15].

[15]Para mais informação sobre este formato, ver www.mapinfo.com.

GML Ainda não há qualquer tipo de suporte para ficheiros GML (segundo o autor, a norma ainda não se encontraria terminada[16]).

xBase (dbf) Existe suporte para leitura e escrita em ficheiros `dbf`. A escrita neste formato já está funcional, porém a leitura ainda não está estável.

geoTIFF Ainda não existe suporte, embora esteja identificado como pertinente e se espere para breve.

TIFF Espera-se que a implementação de operações de leitura e escrita em ficheiros `TIFF` combinados com `twf` da ESRI esteja concluída em breve.

Como se pode verificar, este trabalho ainda se encontra numa fase bastante inicial, o que impede, excepto à custa de bastante esforço, a utilização desta ferramenta. Nomeadamente, tendo sido definido como de grande importância para este trabalho a utilização do formato GML para a transformação de dados, e como esta ferramenta não tem ainda suporte de nenhum tipo para o mesmo, foi considerada como de pouca utilidade (embora provavelmente possa vir a ser utilizada no futuro para permitir a utilização de outros formatos).

Na altura em que ia começar a ser realizada a implementação do novo sistema de conversão, descobriu-se uma nova ferramenta desenvolvida em Java (GML4Java[17]), baseada na aplicação `Xerces` para Java, que gera uma árvore DOM a partir de um documento GML.

4.1.2.3 GML4Java

O GML4Java, desenvolvido pela *Galdos Inc*[18], obtém uma árvore `DOM`, a partir do *parsing* de um ou mais documentos `XML`, verifica se este(s) contêm dados geográficos (segundo a norma GML 2.0) e cria uma nova árvore com uma transformação das marcas existentes nos ficheiros originais, para as características definidas pela norma.

Assim, as classes existentes nesta aplicação têm um mapeamento directo entre o Modelo Abstrato definido pelo OGC e as características existentes nos ficheiros GML.

4.1.2.4 Utilização de GML4Java

Graças a esta ferramenta, foi possível criar um *parser* que, lendo um ou mais documentos GML criava a árvore DOM específica. Com essa árvore criada, a tarefa de criação das bases de dados para Palm tornou-se relativamente simples, bastando para tal percorrer os nós da árvore e, de acordo com o seu tipo, chamar a função que permite transformar os dados para o formato em Palm.

[16]Porém, a verdade é que o OGC já se comprometeu a respeitar a compatibilidade de versões, pelo que o trabalho poderia já ser realizado.

[17]http://gml4java.sourceforge.com/

[18]http://www.galdosinc.com/

Por exemplo, para a obtenção de dados relativos a Coordenadas Geométricas, temos

50a ⟨*Obter as Coordenadas Geométricas* 50a⟩≡

```
if (currDoc.isGeometry(desc.getNamespace(),
                       desc.getLocalName()))
   Escreve((Geometry) construct);
```

Uses `Escreve()` 50b.

em que `currDoc` representa a árvore GML processada previamente e `desc` é o descriptor XML do nó actual. Isto é efectuado para cada tipo específico de característica.

A função `Escreve()`, apresentada no fragmento de código seguinte, processa cada uma das Coordenadas existentes no nó, adicionando a uma estrutura interna que, posteriormente, será serializada para o ficheiro PDB através de uma função que percorre as propriedades da característica.

50b ⟨*Processar Coordenadas Geométricas* 50b⟩≡

```
private static void Escreve(Geometry g) {
String gid = g.getId();
GeometryIterator iBoundIt = g.getInnerBoundaryIterator();
   Escreve(iBoundIt);
CoordinateTupleIterator coordIt = g.getCoordinateTupleIterator();
   Escreve(gid, coordIt);
PropertyIterator pi = g.getPropertyIterator();
   Escreve(gid, pi);
}
```

Defines:
`Escreve()`, used in chunk 50a.

A aplicação resultante da conjugação e análise de cada uma das marcas identificadas pelo *parser* gera três bases de dados PALM. Assim, os dados correspondentes a componentes geométricas (ou seja, os pontos), são inseridos na base de dados do sistema de apresentação de mapas (no caso actual o sistema Mordomo — `mordomo.pdb`); os dados relativos a Pontos de Interesse são armazenados para utilização do sistema Mordomo — `pontosinteresse.pdb`; todos os dados restantes são inseridos na base de dados da aplicação MetaGeo — `metageo.pdb` (ver Figuras 3.3 e 3.2).

Esta divisão revelou-se necessária por forma a manter a compatibilidade com a aplicação Mordomo, já existente. Devido a este sistema ter sido desenvolvido para permitir apenas a visualização e pesquisa sobre dados georreferenciados, a base de dados foi criada sem necessidade de ser actualizada. Porém, visto que o principal objectivo deste trabalho é permitir a actualização de dados georreferenciados, torna-se necessário que a estrutura interna permita essas operações.

A base de dados de pontos de interesse é constituída por (em termos de estrutura):

51 ⟨*Estrutura da Base de Dados de Pontos de Interesse* 51⟩≡

```
char header[60]; // Constituído por 60 zeros
char Identif[8] = '0004MORD'
char zeros[8]; // Constituído por 8 zeros
char NumRegistos = 7;
double datas[6]; (seis datas)

struct Limites {
   double x0;
   double y0;
   double width;
   double height;
} r0; //Limites do Mapa representado
short n1; //Número de Pontos de Interesse
char *Descricoes[n1]; //Descrição associada a cada Ponto de Interesse
struct Coord {
   short x;
   short y;
} coordenadas[n1]; //Coordenadas de cada Ponto de Interesse
short n2; //Categorias existentes
char *Categorias[n2]; //Descrição das Categorias
short PrimeiroPonto[n2]; //Índice do 1º ponto de interesse de cada Categoria
```

O primeiro registo (`r0`) contém os limites do mapa. `(x0, y0)` representa o canto superior esquerdo, `width` o "comprimento" do mapa e `height` a "altura" do mapa.

O segundo registo (`n1`) indica quantos pontos de interesse existem. Os pontos de interesse estão agrupados por categorias, ou seja todos os locais de interesse de uma dada categoria aparecem juntos.

O terceiro registo (`Descricoes`) contém as designações de todos os locais de interesse. Cada designação é uma sequência de caracteres terminada por um byte com o valor 0 (ou seja é uma *string* `C`). O tamanho deste registo é variável, contendo `n1` designações.

O quarto registo (`coordenadas`) armazena as coordenadas de todos os locais de interesse. Assume-se que cada local de interesse é representado por um ponto apenas. As coordenadas são codificadas convertendo os pontos (indicados como *floats*) para *shorts* utilizando os limites. Cada ponto ocupa dois *shorts*, corespondendo o primeiro à coordenada `X`, e o segundo à coordenada `Y`. Este registo tem `n1` pontos.

O quinto registo (`n2`) contém um *short* com o número de categorias.

O sexto registo (`Categorias`) contém `n2` *strings*, cada uma terminada por um byte com valor 0, representando o nome de todas as categorias. As categorias encontram-se ordenadas alfabeticamente.

O sétimo registo (`PrimeiroPonto`) contém o índice do primeiro local de interesse de cada categoria (ou seja, contém `n2` valores). É por isso que os pontos de interesse têm de se encontrar agrupados por categorias (registos `Descricoes` e `coordenadas`). O índice começa em 1 e corresponde à ordem em que os locais de interesse foram armazenados no ficheiro.

Assim, em vez de reformular a aplicação Mordomo original, criou-se uma nova estrutura de dados, já preparada para actualização e sincronização, que será a única sobre a qual se efectuarão operações de modificação, como veremos no Capítulo 6.1.

A base de dados criada para este trabalho é pois bastante mais complexa. O facto de existir actualização de dados, implicou a necessidade de desenvolver um "sistema relacional", embora bastante reduzido em termos de funcionalidade.

O PalmOS permite a utilização de um conjunto de Categorias nas suas Bases de Dados. Estas categorias são geralmente utilizadas mediante o acesso a uma lista, que permite especificar a categoria de um determinado registo. Por exemplo, uma aplicação de E-Mail permite a utilização das categorias de Correio Recebido — *In-box*, Correio Enviado — *Out-box*, entre outras.

Visto que neste caso não há necessidade de utilizar as categorias existentes, estas foram utilizadas para simular o uso de tabelas. Assim, após análise dos requisitos a cumprir pela aplicação, identificaram-se 6 tabelas fundamentais (ver Figura 4.3).

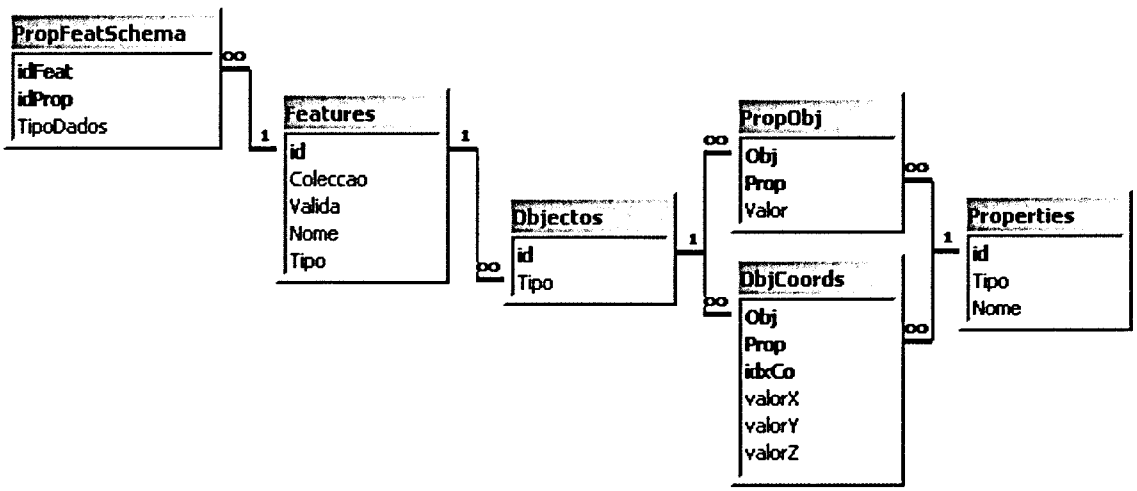

Figura 4.3: Relacionamentos entre as Tabelas do MetaGEO

Foi efectuada a seguinte correspondência entre Categorias:

Features Categoria 0 — representa as Características armazenadas na base de dados;

Properties Categoria 1 — indica as propriedades existentes no esquema original;

Objectos Categoria 2 — contem os objectos definidos;

PropObj Categoria 3 — contem os registos que relacionam os objectos e as propriedades;

ObjCoords Categoria 4 — contem as coordenadas geométricas que permitem a visualização de cada objecto;

PropFeatSchema Categoria 5 — neste momento não é utilizada, porém irá permitir a inclusão de mais dados que, neste momento não são importantes;

Como exemplo, consideremos o seguinte fragmento de código, que poderia corresponder a um documento GML:

53a ⟨*Exemplo de Feature num Documento GML* 53a⟩≡

```
<schoolMember>
   <School>
      <gml:name>Beta</gml:name>
      <address>1673 Balsam St.</address>
      <gml:location>
         <gml:Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
            <gml:coord>
               <gml:X>40.0</gml:X>
               <gml:Y>5.0</gml:Y>
            </gml:coord>
         </gml:Point>
      </gml:location>
   </School>
</schoolMember>
```

Após o processamento por parte do módulo GML2PDB, teríamos na base de dados do MetaGEO os seguintes registos[19], ver Figura 4.3:

53b ⟨*Conteúdos dos registos MetaGEO* 53b⟩≡

```
feat[0] = {1, false, true, "School", "schoolMember"};
prop[0] = {1, 1, "gml:name"};
prop[1] = {2, 1, "address"};
prop[2] = {3, 3, "gml:location"};
prop[2] = {4, 200, "gml:coord"};
obj[0] = {1, 1};
propObj[0] = {1, 1, "Beta"};
propObj[1] = {1, 2, "1673 Balsam St."};
objCoords[0] = {1, 4, 0, 40.0, 5.0, 0.0};
```

[19]Um Tipo de Propriedade 200 indica uma coordenada.

4.2 Sumário

Neste capítulo apresentou-se o método desenvolvido para efectuar a criação de três bases de dados Palm (PDBs) com base num documento GML.

Foram desenvolvidos dois processos distintos, um deles aplicável a documentos elaborados segundo a norma GML 1.0, e utilizando as ferramentas FLEX e BISON. Para utilização com documentos compatíveis com GML 2.0 preferiu-se a utilização de uma ferramenta vocacionada para efectuar a análise, o GML4Java.

Esta ferramenta, apesar de ainda não estar concluída, já permite efectuar as operações principais sobre documentos GML. Para a transformação entre formatos, ficou já provada a sua utilidade.

Agora que temos um mapa num formato que pode ser utilizado pela plataforma móvel, e com um mapa já visualizado, é necessário desenvolver a aplicação central, que permite que se efectue a manutenção dos dados integrados no Palm.

Essa aplicação vai ser apresentada no próximo capítulo.

Capítulo 5

Manipulação de Dados na Plataforma Móvel

Após a criação dos três ficheiros representativos das bases de dados (pdb) e o seu envio para o Palm, por sincronização, a aplicação a desenvolver deve efectuar sobre eles as tarefas normais de manutenção (eliminação de registos, inserção de registos, alteração de dados, pesquisa sobre os dados).

Para que tal seja possível, é necessária a existência de uma aplicação, a funcionar no Palm, que suporte essas operações.

5.1 Desenvolvimento para Sistemas Móveis

O desenvolvimento de aplicações para dispositivos de computação móvel não é tão simples como à primeira vista possa parecer. Devido às restrições da plataforma apresentadas na secção 3.3, é necessário levar em linha de conta um conjunto de factores que não são relevantes quando se desenvolve para sistemas de médio/grande porte.

A quantidade de memória disponível é reduzida, o que obriga a optimizar as aplicações desenvolvidas, relativamente ao tamanho de memória ocupado. No entanto, e visto que estes dispositivos já são lentos[1], é necessário optimizar também relativamente à velocidade de execução. As ferramentas de desenvolvimento permitem a optimização para um destes items, perdendo capacidades relativamente a um, quando se aumenta o grau do outro (i.e. se optimizarem para velocidade a 100% têm pouca optimização para tamanho e vice versa), pelo que algumas das optimizações têm de ser efectuadas no momento da programação[2].

Devido às variáveis envolvidas com a ligação existente, é necessário que as tarefas que necessitam de um meio fiável sejam realizadas no menor tempo possível, aproveitando ao

[1] Quando comparados com os sistemas "normais".

[2] Por esse motivo a linguagem mais utilizada é o C, que permite gerar um código mais eficiente, seguida pelo Java devido à popularidade e (quase) independência da plataforma.

máximo o meio existente no momento (por exemplo na sincronização dos dados). Assim, os algoritmos de sincronização necessitam de ao obter o registo das modificações efectuar imediatamente esse registo no sistema anfitrião, aproveitando a ligação existente no momento.

Visto que a fonte de energia é finita (i.e. pode-se ficar sem energia a meio do trabalho), é necessário que os sistemas implementem operações que evitem a perda de dados em caso de perda de energia (por exemplo efectuando gravação de dados logo que possível, em vez de utilizar um *buffer* de armazenamento temporário, como é efectuado pelos sistemas pessoais).

Desenvolver para um sistema móvel apresenta ainda outras dificuldades. Se é verdade que existem ferramentas capazes de emular uma sessão no sistema móvel (como por exemplo o Emulador de Palm que pode ser usado no sistema operativo Windows), estes emuladores não têm uma compatibilidade a 100%, implicando a reformulações no código após os primeiros testes no dispositivo.

Também, no acto de sincronizar dados, é necessário efectuar testes no dispositivo, e não no Emulador, visto que é necessário testar a ligação através de uma porta série.

5.2 Módulo para manutenção de dados georeferenciados

Após a criação das bases de dados para o sistema Palm, foi necessário desenvolver a aplicação fundamental desta dissertação, cujo objectivo era efectuar a manutenção dos dados georreferenciados.

Após algumas discussões relativamente a aplicações práticas, foi acordado que o caso de estudo a aplicar seria sobre pontos de interesse numa cidade. Estes pontos de interesse, a ser desenvolvidos sobre um mapa do Porto já existente na aplicação Mordomo, seriam implementados para um projecto desenvolvido em cooperação entre o Instituto Superior da Maia e a empresa ParadigmaXis.

Devido a restrições temporais relacionados com a execução do projecto, a aplicação prática seria efectuada por outro sistema que a ParadigmaXis estava já a desenvolver, sendo o resultado deste trabalho de Mestrado aplicado posteriormente, caso se verificasse necessidade de modificação. O formato desenvolvido pela ParadigmaXis não considera tarefas de actualização sobre os pontos de interesse, sendo necessária a gestão nos servidores de dados, o que impede actualizações directamente nos formatos utilizados.

A *interface* a desenvolver para esta aplicação é relativamente simples (como é recomendado pela Palm, Inc.), embora permita aceder a algumas funcionalidades importantes. Na Figura 5.1 podemos ver um exemplo da utilização desta aplicação. A visualização do Mapa pode ser vista na Figura 4.2.

Como foi referido em 4.1.2.4 na página 53, de acordo com o valor do atributo do registo actual, é possível saber qual a estrutura do conteúdo. Torna-se fácil percorrer o conteúdo dos registos de uma categoria utilizando a função `DmQueryNextInCategory(...)` existente no

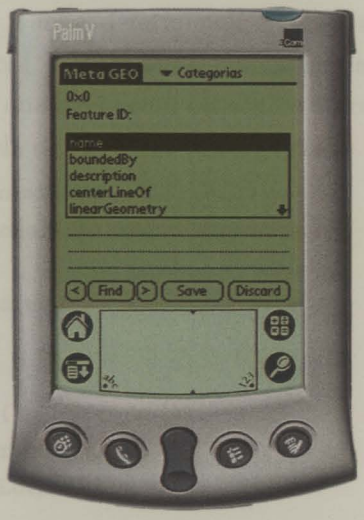

Figura 5.1: Ecrã Principal do MetaGeo

Palm. No fragmento seguinte podemos ver como obter o registo seguinte de uma determinada categoria (i.e. uma tabela tal como foi delineado para esta aplicação).

57 ⟨*Generic Table: Querying a Record* 57⟩≡

```
void Table::setCurrent() {
   MemHandle theRecord;
   void *sr;
   if ((theRecord = DmQueryNextInCategory(db, &idx, categoria)) != NULL) {
      sr = (void *) MemHandleLock(theRecord);
      thisHandle = theRecord;
      m_EOF = m_BOF = false;
   } else
      m_EOF = m_BOF = false;
}
```

Para adicionar um registo a uma determinada categoria, temos que, após adicionar o registo usando `DmAttachRecord(...)`, modificar a categoria. Isto é conseguido através da função `DmSetRecordInfo(..)` que modifica o cabeçalho do registo a modificar. Podemos ver a seguir o exemplo para a tabela de Objectos (`categoria = 2`).

58 ⟨*Generic Table: Inserting new Records* 58⟩≡

```
UInt16 Table::Insere(DmOpenRef db) {
    void *sr;
    UInt16 idx = dmMaxRecordIndex;
    MemHandle newRecord;
    UInt16 attr;
    newRecord = DmNewHandle(db, Tamanho());
    if (newRecord == NULL) return 0;
    sr = (void *) MemHandleLock(newRecord);
    PackRecord(sr);
    MemPtrUnlock(sr);
    DmAttachRecord(db, &idx, newRecord, 0);
    // Aplica a Categoria do Registo (Objecto)
    DmRecordInfo (db, idx, &attr, NULL, NULL);
    attr &= ~dmRecAttrCategoryMask;
    attr |= categoria | dmRecAttrDirty;
    DmSetRecordInfo (db, idx, &attr, NULL);
    return idx; // A posição do registo inserido
}
```

5.3 Sumário

Com o desenvolvimento do módulo descrito neste capítulo, o objectivo principal deste trabalho ficou cumprido. Porém, e visto que para efectuar a manutenção dos dados não estamos a utilizar um formato que seja compatível com os SIG existentes, torna-se necessário reconverter os dados para o formato padrão seleccionado (GML) durante a operação de sincronização.

O processo pelo qual esta transformação é efectuada apresenta-se no capítulo seguinte.

Capítulo 6

Sincronização dos Dados

Como já foi referido anteriormente, a sincronização de dados é uma tarefa fulcral em qualquer sistema para dispositivos de computação móvel. É a única forma existente de "enviar" os dados do dispositivo para um meio mais fiável e sem as restrições impostas a estes dispositivos. Claro que é bastante simples efectuar uma mera cópia dos dados existentes no Palm, mas as aplicações geralmente necessitam de mais que isso.

O investimento necessário para desenvolver um SIG para um computador pessoal que manipule directamente os dados vindos do Palm é uma tarefa complicada, que ultrapassa o âmbito deste trabalho. Assim, a hipótese de criar uma aplicação para um computador pessoal para efectuar a gestão dos dados no formato Palm também parece estar fora de questão. A melhor solução é transformar os dados existentes para um formato padrão (neste momento a maior parte das aplicações reconhece dados compatíveis com `XML`, tal como por exemplo `GML`), permitindo assim que grande número de aplicações possam aceder aos dados.

6.1 Conversão de formato nativo Palm para GML

Por forma a ser possível a transferência dos dados modificados para o computador pessoal onde se encontra armazenado o GML, é necessário reconverter os `PDB` existentes no Palm para o formato original (neste caso GML). Assim, foi necessário desenvolver uma aplicação que, lendo os ficheiros existentes no Palm, processasse uma árvore DOM criada através da análise dos registos modificados no Palm (mais concretamente uma árvore DOM de um GML), para ser posteriormente gravada no formato original (serializada).

Visto que, no âmbito do presente trabalho, as alterações são apenas efectuadas sobre registos de pontos de interesse, verificou-se que não havia necessidade de efectuar a geração do mapa em GML (já que este é estático). Assim, e por forma a ser possível a comparação entre o documento original e os dados modificados, preferiu-se convencionar a divisão entre o ficheiro GML com o mapa e o GML contendo os pontos de interesse.

Para obter as diferenças entre o GML original e os dados modificados, temos diversas

hipóteses, dependendo do local onde foram efectuadas alterações. Assim, o caso mais simples seria considerar que estas eram apenas efectuadas no Palm. O caso mais complexo seria quando muitas réplicas de diferentes origens de dados eram modificadas tanto em Palm como no sistema anfitrião (ou seja, no ficheiro GML original).

Consideremos novamente o exemplo iniciado na secção 4.1.2.4, na página 53.

Imagine-se agora que, no Palm, o atributo `address` de `School` com nome igual a `Beta` foi modificado para "1550 New Balsam St." e que foi eliminado o atributo `gml:name`. O resultado em GML seria:

60 ⟨*Feature Modificada no Palm* 60⟩≡

```
<schoolMember>
   <School>
      <address>1550 New Balsam St.</address>
      <gml:location>
         <gml:Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
            <gml:coord>
               <gml:X>40.0</gml:X>
               <gml:Y>5.0</gml:Y>
            </gml:coord>
         </gml:Point>
      </gml:location>
   </School>
</schoolMember>
```

Por forma a detectar a diferença, existem duas possibilidades:

1. gerar o GML através das bases de dados Palm, efectuando uma comparação entre documentos GML utilizando as técnicas referidas em Chen et al. [2000]; Wang et al. [2000];

2. obter apenas as modificações efectuadas sobre o PDB do Palm, e gerar o GML resultante da sua aplicação sobre o GML original, como defendido por Gossett [2000].

Enquanto a primeira tem a vantagem de identificar alterações efectuadas sobre o GML original, a verdade é que neste caso não é necessário que tal aconteça. Assim, a segunda possibilidade apenas apresenta as modificações efectuadas sobre o sistema a ser sincronizado num determinado momento, sendo efectuada uma "mistura" dos dados.

Graças às funcionalidades existentes na aplicação XML4Java, tal tarefa apresentou-se razoavelmente simples. O processo definido para criação do ficheiro GML consistia nos seguinte passos:

61a ⟨*Processo para criação de GML* 61a⟩≡

```
ArvoreModificados <- Nova ArvoreDOM
ArvoreEliminados <- Nova ArvoreDOM

⟨Obter as modificações efectuadas no Palm 61b⟩

⟨Criar a árvore com o GML original 62a⟩

// Finalmente, gravar a árvore GML resultante para um ficheiro
Serializar( ArvoreGML )
```

61b ⟨*Obter as modificações efectuadas no Palm* 61b⟩≡ (61a)

```
PARA CADA Objecto na Base de Dados
   NoObjecto <- NADA
   SE Objecto é Novo ou foi Modificado ENTAO
      NoObjecto <- ADICIONAR( ArvoreModificados, Nó , NADA, Verdade )
   SENAO SE Objecto foi Eliminado ENTAO
      NoObjecto <- ADICIONAR( ArvoreEliminados, Nó , NADA, Verdade )
   FIM SE
   PARA CADA Propriedade do Objecto
      SE Propriedade é Nova ou foi Modificada ENTAO
         SE NoObjecto É NADA ENTAO
            NoObjecto <- ADICIONAR( ArvoreModificados, Nó, NADA, Verdade )
         FIM SE
         ADICIONAR( ArvoreModificados, NoObjecto, Propriedade, Verdade )
      SENAO SE Propriedade foi Eliminada ENTAO
         SE NoObjecto É NADA ENTAO
            NoObjecto <- ADICIONAR( ArvoreEliminados, Nó, NADA, Falso )
         FIM SE
         ADICIONAR( ArvoreEliminados, NoObjecto, Propriedade, Verdade )
      FIM SE
   FIM
FIM
```

62a ⟨*Criar a árvore com o GML original* 62a⟩≡ (61a)

```
ArvoreGML <- Nova GML4Java.Document
// Eliminar os Nós que foram eliminados no Palm
PARA CADA No na ArvoreEliminados
   NoElim <- Procurar( ArvoreGML, No )
   SE NAO (NoElim é NADA) ENTAO
      ArvoreGML.Eliminar( No )
   FIM SE
FIM
```

⟨*Efectuar as Alterações nos nós modificados no Palm* 62b⟩

62b ⟨*Efectuar as Alterações nos nós modificados no Palm* 62b⟩≡ (62a)

```
PARA CADA No na ArvoreModificados
   NoModif <- Procurar( ArvoreGML, No )
   SE NoModif é NADA ENTAO
      ArvoreGML.Adicionar( No )
   SENAO
      NoModif <- No
   FIM SE
FIM
```

Para criar uma árvore DOM de um documento XML utilizando a aplicação XML4Java, usamos instruções simples, como se pode verificar a seguir:

63 ⟨*Criar Nova Árvore DOM* 63⟩≡

```
try {
   Document doc = new DocumentImpl();
   // Create Root Element
   Element root = doc.createElement("cidadePorto");
   // Create element
   Element item = doc.createElement("nome");
   item.appendChild( doc.createTextNode("Porto") );
   // atach element to Root element
   root.appendChild( item );
   // Create another Element
   item = doc.createElement("Distrito");
   item.appendChild( doc.createTextNode("Douro Litoral") );
   // Attach Element to previous element down tree
   root.appendChild( item );
   item = doc.createElement("População");
   item.appendChild( doc.createTextNode("800.000" ) );
   root.appendChild( item );
   // Add Root to Document
   doc.appendChild( root );
   //Serialize DOM
   OutputFormat    format  = new OutputFormat( doc );
   //Writer will be a String
   StringWriter  stringOut = new StringWriter();
   XMLSerializer    serial = new XMLSerializer( stringOut, format );
   // As a DOM Serializer
   serial.asDOMSerializer();
   serial.serialize( doc.getDocumentElement() );
   //Spit out DOM as a String
   System.out.println( "RESULTADO = " + stringOut.toString() );
} catch ( Exception ex ) {
   ex.printStackTrace();
}
```

6.1.1 Conduta para GML

Após a criação do sistema de conversão do formato interno Palm para o formato original GML, foi necessário desenvolver uma Conduta que efectuasse a cópia das bases de dados para o servidor, executasse a aplicação de geração do GML e posteriormente enviasse para o Palm a versão actualizada das bases de dados[1].

No caso actual, temos uma mistura entre os diversos formatos referidos na secção 3.4. Assim:

- a Base de Dados Geográfica (de uso pelo `Mordomo`) é sincronizada pelo método 3 (transferir para o Palm), visto que as modificações ao mapa são efectuadas no servidor;
- a Base de Dados de Pontos de Interesse é sincronizada pelo mesmo método;
- a Base de Dados de meta-dados deve ser sincronizada nos dois sentidos (método 1), embora no âmbito deste projecto a sincronização vá ser efectuada pelo método 2 (transferir do Palm).

Por forma a reaproveitar código, foram desenvolvidas três condutas, uma para cada Base de Dados, as quais foram posteriormente integradas numa conduta comum (do tipo adaptável), que integra as três primeiras.

6.2 Sumário

Neste momento estão criadas as aplicações necessárias para efectuar:

1. transformação de documentos GML para bases de dados PalmOS (GML2PDB);
2. manutenção de dados georreferenciados no Palm (MetaGeo)[2];
3. transformação dos dados modificados no Palm para um documento GML (PDB2GML).

Para que as transformações fossem simples de realizar, foi implementada uma Conduta que permite a comparação dos registos modificados no Palm com o documento GML original.

Assim, estão cumpridos os objectivos inicialmente propostos.

Avaliam-se, no capítulo seguinte, os resultados obtidos com o desenvolvimento destas três aplicações.

[1] Embora para este caso tal não seja necessário, visto que o único local onde os dados são modificados seja o Palm, optou-se por preparar o sistema para enviar possíveis alterações.

[2] Para a visualização utilizou-se uma aplicação desenvolvida pela ParadigmaXis — Mordomo.

Capítulo 7

Avaliação dos Resultados

Por forma a avaliar o sistema implementado era necessário garantir que os dados existentes no GML original eram apresentados e modificados no Palm. Porém, era também necessário obter um GML válido com o registo das alterações efectuadas que pudesse ser disponibilizado para outras aplicações (por exemplo através da Internet).

Devido à reduzida existência de mapas no formato GML, bem como de aplicações que efectuem operações sobre os mesmos, foi difícil avaliar a aplicação desenvolvida.

Ainda assim, alguns pontos fundamentais foram testados:

- criação dos PDB com base em exemplos de mapas GML
- modificação de dados no dispositivo móvel,
- experimentação num caso real,
- detecção das modificações efectuadas, e
- verificação sintática do ficheiro GML gerado durante a sincronização.

A seguir apresentam-se os resultados de cada um dos testes efectuados.

7.1 Criação dos PDB com base num GML real

Por forma a realizar os primeiros testes sobre a aplicação de criação dos PDB com base no GML, utilizaram-se os esquemas existentes no sítio do OGC, nomeadamente `cambridge.xml` e `schools.xml`.

Os primeiros testes confirmaram que os algoritmos utilizados inseriam correctamente os dados geométricos, bem como os meta-dados. No caso dos dados geométricos, foi efectuada uma comparação entre o mapa apresentado pela aplicação Mordomo no Palm e pelo applet do GeoTools[1], com base no documento cambridge.xml.

[1]Disponível em http://www.ccg.leeds.ac.uk/geotools/demos/gml/

65 ⟨*version/TrabalhoRealizado/cambridge.xml* 65⟩≡

```
⟨Cabeçalho do Documento GML 66a⟩
  <gml:name>Cambridge</gml:name>
  <gml:boundedBy>
    <gml:Box srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
      <gml:coord><gml:X>0.0</gml:X><gml:Y>0.0</gml:Y></gml:coord>
      <gml:coord><gml:X>100.0</gml:X><gml:Y>100.0</gml:Y></gml:coord>
    </gml:Box>
  </gml:boundedBy>

  ⟨Um rio no exemplo GML 66b⟩
  ⟨Uma estrada no exemplo GML 67a⟩
  <cityMember xlink:type="simple" xlink:title="Trinity Lane"
    xlink:href="http://www.foo.net/cgi-bin/wfs?FeatureID=C10239"
    gml:remoteSchema="city.xsd#xpointer(//complexType[@name='RoadType'])"/>
  ⟨A Montanha comentada que dá erro no exemplo GML 70⟩
  <dateCreated>2000-11</dateCreated>
</CityModel>
```

66a ⟨*Cabeçalho do Documento GML* 66a⟩≡ (65)

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- File: cambridge.xml -->
<CityModel xmlns="http://localhost/GML2/examples"
           xmlns:gml="http://localhost/GML2/gml"
           xmlns:xlink="http://www.w3.org/1999/xlink"
           xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
           xsi:schemaLocation="http://localhost/GML2/examples
             http://localhost/GML2/examples/city.xsd">
```

66b ⟨*Um rio no exemplo GML* 66b⟩≡ (65)

```
<cityMember>
  <River>
    <gml:description>The river that runs through Cambridge.</gml:description>
    <gml:name>Cam</gml:name>
    <gml:centerLineOf>
      <gml:LineString srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
        <gml:coord><gml:X>0</gml:X><gml:Y>50</gml:Y></gml:coord>
        <gml:coord><gml:X>70</gml:X><gml:Y>60</gml:Y></gml:coord>
        <gml:coord><gml:X>100</gml:X><gml:Y>50</gml:Y></gml:coord>
      </gml:LineString>
    </gml:centerLineOf>
  </River>
</cityMember>
```

67a ⟨*Uma estrada no exemplo GML* 67a⟩≡ (65)

```
<cityMember>
  <Road>
    <gml:name>M11</gml:name>
      <linearGeometry>
        <gml:LineString srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
          <gml:coord><gml:X>0</gml:X><gml:Y>5.0</gml:Y></gml:coord>
          <gml:coord><gml:X>20.6</gml:X><gml:Y>10.7</gml:Y></gml:coord>
          <gml:coord><gml:X>80.5</gml:X><gml:Y>60.9</gml:Y></gml:coord>
        </gml:LineString>
      </linearGeometry>
    <classification>motorway</classification>
    <number>11</number>
  </Road>
</cityMember>
```

O esquema de validação usado, como se pode ver pelo cabeçalho do documento apresentado no fragmento de código 66a, é o ficheiro http://localhost/GML2/examples/city.xsd, apresentado a seguir.

67b ⟨*version/TrabalhoRealizado/city.xsd* 67b⟩≡

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- File: city.xsd  -->
<schema targetNamespace="http://localhost/GML2/examples"
        xmlns:ex="http://localhost/GML2/examples"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:gml="http://localhost/GML2/gml"
        xmlns="http://www.w3.org/2000/10/XMLSchema"
        elementFormDefault="qualified"
        version="2.03">
  <annotation>
    <appinfo>city.xsd v2.03 2001-02</appinfo>
    <documentation xml:lang="en">
      GML schema for the Cambridge example
    </documentation>
  </annotation>
  ⟨import constructs from the GML Feature and Geometry schemas 68b⟩
  ⟨global element declarations 68a⟩
  ⟨type definitions for city model 69⟩
</schema>
```

68a ⟨*global element declarations* 68a⟩≡ (67b)

```
<element name="CityModel" type="ex:CityModelType"
   substitutionGroup="gml:_FeatureCollection" />
<element name="cityMember" type="ex:CityMemberType"
  substitutionGroup="gml:featureMember"/>
<element name="Road" type="ex:RoadType"
  substitutionGroup="ex:_CityFeature"/>
<element name="River" type="ex:RiverType"
  substitutionGroup="ex:_CityFeature"/>
<element name="Mountain" type="ex:MountainType"
  substitutionGroup="gml:_Feature"/>
<!-- a label for restricting membership in the CityModel collection -->
<element name="_CityFeature" type="gml:AbstractFeatureType" abstract="true"
  substitutionGroup="gml:_Feature"/>
```

68b ⟨*import constructs from the GML Feature and Geometry schemas* 68b⟩≡ (67b)

```
<import namespace="http://localhost/GML2/gml" schemaLocation="feature.xsd"/>
```

69 ⟨*type definitions for city model* 69⟩≡ (67b)

```
<complexType name="CityModelType">
  <complexContent>
    <extension base="gml:AbstractFeatureCollectionType">
      <sequence>
        <element name="dateCreated" type="month"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="CityMemberType">
  <annotation>
    <documentation>
      A cityMember is restricted to those features (or feature
      collections)that are declared equivalent to ex:_CityFeature.
    </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:FeatureAssociationType">
      <sequence minOccurs="0">
        <element ref="ex:_CityFeature"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<complexType name="RiverType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:centerLineOf"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="RoadType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="linearGeometry" type="gml:LineStringPropertyType"/>
        <element name="classification" type="string"/>
        <element name="number" type="string"/>
      </sequence>
```

```
    </extension>
  </complexContent>
</complexType>
<!-- this is just here to demonstrate feature member restriction -->
<complexType name="MountainType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="elevation" type="integer"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Porém, o documento tem uma característica escondida (ou seja comentada):

70 ⟨*A Montanha comentada que dá erro no exemplo GML* 70⟩≡ (65)

```
<!-- a mountain doesn't belong here! Uncomment this cityMember and see
     the parser complain!
<cityMember>
  <Mountain>
    <gml:description>World's highest mountain is in Nepal!</gml:description>
    <gml:name>Everest</gml:name>
    <elevation>8850</elevation>
  </Mountain>
</cityMember>
-->
```

Para validar as ferramentas de *parsing*, removeu-se o comentário da montanha; mas o módulo GML2PDB não se queixou e inseriu a montanha como sendo válida[2].

Tentou-se validar o documento noutro *parser* (Xerces). Também neste caso a montanha foi "digerida" como sendo válida no esquema.

Por último, experimentou-se com o *Internet Explorer 5.00.2920.0000*. Também este não se queixou, o que indica que a maioria dos *parsers* XML não efectuam uma validação semântica (ou seja, não suportam restrições do modelo), mas apenas sintática.

Foi possível detectar o erro, visto que o *parser* GML4Java indicava as características do objecto de uma forma anormal como sendo Unknown Construct.

71 ⟨*Everest is Unknown Construct* 71⟩≡

```
Unknown construct
   http://www.opengis.net/examples:Mountain=
      "                    "
   Unknown construct
      http://www.opengis.net/gml:description=
         "World's highest mountain is in Nepal!"
   Unknown construct
      http://www.opengis.net/gml:name=
         "Everest"
   Unknown construct
      http://www.opengis.net/examples:elevation=
         "8850"
```

Obtinham-se as construções inválidas para o esquema. Apesar de serem inválidas, optou-se por permitir a sua inclusão no Palm, indicando para o efeito no campo Valida da "tabela" Features o valor false, para permitir a sua utilização no futuro.

7.2 Modificação de dados no dispositivo

Os primeiros testes de modificação foram efectuados sobre um sistema reduzido (o documento cambridge.xml) apresentado na página 66. A interface implementada verificou-se adequada e a velocidade de pesquisa e actualização aceitável.

Por forma a efectuar estes testes, efectuaram-se as operações fundamentais de manutenção de dados:

- inserção de novas características;
- eliminação de características existentes[3];

[2]Embora tenha indicado os tipos de uma forma que permitiu considerá-la como inválida, como veremos a seguir.

[3]A actualização de características é um problema que não se coloca, visto que tal é impossível (i.e. não é possível mudar o tipo da característica).

- inserção de novos valores nas propriedades de características;
- modificação de valores existentes em propriedades de características;
- eliminação de valores de propriedades nas características.

Todos estes testes foram realizados, tendo sido corrigidos alguns problemas (*bugs*), entretanto detectados.

Um dos principais problemas verificados nesta fase, está relacionado com o método de eliminação de registos existentes no Palm. Palm Inc. [2000b] refere quatro processos distintos para eliminação de registos:

DmArchiveRecord Quando um registo é arquivado, o bit de estado `eliminado` é activado, mas o espaço ocupado por este não é libertado e o seu `localID` é preservado. Desta forma na próxima vez que a sincronização é efectuada, o registo pode ser armazenado no PC antes de ser definitivamente removido do Palm.

DmDeleteRecord Activa o bit `eliminado` como no item anterior e liberta o espaço que o mesmo ocupa. Porém, não remove a entrada no cabeçalho dos registos, mas marca o `localChunkID` do registo como `NULL`.

DmDetachRecord Esta rotina liberta o registo da base de dados, removendo a entrada do cabeçalho. Ao contrário do anterior, a entrada é removida do cabeçalho, mas o registo não é eliminado.

DmRemoveRecord Remove tanto o espaço ocupado pelo registo como a entrada no cabeçalho da base de dados.

Devido à especificidade da aplicação desenvolvida, como é necessário obter os dados dos registos eliminados, foi necessário utilizar o primeiro método (`DmArchiveRecord`) que permite obter o valor originalmente armazenado em registos eliminados (ou seja é possível desfazer a operação de eliminação sem perda de dados).

7.3 Experimentação do MetaGEO

Após se ter verificado que as operações fundamentais eram realizadas sobre um ficheiro GML simples, foi necessário experimentar o MetaGEO num caso real.

Para tal, o primeiro passo foi tentar obter um mapa completo (o da cidade do Porto), por forma a criar o GML e inserir a informação correspondente no Palm.

Infelizmente, até ao momento não foi possível obter esse mapa. Isto não se revelou como um grande problema[4], visto que a ParadigmaXis já tinha o mapa do Porto em formato

[4]Embora tenha provocado algum atraso no processo.

PalmOS, e visto que a aplicação GML2PDB criava as bases de dados no mesmo formato, utilizou-se esse mapa[5].

Criou-se, porém, um ficheiro GML contendo pontos de interesse seleccionados sobre o mapa da cidade do Porto, relativos ao Projecto "**Percursos Agustinianos: O Porto na obra de Agustina Bessa Luís**", contendo dados georreferenciados de diversos tipos, principalmente texto e fotografias (www.ismai.pt/percursosAgustinianos).

73 ⟨*Excerto do Ficheiro GML do projecto Agustina* 73⟩≡

```
<pontoDeInteresse>
   <descricao>
   Tendo recebido em 1897 o nome de Praça de Mouzinho de Albuquerque, em
   memória do herói de Chaimite e das guerras de África, era, ainda nos começos
   do século XX, uma zona tipicamente rural. Nos nossos dias, apresenta-se como
   o verdadeiro centro da vida empresarial, comercial e cultural do Porto. No
   centro da Praça, merece destaque o Monumento aos Heróis da Guerra
   Peninsular, homenagem à coragem dos portuenses durante as invasões
   francesas.
   </descricao>
   <categoria>via principal</categoria>
   <toponimia>Rotunda da Boavista</toponimia>
   <ponto>
      <gml:Point srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
         <gml:coord>
            <gml:X>23056</gml:X>
            <gml:Y>7568</gml:Y>
         </gml:coord>
      </gml:Point>
   </ponto>
</pontoDeInteresse>
```

[5]Ou seja, assumiu-se que a aplicação GML2PDB criava o PDB geográfico correctamente, o que até agora permanece verdade.

O esquema sobre o qual o GML foi validado é bastante simples, como se pode verificar.

74 ⟨*Esquema GML do projecto Agustina* 74⟩≡

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- File: agustina.xsd  -->
<schema targetNamespace="http://localhost/GML2/examples"
        xmlns:ex="http://localhost/GML2/examples"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:gml="http://localhost/GML2/gml"
        xmlns="http://www.w3.org/2000/10/XMLSchema"
        elementFormDefault="qualified"
        version="0.2">
  <annotation>
    <appinfo>agustina.xsd v0.2 2001-11</appinfo>
    <documentation xml:lang="en">
      GML schema for Agustina project
    </documentation>
  </annotation>
  <import namespace="http://localhost/GML2/gml" schemaLocation="feature.xsd"/>

  <element name="pontoDeInteresse" type="ex:piType"
    substitutionGroup="gml:featureMember"/>

  <element name="_PontoDeInteresse" type="gml:AbstractFeatureType"
           substitutionGroup="gml:_Feature"/>

  <complexType name="piType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element name="descricao" type="string"/>
          <element name="categoria" type="string"/>
          <element name="toponimia" type="string"/>
          <element name="ponto" type="gml:PointPropertyType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
```

Efectuou-se a geração dos PDB para Palm com sucesso e sincronizaram-se os dados. O módulo MetaGeo foi utilizado para se efectuar actualizações nos dados[6]. As actualizações foram diversas, sempre referenciando geometricamente os locais e atribuindo-se informação extra (ou seja, dados georreferenciados).

Estas tarefas decorreram com sucesso. Era altura de verificar se o sistema conseguia regenerar o GML com as alterações introduzidas.

7.4 Detecção das modificações efectuadas

No servidor de dados onde se encontrava o GML original[7], acoplou-se o Palm à base (*craddle*) e pressionou-se o botão de sincronização (*HotSync*). O sistema iniciou o processo de sincronização, que aparentemente estava a decorrer sem problemas. Quando o sistema terminou, tudo estava OK.

Algumas das alterações foram escritas sobre o ficheiro de *Log*, nomeadamente actualização e eliminação de dados.

Todas as modificações estavam referidas. Porém, o sistema estava um pouco lento, sendo verificado que o algoritmo de comparação entre as árvores DOM geradas a partir dos PDB existentes no Palm e do ficheiro GML original, não se encontrava optimizado. Porém, foi perferido incorporar a melhoria numa versão futura.

7.5 Correcção do ficheiro GML gerado

Faltava, por fim, verificar se o ficheiro GML gerado era válido de acordo com o esquema definido para a aplicação.

Para validar o GML gerado, utilizou-se o *parser* Xerces, que não indicou problemas com o mesmo. A seguir, visualizou-se o documento no *Internet Explorer*, que também não indicou problemas.

Por fim, verificou-se se o *parser* específico para GML GML4Java continuava a considerar o documento gerado válido. Também este teste foi efectuado com sucesso (i.e. o *parser* não indicou qualquer problema de sintaxe no documento gerado, embora o problema semântico continuasse a existir).

Considerou-se, assim, que o GML gerado pelo sistema de sincronização estava correctamente delineado.

[6]Por forma a saber as coordenadas correctamente acoplou-se um sistema de GPS ao Palm.
[7]Poderia ser noutra máquina que tivesse ligação através da rede a este computador.

Capítulo 8

Conclusões

Após o desenvolvimento e avaliação da aplicação que permite efectuar a manutenção de dados georreferenciados, torna-se necessário rever os objectivos delineados inicialmente para o trabalho. Foi possível demonstrar que a tecnologia existente já permite a manutenção de dados georreferenciados *in loco*, sem necessidade de efectuar múltiplas operações, como até agora acontecia.

Neste capítulo apresentam-se os objectivos iniciais, sendo indicado para cada um o seu grau de satisfação.

8.1 Resultados Alcançados

Já se verificou anteriormente que o principal objectivo do trabalho relatado nesta dissertação foi alcançado, graças ao desenvolvimento dos módulos da aplicação MetaGeo.

Este foi, aliás, o efeito mais visível da presente dissertação. Porém, algo mais foi alcançado.

Foram analisados os dois Sistemas Móveis mais utilizados no momento, sendo avaliadas as suas diferenças, vantagens e desvantagens. Destes foi seleccionado para plataforma de aplicação o Palm.

Tanto quanto sabemos, foi criada uma das primeiras aplicações académicas com suporte para transformação de dados de e para o formato GML.

Da mesma forma, foram analisadas as implicações da sincronização de dados, principalmente quando se estão a comparar documentos em formatos distintos. Daqui saíram dois processos distintos, que poderiam ser igualmente aplicados, tendo sido seleccionado o que pareceu, para este caso concreto, ser mais favorável.

Estudaram-se formas de, utilizando a plataforma seleccionada, efectuar chamadas entre aplicações distintas, sendo seleccionada uma que permite desenvolver uma aplicação que tanto pode ser utilizada em separado (*stand-alone*) como interagindo com uma outra (no caso presente com a aplicação `Mordomo`).

8.2 Revisão dos Objectivos

Com o desenvolvimento da aplicação MetaGEO, constituída pelos módulos `GML2PDB`, `MetaGeo`, `PDB2GML` e `CondutaGML`, foram cumpridas as operações de manutenção de dados georreferenciados. Este sistema utiliza internamente uma base de dados, específica para o sistema Palm, embora permita a transformação de e para o formato padrão apresentado pelo OGC.

Este formato (GML) foi avaliado relativamente às possibilidades de utilização. A evolução do formato foi referida, sendo indicadas as características a esperar para a próxima versão do formato. A aplicação desenvolvida suporta o formato GML 2.0.

Foram identificados e resolvidos os problemas específicos dos dispositivos de computação móvel, nomeadamente os problemas colocados durante a sincronização e reconciliação de dados e as restrições existentes durante o desenvolvimento. Foram principalmente levados em linha de conta os problemas relativos à reduzida quantidade de memória.

O sistema desenvolvido foi testado num sistema real, através da sua integração no projecto **Percursos Agustinianos: O Porto na obra de Agustina Bessa Luís.**

Os resultados alcançados foram apresentados e foi testado o funcionamento de todos os módulos desenvolvidos.

Daqui se pode concluir que os objectivos inicialmente propostos foram alcançados.

8.3 Trabalhos Futuros

O sistema, apesar de no momento permitir executar as funcionalidades essenciais propostas, poderia ser melhorado. Durante o trabalho foram desde logo encontrados alguns problemas que ainda não estão resolvidos.

Destes, o que se apresenta como sendo o mais essencial é o tratamento da sincronização de múltiplas réplicas, ao invés de apenas duas como é actualmente. Algumas técnicas que podem ser utilizadas foram já apresentadas. Outra das técnicas que permitiriam a resolução deste problema relaciona-se com a utilização de sistemas de controlo de versões, como por exemplo `RCS` [Tichy 1985]. A aplicação de sistemas de controle de versões para sincronização de réplicas incompatíveis foi já descrita por Agrawal e Sengupta [1989] e Wu et al. [1993].

O sistema de divisão em rectângulos (*tiles*) apresentado na secção 4.1.1 (página 42) é atraente, mais para o sistema de visualização de cartografia que para o sistema de manutenção. Porém, mais algum trabalho deverá ser efectuado sobre este tópico.

O algoritmo de comparação, que já se verificou estar aquém do esperado em termos de desempenho, como foi referido na secção 7.4 (página 75), deve ser reformulado. Nomeadamente, o processo de criação da árvore DOM de GML com as alterações impede que seja utilizado o algoritmo referido por Wang et al. [2000].

Outros melhoramentos possíveis incluem:

visualizar o fragmento do mapa onde consta o ponto a ser actualizado que permitiria identificar, durante as operações de actualização, a envolvente cartográfica do ponto, existindo assim mais informação disponível para o utilizador;

utilizar hiperligações através de dados do tipo XLink e XPointer com ligação directa através da Internet a sítios Web onde poderia ser capturada mais informação referente ao ponto (por exemplo, obter actualizações efectuadas no GML original e disponíveis via Web);

obter dados directamente de servidores Web como proposto pelo OGC, por forma a ser possível obter dados de diferentes sítios, sem necessidade de efectuar uma operação de pesquisa e actualização no local onde os dados se encontravam originalmente [Cox 2001; Open GIS Consortium 1998b; Tom 1994; Wei et al. 1999].

Também seria conveniente que a aplicação fosse experimentada noutras situações (por exemplo, em que seja necessário actualizar não apenas um ponto, mas um polígono, como um terreno no PDM). Não foi possível realizar este último teste embora, em teoria, o sistema deva funcionar correctamente.

8.4 Conclusão

A vulgarização dos Sistemas de Computação Móveis, como por exemplo a plataforma Palm, seleccionada a título de exemplo, obriga à redefinição das tarefas executadas manualmente.

Não sendo propósito desta dissertação apresentar novos modelos de trabalho para os utilizadores de dados na área geográfica, foram indicadas pistas sobre como poderão (e deverão) ser desenvolvidas aplicações que permitam reformular esses processos de trabalho.

A aplicação desenvolvida, constituída por quatro módulos distintos e adaptáveis a novas situações, cumpriu os objectivos propostos. Alguns pontos que foram surgindo no decorrer do projecto foram incorporados, tendo outros sido indicados como melhorias.

Esta aplicação permite efectuar uma gestão de dados georreferenciados, sendo a sua área de aplicação bastante abrangente.

Esta tecnologia poderia ser aplicada, por exemplo, na criação de uma infra-estrutura de consulta através de Postos de Informação Municipal, existindo anotação de locais de interesse para criação de mapas virtuais.

Outra das aplicações possíveis seria para fiscalização de obras públicas.

Referências

Agrawal, Divyakant e Soumitra Sengupta (1989). Modular Synchronization in Multiversion Databases: Version Control and Concurrency Control. pág. 408–417. ACM Press.

Ball, Thomas e Fred Douglis (1996, Janeiro). Tracking and viewing changes on the Web. Em *Proceedings of 1996 USENIX Technical Conference*, San Diego, CA, pág. 165–176.

BBN Technologies (2001, Setembro). *What is OpenMap?* BBN Technologies.

Beckmann, Norbert, Hans-Peter Kriegel, Ralf Schneider, e Bernhard Seeger (1990). The R*-tree: an efficient and robust access method for points and rectangles. Em *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, pág. 322–331. ACM Press.

Berk, Elliot (1996, Dezembro). HtmlDiff: A Differencing Tool for HTML Documents. Student Project, COS 461, Princeton University.

Box, Don, Aaron Skonnard, e John Lam (2000). *Essencial XML.* Addison Wesley.

Bray, Tim (2001, Junho). What is RDF?
Disponível na Web em http://www.xml.com/lpt/a/2001/01/24/rdf.html.

Brinkhoff, Thomas, Hans-Peter Kriegel, e Bernhard Seeger (1993). Efficient processing of spatial joints using R-trees. Em *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pág. 237–246. ACM Press.

Center for Computational Geography (2001, Novembro). *GeoTools.* Center for Computational Geography.

cgal.org (2000, Outubro). *CGAL: Release 2.2.* cgal.org. Diversos Manuais do Produto.

Champin, Pierre-Antoine (2001, Abril). RDF Tutorial.
Disponível na Web em http://www710.univ-lyon1.fr/~champin/rdf-tutorial/.

Chen, Yih-Farn, Fred Douglis, Huale Huang, e Kien-Phong Vo (2000). TopBlend: An Efficient Implementation of HtmlDiff in Java.
Disponível na Web em http://www.research.att.com/~chen/topblend/paper/index.html.

Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Bases. *Communications of the ACM 13*(6), 377–387.

Cox, Simon (2001, Junho 28). Geologic Data Transfer Using XML.
Disponível na Web em http://www.kgs.ukans.edu/Conferences/IAMG/Sessions/I/cox.html.
CSiro Exploration & Mining - IAMG2001.

Cox, Simon, Adrian Cuthbert, Ron Lake, e Richard Martell (2001, Fevereiro). Geography Markup Language (GML) 2.0 - OGC Recommendation.
Disponível na Web em http://www.opengis.net/gml/01-029/GML2.html.

Demers, Michael N. (2000). *Fundamentals of Geographic Information Systems* (second ed.). John Wiley & Sons.

Donnely, Charles e Richard Stallman (1995, Novembro). *Bison: The YACC-Compatible Parser Generator, Version 1.25 (User Manual).* Free Software Foundation.

Douglis, Fred, Thomas Ball, Yih-Farn Chen, e Eleftherios Koutsofios (1998, Janeiro). The AT&T Internet Difference Engine: Tracking and Viewing Changes on the Web. Em *World Wide Web*, Número 1, pág. 27–44. Baltzer Science Publishers.

Federal Geographic Data Committee (2000, Maio). *Content Standard for Digital Geospatial Metadata Workbook (For use with FGDC-STD-001-1998) — Version 2.0.* Federal Geographic Data Committee. Disponível na Web em http://www.fgdc.gov/publications/documents/metadata/workbook_0501_bmk.pdf.

Galdos Systems, Inc. (2001, Março). *GML 2.0 — Enabling the Geo-spatial Web.* Galdos Systems, Inc. Disponível na Web em http://xml.coverpages.org/Galdos-GML2-EnablingTheGeoSpatialWeb.pdf.

Gossett, Brent (2000, Janeiro 24). *Conduit Programmer's Companion for Windows.* 5400 Bayfront Plaza — Santa Clara, CA — 95052 USA: Palm Computing, Inc. Document Number 3013-001.

Gutman, A. (1984, Junho). R-trees: A Dynamic Index Structure for Spatial Searching. Em *Proceedings of the 1984 ACM-SIGMOD Conference*, Boston, Mass, pág. 47–57. ACM Press.

Idris, Nazmul (1999, Maio 22). Should I use SAX or DOM?
Disponível na Web em http://65.1.136.127/developerlife/saxvsdom/file.pdf.

Knuth, Donald Ervin (1992). *Literate Programming*, Volume 27 de *CSLI Lecture Notes.* Center for the Study of Language and Information, Leland Stanford Junior University.

Lake, Ron (2001, Março). Introduction to Geography Markup Language (GML), Part 1 in GML Series. Disponível na Web em http://www.jlocationservices.com/company/galdos/articles/introduction_to_gml.htm.

Liefke, Hartmut e Dan Suciu (2000). XMill: an efficient compressor for XML Data. Em *Proceedings of the 2000 ACM SIGMOD on Management of data*, pág. 153–164. ACM Press.

Matos, João Luís (2001, Março). *Fundamentos de Informação Geográfica.* LIDEL.

MicroImages, Inc. (1997). *Announcing TNTlite.* MicroImages, Inc.

Microsoft. English Query.
Disponível na Web em http://www.microsoft.com/sql/evaluation/features/english.asp.

Microsoft. Meta Data Services.
Disponível na Web em http://www.microsoft.com/sql/techinfo/BI/mds.asp.

Neto, Pedro Leão (1998, Abril). *Sistemas de Informação Geográfica.* FCA.

Open GIS Consortium (1998a). *The OpenGIS Abstract Specification Model. Topic 0: Abstract Specification Overview* (3 ed.). Wailand, MA 01778: Open GIS Consortium. editor: Core Task Force, disponível na Web em http://www.opengis.org/techno/specs.htm.

Open GIS Consortium (1998b). *The OpenGIS Abstract Specification Model. Topic 10: Transfer Technology* (3 ed.). Wailand, MA 01778: Open GIS Consortium. editor: The Transfer Technology Working Group, disponível na Web em http://www.opengis.org/techno/specs.htm.

Open GIS Consortium (1998c). *The OpenGIS Abstract Specification Model. Topic 11: Metadata* (3 ed.). Wailand, MA 01778: Open GIS Consortium. editor: Core Task Force, disponível na Web em http://www.opengis.org/techno/specs.htm.

Open GIS Consortium (1998d). *The OpenGIS Abstract Specification Model. Topic 5: The Open GIS Feature and Feature Collections* (3 ed.). Wailand, MA 01778: Open GIS Consortium. editor: Feature Special Interest Group, disponível na Web em http://www.opengis.org/techno/specs.htm.

Open GIS Consortium (1998e). *The OpenGIS Abstract Specification Model. Topic 7: Earth Imagery* (3 ed.). Wailand, MA 01778: Open GIS Consortium. editor: Coverages Working Group and Pixel Manipulation Working Group, disponível na Web em http://www.opengis.org/techno/specs.htm.

Open GIS Consortium (1998f). *The OpenGIS Abstract Specification Model. Topic 8: Relations Between Features* (3 ed.). Wailand, MA 01778: Open GIS Consortium. editor: Core Task Force and Feature SIG, disponível na Web em http://www.opengis.org/techno/specs.htm.

Open GIS Consortium (2000). *OGC Technical Specifications.* Open GIS Consortium.

Palm Inc. (2000a, Maio). *Palm File Format Specification.* Palm Inc. Document Number 3008-003.

Palm Inc. (2000b, Junho). *Palm OS SDK Reference.* Palm Inc. Document Number 3003-003.

Paxson, Vern (1995, Março). *Flex: A fast scanner generator, Edition 2.5 (User Manual).*

Puma Technologies (1999, Abril). *Invasion of the Data Snatchers.* Puma Technologies.

Satyanarayanan, M. (1996). Fundamental challenges in mobile computing. Em *Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing*, pág. 1–7. ACM Press.

Satyanarayanan, M., J. J. Kistler, P. Kumar, M. E. Okasaki, E. H. Siegel, e D. C. Steere (1990, Abril). Coda: A highly available file system for a distributed workstation environment. Em *IEEE Trans. Comput.*, Número 39.4, pág. 447–459. IEEE.

Sousa, Alexandre Valente (2001, Novembro). Literate Programming applied to Software Maintenance and Teaching Computer Science. *Perspectivas XXI 4*(8), 101–120. ISSN 0874-274X.

SyncML (2001). Building an Industry-Wide Mobile Data Synchronization Protocol. Disponível na Web em http://www.syncml.org/download/whitepaper.pdf.

The Apache Software Foundation (2001). *Manual de Xerces 1.5.1 para C++.* The Apache Software Foundation.

Tichy, Walter F. (1985, Julho). RCS — A System for Version Control. *Software Practice & Experience 15*(7), 637–654.

Tom, Henry (1994, Setembro). The Geographic Information Systems (GIS) Standards Infrastructure. *StandardView 2*(3), 133–142.

W3C (2000a). eXtensible Markup Language. Disponível na Web em http://www.w3.org/XML/.

W3C (2000b). XML Schema part 0: Primer. W3C Candidate Recomendation. Disponível na Web em http://www.w3.org/TR/xmlschema-0/.

W3C (2000c). XML Schema part 2: Datatypes. W3C Candidate Recomendation. Disponível na Web em http://www.w3.org/TR/2000/CR-xmlschema-2-20001024/.

Wang, Yuan, David J. DeWitt, e Jin-Yi Cai (2000). X-Diff: A Fast Change Detection Algorithm for XML Documents.
Disponível na Web em http://citeseer.nj.nec.com/449452.html.

Wei, Zu-Kuan, Young-Hwan Oh, Jae-Dong Lee, Jae-Hong Kim, Dong-Sun Park, Young-Geol Lee, e Hae-Ypung Bae (1999). Efficient spatial data transmission in Web-based GIS. Em *Proceedings of the second international workshop on Web information and data management*, pág. 76–81. ACM Press.

Winter, Stephan (1998, Novembro). Bridging Vector and Raster Representation in GIS. Em *ACM GIS '98*. ACM Press.

Wu, Kun-Lung, Philip S. Yu, e Ming-Syan Chen (1993). Dynamic Finite Versioning: An Effective Versioning Approach to Concurrent Transaction and Query Processing. Em *Proceedings of the Nineth International Conference on Data Engineering, April 19-23, 1993, Vienna, Austria*, pág. 577–586. IEEE Computer Society.
Disponível na Web em http://dblp.uni-trier.de/text/2-6/ICDE93/P577.pdf.

Wu, Y. (2000a, Setembro). R2V Conversion: Why and How? Em *GeoInfomatics*, Número 6, pág. 28–31.
Disponível na Web em http://www.ablesw.com/r2v/gi0900.html.

Wu, Y. (2000b, Setembro). *Raster, Vector, and Automated Raster-to-Vector Conversion.* Moving Theory into Practice: Digital Imaging for Libraries and Archives. RLG, Cornell Univ. Library.

Yu, J. e D. DeWitt (1996, Setembro). Processing Satellite Images on Tertiary Storage: A Study of the Impact of Tile Size on Performance.
Disponível na Web em http://citeseer.nj.nec.com/yu96processing.html.

Glossário

Document Type Definition (DTD) Define a gramática e as marcas válidas para uma formatação específica XML.

eXtensible Markup Language (XML) Uma Linguagem de Marcação que permite hiperligações complexas, e permite que os utilizadores definam os seus próprios elementos e marcas.

Norma disponível na Web em http://www.w3.org/TR/REC-xml/.

Geographic Markup Language (GML) Linguagem de marcação que especifica um conjunto de regras para a transformação de dados georreferenciados.

Norma disponível na Web em http://opengis.net/gml/index.htm.

Geographic Information System (GIS) A principal tecnologia que motiva interesse nos Sistemas de Gestão de Bases de Dados Espaciais (*Spatial Database Management Systems — SDBMS*. Fornece um mecanismo conveniente para a análise e visualização de dados geográficos.

Hypertext Markup Language (HTML) Uma linguagem de marcação de texto, desenhada para disponibilizar funcionalidades hipermédia na Internet. Os elementos HTML permitem definir, formatar e melhorar a aparência de páginas na Web.

Norma disponível na Web em http://www.w3.org/TR/html401/.

Open GIS Consortium(OGC) Organização constituída pelos principais implementadores de produtos GIS, com o objectivo de formular normas para toda a indústria.

Web page em http://www.opengis.org/.

Resource Description Framework (RDF) Uma norma para descrever metadados em XML.

Norma disponível na Web em http://www.w3.org/TR/REC-rdf-syntax/.

Standard Generalized Markup Language (SGML) Uma linguagem de marcação de texto especificada pela (*International Standard Organization — ISO*). XML, HTML e GML são subconjuntos de SGML.

World Wide Web Consortium (W3C) Uma organização que define normas para a internet (WWW) e contribui para XML, HTML, XSD e outros padrões.

Web page em http://www.w3.org/.

XML Linking Language (XLink) Uma linguagem que permite que documentos XML criem e descrevam ligações entre recursos na Web.

Norma disponível na Web em http://www.w3.org/TR/xlink/.

XML Schemas (XSD) Uma linguagem usada para definir, descrever e catalogar vocabulários XML para classes de documentos XML.

Norma disponível na Web em http://www.w3.org/TR/xmlschema-0/.

Índices

Fragmentos de Código

⟨*Processo para criação de GML* 61a⟩ 61a
⟨*Resultado para o Ficheiro de Exemplo* 43c⟩ 43c
⟨*type definitions for city model* 69⟩ 67b, 69
⟨*Um rio no exemplo GML* 66b⟩ 65, 66b
⟨*Uma estrada no exemplo GML* 67a⟩ 65, 67a
⟨*version/FundTeorica/exemplo.xsd* 27⟩ 27
⟨*version/TrabalhoRealizado/cambridge.xml* 65⟩ 65
⟨*version/TrabalhoRealizado/city.xsd* 67b⟩ 67b

Citações

Agrawal e Sengupta [1989], 35, 77
BBN Technologies [2001], 10
Ball e Douglis [1996], 36
Beckmann et al. [1990], 19
Berk [1996], 36
Box et al. [2000], 27
Bray [2001], 24, 25
Brinkhoff et al. [1993], 19
Center for Computational Geography [2001], 10, 48
Champin [2001], 24
Chen et al. [2000], 36, 60
Codd [1970], 40
Cox et al. [2001], 26, 42
Cox [2001], 22, 78
Demers [2000], 18
Donnely e Stallman [1995], 43
Douglis et al. [1998], 36
Federal Geographic Data Committee [2000], 23
Galdos Systems, Inc. [2001], 28, 29
Gossett [2000], 37, 60
Gutman [1984], 19
Idris [1999], 47
Knuth [1992], 13
Lake [2001], 24, 26
Liefke e Suciu [2000], 41
Matos [2001], 16–18
MicroImages, Inc. [1997], 10
Microsoft [a], 24
Microsoft [b], 24
Neto [1998], 19
Open GIS Consortium [1998a], 15, 19, 20
Open GIS Consortium [1998b], 29, 78
Open GIS Consortium [1998c], 22, 23
Open GIS Consortium [1998d], 21
Open GIS Consortium [1998e], 20
Open GIS Consortium [1998f], 29
Open GIS Consortium [2000], 19
Palm Inc. [2000a], 40
Palm Inc. [2000b], 72
Paxson [1995], 43
Puma Technologies [1999], 11
Satyanarayanan et al. [1990], 35
Satyanarayanan [1996], 34, 35
Sousa [2001], 14
SyncML [2001], 35
The Apache Software Foundation [2001], 47, 48
Tichy [1985], 77
Tom [1994], 78
W3C [2000a], 24
W3C [2000b], 25, 26
W3C [2000c], 27
Wang et al. [2000], 36, 60, 77
Wei et al. [1999], 78
Winter [1998], 18, 19
Wu et al. [1993], 35, 77
Wu [2000a], 19
Wu [2000b], 19
Yu e DeWitt [1996], 42
cgal.org [2000], 10

Remissivo