



Faculdade de Engenharia  
**Universidade do Porto**

# **Adaptação de conteúdos Web para o ambiente WAP**

**Arlindo Costa dos Santos**

Licenciado em Engenharia Electrotécnica pela  
Universidade de Trás-os-Montes e Alto Douro - UTAD

Dissertação submetida para a satisfação parcial dos requisitos de  
**Grau de Mestre em Tecnologia Multimédia**

Dissertação realizada sob a supervisão do  
**Professor Doutor Eurico Carrapatoso**  
do Departamento de Engenharia Electrotécnica e de Computadores da  
Faculdade de Engenharia da Universidade do Porto – FEUP

**Porto – Março de 2002**

# Resumo

Um dos actuais desafios da Internet surge a partir do momento que a tecnologia WAP permite a utilização de dispositivos móveis para aceder à informação disponível na Internet formatada de acordo com uma especificação própria.

Mas os dispositivos móveis apenas podem aceder a conteúdos multimédia projectados de acordo com uma especificação que difere daquela que é usada para disponibilizar conteúdos para os computadores pessoais. E como um dos actuais entraves para o sucesso da tecnologia WAP é sem dúvida a escassez de conteúdos quando comparada com quantidade de informação Web. É portanto necessário criar mecanismos de enriquecimento através da adaptação da informação existente na Web ou simplesmente através do desenvolvimento de conteúdos próprios.

Esta dissertação tem como objectivo apresentar uma aplicação responsável pela adaptação automática de conteúdos disponibilizados na World Wide Web para o ambiente WAP, e assim permitir que os utilizadores móveis possam aceder a informação que até agora era acessível apenas a computadores pessoais. Para efectuar o desenvolvimento dessa aplicação foi necessário investigar os métodos e técnicas associados ao processo de conversão da linguagem de formatação dos documentos da Web (HTML) para a linguagem de formatação normalizada para o ambiente WAP (WML).

Palavras-chave: Conversão de HTML para WML; XSL; XML; Segmentação; WAP; World Wide Web; Wireless; Acesso móvel.

# Abstract

One of the current challenges of the Internet derives from the fact that the WAP technology allows the use of mobile devices to access information available on the Internet shaped according to a specific standard.

However mobile devices only access multimedia content if it is configured according to a standard which differs from that used for personal computers. And one of the existing obstacles for the success of WAP technology is, without any doubt, the lack of content when compared to the quantity of information available in the Web. It is therefore necessary to create packages to adapt the existing Web information or simply to allow the development of dedicated content.

This thesis has got the main purpose of introducing a package for the automatic conversion of content available on the World Wide Web to the WAP environment, and thus allow mobile users to have access to information that until now was only available through personal computers. In order to accomplish the development of this package, it was necessary to investigate methods and techniques that may be used in the process of converting Web documents language (HTML) to WAP standard language (WML).

Key words: HTML conversion to WML; XSL; XML; Segmentation; WAP; World Wide Web; Wireless; Mobile access.

# Résumé

L'un des actuels défis de l'internet apparaît dès le moment où la technologie WAP permet l'utilisation de dispositifs mobiles pour accéder à l'information disponible à l'internet dont la formatation suit une spécification déterminée.

Mais les dispositifs mobiles peuvent seulement accéder à des contenus multimédias avec une spécification qui est différente de celle qui est usée pour mettre à la disposition des contenus d'ordinateurs personnels. Et comme un des actuels obstacles pour le succès de la technologie WAP est, sans doute, la rareté des contenus comparativement à la quantité d'information Web, il faut, donc, créer des mécanismes d'enrichissement avec l'adaptation de l'information Web existante ou simplement avec le développement de contenus spécifiques.

Cette dissertation a pour objectif présenter une application responsable par l'adaptation automatique de contenus mis à disposition à la World Wide Web pour WAP, et, de cette manière, permettre que les utilisateurs de mobiles puissent accéder à l'information qui jusqu'ici était seulement accessible par des ordinateurs personnels. Pour effectuer le développement de cette application, il a été nécessaire de chercher les méthodes et les techniques associés au procès de conversion du langage utilisé pour formater des documents de la Web (HTML) en langage de formatation normalisée de WAP (WML).

Mots clef: Conversion HTML à WML; XSL; XML; Segmentation; WAP; World Wide Web; Wireless; L'accès mobile;

# Agradecimentos

Os agradecimentos e dedicatória são uma parte integrante das regras da elaboração de um relatório, e por isso quero começar por prestar homenagem aos meus pais, e agradecer-lhes todo o amor que me deram ao longo destes 28 anos. São apenas palavras que não podem substituir o grande OBRIGADO!

Apesar da sua natureza individual, vários foram aqueles que contribuíram, directa ou indirectamente, para a sua concretização. Cumpre-me agora agradecer a preciosa colaboração.

Ao meu orientador agradeço a orientação desta dissertação, bem como também a estima e simpatia com que sempre o fez.

Ao Instituto Politécnico de Bragança e a todos os colegas, agradeço as facilidades concedidas para a realização desta dissertação.

A todos os meus amigos agradeço do fundo do coração aquilo que me ensinaram, os conselhos que me deram e a forma como me têm ajudado a olhar e sentir o mundo que nos rodeia.

À Ana, agradeço toda a compreensão, carinho e apoio ao longo destes últimos anos, e espero que continue a ser a minha grande amiga.

Um obrigado a todas as pessoas que directamente ou indirectamente me ajudaram e ajudam a viver, e em especial aos meus antigos professores.

# Índice

RESUMO .....	II
ABSTRACT .....	III
RESUME.....	IV
AGRADECIMENTOS .....	V
ÍNDICE .....	VI
LISTA DE FIGURAS.....	IX
LISTA DE TABELAS .....	X
ACRÓNIMOS .....	XI
1 INTRODUÇÃO.....	1
1.1 Contexto .....	1
1.2 Objectivos .....	2
1.3 Organização da dissertação .....	3
2 REDES E ARQUITECTURAS DE COMUNICAÇÃO.....	4
2.1 Infra-estruturas de comunicação .....	4
2.1.1 Internet .....	4
2.2 Tecnologias Wireless .....	5
2.2.1 Global System Mobile .....	6
2.2.2 General Packet Radio Service .....	8
2.2.3 Universal Mobile Telecommunications System .....	9
2.3 Architecturas de comunicação .....	11
2.3.1 World Wide Web .....	11
2.3.1.1 Contexto histórico .....	11
2.3.1.2 Arquitectura .....	12
2.3.2 Wireless Application Protocol .....	14
2.3.2.1 Compreender a tecnologia .....	14
2.3.2.2 Arquitectura .....	17
2.3.2.3 O futuro do WAP .....	18
3 EVOLUÇÃO DAS LINGUAGENS DE MARKUP .....	21
3.1 Definição de markup .....	21
3.2 Linguagens de markup .....	22
3.2.1 Standard Generalized Markup Language.....	24
3.2.2 HyperText Markup Language .....	24
3.2.2.1 Estrutura do documento .....	25

3.2.2.2	Cascading Style Sheets .....	26
3.2.2.3	Aspectos críticos da linguagem.....	27
3.2.3	eXtensible Markup Language .....	28
3.2.3.1	Motivação.....	28
3.2.3.2	Especificação.....	29
3.2.3.3	Estrutura dos documentos .....	30
3.2.3.4	Transformação com XSL .....	33
3.2.4	eXtensible HyperText Markup Language.....	35
3.2.4.1	Motivação.....	35
3.2.4.2	Diferenças entre HTML e XHTML .....	35
3.2.5	Wireless Markup Language .....	37
3.2.5.1	Características .....	37
3.2.5.2	Estrutura do documento .....	39
3.2.5.3	Formatos complementares .....	40
4	ADAPTAÇÃO DE HTML PARA WML .....	42
4.1	Estado da arte .....	43
4.2	Métodos de conversão.....	44
4.2.1	Automática .....	45
4.2.2	Manual .....	46
4.2.3	Personalizada .....	46
4.2.4	Integrada.....	47
4.3	Problemática da conversão.....	47
4.3.1	Localização do conversor automático .....	47
4.3.2	Segmentação do conteúdo.....	48
4.4	Técnicas aplicadas à conversão.....	48
5	REQUISITOS DE UM SISTEMA DE CONVERSÃO.....	54
5.1	Análise de trabalhos relacionados.....	54
5.1.1	Projectos académicos .....	54
5.1.2	Projectos comerciais.....	57
5.2	Requisitos do sistema.....	61
5.3	Metodologia de desenvolvimento .....	62
6	DESENVOLVIMENTO DO SISTEMA CONVERSOR .....	65
6.1	Tabela de correspondência directa.....	65
6.1.1	Formulário para introdução da URL .....	67
6.1.2	Pedido da página HTML.....	67
6.1.3	Formatação correcta da página HTML .....	68
6.1.4	Conversão para WML .....	69
6.1.5	Eliminação de conteúdo irrelevante .....	70
6.1.6	Formatação correcta do documento WML .....	73
6.1.7	Eliminação de informação nula.....	74
6.1.8	Conversão dos caracteres especiais.....	75
6.1.9	Adição da acessibilidade na navegação .....	75

6.1.10	Eliminação da duplicação de elementos nos sub-nodos .....	76
6.1.11	Adaptação das listas de itens.....	79
6.1.12	Limitação do tamanho do deck .....	80
6.1.13	Inserção de parágrafos nos cards.....	82
6.1.14	Guardar cada deck.....	83
6.2	Conversão com o documento XSL .....	83
7	CONCLUSÕES.....	86
7.1	Síntese .....	86
7.2	Discussão dos resultados.....	87
7.3	Considerações finais .....	87
	REFERÊNCIAS .....	90
	ANEXO A – EXEMPLOS DE CONVERSÕES DE PÁGINAS HTML .....	96
	Exemplo 1 .....	96
	Página HTML .....	96
	Deck WML .....	97
	Exemplo 2 .....	98
	Página HTML .....	98
	Deck WML .....	98
	Exemplo 3 .....	99
	Página HTML .....	99
	Deck WML .....	99
	ANEXO B – DOCUMENTO XSL .....	101
	ANEXO C – EXEMPLOS DE DISPOSITIVOS MÓVEIS .....	105
	ANEXO D – EXCERTOS PRINCIPAIS DO CÓDIGO FONTE.....	108



# Lista de Figuras

Figura 2.1 - Evolução das redes <i>wireless</i> .....	6
Figura 2.2 - Estatísticas do estado actual do GSM (número de utilizadores) .....	8
Figura 2.3 - Modelo de arquitectura cliente-servidor .....	13
Figura 2.4 - Modelo WAP .....	18
Figura 3.1 - Evolução das linguagens de <i>markup</i> .....	23
Figura 3.2 - Formato utilizado para os vários dispositivos de saída .....	28
Figura 3.3 - Processo de construção do documento de saída .....	33
Figura 3.4 - Processo de transformação XSLT .....	34
Figura 3.5 - Evolução para o XHTML .....	36
Figura 4.1 - Conversão de um documento em vários .....	42
Figura 4.2 - Acesso à informação no formato WML e HTML .....	43
Figura 4.3 - Filtro de informação HTML/WML .....	44
Figura 4.4 - Conversor personalizado .....	46
Figura 4.5 - Localização do conversor. ....	48
Figura 4.6 - Estrutura de conversão de uma página HTML bem estruturada para um <i>deck</i> WAP .....	49
Figura 4.7 - Estrutura de conversão de uma página HTML mal estruturada para um <i>deck</i> WAP .....	50
Figura 5.1 – Ecrã de desenho do formulário .....	63
Figura 6.1 – Fluxograma utilizado com a aplicação da tabela de correspondência ...	66
Figura 6.2 - Formulário de introdução de dados .....	67
Figura 6.3 - Resultado final, após efectuar o pedido da página .....	68
Figura 6.4 - Adição do componente TidyCOM .....	68
Figura 6.5 - Conversão de HTML para WML .....	69
Figura 6.6 - Eliminação de conteúdo irrelevante .....	71
Figura 6.7 - Exemplos de informação relevante e irrelevante .....	72
Figura 6.8 - Esquema de eliminação de parte da informação .....	73
Figura 6.9 - Processo de eliminação de informação nula.....	74
Figura 6.10 - Resultado da URL composta.....	76
Figura 6.11 - Método 1: Ignorar os elementos.....	77
Figura 6.12 - Método 2: Substituir os elementos .....	78
Figura 6.13 – Método 3: Criar novo <i>card</i> .....	78
Figura 6.14 - Divisão do documento em vários <i>decks</i> .....	80
Figura 6.15 – Análise dos elementos do <i>deck</i> WML .....	81
Figura 6.16 - Agrupar os <i>cards</i> em <i>decks</i> .....	81
Figura 6.17 – Divisão do <i>card</i> .....	82
Figura 6.18 – Processo de divisão dos <i>decks</i> . ....	82
Figura 6.19 – Fluxograma utilizado com a aplicação do documento XSL.....	85

# Lista de Tabelas

Tabela 3.1 - Evolução das várias versões do HTML .....	25
Tabela 3.2 - Estrutura e documento HTML .....	26
Tabela 3.3 - Estrutura e documento HTML com CSS .....	27
Tabela 3.4 - Documento XML .....	30
Tabela 3.5 - Documento DTD.....	31
Tabela 3.6 - Documento XSL .....	32
Tabela 3.7 - Documento HTML .....	33
Tabela 3.8 - Limitação do tamanho do <i>deck</i> nos dispositivos WAP.....	38
Tabela 3.9 - Documento WML .....	39
Tabela 4.1 - Sub-documentos HTML com tabelas. ....	51
Tabela 6.1 - Tabela de correspondências directa .....	70
Tabela 6.2 - Elementos com informação irrelevante. ....	72
Tabela 6.3 - Elementos que podem ser incorporadas nos <i>sub-nodos</i> .....	73
Tabela 6.4 - Elementos com informação visível .....	74
Tabela 6.5 – Elementos com informação não divisível .....	79
Tabela 6.6 - Exemplo de uma página HTML com listas .....	80

# Acrónimos

ANSI	American National Standards Institute
ASCII	American Standard Code for Information Interchange
ASP	Active Server Pages
CERN	European Organization for Nuclear Research
CHTML	Compact HyperText Markup Language
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DOM	Document Object Model
DSSSL	Document Style Semantics and Specification Language
DTD	Document Type Definition
ESTI	Document Style Semantics and Specification Language
GIF	Graphic Interchange Format
GPRS	General Packet Radio Service
GSM	Global System Mobile
HDML	Handheld Device Markup Language
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
I/O	Input/Output
IEC	International Engineering Consortium
IP	Internet Protocol
ISO	International Standards Organization
ISP	Internet Service Provider
ITU	International Telecommunication Union
JPEG	Joint Photographic Expert Group
JSP	Java Server Pages

NCSA	National Center for Supercomputing Applications
PC	Personal Computer
PDA	Personal Digital Assistant
RAM	Random Access Memory
REDIS	Rede Digital com Integração De Serviços
RFC	Request For Comments
ROM	Read-Only Memory
SGML	Standard Generalized Markup Language
SHTML	Small-HyperText Markup Language
TCP	Transport Control Protocol
TIC	Tecnologias da Informação e Comunicação
TTA	Telecommunications Technology Association
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WAP	Wireless Application Protocol
WBMP	Wireless Bitmap Format
WBXML	WAP Binary eXtensible Markup Language
WML	Wireless Markup Language
WSP	Wireless Session Protocol
WWW	World Wide Web
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformation

# 1 INTRODUÇÃO

## 1.1 Contexto

O acesso à Internet a baixo custo a partir das redes *wireless* e a explosão do número de dispositivos móveis com capacidade de comunicação estão a provocar uma mudança na forma como se desenvolvem as aplicações informáticas para esses dispositivos e nas expectativas dos utilizadores relativamente às funcionalidades que estes oferecem [Silva 99].

A Internet móvel adoptou o protocolo Wireless Application Protocol (WAP) para estender a Internet para além dos computadores pessoais. Embora muito se tenha falado sobre as suas limitações e falhas, tal facto também ocorreu com a Internet na sua fase inicial [Nielsen 00], e hoje ela tornou-se uma ferramenta fundamental para as mais diversas actividades.

O objectivo primordial desta tecnologia é fornecer informação aos utilizadores móveis, em qualquer lugar e a qualquer hora, mas a quantidade e qualidade da informação ainda são pouco significativas. Em Abril de 2000 o número de páginas WAP estimava-se entre 50 000 e 1 500 000 [Aheart 00] e, segundo o WAP Forum [WAP 01], em Maio de 2001 existiam cerca de 12 000 *sites* WAP em mais de 100 países. Em contrapartida, estima-se em cerca de 8 milhões o número de *web sites* em 2001 [OLC 01], e num futuro próximo, de acordo com a IDC [IDC 01], o número de páginas *web* ascenderá a cerca de 8.000 milhões. A acrescentar a estes números, foi efectuada uma pesquisa pela palavra “HTML” e “WML” em [www.google.com](http://www.google.com), no dia 18 de Dezembro de 2001, e obteve-se um universo de 270 000 000 páginas que tinham associadas a palavra “HTML” e 549 000 páginas associadas à palavra “WML”, numa razão de aproximadamente 500:1. A mesma operação foi efectuada

em [www.altavista.com](http://www.altavista.com) e os resultados obtidos foram de 56 551 728 para a palavra “HTML” e de 222 262 para a palavra “WML” numa razão de 250:1. Estes dados reflectem o estado de imaturidade da Internet móvel [Aheart 00] comparado com a actual Internet.

De acordo com estes dados pode-se concluir que a probabilidade de um utilizador móvel desejar obter informação (HTML) da Web num formato não suportado pelo telemóvel (WML) é significativa.

## 1.2 Objectivos

O objectivo da dissertação consistiu no desenvolvimento de um sistema de adaptação da informação disponível na Web para o ambiente WAP, com a finalidade de oferecer através de terminais WAP conteúdos apenas acessíveis a dispositivos com características diferentes dos telemóveis. Este sistema efectua a tarefa descrita sempre que o utilizador móvel faz uso do seu telemóvel para aceder à informação a partir de um serviço *on-line* disponível num *web site* conhecido.

Para atingir o objectivo proposto foi necessário um conjunto de tarefas de suporte:

- Estudar as várias gerações das redes *wireless*;
- Definir as arquitecturas WAP e World Wide Web;
- Analisar a evolução da linguagem de formatação de conteúdos;
- Implementar uma aplicação para converter o conteúdo Web para o WAP, com base num conjunto de métodos e técnicas, implementadas ou não, de trabalhos relacionados.

O processo de conversão de documentos HTML em documentos WML que serão visualizados nos dispositivos WAP, não passa somente pela alteração de *tags*. Esta dissertação investiga também os problemas associados ao processo de conversão e apresenta uma solução para o desenho e implementação do sistema.

### 1.3 Organização da dissertação

A dissertação está organizada em sete capítulos seguindo uma ordem temporal de execução.

O actual capítulo apresenta uma introdução ao projecto de investigação, bem como o contexto do trabalho e os principais objectivos.

No capítulo 2 são abordadas no contexto histórico-evolutivo as várias tecnologias de comunicação. No início são descritas as infra-estruturas de suporte à Web (Internet), e as infra-estruturas de rede *wireless*. Neste capítulo são também apresentadas as arquitecturas associadas aos dois tipos de rede: World Wide Web e WAP.

No terceiro capítulo procura-se descrever com detalhe a evolução das várias linguagens de *markup*, dando especial atenção às características das linguagens HTML e WML.

No quarto capítulo é apresentado o problema da existência de conteúdo que não está disponível para os dispositivos móveis. As soluções apontadas para resolver o problema passam pela adaptação de conteúdos Web para os vários dispositivos de saída independentemente das suas características.

No quinto capítulo são apresentados os requisitos necessários e a metodologia de desenvolvimento de um sistema conversor com base na análise de trabalhos desenvolvidos a nível académico e comercial.

No sexto capítulo são descritas as várias fases de desenvolvimento do sistema de adaptação de conteúdos, com base nas linhas gerais apresentadas no capítulo quinto.

Por fim, a conclusão apresenta uma breve revisão da dissertação, as impressões pessoais acerca do tema, assim como os resultados obtidos. E termina com as perspectivas futuras acerca da continuidade do projecto.

## 2 REDES E ARQUITECTURAS DE COMUNICAÇÃO

A rede *wireless* e a Internet são dois tipos de infra-estruturas que possibilitam o acesso a dados, e as quais estão associadas a dois tipos de arquitecturas de comunicação: World Wide Web e WAP. Enquanto que o World Wide Web permite o acesso a partir de um dispositivo ligado fisicamente à rede, como por exemplo um computador pessoal, o WAP estende o acesso aos dispositivos móveis. Neste capítulo apresenta-se o enquadramento geral das várias tecnologias num contexto histórico evolutivo.

### 2.1 Infra-estruturas de comunicação

#### 2.1.1 Internet

A Arpanet, criada pelo Departamento de Defesa dos Estados Unidos no início dos anos 70, interligava vários centros militares e de investigação com objectivos de defesa na época da Guerra Fria [Hahn 96]. A tecnologia desenvolvida permitia a comunicação entre diferentes sistemas de computação e possibilitou a incorporação de outras redes experimentais que foram surgindo ao longo do tempo.

Nos anos 80, a National Science Foundation, utilizando a tecnologia desenvolvida pela Arpanet, criou uma rede de alta velocidade que permitiu aos centros de investigação e universidades ter acesso aos seus supercomputadores. Esta interligação entre as diferentes redes de computadores veio a ser a Internet.



Actualmente a Internet é a mais importante de todas as redes, caracterizando-se por uma enorme diversidade de utilizadores e aplicações e por um ritmo de crescimento surpreendente, sendo acedida diariamente por milhões de pessoas em todo o mundo. Se por um lado esta característica a transforma num óptimo meio para a divulgação de informação, por outro lado exige ponderação na obtenção da informação, uma vez que esta pode ser fornecida por qualquer entidade ou pessoa.

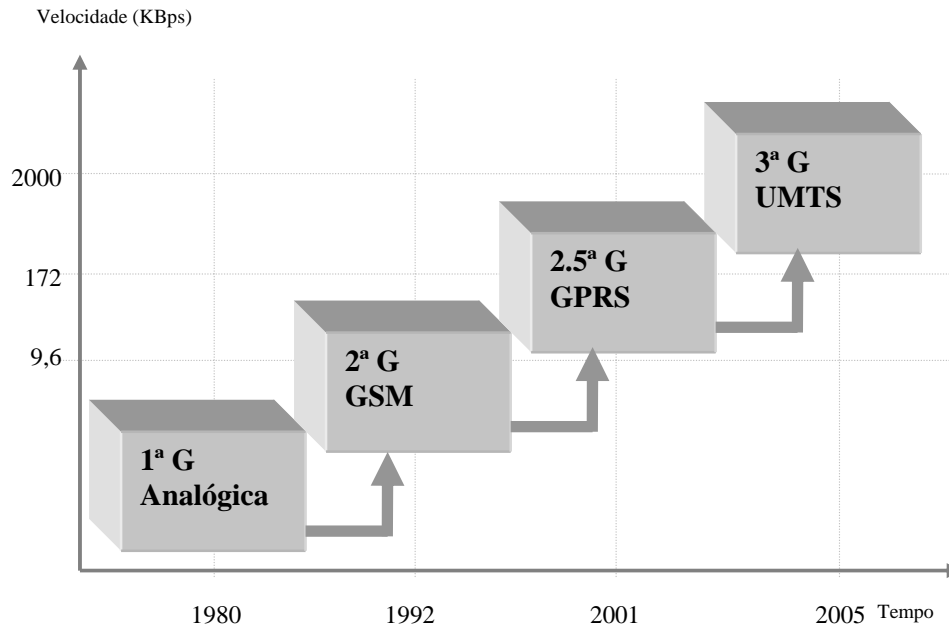
Para interagir com a Internet são necessárias ferramentas que permitem criar a interacção do utilizador com a informação. Estas ferramentas interagem de duas formas:

- Na primeira o utilizador acede à informação. Enquadram-se aqui todas as ferramentas através das quais é possível disponibilizar informação, tais como o Gopher, World Wide Web, File Transfer Protocol e Telnet;
- Na segunda, a informação é enviada directamente ao utilizador. Incluem-se aqui o correio electrónico e as listas de grupos de discussão.

## 2.2 Tecnologias Wireless

As redes *wireless*, de acordo com as características e serviços oferecidos, são divididas em várias gerações como ilustra a figura 2.1.

A primeira geração oferecia serviços de transmissão de voz baseados no sistema analógico. Posteriormente, na segunda geração procedeu-se à optimização da qualidade de transmissão ao adoptar o sistema digital com débitos de transmissão de 9,6Kbps. A partir da segunda geração e meia é possível oferecer vários serviços multimédia com débitos de transmissão máximos na ordem dos 177,2 Kbps [Hjelm 00]. A terceira geração permitirá o acesso a um conjunto mais alargado de serviços multimédia com um débito de transmissão máxima de 2 Mbps.

Figura 2.1 - Evolução das redes *wireless*

### 2.2.1 Global System Mobile

No início dos anos 80 apareceram na Europa os primeiros sistemas de comunicações móveis. Cada país procedeu à instalação de redes locais analógicas que eram específicas para cada um. Tal acção implicava que um telemóvel apenas funcionava no país onde era adquirido devido à incompatibilidade dos vários sistemas. As desvantagens estendiam-se às próprias capacidades das redes, que sendo limitadas levavam à estagnação do número de utilizadores.

Em 1982 nasceu a Conférence des Administrations Européennes des Postes et Télécommunications para poder colmatar as desvantagens. A sua primeira tarefa foi afirmar a necessidade de uma acção concertada para garantir a sobrevivência económica dos sistemas móveis. Foi então criada o Groupe Spéciale Mobile com o objectivo de desenvolver especificações técnicas para uma rede europeia de telecomunicações móveis capaz de suportar os futuros assinantes do novo serviço.

O grupo [GSM 01] definiu os seguintes requisitos para a 2ª geração de telemóveis:

- Boa qualidade de serviço;
- Terminais e serviços baratos;
- *Roaming* internacional;
- Possibilidade de utilização de terminais portáteis;
- Possibilidade de suportar futuros serviços;
- Eficiência na utilização de espectro;
- Compatibilidade com os sistemas RDIS.

Desde o princípio que o grupo acreditava que a nova norma devia utilizar a tecnologia digital, ao invés da analógica, e operar na frequência de 900 MHz. Assim seria possível oferecer uma transmissão de alta qualidade e utilizar com mais eficiência o espectro disponível, possibilitando mais chamadas, e ao mesmo tempo mais seguras, e ainda a adopção de terminais mais pequenos e mais baratos.

Apesar do sistema ter nascido de uma necessidade económica, tal iniciativa necessitava de apoios políticos para ser concretizada. As autoridades reconheceram a validade do projecto, e em 1984 a Comissão Europeia deu o seu apoio formal. Em 1989 o projecto teve um novo impulso com a criação do European Telecommunications Standards Institute e no ano seguinte foi divulgada a maioria das especificações. Entretanto, a partir de 1992 este sistema começou a ser adoptado a nível mundial e em Novembro de 1995 o significado da sigla GSM foi alterado para Global System Mobile.

Em Portugal [ANACOM 02] o GSM foi lançado no segundo semestre de 1992 com a TMN e a Telecel. A Optimus entrou no mercado em 1998.

Actualmente, segundo dados da GSM Association de Janeiro de 2002 [GSM 01], fazem parte desta associação 482 membros, desde fornecedores de equipamento,

fornecedores de serviços e conteúdo, a operadores de redes de telemóveis. O número de clientes ascende a cerca de 646 milhões em todo o mundo, espalhados por 172 países, e o número de mensagens SMS enviadas por mês situa-se em valores próximos dos 30 biliões.

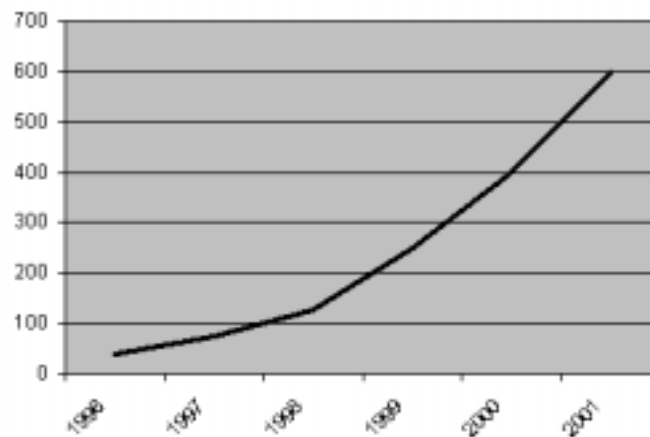


Figura 2.2 - Estatísticas do estado actual do GSM (número de utilizadores)

### 2.2.2 General Packet Radio Service

Apesar dos fabricantes já se encontrarem a desenvolver a terceira geração de telemóveis designado por Universal Mobile Telecommunications System (UMTS) apresentada na secção 2.3.3, existem outras tecnologias que procuram fazer a ponte entre a 2ª e 3ª geração. O General Packet Radio Service (GPRS) é um destes novos avanços apelidados de 2.5ª G devido ao facto de possibilitarem um maior leque de opções quando comparados com os sistemas de 2ª geração. Mas situam-se aquém das características que o UMTS oferece.

O GPRS é um sistema apresentado pelo GSM Association [GSM 01] que permite uma maior capacidade de transmissão de dados. Em comparação com a rede GSM, cuja velocidade é estabelecida pelas operadoras em 9.6 Kbps, o GPRS permite em termos teóricos, uma velocidade máxima de 177.2 Kbps caso utilize todos os recursos do sistema. Outra vantagem refere-se ao facto de as ligações com o servidor serem feitas instantaneamente, pois os utilizadores estão continuamente conectados,

sendo a taxa  o do servi  o efectuada com base na informa  o transmitida/recebida, ao inv  s de ser contabilizado o tempo que se est   ligado.

Apesar do salto que esta tecnologia representa em termos de velocidade e de capacidade, as limita  es existentes em termos de rede impedem que a velocidade m  xima possa ser alcan  ada, sendo a velocidade de transmiss  o, em termos pr  ticos, de 28 Kbps.

Em termos de servi  os, o GPRS oferece aos utilizadores maior mobilidade e obten  o da informa  o sobre a actual localiza  o f  sica do dispositivo. A combina  o destas caracter  sticas fornece uma vasta gama de poss  veis aplica  es que podem ser oferecidas [Cisco 01]. Entre os v  rios servi  os oferecidos pelo GPRS incluem-se:

- Chat;
- *Web browsing*;
- WAP sobre GPRS;
- Imagens;
- Partilha/transfer  ncia de documentos;
- Correio electr  nico;
-   udio.

### **2.2.3 Universal Mobile Telecommunications System**

A evolu  o do GSM surge com o Universal Mobile Telecommunications System (UMTS), apelidada de terceira gera  o. Integrado num projecto que tem como objectivo criar uma norma global ao pretender construir e melhorar a capacidade das actuais tecnologias m  veis, oferecer um aumento da largura de banda e disponibilizar um n  mero maior de servi  os multim  dia.

Em Dezembro de 1998, as entidades regionais reguladoras das telecomunicações (ESTI na Europa, ARIB e TIC no Japão, ANSI nos EUA e TTA na Coreia) chegaram a um acordo denominado Third Generation Partnership Project.

Para além das funcionalidades básicas como telefonar, enviar/receber mensagens ou transmissão de vídeo e áudio, o UMTS permitirá o acesso à Internet a uma velocidade mais rápida que os modems normais, podendo atingir o valor máximo de 2 Mbps. Informação, comércio e entretenimento multimédia estarão disponíveis nos ecrãs, num sistema que integrará as redes de telecomunicações móveis, fixas e por satélite. Para além do *roaming* à escala mundial, o UMTS permitirá ainda a convergência dos vários tipos de redes existentes.

Segundo a Comissão Europeia, os serviços UMTS deverão possuir as seguintes características:

- Serviços personalizados e fáceis de usar, atingindo as preferências e necessidades dos utilizadores;
- Preços acessíveis e competitivos;
- Grande variedade de terminais com preços acessíveis ao mercado e que suportem os serviços avançados do UMTS;
- Capacidade multimédia e uma grande mobilidade;
- Acesso eficiente à Internet;
- Mobilidade entre os vários sistemas UMTS (permitindo o acesso às redes UMTS terrestres e de satélite);
- Compatibilidade entre o sistema GSM e o UMTS, devendo os terminais possuir *dual band* ou poderem funcionar em ambos os sistemas.

Esta nova tecnologia deverá alterar radicalmente a forma como são usados os telemóveis. As pessoas terão o telemóvel mais tempo diante dos olhos do que encostado ao ouvido, pois este passará a ser um dispositivo multimédia, como a televisão ou o computador.

Em Portugal [ANACOM 02] está previsto que em 2002 as operadoras licenciadas (Optimus, TMN, Telecel, Oni) disponibilizem o novo serviço à medida que procederem à instalação da nova rede. Os regulamentos do concurso efectuado estipulam que cada operadora de UMTS possua uma taxa de cobertura da população nacional até 20 por cento no fim do primeiro ano de vigência de licença, 40 por cento no final do terceiro e 60 por cento no final do quinto ano.

## **2.3 Architecturas de comunicação**

Convencionalmente, a maioria dos utilizadores que têm acesso à Internet usam o computador pessoal, e através de uma interface obtêm informações disponíveis em servidores, ou utilizam o espaço da Internet para comunicar. Estes serviços foram possíveis a partir de 1989 com a World Wide Web. Porém, o modo de vida das pessoas alterou-se. O desenvolvimento de dispositivos móveis como o telemóvel ou os PDAs e a criação de um protocolo designado por Wireless Application Protocol (WAP) tornou possível aceder a um conjunto de serviços limitados apenas a dispositivos ligados fisicamente à Internet.

### **2.3.1 World Wide Web**

#### **2.3.1.1 Contexto histórico**

O W3C considera que o ano de 1945, quando Vannevar Bush [Bush 45] escreveu um artigo sobre a estrutura de funcionamento do sistema Memex, foi o ponto de partida da actual World Wide Web. O Memex funcionava como um armazém onde era guardada informação acerca de livros, ficheiros, actas, etc. Cada elemento de informação seria visualizado através da introdução de um código único e, mais importante que isso, era possível anotar as ligações entres vários tipos de informação, como por exemplo, ao aceder a um artigo era possível visualizar a fotografia do autor.

Em 1989 a rede mundial de dados já existia. A Internet evoluiu para além dos propósitos originais como resultado do seu uso por parte da comunidade científica internacional que necessitava de novos sistemas de distribuição da informação. Este era um dos objectivos que Tim Berners-Lee [Berners 89] delineou em 1989 quando apresentou aos seus superiores do CERN a proposta original para o projecto World Wide Web.

World Wide Web, abreviada para “Web”, “WWW” ou somente “W3”, significa algo como “rede global”. O CERN define-o como um "sistema hipermédia distribuído" [Boutell 94]. No princípio foi idealizado como um meio para distribuição de informação entre equipas de investigadores geograficamente dispersos e, dirigia-se em especial à comunidade de físicos de altas energias vinculados ao CERN [Berners 92].

Meses após a proposta original do CERN, a National Center for Supercomputing Applications (NCSA) iniciou um projecto para criar uma interface para a Web. Uma das missões do NCSA era auxiliar a comunidade científica na produção de aplicações. Outro dos seus objectivos era investigar novas tecnologias na esperança de que interesses comerciais surgissem. Assim, iniciou-se o desenvolvimento de uma interface versátil, multi-plataforma para a Web a que chamaram Mosaic. Esta interface designada por *browser* deveria ser capaz de gerir qualquer informação da Web através da interface gráfica orientada ao rato, de exibir texto em várias fontes, dar suporte a áudio, a vídeo e gráficos, e suportar os formulários electrónicos interactivos com vários elementos.

### **2.3.1.2 Arquitectura**

A arquitectura World Wide Web apresentada na figura 2.3 segue o modelo cliente-servidor: um paradigma de divisão do trabalho informático, em que o cliente efectua requisições de serviços a um servidor de Internet e este responde com os dados pedidos [RFC1392 01].



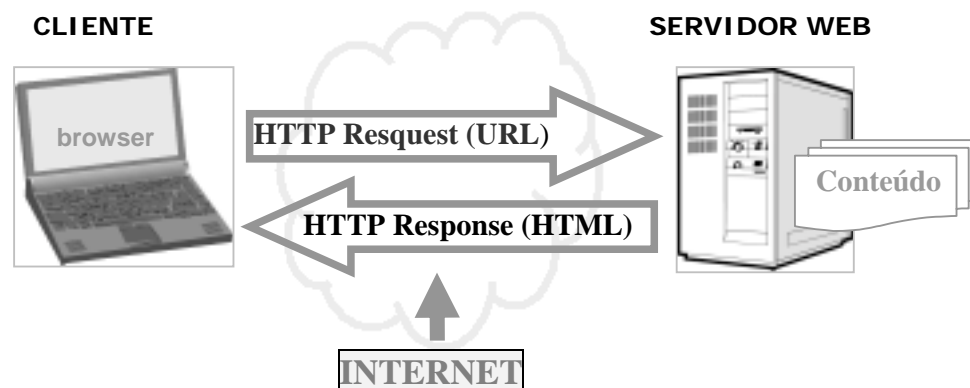


Figura 2.3 - Modelo de arquitectura cliente-servidor

Na Web os postos de trabalho são clientes que solicitam documentos hipertextos aos servidores. Para colocar em funcionamento um sistema como este é necessário:

- **Protocolo de comunicação normalizado:** projectar e implementar um protocolo que permita realizar a comunicação entre o cliente e o servidor. Este protocolo denomina-se por HyperText Transfer Protocol (HTTP).
- **Servidor:** um computador que forneça serviços designado por servidor; pode ser classificado em três categorias:
  - **Servidor Web:** servidor onde um dado recurso reside.
  - **Proxy:** um programa intermediário que actua como cliente ou servidor. Este tipo de servidor reside entre os clientes e os servidores que não têm meios de comunicação directa. Um *proxy* efectua os pedidos como se ele fosse o cliente de origem.
  - **Gateway:** um servidor que actua como intermediário para outro servidor. Um *gateway* recebe os pedidos como se ele fosse o servidor de origem do recurso solicitado.
- **Normalização do conteúdo:** definição de uma linguagem para representar o hipertexto que inclui a informação sobre a estrutura e formato de

representação, para que os *browsers* a possam processar correctamente. Esta linguagem designa-se por HyperText Markup Language (HTML).

- **URL:** todos os servidores e conteúdos na Web são referenciados de acordo com a norma da Internet, conhecido por Uniform Resource Locator.
- **Browser:** aplicação cliente que resolveu o problema do acesso à informação que está armazenada e acessível. Esta informação é apresentada sob a forma de texto, gráficos, áudio, vídeo, etc.

Tal infra-estrutura permite o uso fácil por parte dos utilizadores e aplicações da informação presente na Web.

### 2.3.2 Wireless Application Protocol

Em 1995, quando a Ericsson iniciou um esforço para desenvolver um protocolo global que pudesse oferecer vários serviços para a rede *wireless*, outras empresas seguiram os seus passos no desenvolvimento de outras tecnologias proprietárias para competir neste mercado. Neste grupo incluíram-se grandes empresas como a Nokia e a Phone.com (conhecida no passado por Unwired Planet).

Com o número de empresas a crescer, tornou-se óbvio que a cooperação entre as várias organizações era o único caminho para que as empresas conseguissem obter lucros com este projecto [Hjelm 00], pois facilitava uma maior aceitação por parte de todas as entidades envolvidas, desde os utilizadores a fornecedores de serviços e produtos. Após um período de intenso desenvolvimento e de negociações, esta iniciativa culminou na formação do WAP Forum em 26 de Junho de 1997. As melhores ideias foram aproveitadas e daí resultou a publicação do WAP 1.0 [WAP 98] em Fevereiro de 1998.

#### 2.3.2.1 Compreender a tecnologia

O WAP Forum definiu um conjunto de metas [WAP 01] para promover a tecnologia:

- Incentivar a criação de conteúdos da Internet e serviços de dados avançados para dispositivos móveis;
- Criar uma especificação global usada pelas diferentes tecnologias *wireless*;
- Permitir a criação de conteúdos e aplicações acessíveis aos vários tipos de redes *wireless*;
- Especificar um sistema aberto às tecnologias existentes.

A maioria das tecnologias desenvolvidas para a Internet são destinadas a computadores pessoais com alto poder de processamento, grande quantidade de memória, razoável largura de banda e redes geralmente fiáveis. Por este motivo a utilização de dispositivos móveis obrigou à criação de uma especificação que pudesse manter a independência dos dispositivos, assegurar a interoperabilidade, e ter em atenção as diferenças de rede e dispositivos.

#### **(1) Independência dos dispositivos**

A especificação WAP define a funcionalidade mínima dos dispositivos e estes devem ser projectados para aceitar funcionalidades extras sobre este mínimo.

A independência do dispositivo oferece um conjunto de benefícios:

- Aplicações desenvolvidas para um dispositivo que segue as especificações WAP podem ser usadas noutros dispositivos que implementam a mesma especificação;
- Os programadores não necessitam de escrever versões separadas de código para dispositivos diferentes.

#### **(2) Assegurar a interoperabilidade**

A especificação WAP foi projectada para promover a interoperabilidade fácil e aberta entre os seus componentes fundamentais. Fornecedores de serviços podem escolher o equipamento e o *software* de múltiplos vendedores compatíveis com a especificação WAP, seleccionando cada parte da solução apropriada para o serviço.

A interoperabilidade oferece benefícios claros para os fabricantes de dispositivos portáteis e da infra-estrutura do Internet Service Provider. Se os dispositivos estiverem de acordo com a especificação WAP podem comunicar com qualquer servidor WAP compatível, indiferentemente do fabricante. Igualmente, os fabricantes de servidores WAP compatíveis estão certos de que qualquer dispositivo WAP comunica correctamente com os seus servidores.

### **(3) A diferença da rede**

As redes *wireless* apresentam um ambiente de comunicação mais limitado, comparado com as redes da Internet. Por causa de limitações de potência, espectro disponível e mobilidade, redes de dados *wireless* tendem a ter [WAP 98]:

- Menor largura de banda;
- Menor estabilidade de conexão;
- Disponibilidade imprevisível.

Uma possível solução deve poder superar estas limitações de rede e ainda oferecer uma experiência satisfatória ao utilizador.

### **(4) Diferenças dos dispositivos**

Os dispositivos móveis apresentam várias limitações quando comparados com os computadores pessoais. Devido a limitações fundamentais de consumo e ergonomia, estes dispositivos tendem a ter:

- CPUs menos poderosos;
- Menos memória (ROM e RAM);
- Consumo de potência restringido;
- Dispositivos de saída limitados;
- Dispositivos de entrada de dados limitados.

### 2.3.2.2 Architectura

A principal característica do WAP é a utilização de vários elementos da arquitectura cliente-servidor, acrescidos de um bloco intermédio designado por Gateway WAP. Esta arquitectura oferece uma plataforma de serviços flexível com o propósito de acomodar o acesso ao conjunto de informações oferecidas pela Web [WAP 98].

O WAP define um conjunto de normas que permitem a comunicação entre terminais móveis e servidores Web, incluindo:

- **URL:** a norma da Web é usada para identificar o conteúdo WAP nos servidores de origem;
- **Normalização do conteúdo:** todo o conteúdo WAP tem um formato específico que permite aos *browsers* processá-lo correctamente. O formato especificado para este modelo é o Wireless Markup Language. Foi também criada uma especificação para a linguagem *script* de cliente designada por WMLscript baseada no ECMAScript, análoga ao JavaScript da Web;
- **Protocolos de comunicação:** o protocolo de comunicação WAP, designado por Wireless Session Protocol (WSP), possibilita a comunicação entre o *browser* do terminal móvel e o Gateway WAP e o protocolo HTTP possibilita a comunicação entre este e o servidor Web.

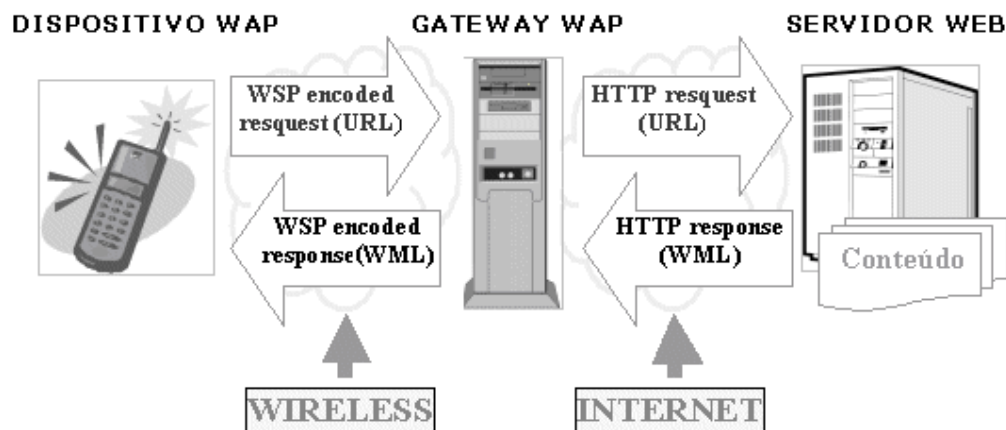


Figura 2.4 - Modelo WAP

Como se pode ver na figura 2.4, o Gateway WAP está posicionado entre o servidor Web e o dispositivo WAP. Este bloco efectua a tarefa de um *proxy* quando traduz as requisições da rede *wireless* para a Internet, com o propósito de facilitar a comunicação. Este bloco também actua como *gateway* porque recebe os dados do servidor Web. Com a adição desta funcionalidade, conteúdos, aplicações e serviços podem ser oferecidos em servidores Web e desenvolvidos com as tecnologias próprias do meio. Um outro módulo incorporado neste bloco designa-se por codificador de conteúdo, responsável pela compressão dos dados transmitidos entre o *gateway* e cliente. O formato final é designado por WAP Binary eXtensible Markup Language (WBXML). Este processo vem dar resposta à limitação da largura de banda utilizada na rede *wireless* e características dos próprios dispositivos WAP.

### 2.3.2.3 O futuro do WAP

As opiniões dividem-se. Existem aqueles que opinam que o WAP tem os dias contados e outros que acreditam que esta tecnologia será eterna e alvo de muitas mudanças e melhorias. Alguns invocam que as características dos dispositivos WAP com tamanho reduzido e dificuldade na entrada de dados, impossibilitam a sua utilização. No entanto, segundo dados fornecidos pelo Strategis Group [Strategis 01], o número de utilizadores com telemóveis ultrapassa já os 500 milhões e o Gartner

Group [Gartner 01] estima que em 2005 os telemóveis serão de longe o aparelho mais usado para aceder à Internet, ultrapassando os mil milhões de utilizadores, e em 2003 95% dos telemóveis vendidos incluirão a tecnologia WAP.

De acordo com estudos [Nielsen 00] realizados nesta área, o uso dos telemóveis não é entrave ao crescimento do WAP, mas sim a falta de serviços e conteúdos *wireless*.

Apesar das dificuldades iniciais que conduziram a uma adaptação lenta, e considerando que a infra-estrutura que suporta o WAP é independente, as novas infra-estruturas móveis como o GPRS e o UMTS poderão aumentar os benefícios da tecnologia WAP principalmente em termos de velocidade. Outro factor de incerteza no modelo WAP diz respeito à linguagem de formatação de documentos: o WML. Como os próximos dispositivos terão mais memória e poder de processamento, então será possível utilizar a especificação de uma nova linguagem, designada por eXtensible HyperText Markup Language (XHTML), que foi desenhada para formatar o conteúdo da Web, em substituição do “extinto” HTML. Esta funcionalidade provocaria o abandono do WML, pois este tornar-se-ia obsoleto.

O WAP é o protocolo mais desenvolvido para redes *wireless*, mas existem outros protocolos como o iMode, construído sobre a arquitectura proprietária da NTT, e o AnyWeb, da Samsung. Cada um desenvolveu um modelo próprio de visualização dos documentos. O WAP utiliza o WML, o iMode utiliza o Compact HTML e o AnyWeb tem o Small HTML. Os *browsers* estão geralmente embutidos directamente no dispositivo móvel e cada modelo tem diferenças.

Em Agosto de 2001 surgiu o WAP 2.0 [WAP2 01] com objectivos bem definidos no sentido de acompanhar as evoluções do mundo *wireless* e ao mesmo tempo responder às preocupações dos vários intervenientes. Os vários modelos referidos convergiram para o WAP 2.0 como resultado de uma iniciativa conjunta entre o W3C e o WAP Forum, contribuindo deste modo para uma especificação global, integrada com as iniciativas da Internet.

Assim, tal como a especificação WAP 1.0 de 1998, a nova especificação suporta os protocolos de comunicação da Internet, Transport Control Protocol (TCP) e HTTP, permite que os dispositivos móveis acessem ao conteúdo da Web, e inclui o suporte para as novas tecnologias como o GPRS e o UMTS.

A nova linguagem de normalização do conteúdo, designada por WML2 é baseado no XHTML. Esta especificação permite que os documentos escritos em XHTML sejam visualizados nos dispositivos que suportam o WML2 e foram também criados mecanismos de visualização dos documentos WML1 nos *browsers* WML2. O WAP 2.0 suporta apresentações de conteúdo de um novo sub-conjunto de Cascading Style Sheets, designado por Cascading Style Sheets Mobile Profile.

Nas versões anteriores, o bloco referenciado por Gateway WAP é um elemento fundamental na comunicação entre o servidor e o cliente. Na nova especificação não é necessária a sua inclusão desde que na comunicação entre o cliente e o servidor seja usada o protocolo HTTP/1.1. No entanto a adição do Gateway WAP pode otimizar o processo de comunicação e providenciar serviços tais como privacidade e localização.



## 3 EVOLUÇÃO DAS LINGUAGENS DE MARKUP

Um dos objectivos do trabalho apresentado nesta dissertação foi efectuar o estudo das várias linguagens de *markup* utilizadas no processo de desenvolvimento do sistema conversor. Assim, pretende-se neste capítulo apresentar as linguagens associadas à formatação de conteúdos Web e a linguagem associada à tecnologia WAP.

A eXtensible Markup Language (XML), também será apresentada como um linguagem que não está associada a qualquer arquitectura em particular, mas como um solução de integração de conteúdos acessíveis por diferentes tipos de dispositivos e arquitecturas.

### 3.1 Definição de markup

O *markup* é definido como uma forma de texto adicionada a um documento para transmitir informação ao utilizador no formato físico ou no formato digital [Whatis 01]. Este conceito foi utilizado inicialmente na impressão em papel para permitir o diálogo entre o compositor e o tipógrafo. Philip Gaskell [Gaskell 95] refere:

*“...muitos exemplos de cópias de impressão sobreviveram do tempo da prensa manual, algumas delas com anotações, ...”.*

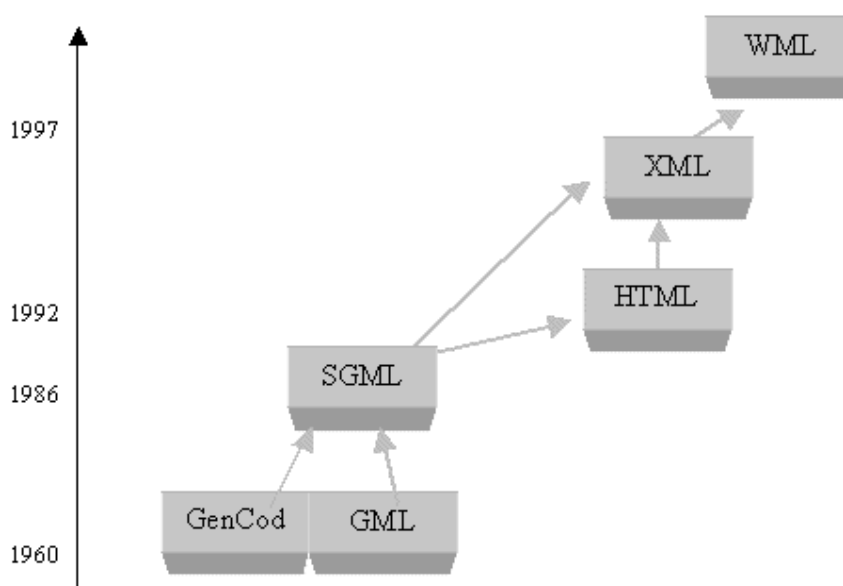
Segundo Ressler [Ressler 97], o *markup* pode ser classificado em três categorias diferentes:

- O *markup* específico indica ao sistema a forma de apresentação do conteúdo do documento. Tipicamente são instruções para formatar a secção do texto a negrito ou centrado e com um tamanho específico.
- O *markup* generalizado, também apelidado de descritivo, descreve o documento ao sistema e não informa o sistema como o documento deve ser visualizado. O SGML é um exemplo deste tipo.
- O *content markup* usa o *markup* generalizado para descrever os elementos semânticos de um documento. Por exemplo, um documento tem vários parágrafos e por sua vez estes têm várias palavras. As *tags* <DOCUMENTO> e <PARAGRAFO> descrevem o conteúdo, não a estrutura.

### 3.2 Linguagens de markup

Desde os primórdios da existência da informática que a manipulação e troca de documentos electrónicos têm sido uma preocupação dos criadores de aplicações. Ao longo dos anos assistiu-se ao aparecimento e à evolução de múltiplos programas que visavam o tratamento electrónico de documentos, sejam eles processadores de texto ou programas de aplicação gráfica [Coelho 00].

Nos anos 60, a Graphic Communications Association definiu uma linguagem designada por GenCode para criar anotações nos documentos dos seus clientes que utilizavam diferentes aplicações e geravam diferentes formatos. Esta solução permitiu integrar os vários documentos provenientes desses clientes. Num esforço paralelo, a IBM desenvolveu outra linguagem designada por Generalized Markup Language na tentativa de resolver os problemas internos de publicação electrónica [Goldfarb 90].

Figura 3.1 - Evolução das linguagens de *markup*

Como os tipos de documento começaram a proliferar, tendo cada um requisitos diferentes e exigindo por isso um conjunto de anotações diferentes, a necessidade de publicar e manipular de uma forma normalizada cada tipo de documento também foi crescendo. No princípio dos anos 80, os representantes do GenCode e da Generalized Markup Language juntaram-se e formaram o grupo designado por Computer Languages for the Processing of Text que tinha como objectivo normalizar o método de especificação, definição e utilização de anotações em documentos [Ressler 97]. Assim, em 1986 foi lançada a Standard Generalized Markup Language (SGML), que no fim da década de 80 era aceite em algumas instituições de grande dimensão.

No laboratório suíço do CERN, Tim Berners-Lee procurava encontrar um sistema de informação que lhe permitisse uma fácil consulta da documentação, bem como a partilha de informação entre vários utilizadores [Coelho 00]. O problema que Lee estava a procurar resolver resultava do facto de ter muita documentação espalhada por múltiplos formatos digitais. Então a alternativa passou pela normalização de um sistema e, com base no SGML, escolheu um conjunto de anotações que adoptou como a linguagem da sua aplicação e criou o HyperText Markup Language (HTML).

Desde 1992 o HTML evoluiu para uma sintaxe *ad-hoc* e, mesmo assim, na versão HTML 4.0 [HTML40 97] lançada em Junho de 1997 fornece apenas um conjunto limitado de *tags*. Nesse momento surgiram razões para viabilizar o aparecimento do eXtensible Markup Language (XML).

### 3.2.1 Standard Generalized Markup Language

O SGML é uma linguagem que passa pela decomposição do documento em 3 vertentes: estrutura, estilo e conteúdo [Goldfarb 86]. A estrutura é visualizada pelo utilizador respeitando determinadas convenções, como os capítulos, as secções, os títulos, os parágrafos, etc. (exemplo: "isto aqui é um título", "este pedaço de texto, entre esta linha aqui e esta linha ali, é um capítulo", "isto é um índice", etc.). Esta estrutura é definida pelo estilo. Estrutura e estilo permitem o suporte do conteúdo. Em função do tipo de conteúdo devem ser definidos em primeiro lugar a estrutura e o estilo em seguida.

Em termos de SGML, começa-se por descrever a estrutura a adoptar para um dado tipo de conteúdo através de um documento designado por Document Type Definition. A cada elemento estrutural corresponde uma *tag* que deve ser incluída no documento. Normalmente estas *tags* marcam o princípio e o fim de cada bloco de texto. Por exemplo, de acordo com as especificações SGML, o texto deve ter uma *tag* <P> no início do texto e </P> para indicar o final do texto.

### 3.2.2 HyperText Markup Language

O HTML é uma linguagem de definição de documentos e de especificação de ligações (*hyperlinks*) entre os documentos que surgiu em 1992 a partir do SGML.

Nas versões iniciais do HTML, as páginas Web apresentavam limitações, tanto em termos de criatividade gráfica, como em termos de conteúdo. No entanto, após a introdução de aplicações designadas por *browsers*, os utilizadores aprenderam a exigir algo mais [Coelho 00]. Depois de um início algo controverso, com vários fabricantes a especificarem as suas próprias linguagens, foi criado em 1992 o grupo

W3C [W3C 01] responsável pela especificação das sucessivas versões da linguagem HTML. Na tabela 3.1 são apresentadas algumas datas importantes segundo o W3C.

Tabela 3.1 - Evolução das várias versões do HTML

<b>1992</b>	Definição original do HTML
<b>1993</b>	Introdução na especificação de algumas definições como tabelas e formulários.
<b>1994</b>	HTML 2.0.
<b>1995/1996</b>	Surge o HTML 3.2, que representa a primeira grande evolução da linguagem. Os novos browsers implementam as suas especificações.
<b>1997</b>	O HTML 4.0 define a separação da apresentação da estrutura com <i>style sheets</i> .
<b>1999</b>	Definição do HTML 4.01, com pequenas correcções da versão anterior.

Actualmente quase todas as páginas disponíveis na Internet são apresentadas no formato HTML. Estimava-se em cerca de 8 milhões o número de *web sites* em 2001 [OLC 01], e no futuro próximo, de acordo com a IDC [IDC 01], estarão disponíveis 8.000 milhões de páginas HTML.

### 3.2.2.1 Estrutura do documento

O projecto dos documentos HTML tem duas características importantes:

- Os documentos HTML são produzidos para fornecer uma estrutura lógica da informação destinada à apresentação de páginas na Web;
- A linguagem HTML contém um número fixo de *tags* para definir a estrutura do documento, cada *tag* tem a sua semântica já definida e a utilização de Cascading Style Sheets (CSS) permite separar a estrutura da apresentação.

Todas as *tags* HTML são delimitadas pelos caracteres menor “<” e maior “>” e são interpretadas pelo *browser*, a fim de permitir a visualização do conteúdo.

Qualquer outra informação que não seja uma *tag* deve ser texto. Na tabela 3.2 é apresentado um exemplo genérico de um documento HTML.

Tabela 3.2 - Estrutura e documento HTML

```
<html>
<head>
<title>Exemplo</title>
</head>
<body bgcolor="#000000">
<h1 align=center> Exemplo de uma página
HTML </h1>
<ul><li>HTML</li></ul>
<font size=4 color="#FF0000">
Texto com cor vermelha!
</font>
</body>
</html>
```

### 3.2.2.2 Cascading Style Sheets

A criação de qualquer documento HTML implica considerar tanto a estrutura da informação como a forma de o apresentar. As primeiras versões do HTML especificavam normas para formatar a informação e definiam as várias partes do documento: cabeçalho, parágrafos, listas, etc.

A especificação da linguagem HTML sofreu evoluções que permitiram a introdução de ajudas ao nível da apresentação. Estas evoluções foram mais sentidas quando a Microsoft e a Netscape introduziram elementos proprietários que influenciavam a forma como a informação era apresentado por cada um dos *browsers*. Estas acções conduziram a um caos porque um documento HTML apresentava diferenças na apresentação em diferentes *browsers*. O caminho seguido consistiu em separar a estrutura da apresentação com a introdução de folhas de estilo em cascata, denominadas por Cascading Style Sheets (CSS).

As CSS são conjuntos de elementos próprios que especificam a formatação, posição e visualização da informação. Assim, quando o documento HTML

apresentado na tabela 3.3 é processado pelo *browser*, este ajusta o conteúdo de acordo com as regras impostas pelas folhas de estilo do documento apresentadas na mesma tabela.

Tabela 3.3 - Estrutura e documento HTML com CSS

```
<html>
<title>Exemplo</title>
</head>
<style type="text/css">
Body { color=#000000; background:#FFFFFF;}
H1 { color=#FF0000; }
</style>
<body>
<h1> Exemplo de uma página HTML </h1>
</html>
```

### 3.2.2.3 Aspectos críticos da linguagem

A preocupação principal dos mentores do HTML consistia em permitir apresentar a estrutura e o conteúdo dos documentos em diferentes plataformas de computadores para que os documentos pudessem ser aceites pelo número máximo de utilizadores [W3C 01]. No entanto o HTML sofre algumas críticas relativamente ao modelo de concepção dos documentos.

O HTML nasceu como uma linguagem simples, com *tags* que ilustravam a estrutura do documento ao invés da sua apresentação, e, à medida que novas versões foram especificadas, adicionaram-se formas de controlar a apresentação. Chegou-se a um patamar em que as páginas são escritas para um *browser* específico e para uma dada versão. O programador não tem controlo sobre o efeito final do documento, que pode variar de acordo com as características técnicas do *browser*, dependendo por exemplo:

- Dimensões da área de visualização;
- Versão da especificação do HTML suportado pelo *browser*;
- Marca e versão do *browser*;

- Limitações dos diferentes dispositivos.

Por outro lado, muitos programadores e aplicações tendem a tirar partido da permissividade dos *browsers* para criar conteúdos HTML com pouco rigor sintático, o que pode trazer problemas na migração de conteúdos [Coelho 00].

### 3.2.3 eXtensible Markup Language

#### 3.2.3.1 Motivação

Quando é solicitado um documento da Web, a informação relativa ao nome e a versão do *browser* está agregada ao pedido e é portanto conhecida pelo servidor Web. Uma vez na posse destes dados, a informação a fornecer ao cliente deverá ter em consideração as características do *browser*. Assim, no limite seria necessário um documento para cada tipo de *browser*. Mas, de acordo com a figura 3.2, se em vez de serem criados  $n$  documentos, fosse criado apenas um, os custos de manutenção decresciam para  $1/n$ . A tecnologia XML é a solução indicada para resolver este problema.

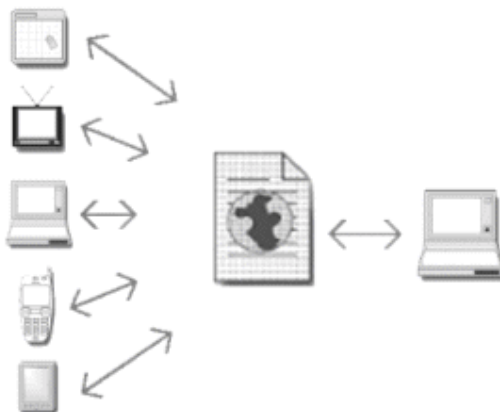


Figura 3.2 - Formato utilizado para os vários dispositivos de saída



### 3.2.3.2 Especificação

A linguagem XML é um formato normalizado pelo W3C em Fevereiro de 1998 [XML10 00] que descreve a estrutura da informação e conteúdo independente de qualquer aplicação, sistema operativo ou base de dados. É uma tecnologia de conteúdos alternativa que complementa o HTML e não o substitui [Coelho 00].

O XML foi introduzido com o propósito de recuperar o poder e flexibilidade do SGML sem a complexidade do HTML. Sendo um subconjunto simplificado do SGML, retém as características mais comuns e remove muitas das complexidades com alto custo de implementação.

Existem objectivos básicos para o XML definidos pelo W3C [XML10 00]:

- Utilizável na Internet;
- Suporte de um grande leque de aplicações;
- Compatível com o SGML;
- Fácil escrever programas que processam os documentos XML;
- Número de características opcionais deverá ser mantido num valor mínimo, que idealmente será zero;
- Documentos XML deverão ser claros e de fácil leitura;
- O desenvolvimento de documentos XML deverá ser rápido, formal e conciso.

Desde 1998 que se assiste a uma migração para tecnologias baseadas no XML. Afinal de contas o XML facilita a tarefa dos programadores na medida em que é possível separar a estrutura da apresentação, e diminui os custos associados à manutenção.

### 3.2.3.3 Estrutura dos documentos

A linguagem XML assemelha-se à linguagem HTML, mas permite definir novo vocabulário ao adicionar *tags* personalizadas. O exemplo da tabela 3.3 apresenta duas notícias que integram um bloco limitado pelas *tags* <corpo> e </corpo>. De referir que é também possível associar a cada *tag* atributos como o exemplo da tabela 3.4 apresenta.

Tabela 3.4 - Documento XML

```
<jornal dia="1 de Janeiro de 2001">
<noticia titulo = "Porto 2001 com novo programa" autor = "J. Rodrigues">
<corpo>
<p>A directora do Porto 2001 apresentou um novo programa no passado dia 25.</p>
<p>Com o novo programa, a directora pretende incentivar os jovens a participar.</p>
</corpo>
</noticia>
<noticia titulo = "Visita do Primeiro Ministro" autor = "R. Santos">
<corpo>
<p> O Primeiro Ministro encontra-se em Ponta Delgada.</p>
</corpo>
</noticia>
</jornal>
```

O documento XML complementa-se principalmente pelas seguintes especificações:

- Document Type Defintion (DTD);
- Extensible Style Language (XSL).

#### (1) Document Type Defintion

Um DTD define regras para a especificação de uma classe de documentos, tais como:

- Os elementos que podem estar contidos no documento. Por exemplo, um jornal possui notícias e estas contêm informação textual;

- Os atributos de cada elemento. Por exemplo, uma notícia pode conter informação acerca do autor e o título;
- A relação hierárquica entre os vários elementos.

Um DTD possui uma propriedade importante: apenas a estrutura lógica de um documento é descrita, não sendo fornecida informação sobre a semântica da apresentação do documento.

Tabela 3.5 - Documento DTD

```
<!ELEMENT jornal (noticia+)>
<!ATTLIST jornal dia CDATA>

<!ELEMENT noticia (corpo) >
<!ATTLIST noticia autor CDATA #REQUIRED>
<!ATTLIST noticia titulo CDATA #REQUIRED>

<!ELEMENT corpo (p) >
<!ELEMENT p ANY >
```

O exemplo apresentado na tabela 3.5 ilustra a definição do documento da tabela 3.3. O elemento “jornal” pode conter um ou mais elementos filhos do tipo “noticia”, e o atributo dia deve ser constituído apenas por texto. O elemento “noticia” apenas pode conter elementos filhos do tipo “corpo” e os atributos devem ser do tipo texto que é obrigatório preencher. O elemento “corpo” apenas pode conter elementos filhos do tipo “p”. Finalmente, o elemento “p” pode albergar qualquer tipo de conteúdo.

## (2) Extensible Style Language

O XSL é uma linguagem utilizada para acrescentar aspectos de apresentação aos elementos de um documento XML. Foi projectada para formatar os dados do documento XML de um modo semelhante ao CSS, usado para formatar documentos HTML.

Desta forma é possível criar múltiplas representações da mesma informação a partir de vários documentos XSL diferentes, aplicados a um único documento XML.

Para exemplificar esta tecnologia utiliza-se o exemplo do documento XML anterior, “noticias.xml”, para apresentar as notícias diárias no *browser web*. De acordo com o exemplo da tabela 3.6, são colocados inicialmente os cabeçalhos e o título será igual ao atributo “dia” do elemento “jornal”; em seguida são processadas todas as notícias de forma individual.

Tabela 3.6 - Documento XSL

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="jornal">
<html>
<head>
<title><xsl:value-of select="@dia"/> </title>
</head>
<body>
<xsl:apply-templates/>
</body>
</html>
</xsl:template>

<xsl:template match="noticia">
<p align="left"><b><xsl:value-of select="@titulo"/><xsl:value-of
select="@autor"/></b></p>
<xsl:apply-templates/>
</xsl:template>

<xsl:template match="corpo">
<xsl:apply-templates/>
</xsl:template>

<xsl:template match="p">
<p align="left">
<xsl:apply-templates/>
</p>
</xsl:template>

</xsl:stylesheet>
```

O que se visualiza corresponde ao documento HTML da tabela 3.7, o qual obedece às regras de formatação definidas pelo documento XSL anterior.

Tabela 3.7 - Documento HTML

```
<html>
<head>
<META http-equiv="Content-Type" content="text/html">
<title>1 de Janeiro de 2001</title>
</head>
<body>
<p align="left"><b>Porto 2001 com novo programa(J. Rodrigues)</b></p>
<p align="left">A directora do Porto 2001 apresentou um novo programa no passado dia
25.</p>
<p align="left">Com o novo programa, a directora pretende incentivar os jovens a
participar.</p>
<p align="left"><b>Visita do Primeiro Ministro(R. Santos)</b></p>
<p align="left">O Primeiro Ministro encontra-se em Ponta Delgada.</p>
</body>
</html>
```

### 3.2.3.4 Transformação com XSL

Embora se refira o XSL unicamente como linguagem, a verdade é que existem dois tipos de conceitos: o de transformação XSL, também designado pelo W3C por eXtensible Stylesheet Language Transformation (XSLT), enquanto que o outro consiste na utilização do XSL como linguagem de formatação [Coelho 00].

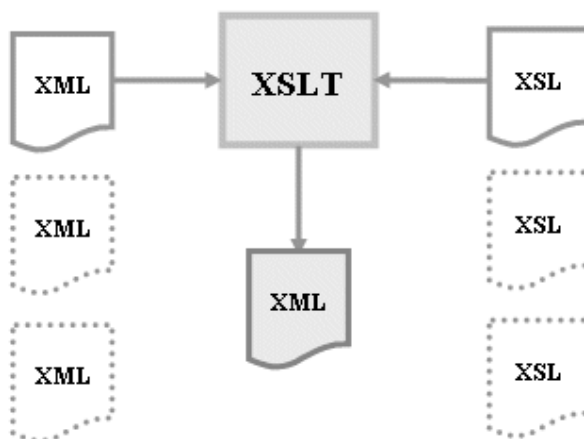


Figura 3.3 - Processo de construção do documento de saída

De acordo com a figura 3.3, uma transformação XSLT lê os dados dos documentos XML e XSL. Baseado nas instruções do documento XSL, é criado um novo documento XML. Embora o XSLT tenha sido especificado para manipular apenas documentos baseados no XML, também suporta documentos de saída como o HTML [Harold 99].

O processo de transformação XSLT pode ser utilizado na máquina cliente ou na máquina servidor. Os *browsers* como o Microsoft Internet Explorer estão habilitados a processar e apresentar documentos XML. Porém, este processo requer recursos que são escassos em dispositivos como os telemóveis [Sergeant 00]. Assim, a transformação efectuada no servidor é considerada como a técnica mais fidedigna, na medida em que se sabe *à priori* que o documento final está correctamente formatado de acordo com as características do cliente, porque é possível seleccionar o XSL específico para cada um e, por outro lado, é mais fácil optimizar uma máquina servidor do que múltiplas máquinas clientes.

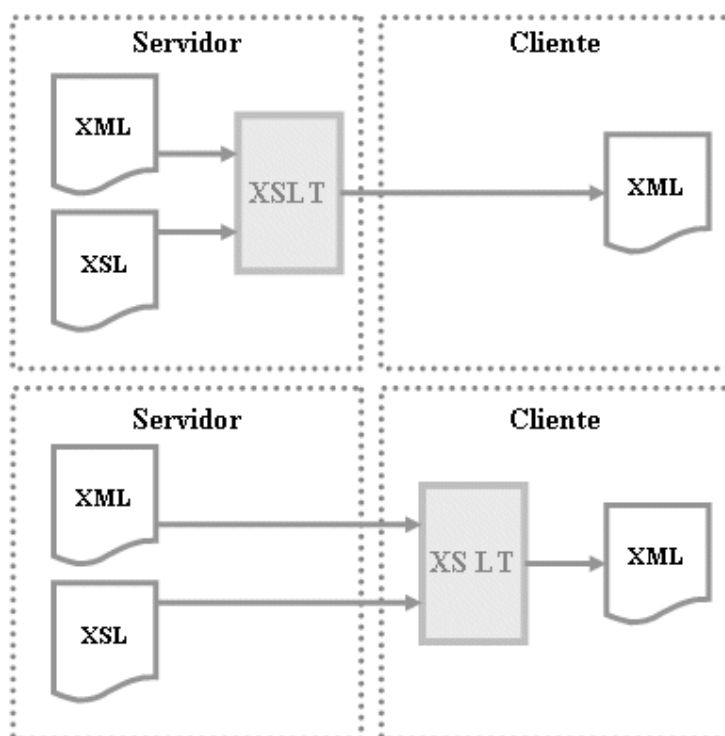


Figura 3.4 - Processo de transformação XSLT

## 3.2.4 eXtensible HyperText Markup Language

### 3.2.4.1 Motivação

O HTML nunca foi especificado para fazer aquilo que um documento XML pode fazer. Ao olharmos para a linguagem HTML e para as respectivas evoluções, constatamos que começou como uma linguagem estrutural e evoluiu para uma fase onde é possível incluir *markup* que afecta a apresentação e não a estrutura [Valentine 01]. A rivalidade entre as várias empresas que fornecem *browsers*, nomeadamente a Microsoft e a Netscape, provocou o caos porque foram introduzidos novos elementos proprietários que aumentaram o grau de dificuldade para programadores.

Hoje o HTML possui um conjunto de limitações, principalmente para os programadores de conteúdo e os fornecedores de aplicações baseados na Web:

- Elementos proprietários criam código com formatação incorrecta, o que torna difícil a sua manutenção, e ao mesmo tempo produz atrasos na visualização;
- Alguns elementos não são optimizados e não existem garantias de funcionamento em várias plataformas;
- Grandes quantidades de *markup* dificultam a descrição dos dados e a sua conversão.

Uma opção é usar a linguagem eXtensible HyperText Markup Language (XHTML), que segundo Graham [Graham 00] é a próxima geração do HTML. Esta linguagem elimina vários problemas estruturais do HTML, combinando os melhores aspectos do HTML e XML numa única tecnologia.

### 3.2.4.2 Diferenças entre HTML e XHTML

Para converter um documento HTML para XHTML é necessário, numa primeira fase, identificar as diferenças entre as duas linguagens. Existe uma verdade

fundamental que qualquer programador de conteúdos tem que entender: o HTML não será reformulado e em sua substituição apareceu a versão 1.0 do XHTML com alterações suaves ao nível da sintaxe, tendo como objectivo tornar mais fácil a migração de documentos HTML4 ou anteriores para XHTML.

Sendo o XHTML o próximo passo evolutivo na família HTML, os elementos da linguagem XHTML são os mesmos da linguagem HTML; apenas a filosofia é alterada com o propósito de a tornar uma linguagem aberta e baseada no XML [Valentine 01].

O XHTML foi recomendado como uma norma pelo W3C em Dezembro de 2000 [XHTML 00] e inclui características que visam a melhoria da acessibilidade, o enriquecimento do conteúdo, a clareza e a compatibilidade entre os vários *browsers* de diferentes dispositivos (PDAs, PCs, telemóveis). Estas características permitem separar a apresentação da estrutura, promovem a formatação correcta dos documentos e facilitam a visualização destes documentos nos dispositivos móveis.

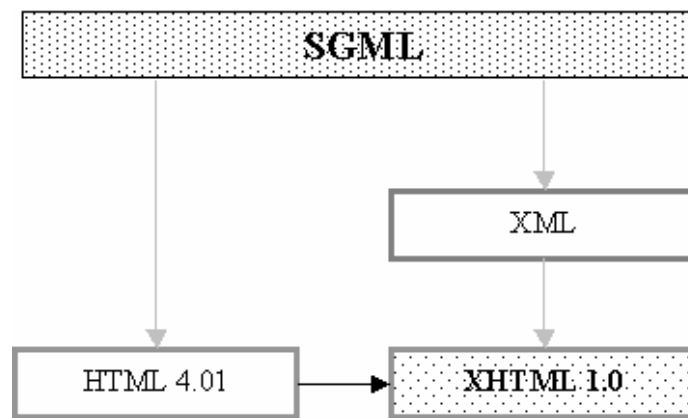


Figura 3.5 - Evolução para o XHTML

A formatação de um documento XHTML apresenta basicamente as seguintes diferenças em relação à formatação do documento HTML, como por exemplo:

- O XHTML é *case sensitive*. Todas as *tags* e os nomes dos atributos devem ser escritos com caracteres minúsculos. Ou seja não é possível ter <P ALIGN = "right">, mas sim <p align = "right">;



- As *tags* `<body>` e `<head>` não podem ser omitidas;
- Todos os atributos devem ter um valor;
- Todos os atributos devem ser citados ( ' ou " ), isto significa que por exemplo `<table border=2>` deve ser substituído por `<table border ="2">`;
- Todas as *tags* devem ser fechadas correctamente. Por outras palavras, uma secção que começa `<p><i>` deve ser fechado com `</i></p>`, ao invés de `</p><i>`;
- Todas as *tags* nulas devem ser fechadas explicitamente ou implicitamente;
- Todas as *tags* vazias devem ser terminadas. Num documento HTML é válido ter `<hr>` e `<br>`, num documento XHTML é válida a sintaxe `<hr />` e `<br />`;
- O elemento “title” deve ser o primeiro da secção `<head>`.

### 3.2.5 Wireless Markup Language

No início da década de 90, a Unwired Planet criou a linguagem Handheld Device Markup Language (HDML) para servir de norma de desenvolvimento de aplicações, conteúdos e serviços para as redes *wireless*. Em 1997 algumas entidades associaram-se e fundaram o WAP Forum e criaram uma nova linguagem designada por Wireless Markup Language (WML) que tem origens na linguagem HDML, e baseada no XML.

#### 3.2.5.1 Características

São várias as características da linguagem WML tendo cada uma delas sido desenvolvida para se adequar às peculiaridades dos dispositivos WAP.

##### (1) Deck e card

Um documento WML é conhecido como um *deck* que agrega várias interacções com o utilizador, conhecidos como *cards*. Este modelo permite que

múltiplos ecrãs sejam carregados para o cliente numa única requisição. Um *card* é a menor unidade visualizada pelo dispositivo WAP [WML 98] e o tamanho depende das características de cada dispositivo como mostra a tabela 3.8.

Tabela 3.8 - Limitação do tamanho do *deck* nos dispositivos WAP

Browser	Tamanho máximo
UP.Browser 3.2	1.492 Bytes
UP.Browser 4.x	2.048 Bytes
Ericson R320	Aproximadamente 3.000 Bytes
Ericson R380	Aproximadamente 3.500 Bytes
Ericson MC218	Mais de 8.000 Bytes
Nokia 7110	Aproximadamente 1.500 Bytes

## (2) Texto e imagem

O WML fornece aos programadores várias formas de introdução de textos e imagens que serão visualizadas pelos utilizadores.

## (3) Entrada de dados

A linguagem suporta vários elementos que permitem ao utilizador introduzir dados e podem ser combinados num ou mais *cards*. O WML possui um pequeno conjunto de ferramentas de entrada e suporta validação de dados através da utilização de *scripts* na máquina cliente

## (4) Suporte internacional

O conjunto de caracteres do WML obedece à norma ISO/IEC-10646.

## (5) Independência da interface do utilizador

A especificação abstracta do *layout* e da apresentação do WML permite aos fabricantes de dispositivos controlarem o projecto da Interface do utilizador para os seus dispositivos.

## (6) Optimização para a largura de banda

A linguagem inclui tecnologias para otimização da comunicação entre o dispositivo e o servidor ao permitir que múltiplos *cards* sejam pedidos numa única transferência.

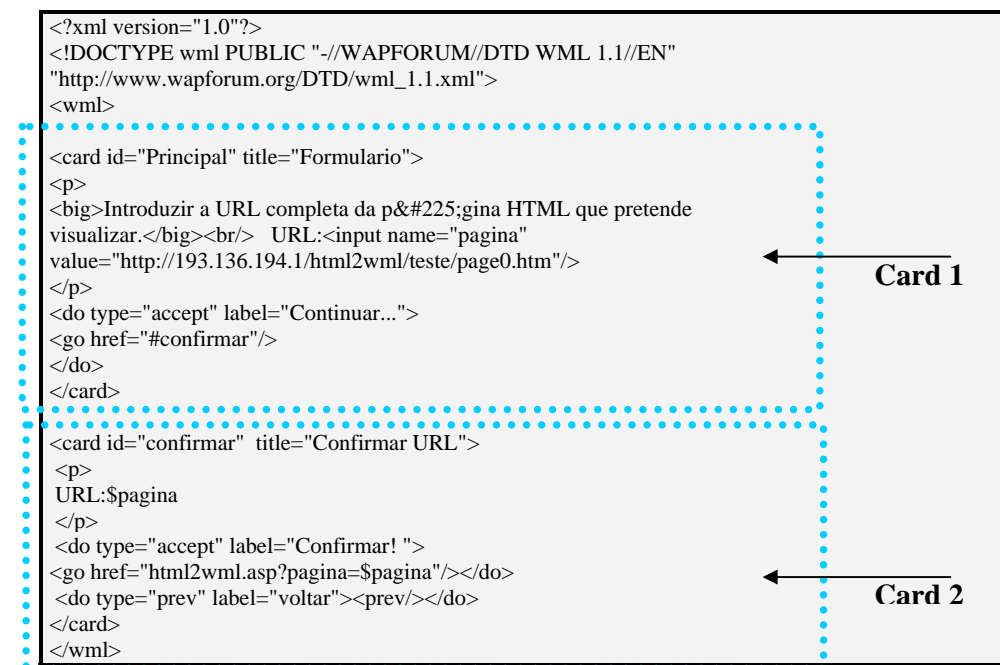
### (7) Manutenção do estado e do contexto

Para maximizar os recursos da rede, o WAP permite que a transmissão de dados entre documentos WML. Os dispositivos móveis podem armazenar em *cache* as variáveis e os documentos WML. Além disso, o valor de cada variável pode ser passado entre vários *cards* do mesmo *decks*.

### 3.2.5.2 Estrutura do documento

A linguagem, tal como o HTML, baseia-se no sistema de *tags*. Permite a formatação de conteúdos como imagens e textos que são acedidos pelos dispositivos WAP. Basicamente um documento WML possui uma estrutura idêntica à apresentada na tabela 3.9.

Tabela 3.9 - Documento WML



### 3.2.5.3 Formatos complementares

A linguagem WML obrigou à especificação de novos formatos com o objectivo de complementar o acesso à informação: foi especificada uma linguagem de *script* utilizada na máquina cliente; foi incluído um novo formato de imagem ajustado aos dispositivos móveis e rede *wireless*; e por fim, foi necessário comprimir a informação num formato conhecido pelo cliente para colmatar a limitação da taxa de transmissão da rede.

#### (1) WMLscript

O WMLscript é uma linguagem de *script* baseada no ECMAScript que realça as facilidades de navegação e apresentação do WML, à semelhança da finalidade do JavaScript para os documentos HTML. Além de suportar a manipulação da interface, adiciona inteligência ao cliente, fornece mecanismos de acesso eficientes aos dispositivos e seus periféricos, e ainda reduz os pedidos aos servidores [Forta 01].

O WMLscript pode adicionar funcionalidades ao WML como por exemplo:

- Validar os dados introduzidos pelo utilizador no formulário;
- Aceder a funções de manipulação do dispositivo, como fazer uma chamada ou enviar mensagens;
- Enviar mensagens localmente;
- Permitir a configuração do dispositivo.

#### (2) WAP Binary XML Content Format

O WML pode ser representado através de um modelo de compressão binária designado por WAP Binary XML Content Format (WBXML), que consiste na associação dos elementos de um *card* a uma tabela de *tokens* pré-definidos. O objectivo deste processo é reduzir o tamanho do documento XML na transmissão para dar resposta às próprias limitações dos dispositivos WAP e da rede *wireless*.

#### (3) Wireless Bitmap Format

O formato Wireless Bitmap Format (WBMP) foi introduzido na tecnologia WAP e é caracterizado por possuir apenas *1-bit* por *pixel* de informação, distinto do GIF e JPEG em termos de *palettes* de cores e em termos de compressão.

Devido às características inerentes do próprio dispositivo WAP o tamanho de cada *deck* é limitado e, como consequência, os gráficos WBMP também sofrem essas limitações. Outro constrangimento é o tamanho limitado do ecrã de visualização.

## 4 ADAPTAÇÃO DE HTML PARA WML

Falar da adaptação dos conteúdos existentes na Web para os dispositivos WAP é falar da adaptação da linguagem de normalização de conteúdos da Web, o HTML, para a linguagem normalizada para o ambiente WAP, o WML. Mas a adaptação pode ser utilizada noutros domínios, nomeadamente a adaptação de qualquer documento de um formato para outro formato diferente. Este processo faz parte do quotidiano de qualquer pessoa que utiliza um computador pessoal, como por exemplo converter um documento da Microsoft Word 97 para Microsoft Word 2000, ou então para o formato PDF, como se pode ver na figura 4-1. Estas tarefas são realizadas ocasionalmente, uma a uma e não de forma automática. Mas se o utilizador pretendesse converter  $n$  documentos, com  $n$  suficientemente grande, tornava-se imperativo desenvolver um algoritmo automático, com ou sem a ajuda do utilizador, para reduzir o tempo de conversão.

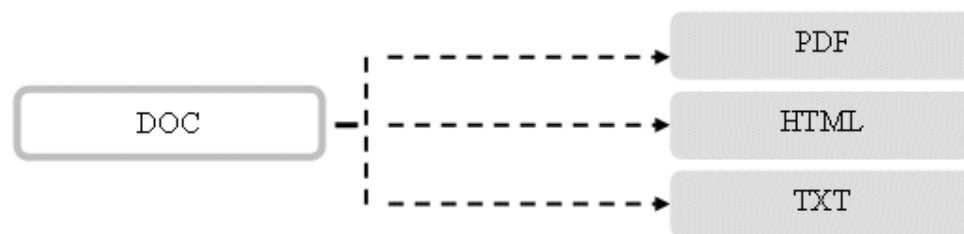


Figura 4.1 - Conversão de um documento em vários

De acordo com a informação referida no capítulo 1, a quantidade e a qualidade dos conteúdos disponíveis no formato WML são pequenas quando comparadas com

as páginas HTML. Se estas páginas fossem acessíveis aos dispositivos WAP, tornar-se-ia vantajoso tanto para os utilizadores como para os fornecedores de conteúdos.



Figura 4.2 - Acesso à informação no formato WML e HTML

## 4.1 Estado da arte

Os fornecedores de conteúdos Web estão a ser pressionados para converterem os conteúdos existentes do formato HTML para o formato WML de forma a serem visualizados nos dispositivos móveis. Esta tendência inicial de “injecção” de conteúdo no formato WML está errada segundo Nielsen [Nielsen 00], e sacrifica a usabilidade para reduzir tempo de comercialização. A maioria dos *web sites* disponibilizados no formato WML são desenvolvidos à custa da simples conversão das páginas de HTML desenhadas para computadores pessoais [Notess 00], sem ter em consideração as diferenças entre os vários tipos de dispositivos de saída.

A maioria dos utilizadores com acesso à Internet usam computadores pessoais; porém, actualmente os utilizadores com dispositivos móveis podem navegar na Internet para aceder a páginas desenvolvidas especialmente para esse tipo de dispositivos [Cummisky 01][Harm-Jam 01].

Hoje, a Internet e os telemóveis fazem parte vida diária das pessoas e muitos esperam que a Internet móvel seja mais fácil de usar. Segundo estudos [Nielsen 00] realizados nesta área, o uso dos telemóveis não é um entrave à utilização da tecnologia WAP, mas sim a falta de serviços. Uma razão para o fraco desempenho é que os programadores não têm em consideração o modelo WAP. De acordo com um

documento do grupo W3C de 15 Março de 1999 [HTML40 99], há muitas características do telemóvel que diferem do computador pessoal (referenciado no capítulo 2), tanto em termos de restrições de hardware e largura de banda, como ao nível da interface.

Em contraste com a grande quantidade de páginas HTML, só algumas páginas WML estão actualmente disponíveis, um pouco à semelhança das dificuldades encontradas pela Web no início os anos 90 [Nielsen 00]. Também existe uma falta de consonância entre os vários fabricantes de dispositivos, resultando numa incompatibilidade séria entre os vários modelos [WAP 01]. Devido a este facto, manter páginas WML acarreta mais custos pois é necessário criar os *decks* de forma a serem visualizados pelos diferentes modelos.

## 4.2 Métodos de conversão

Existem milhões de páginas HTML na Internet e algumas delas são obviamente úteis para os utilizadores de telemóveis. Então a solução seria aproveitar essa informação, e fornecer-la aos dispositivos móveis, utilizando para o efeito um filtro de informação.

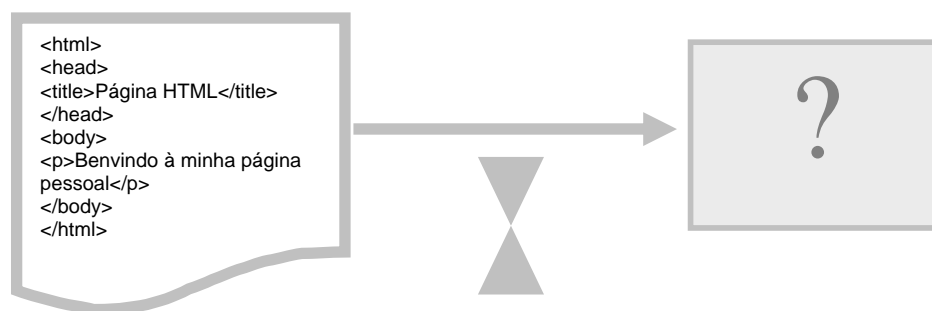


Figura 4.3 - Filtro de informação HTML/WML

Embora já existam no mercado ferramentas comerciais e não comerciais que permitem filtrar a informação ou, por outras palavras, realizar a conversão entre os vários formatos de linguagem de *markup*, essas não representam um processo



mágico [Rysavy 01], pois não será efectivamente fácil condensar a grande quantidade de informação visualizada em ecrãs com dimensões superiores a 640x480 *pixels* e com milhões de cores para os pequenos ecrãs monocromáticos dos dispositivos móveis.

Segundo Arehart [Arehart 2000], existem actualmente quatro perspectivas sobre a adaptação do conteúdo da Web para o WAP.

1. Automática: converter de forma automática os documento HTML para WML;
2. Manual: reescrever o conteúdo das páginas HTML e as aplicações para suportarem os dispositivos móveis;
3. Personalizada: incluir o *deck* WML na página HTML correspondente;
4. Integrada: combinar a conversor automática com o conversor baseado nos documentos XML e XSL.

#### **4.2.1 Automática**

O conversor automático envolve o desenvolvimento de *software* que possa converter um documento projectado para um dispositivo de saída diferente do dispositivo móvel e adaptá-lo de forma a ser visualizado neste dispositivo. Existem no mercado alguns conversores que suportam a adaptação dos conteúdos [Arehart 2000]. As vantagens advêm do facto de acelerar o processo da comercialização, do baixo custo e de serem independentes do desenho original da página HTML. As desvantagens também são visíveis na medida em que adicionam tempo de processamento e a avaliação da usabilidade no resultado final é imprevisível.

Os conversores automatizados podem ser classificados de acordo com a inclusão ou não de configuração. Empresas como a Phone.com, Argo ActiGate e a UC Berkley Pythia incluem conversores automáticos no Gateway WAP cujos algoritmos são os mesmos para todas as páginas. Este processo diminui o tempo de comercialização e o tempo de processamento. A configuração de um conversor pressupõe que cada página HTML é tratada como única, com o intuito de produzir

melhores resultados. As ferramentas Oracle Portal-to Go, Spyglass Prism e Orchid WebShaper enquadram-se neste tipo.

### 4.2.2 Manual

O conversor automático será sempre uma solução temporária e, com o decorrer do tempo, as soluções que utilizam XML para descrever o conteúdo e os documentos XSL para descrever a apresentação farão parte do quotidiano dos fornecedores de conteúdos.

Para disponibilizar conteúdos para os dispositivos WAP, o programador deve escrever outras páginas, estáticas ou dinâmicas, de forma a fornecer uma estrutura e conteúdo de acordo com o meio de divulgação. Embora esta solução seja fácil para *web sites* pequenos, para *web sites* complexos, a solução acarreta um aumento substancial dos custos.

### 4.2.3 Personalizada

Esta solução permite aceder a uma página HTML e ao mesmo tempo o utilizador tem acesso ao *deck* WML correspondente embutido no código HTML sem ter necessidade de fazer qualquer alteração na configuração. Assim que a página HTML é obtida, um aplicativo procura uma anotação que indica se a página possui *tags* personalizadas para a tradução. No caso de resposta afirmativa, é carregado o código personalizado que identifica o documento WML.

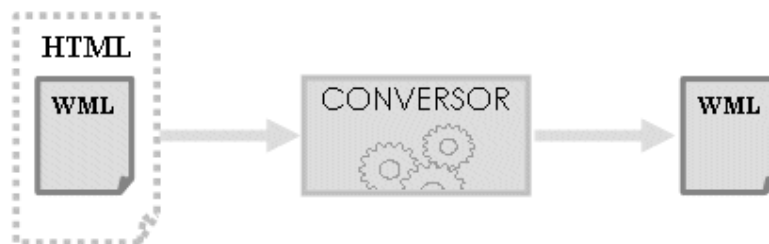


Figura 4.4 - Conversor personalizado

#### 4.2.4 Integrada

Uma solução integrada que combina um conversor automatizado e um conversor baseado nos documentos XML e XSL oferece garantias para uma razoável adaptação das páginas HTML para páginas WML. O *software* da Sun Microsystems iPlanet Wireless Server<sup>TM</sup> [IPlanet 01] segue esta linha de orientação, permitindo ainda a configuração de forma a apresentar melhores resultados.

Esta solução também pode ser adoptada no lado do programador quando existe necessidade de desenvolver páginas para outro formato. O desenvolvimento de um produto que dá ao programador a possibilidade de conhecer apenas as características básicas da linguagem e é responsável pela conversão automática seguindo determinadas regras, configuradas ou não pelo programador, permite que o processo de desenvolvimento de conteúdos em formato WML seja facilitado com a diminuição do tempo de execução e, como consequência, o número de páginas disponíveis aumentaria.

### 4.3 Problemática da conversão

#### 4.3.1 Localização do conversor automático

Existem incertezas onde deveria colocado o conversor: servidor Web, Gateway WAP ou mesmo servidor *proxy*? O servidor Web pode ser uma solução, pois este armazena documento WML e outras aplicações adicionais. Mas esta não é uma tarefa típica de um servidor Web. A conversão interna no Gateway WAP pode ser também uma solução, mas esta especificação deveria ser normalizada para todos os fornecedores de Gateway WAP. O servidor *proxy*, instalado entre o servidor Web e o Gateway WAP, pode conter o conversor [Móro 2001].

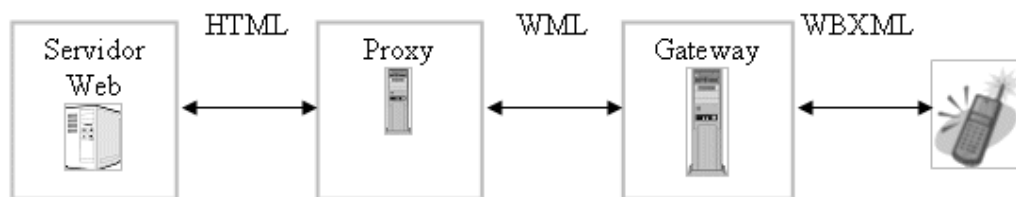


Figura 4.5 - Localização do conversor.

### 4.3.2 Segmentação do conteúdo

Além da conversão descrita, ainda há outro item problemático: a segmentação do conteúdo resultante. O tamanho de cada *deck* WML, devido às limitações do dispositivo (memória, processamento, hardware, etc.), deve ser um elemento a ter em consideração. Por exemplo o Nokia 7110 apenas pode movimentar cerca de 1,4 KB de informação compactada, o Nokia 9210 cerca de 8 KB. Ao invés, os *browsers* dos computadores pessoais não limitam o tamanho da página. Isto levanta uma pergunta pertinente: como proceder à tarefa da segmentação? Seguindo a lógica, só é possível segmentar o conteúdo de forma adequada para cada dispositivo cliente se à *priori* existir informação sobre a sua capacidade. Porém, se o dispositivo actualizar as suas características haveria necessidade de actualizar a aplicação. Na prática é adoptado o limite do dispositivo menos capaz [Móro 2001].

Segmentar o conteúdo pressupõe dividi-lo em vários *decks* para apresentar o resultado nas condições desejadas. Como cada *deck* WML pode ser dividido em vários *cards*, a limitação também se lhes aplica.

## 4.4 Técnicas aplicadas à conversão

O filtro de informação passa pela aplicação de um conjunto de técnicas que têm como objectivo entregar ao utilizador final apenas informação útil. Sob este ponto de vista é imperativa a remoção de informação não relevante e, ao mesmo tempo, a informação a visualizar deve ser adequada às características do dispositivo

de saída. Assim a conversão deve considerar algumas restrições sumariadas a seguir [HTML40 99] [Openwave 01] [Hjelm 00]:

### (1) Níveis de abstracção

Os dados devem ser fáceis de visualizar a um nível superficial e devem ser criados mecanismos de acesso rápido a um nível com menor abstracção. Se são implementados vários níveis de abstracção, então deve ser apresentado ao utilizador esse nível de abstracção.

### (2) Conversão da estrutura

Devido às diferentes características de visualização, é necessário ignorar a estrutura de cada página HTML. Considerando que uma página HTML está dividida em várias secções e está correctamente formatada, a conversão é relativamente fácil: a cada secção corresponde um *card* e usam-se *hyperlinks* para navegar entre os vários *cards*. Porém, devido à limitação de tamanho de cada *deck*, este método pode não ser apropriado.

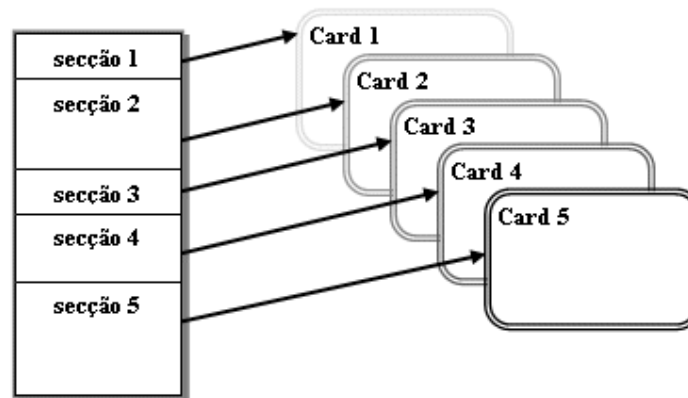


Figura 4.6 - Estrutura de conversão de uma página HTML bem estruturada para um *deck* WAP

Infelizmente a maioria das páginas HTML presentes na Internet ou não estão correctamente formatadas ou é impossível definir secções. A violação sintáctica do HTML pode ser corrigida por algum *software*, como por exemplo o HTML Tidy

[Raggett 01]. Porém existem ambiguidades semânticas que necessitam do envolvimento do programador WAP; caso contrário o resultado da conversão poderá oferecer páginas que não obedecem às exigências dos dispositivos e poderá comprometer a apresentação da informação.

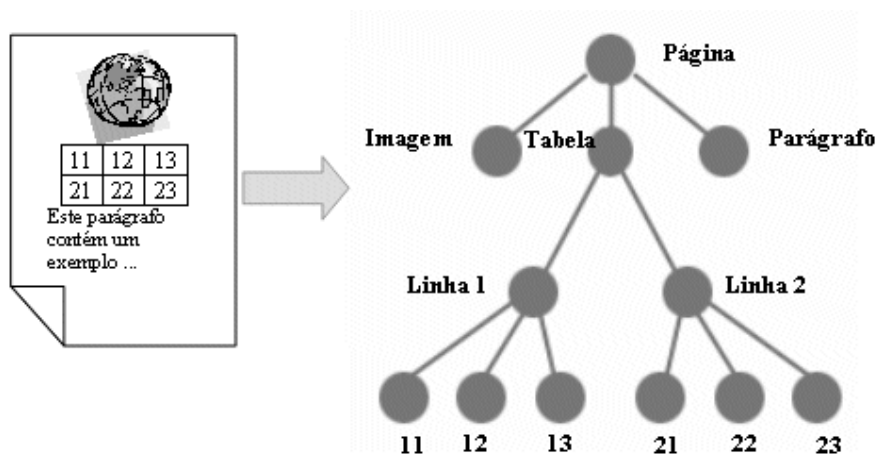


Figura 4.7 - Estrutura de conversão de uma página HTML mal estruturada para um *deck* WAP

O conteúdo das páginas HTML pode ser estruturado segundo a tipologia de uma árvore, em que cada *nodo* representa um elemento HTML como ilustra a figura 4.7. Desta forma é possível criar mecanismos que mantêm a estrutura de informação e alteram a estrutura de apresentação.

### (3) Formulários

De acordo com Nielsen [Nielsen 00], conteúdo com mais de duas linhas pode originar a perda de dados importantes quando são usados simultaneamente os campos de *input* e *select*. De acordo com esta conclusão, deve-se limitar o uso desses campos.

Os dispositivos WAP também suportam formulários, mas estes têm a limitação do mecanismo de entrada de dados, sendo necessário minimizar a acção do utilizador.

### (4) Informação associada à apresentação

Os dispositivos móveis, devido ao seu ecrã pequeno, limitam a capacidade de visualização e por conseguinte o uso de CSS torna-se irrelevante. Assim o conversor deve ter em atenção que o conteúdo deve ser legível sem CSS.

### (5) Tabelas

O uso de tabelas gera na maioria dos dispositivos móveis perda de informação pois estes não têm *scroll* horizontal. Se para tabelas simples o processo de reestruturação é relativamente fácil, o mesmo não se pode dizer para tabelas com algum grau de complexidade. Para tabelas com uma estrutura simples pode-se aplicar uma técnica que consiste em incorporar num único *card* os *hyperlinks* para cada coluna de informação a apresentar num novo *card*.

O Digestor [Bickmore 99] é um sistema que converte automaticamente conteúdo de documento Web para dispositivos móveis (PDAs, dispositivos WAP, etc.). O sistema aplica uma técnica que consiste em considerar cada célula da tabela como um sub-documento.

Tabela 4.1 - Sub-documentos HTML com tabelas.

1	2	3
	4a	
	4.1	4.2
	4b	
	5	6

### (6) Frames

As *frames* podem ser úteis, mas nos dispositivos com ecrãs pequenos só é possível visualizar uma *frame* de cada vez.

Caso as *frames* sejam usadas nas páginas HTML, é necessário criar mecanismos de visualização dos conteúdos, como por exemplo atribuir a cada *frame* um nome para que o utilizador a possa seleccionar e aceder à informação.

## **(7) Imagens**

Nem todos os dispositivos móveis suportam imagens e, mesmo naqueles que as suportam, são pequenos ícones monocromáticos sem informação visual no formato de WBMP. Existe *software* disponível com capacidade de conversão das imagens suportadas pela Web para o formato WBMP [Teraflops 01]. Porém a conversão e apresentação devem ter em consideração que estas são limitadas em tamanho, devido às limitações do ecrã e do *deck*.

Normalmente a técnica a aplicar passa pela substituição da *tag* <IMG> pelo atributo *alt*, ou então redimensionar a imagem de acordo com o dispositivo de saída.

## **(8) Texto**

Devem-se usar frases curtas e simples de modo a revelar o nível seguinte de abstracção e evitar que o utilizador aumente de nível. Este é um princípio básico da acessibilidade na Internet [Hsu 94] e também deve ser um princípio básico nos dispositivos móveis.

A maioria dos ecrãs dos dispositivos WAP só é capaz de exibir aproximadamente 20 caracteres por linha. No entanto, o resultado da conversão pode gerar parágrafos ou frases de texto demasiados grandes, sendo necessárias múltiplas acções do utilizador para visualizar todo o conteúdo.

Alguns *browsers* WAP são capazes de exibir texto com fluxo automático. Mas outras técnicas podem se aplicadas de forma a minimizar a intervenção do utilizador, como por exemplo dividir o texto em vários *cards* ou para cada parágrafo, substituir o texto inicial por um *hyperlink* que aponta para o texto integral.

## **(9) Hyperlinks**

Uma grande vantagem das páginas HTML são os *hypelinks*, que permitem manter uma ligação aberta para outros *web sites*. Num PC com o apontador do *mouse* pode-se facilmente activar uma hiperligação; no entanto, nos dispositivos WAP esta operação torna-se mais incómoda.



Muitos conversores automáticos produzem *cards* de WML com texto e *links* misturados. Uma das técnicas a aplicar será extrair os *hyperlinks* para um *card* único e criar mecanismos de acesso ao *card*, permitindo que o utilizador visualize o conteúdo textual separado dos *hyperlinks*.

#### **(10) Scripts e eventos**

Os dispositivos têm limites de memória e processamento e como tal muitos *scripts* e eventos não são suportados, como o JavaScript. Existe uma solução que passa pela conversão das linguagens de *script* para permitir a sua adaptação ao novo ambiente. Esta tarefa é crucial quando o programador Web faz uso do *script* para manipular os dados do formulário. No entanto, este processo é irrealista se pensarmos que cada programador tem o seu estilo de programação e se pensarmos que a maior parte dos *scripts* de cliente apenas têm acção directa sobre a apresentação.

#### **(11) Desenho do card**

O ecrã é muito pequeno e compacto, o que implica que os *cards* devem ser desenhados de forma a minimizar o número de acções do utilizador e limitar o tamanho.

## 5 REQUISITOS DE UM SISTEMA DE CONVERSÃO

O desenvolvimento de um sistema de conversão é uma tarefa que envolve à *priori* a definição dos requisitos a que deve obedecer. Deste modo pretende-se analisar os projectos relacionados, desenvolvidos com cariz académico ou comercial, com o objectivo de conhecer o estado actual de implantação deste tema.

Finalmente, com base nessa análise, é apresentada a metodologia a utilizar na implementação.

### 5.1 Análise de trabalhos relacionados

#### 5.1.1 Projectos académicos

##### (1) Kansas State University

Deep Kapadia [Kapadia 01] da Universidade do Estado do Kansas apresenta três métodos para a implementação do conversor:

- O primeiro método, desenvolvido em Java, lê o documento HTML, em seguida filtra os dados e adiciona as tags WML. Finalmente guarda o documento com a extensão wml;
- O segundo método utiliza o documento XSL. A conversão desenvolvida em Java recorre ao parser Xerces [Apache 01];

- O terceiro método é baseado em *servelets* do projecto Cocoon [Apache 01].

O autor conclui que a segunda implementação produz melhores resultados, tanto em termos de velocidade de processamento, como em simplicidade.

## (2) Maddingue

Sébastien Aperghis-Tramoni [Aperghis 01] desenvolveu em 2001 um programa em Perl com o nome Html2Wml 0.4.1, que converte documentos HTML, estáticos ou dinâmicos, em *decks* WML.

O Html2Wml apresenta as seguintes características:

- Análise dos *hyperlinks*;
- Limita o tamanho dos *decks*, subdividindo a informação original em vários documentos;
- Substituição da imagem pelo atributo *alt* da tag `<img>`;
- Conversão dos caracteres especiais para ASCII;
- Elimina a informação redundante, como os espaços vazios, parágrafos, etc.;
- Não suporta *frames*.

## (3) Durham

Uma publicação [Cornelius 01] do Departamento Ciências da Computação da Universidade de Durham descreve técnicas associadas à conversão de documentos HTML em XML. A primeira requer uma ferramenta que efectue a conversão sempre que o documento HTML é alterado e a segunda técnica é executada no servidor Web apenas quando é pedida a página HTML.

Finalmente, na posse do documento XML, o *browser* processa-o, usando o documento XSL.

## (4) LazyWAP v.0.5

LazyWAP [Baikov 00] é um programa grátis desenvolvido em PHP e escrito por Mike Baikov. O método envolve somente a substituição de *tags* HTML por *tags* WML e apresenta outras características:

- Os documentos convertidos não contêm tabelas ou *scripts*;
- Os *hyperlinks* são relativos, permitindo a conversão automática dos documentos referenciados;
- Os *decks* com um tamanho maior que 4000 bytes são divididos em vários *decks*.

#### **(5) VTT**

Na 9ª Conferência Internacional da WWW em 15 de Maio de 2000, realizada em Amesterdão, o grupo VTT Information Technology [Kaasinen 00] publicou um *paper* onde apresenta técnicas para o conversor.

A conversão é implementada no servidor *proxy* com o propósito de minimizar a informação, modificando o conteúdo com base nas preferências do utilizador e do sistema de processamento. Inicialmente o documento HTML é validado e corrigido e depois é redesenhado em *decks* e *cards*. O método utilizado tem em consideração as tabelas, formulários, *frames* e imagens.

Os autores referem que a conversão automática de páginas HTML que usam tabelas torna-se mais difícil, e que os botões de *submit* dos formulários causam também problemas, pois alguns podem ser imagens. Ou seja, muitas das páginas HTML disponíveis na Web depois de convertidas em WML criam dificuldades de visualização nos dispositivos com ecrãs pequenos.

#### **(6) Wireless Developer Network**

Um artigo publicado no WDN [Lee 01] explica como converter XML para WML usando XSLT. A conversão referida usa a tecnologia ASP e o objecto designado por Document Object Model (DOM) MSXM2 para apresentar o resultado.

De salientar o facto de que o documento de entrada deve ser formatado de acordo com as regras do XML.

A inclusão deste artigo justifica-se devido ao facto de incluir métodos e termos que não constam nos anteriores projectos.

### **5.1.2 Projectos comerciais**

#### **(1) ZAP2WAP**

Em 2000, Jataayu Software [Jataayu 01], empresa da Índia e subsidiária da Integra MicoSystems, anunciou o lançamento do serviço ZAP2WAP que permite converter automaticamente HTML em WML.

O sistema é baseado num conjunto de parâmetros configuráveis (converte gráficos/remove gráficos, converte as *frames* em *cards*, etc.), e ajuda a otimizar o resultado que é armazenado no servidor Web. Este sistema pode ser executado no servidor Web ou no Gateway WAP.

#### **(2) Phone.Com WAP Gateway**

A Phone.com [Phone 01] inclui um conversor automático que extrai o título, texto e *hyperlinks* da página HTML e apresenta os resultados no dispositivo WAP. Se o conteúdo é superior a aproximadamente 1.4 KB, é dividido em vários *decks*. Este conversor está incorporado no Gateway e não pode ser disponibilizado separadamente.

#### **(3) MetaWrap**

O MetaWrap [Metawrap 01] é uma aplicação que permite a conversão automática do conteúdo Web para os múltiplos dispositivos de saída (PDs, PCs, telemóveis) e o resultado é formatado de acordo com a especificação da linguagem de *markup* apropriada para dispositivo.

A aplicação oferece um conjunto de características tais como:

- Manipulação de texto, permitindo alterar a sua formatação, a cor e o tamanho;
- Manipulação da imagem de forma a entregá-la em condições aceitáveis aos dispositivos de saída. Os dispositivos WAP recebem a imagem redimensionada em tamanho e no número de cores, de acordo com a capacidade do dispositivo;
- Manipulação da interface, com o redimensionamento das páginas;
- Configuração do processo de acordo com o *browser*, com o nível de complexidade do *web site*, a própria página, ou ainda de uma forma mais particular com os elementos da página;

#### **(4) ActiveGate**

O Gateway Activate da Argo [Argo 01] incorpora a aplicação conversor que permite o acesso a informação Web a qualquer utilizador de dispositivos móveis. O resultado visualizado segue um processo específico para cada dispositivo.

Tal como o aplicativo da Phone.com, este permite dividir o conteúdo em vários *decks* e suporta *frames*. Possíveis erros de formatação do conteúdo HTML são corrigidos pela aplicação e a página é formatada de acordo com as regras de um documento XML.

#### **(5) Portal-to-Go**

Esta aplicação da Oracle [Oracle 01] não se limita apenas à conversão de HTML para WML, mas permite configurar múltiplas entradas e saídas, utilizando para o efeito Java Server Pages (JSP) como linguagem de programação e XSL como linguagem de *markup* para formatar o conteúdo armazenado no documento XML.

A URL da página é introduzida e a aplicação divide a página em secções, tais como título, parágrafo, *links*, imagens, tabelas. O utilizador poderá remover, alterar cada elemento, ou então criar novos elementos. Para formatar a saída, o Portal-to-Go completa a conversão com o documento XSL, o qual ao ser aplicado ao documento

XML, oferece o conteúdo a vários tipos de dispositivos. O tamanho máximo de cada *deck* pode ser também configurado.

#### **(6) Spyglass' Prism**

O Spyglass' Prism [OpenTV 01] suporta a conversão de documentos HTML, bem como a configuração para permitir a funcionalidade em vários dispositivos (Windows CE, Pocket PC, Palm Pilots, e telemóveis WAP).

O Prism oferece quatro tipos de conversão:

- Conversão automática de texto e imagem – transformação do conteúdo existente para outro formato. E, relativamente às imagens, permite a alteração da *palette* de cores, resolução, qualidade do JPEG, e outras funcionalidades gráficas como a conversão para o formato WBMP;
- Extração do conteúdo – elimina conteúdo irrelevante do documento fonte. Este processo pressupõe o conhecimento prévio por parte do programador da estrutura e conteúdo;
- Conversão da linguagem de *markup* – permite a conversão de uma linguagem de *markup* para outra, substituindo as *tags* do documento de entrada pelas *tags* apropriadas do documento de saída.
- Conversão personalizada – permite a criação personalizada de outros tipos de conversão, que podem ser desenvolvido em C ou C++.

#### **(7) WebSphere Transcoding Publisher**

É um *software* da IBM [IBM 01] que apresenta o conteúdo baseado na informação associada ao pedido, tal como as características do dispositivo, da rede, ou as preferências do utilizador.

O *software* contém *plug-ins* responsáveis por cada tarefa. Por exemplo, um dos *plug-ins* selecciona e aplica o documento XSL apropriado a cada dispositivo. Outros extraem os elementos mais significativos das aplicações ou dos dados da Web para dispositivos com ecrãs mais pequenos.

O conversor possui outras características:

- Aplicação baseada em Java;
- Software disponível para Linux, Solaris, Microsoft Windows NT e Windows 2000;
- As imagens com formatos comuns da Web (GIF e JPEG) podem ser convertidas para outros formatos, redimensionadas e alteradas as *palettes*;
- Substitui as imagens por *hyperlinks*, remove objectos como *Flash* ou *JavaScript* que não são suportados pelo dispositivo cliente;
- Converte o documento HTML num documento XML (sem erros de formatação), e aplica métodos que modificam o conteúdo de acordo com as exigências do pedido;
- Suportar a segmentação dos *decks*.

#### **(8) ScoutWeb**

A Aether Systems [Aether 01] é responsável pelo desenvolvimento do software ScoutWeb que publica os dados baseados na Web em múltiplas plataformas móveis (Palm, Pocket e WAP). Suporta HTML, WML e XHTML e vários níveis de conversão de imagem de acordo com as características dos intervenientes.

#### **(9) Atinav WAP Gateway**

Este Gateway [Aveaccess 01] permite o acesso e conversão de qualquer recurso da Internet para os dispositivos WAP. As principais características do Atinav WAP Gateway são:

- Conversão de imagens como os formatos GIF e JPEG para o formato WBMP;
- Quando uma página HTML é pedida a um servidor Web a partir de um determinado *browser*, um módulo do Gateway tem a tarefa de conversão da



página fonte para uma página WML. Se o tamanho da página é maior que o limite especificado, então a página será dividida em vários *decks*;

- Uma das características mais poderosas do HTML são os formulários na medida em que estes adicionam interacção com o utilizador. O módulo do Gateway efectua a conversão de todos os elementos do formulário para a especificação correspondente da página WML;
- Optimização para dispositivos móveis.

## 5.2 Requisitos do sistema

De acordo com o estudo realizado nos capítulos anteriores e com a análise realizada a partir dos trabalhos relacionados, existe um conjunto de exigências naturais que devem servir como orientadoras no desenvolvimento de um sistema de conversão:

- Apresentação dos *decks* WML em vários *cards* tendo em atenção as normas da acessibilidade e usabilidade;
- Os *decks* de WML devem ser limitados em tamanho;
- Converter os elementos multimédia presentes no documento HTML para o documento WML;
- Qualquer que seja o documento de entrada, o documento de saída deve estar formatado correctamente.

Fundamentalmente estamos perante a existência de duas linguagens diferentes. Uma (HTML) que é altamente irregular e outra (WML) que foi projectada de acordo com regras bem delineadas. Então um dos requisitos do sistema é tornar o formato dos documentos Web “inteligentes”, isto é, reestruturar a informação de forma a ser bem interpretada.

### 5.3 Metodologia de desenvolvimento

De acordo com a análise efectuada, o sistema a desenvolver poderá abranger diferentes áreas de utilização: aplicações *offline*, módulo embutido no Gateway WAP, ou um serviço fornecido pelo servidor WAP.

O desenvolvimento de uma aplicação *offline* é útil para o programador se o objectivo é efectuar a conversão automática ou semi-automática do conteúdo dos *web sites* criados. Um sistema desta natureza facilita o trabalho de duplicação da informação para os dois tipos de dispositivos: PC e telemóveis.

O módulo para o Gateway WAP converte o conteúdo dos documentos HTML e outros elementos multimédia embutidos, sempre que o utilizador os pretenda visualizar no seu dispositivo uma página HTML.

Por fim, pode ser disponibilizado um serviço aos utilizadores WAP que tenham interesse em visualizar as páginas HTML. Desta forma, sempre que um utilizador pretenda aceder a informação Web deverá, na fase inicial, aceder ao servidor que fornece este serviço e efectuar a escolha da página HTML.

Com base nos três exemplos práticos, o sistema a desenvolver tem como objectivo fornecer o serviço apresentado no último exemplo. Assim, é necessário especificar um conjunto de etapas para efectuar o desenvolvimento.

- Formulário para introdução da URL;
- Acesso à página HTML;
- Conversão para XHTML;
- Conversão de XHTML para WML;
- Segmentação em vários *decks* e *cards*;
- Visualização do resultado.

### (1) Formulário para introdução da URL

Com o apoio de uma ferramenta que permite a edição de documentos WML, implementa-se um documento que serve para introdução da URL da página HTML. O seu desenho deve seguir as normas de acessibilidade e usabilidade para os dispositivos WAP, descritas no capítulo 4.

A screenshot of a WML form. It contains a text box with the label "Introduzir a URL completa da página HTML que pretende visualizar:". Below this, there is a label "URL:" followed by a text input field.

Figura 5.1 – Ecrã de desenho do formulário

O servidor no qual o sistema conversor está armazenado recebe o pedido do formulário anterior e, depois de preencher o campo da URL, o utilizador dá ordem para iniciar o processo de conversão.

### (2) Acesso à página HTML

O sistema acede ao servidor Web para fazer o pedido da página HTML e esta é enviada para o servidor WAP.

Esta fase não consiste apenas em aceder ao conteúdo textual do documento, mas também em aceder a todos os elementos multimédia referenciados na página, como imagens. Ou seja, se um documento possui  $n$  elementos multimédia, o sistema deve efectuar  $n+1$  pedidos e guardar toda a informação no servidor WAP.

### (3) Conversão para XHTML

O terceiro passo consiste em converter o documento mal formatado num documento bem formatado: em XHTML.

Como foi apresentado no capítulo 3, o XHTML usa os mesmos elementos e atributos que o HTML, sendo apenas necessário actualizar a sintaxe. Tihel [Tihel, 2001] adianta dois processos: manual e automático. Qualquer que seja o processo

utilizado, o documento XHTML deve ser criado segundo a especificação definida pelo W3C.

A conversão manual implica um acréscimo insustentável de custos para efectuar a conversão de todas as páginas necessárias e, por isso, a melhor opção passa pela escolha de um processo automático.

#### **(4) Conversão de XHTML para WML**

O quinto passo é em princípio directo de conversão entre duas linguagens com regras bem definidas e comuns. Tendo em atenção que o documento XHTML contém a informação do conteúdo e da apresentação para os PCs, o objectivo neste passo é apresentar o conteúdo ao utilizador no formato WML para os dispositivos WAP.

#### **(5) Segmentação em vários decks e cards**

De acordo com o ponto 4.2.2 do capítulo quarto, é necessário proceder à segmentação do conteúdo em vários *decks* e *cards*, com o propósito de contornar as limitações impostas pelos dispositivos WAP e apresentar o resultado final em condições de ser visualizado de forma fácil e clara.

Como a segmentação implica a divisão da informação, então torna-se imperativa a criação de mecanismos que permitam a navegação entre os vários elementos.

#### **(6) Visualização do resultado**

Finalmente o sexto passo consiste em enviar o conteúdo WML para o utilizador. Pelo facto da conversão resultar em vários *decks* WML, o processo de envio do conteúdo passa pelo envio de vários documentos.

## 6 DESENVOLVIMENTO DO SISTEMA CONVERSOR

O objectivo deste capítulo consiste na apresentação dos módulos do sistema de conversão automático, que são descritos de acordo com a metodologia elaborada no capítulo 5 e, ao mesmo tempo também são apresentados os resultados obtidos com a adição de cada módulo.

O desenvolvimento do sistema implicou a utilização de um conjunto de algoritmos traduzidos para uma ou várias linguagens de programação. De salientar que foram usadas várias linguagens de programação (ASP, PHP e Perl) e aplicadas duas perspectivas para efectuar a adaptação: a primeira utiliza apenas uma tabela de correspondências directas entre o HTML e o WML, enquanto que a segunda utiliza o documento XSL e a transformação XSL designada por XSLT.

A aplicação desenvolvida e apresentada neste capítulo tem como objectivo oferecer o serviço de conversão de documentos HTML para WML aos utilizadores móveis, como está referenciado no capítulo 5. Esta aplicação deverá ser armazenada num servidor WAP disponível na Internet.

### 6.1 Tabela de correspondência directa

São apresentados nas seguintes secções os módulos implementados que têm como objectivo a adaptação usando a tabela de correspondência directa entre o HTML e o WML.

A figura 6.1 apresenta o fluxograma utilizado nesta primeira perspectiva:

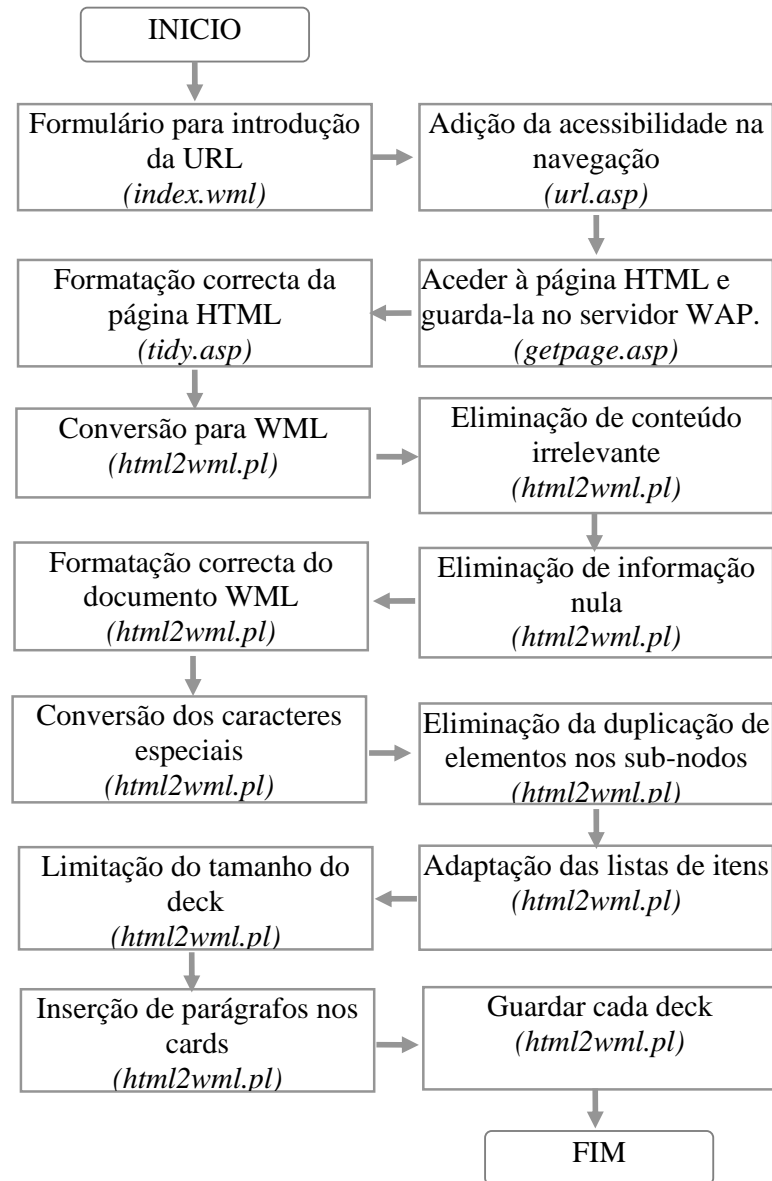
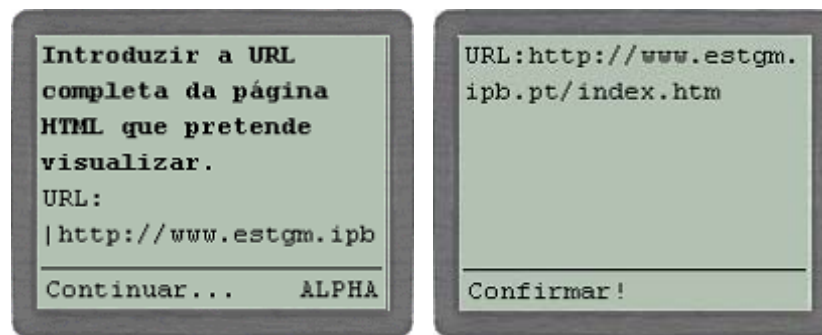


Figura 6.1 – Fluxograma utilizado com a aplicação da tabela de correspondência

Cada um dos módulos referenciados no fluxograma da figura 6.1 será descrito nas secções a seguir, e a sua ordem de execução segue as etapas apresentadas na metodologia de desenvolvimento referenciadas no capítulo 5. Apesar de serem apresentadas seis etapas no capítulo anterior, pressuponha que o número de módulos

a desenvolver cifrava-se em seis. No entanto existiu necessidade de acrescentar novos módulos com o objectivo de promover melhores resultados ou colmatar falhas verificadas durante a fase de desenvolvimento.

### 6.1.1 Formulário para introdução da URL



The figure shows two sequential screens of a WML form. The first screen, labeled 'ALPHA' at the bottom right, contains the text: 'Introduzir a URL completa da página HTML que pretende visualizar.' followed by 'URL:' and a text input field containing 'http://www.estgm.ipb'. Below the input field is a button labeled 'Continuar...'. The second screen contains the text: 'URL:http://www.estgm.ipb.pt/index.htm' and a button labeled 'Confirmar!' at the bottom.

Figura 6.2 - Formulário de introdução de dados

O desenvolvimento de um sistema que satisfaça os objectivos estipulados na análise, necessita de ter uma porta de entrada sob a forma de um *deck* WML. O *deck* inclui um formulário com o campo para a introdução da URL da página HTML.

### 6.1.2 Pedido da página HTML

Este módulo foi desenvolvido em PHP e tem como objectivo efectuar o pedido da página HTML ao servido Web e guardá-la no servidor WAP. De salientar que nesta fase é efectuado apenas um pedido, ou seja, os elementos multimédia incorporados na página, tal como as imagens, são ignorados. Caso não seja possível aceder à página, a mensagem de erro da figura 6.3 é visualizada assim como a solução do problema.

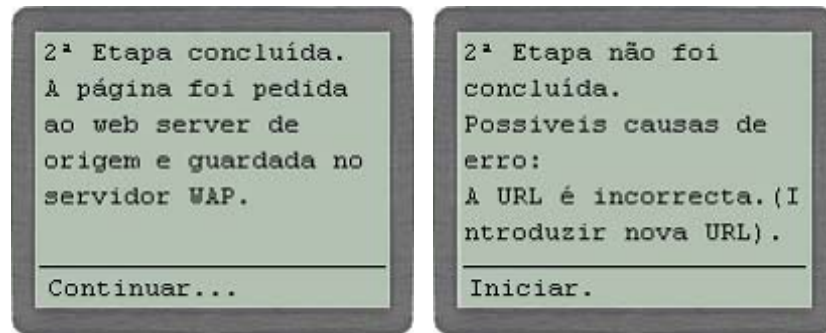


Figura 6.3 - Resultado final, após efectuar o pedido da página

### 6.1.3 Formatação correcta da página HTML

De acordo com o texto do capítulo 3, o XHTML combina os melhores aspectos de HTML e XML numa única tecnologia e, porque é parecida com o HTML, é possível migrar os documentos HTML em novos documentos. Mas esta tarefa de formatação deve obedecer a algumas regras apresentadas no capítulo 3.

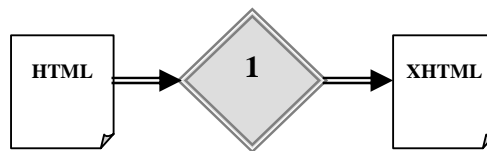


Figura 6.4 - Adição do componente TidyCOM

É necessário acrescentar um módulo referenciado com o elemento 1 na figura 6.4 para resolver o problema das páginas mal formatadas. Este módulo desenvolvido em ASP utiliza o ActiveX designado por TidyCOM desenvolvido por André Blavier [Blavier 01] com base na aplicação HTML Tidy de Dave Raggett's [Raggett 01] e recomendada pelo W3C como uma possível solução para efectuar a tarefa descrita. Este componente realiza um conjunto de tarefas que têm como objectivo tornar o documento “inteligente”, isto é, formatado de acordo com as regras de um documento XML. Entre outras tarefas, permite:

- Analisar e corrigir as *tags* de fim;
- Converter as *tags* para minúsculas;



- Adicionar os caracteres ‘ ou ’ entre os valores dos atributos.

Além das funcionalidades acima descritas, permite ainda a configuração personalizada da saída (converter para XML ou XHTML, converter os caracteres especiais, etc.).

### 6.1.4 Conversão para WML

A partir do momento que se adquire o conhecimento de cada linguagem (XHTML e WML) e das diferenças entre elas, a tarefa imediata é implementar uma tabela de correspondência. A cada *tag* HTML corresponde uma *tag* WML.

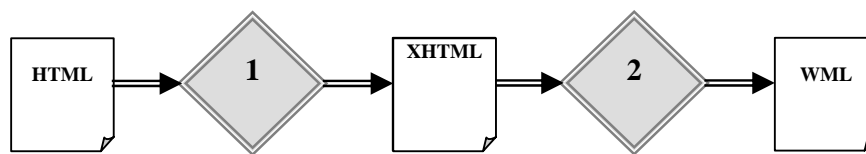


Figura 6.5 - Conversão de HTML para WML

O módulo desenvolvido em Perl realiza a conversão da página de acordo com as técnicas enunciadas no capítulo 4 e envia o resultado para o dispositivo WAP. As tarefas deste módulo podem ser divididas em duas partes:

- Guardar cada elemento numa estrutura com a tipologia de uma árvore, em que cada *nodo* contém a informação de cada elemento;
- Converter cada *tag* HTML para a *tag* correspondente da especificação WML. No entanto, algumas não foram convertidas porque não existe qualquer base de comparação, ou devido a factores relacionados com o aspecto, ou por ser em termos técnicos uma solução inviável.

Na tabela 6.1 são listadas as *tags* HTML que são analisadas e convertidas para *tags* WML.

Tabela 6.1 - Tabela de correspondências directa

HTML	WML
<FORM>	" {--Formulário (acção   nome)--} 
<A>	<a href="hyperlink">
</A>	</a>
</U>	</u>
<U>	<u>
<P>, <DIV>, <OL>, <UL>, <CENTER>, <BLOCKQUOTE>	<p>
</P>, </DIV>, </OL>, </UL>, </CENTER>, </BLOCKQUOTE>	</p>
<TABLE>	<table columns="1">
<TR>	<tr><td>
</TR>	</td></tr>
</TABLE>	</table>
</TD>	 
<B>	<em>
</B>	</em>
<I>	<i>
</I>	</i>
<H1>, <H2>	<strong>
</H1>, </H2>	</strong>
<H3>, <H4>	<em>
</H3>, </H4>	</em>
<H5>, <H6>	<b>
</H5>, </H6>	</b>
<FRAME>	<a href="[nome da página]">[nome do frame]</a>
<IMG>	Substituição pelo texto <b>"*img*"</b>

Este processo apresenta resultados positivos para páginas HTML simples e bem formatadas. No entanto, este não é o panorama actual, pois a maioria delas têm algum nível de complexidade.

### 6.1.5 Eliminação de conteúdo irrelevante

O conteúdo das páginas HTML é convertido através da tabela de correspondências. Este método pode permitir a conversão da informação irrelevante que aumenta o tamanho dos dados transmitidos. Se um determinado elemento do *nodo* não tem correspondência, deve ser eliminado assim como todos os *sub-nodos*

como é exemplificado na figura 6.6. Mas por vezes a eliminação dos *sub-nodos* não é a melhor opção porque representam informação importante, apesar de o elemento do *nodo* não ter correspondência.

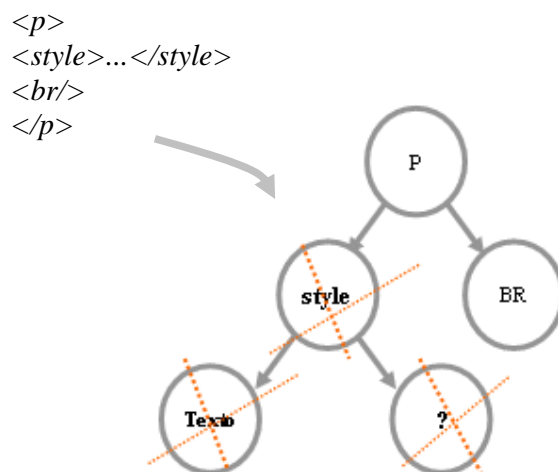


Figura 6.6 - Eliminação de conteúdo irrelevante

Este processo é aceitável para situações idênticas ao exemplo 1 da figura 6.7. Mas ao analisarmos o exemplo 2, se a tabela ignorar o elemento `<CENTER>`, a informação do bloco é ignorada. No entanto esta informação deve e pode ser visualizada. No exemplo 3 são usadas as *tags* do formulário e neste caso a informação agregada deve ser ignorada pelo simples facto do actual sistema não suportar este tipo de informação.

**EXEMPLO 1:**

```
<STYLE FPROLLOVERSTYLE>
a:hover      { text-decoration: none }
</STYLE>
```

**EXEMPLO 2:**

```
<CENTER>
Informação importante para os dispositivos com
o protocolo WAP.
</CENTER>
```

**EXEMPLO 3:**

```
<FORM NAME="pedido" ACTION="enviarpedido.asp"
METHOD="POST">
  nome :<br>
  <INPUT TYPE="text" NAME="nome" SIZE="20">
  descrição :<br>
  <TEXTAREA ROWS="5" NAME="descricao"
  COLS="40"></TEXTAREA>
  <INPUT TYPE="image" BORDER="0"
  SRC="btnenviar.gif" SIZE="20" NAME="submit"
  WIDTH="119" HEIGHT="35">
</FORM>
```

Figura 6.7 - Exemplos de informação relevante e irrelevante

Com base nos exemplos apresentados e respectiva conclusão, o módulo desenvolvido em Perl ignora os elementos dos sub-nodos quando conhece à *priori* o resultado final. Foi criada uma tabela que especifica os elementos que devem ser eliminados bem como o seu conteúdo.

Tabela 6.2 - Elementos com informação irrelevante.

Elementos
form (formulário)
script (programação)
style (elemento que afecta a apresentação - CSS)

*PARA CADA nodo FAZER*

*SE os sub-nodos devem ser eliminados ENTÃO*

*Limpar o conteúdo*

*FIMSE*

*FIMPARA*

### 6.1.6 Formatação correcta do documento WML

Este módulo detecta erros de formatação no documento final e corrige-os. Esses erros resultam da aplicação da técnica referida na alínea 6.1.4. Ora o documento WML deve ser formatado de acordo com regras definidas na especificação desenvolvida pelo WAP Forum.

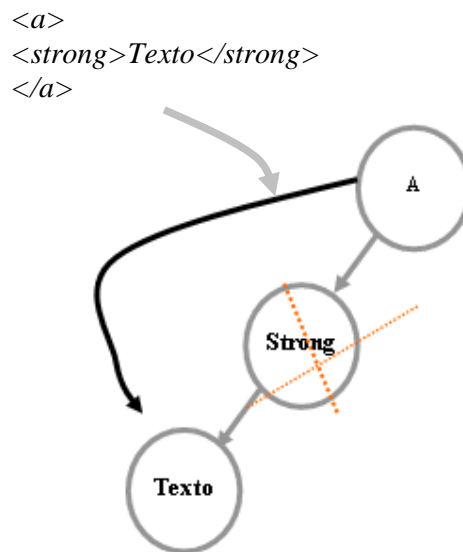


Figura 6.8 - Esquema de eliminação de parte da informação

A aplicação da referida técnica permite que alguns elementos sejam incorporados nos *sub-nodos*, o que impossibilita a correcta visualização do resultado, tal como ilustra a figura 6.8.

O algoritmo implementado em Perl permite a configuração (tabela 6.3) destas situações na medida em que limita a inclusão dos elementos que provocam um resultado incorrecto.

Tabela 6.3 - Elementos que podem ser incorporadas nos *sub-nodos*

Elementos	Elementos permitidos
a (hyperlink)	img (imagem), texto, br (break)

### 6.1.7 Eliminação de informação nula

O resultado final pode apresentar informação nula ou desnecessária. Por exemplo, se uma página apresenta 5 <br/> são visualizadas 4 linhas sem informação. Neste caso específico, apenas são necessárias 2 duplicações para separar a informação. Outra situação verifica-se quando não existe nos *sub-nodos* informação visível para o utilizador. Neste caso o *nodo* e todos os *sub-nodos* são eliminados.

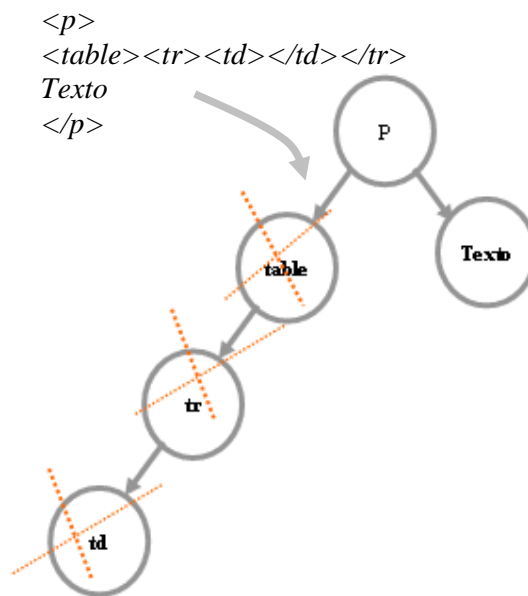


Figura 6.9 - Processo de eliminação de informação nula

O sistema incorpora um módulo desenvolvido em Perl que permite configurar os elementos que representam informação visível, como é apresentado na tabela 6.4.

Tabela 6.4 - Elementos com informação visível

Elementos
a (hyperlink)
Texto
img (imagem)

O resultado final, com a aplicação deste algoritmo apresenta duas vantagens significativas:

- Diminuição do tamanho da informação;
- Visualização de informação correcta, com o mínimo de espaços nulos.

### 6.1.8 Conversão dos caracteres especiais

Os caracteres especiais não são visíveis na forma mais correcta, então existiu a necessidade de incluir um módulo em Perl responsável pela conversão desses caracteres para a notação ASCII. Por exemplo o “á” deve ser substituído por “&#225;”.

De salientar que ao longo do desenvolvimento de qualquer página WML, os caracteres especiais mereceram uma particular atenção, sendo substituídos pelos respectivos códigos ASCII

“.. Introduzir a URL completa da p&#225;gina HTML..”

Outro aspecto a ter em consideração no desenvolvimento de conteúdo em formato WML é a visualização do carácter “\$”. Segundo a especificação WML, este carácter identifica uma variável: \$(var). Deste modo, todos estes caracteres que não identificam qualquer variável devem ser substituídos por dois caracteres - “\$\$”.

### 6.1.9 Adição da acessibilidade na navegação

Suponhamos que o utilizador introduz a localização completa da página X(URL), e aplica a conversão, e esta inclui *hyperlinks* para a página P e para o *web site* S. Caso pretenda aceder à informação da página P, deve introduzir novamente a URL completa. E o mesmo acontece com o *web site* S.

Um módulo desenvolvido em ASP recebe a URL do formulário inicial e faz a gestão adequada dos *hyperlinks* para evitar que o utilizador tenha de introduzir novamente a URL enquanto permanecer no mesmo *web site* ou noutra a que acedeu por intermédio do anterior. Por exemplo a página “index.htm”, com ao URL <http://www.estgm.ipb.pt/index.htm>, possui um *hyperlink* para a página “cursos.htm”. Então não será necessário introduzir novamente a URL completa. Este módulo

permite a junção correcta da URL e o resultado seria dado pela seguinte URL:  
<http://www.estgm.ipb.pt/cursos.htm>.

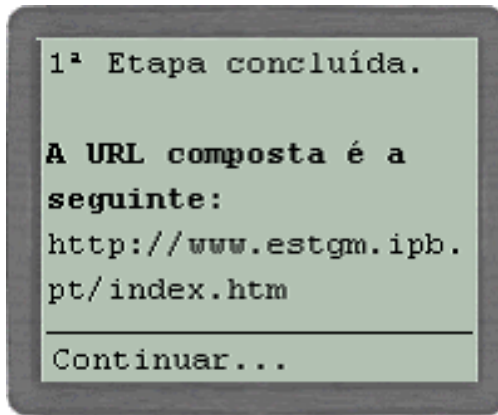


Figura 6.10 - Resultado da URL composta

A introdução desta funcionalidade implicou alterar o processo de conversão de XHTML para WML. Todos os *hyperlinks* encapsulados no *deck* WML são substituídos de forma a permitir que o sistema identifique a URL composta quando o utilizador activar o *hyperlink*. Tomando em consideração o exemplo anterior, o elemento “a” com o atributo href=“cursos.htm” deve ser substituído por href=“url.asp?pagina=cursos.htm”.

### 6.1.10 Eliminação da duplicação de elementos nos sub-nodos

Segundo a especificação de um documento XML, não é possível ter nos *sub-nodos*, elementos iguais a qualquer *nodo* superior. No entanto o processo de conversão pode disponibilizar documentos WML incorrectos e por isso é necessário criar mecanismos para evitar o problema da duplicação que passa por uma ou várias tarefas:

- Ignorar a informação duplicada;
- Substituir todos os elementos duplicados por outros que evitam esta situação;



- Criar um novo *card* com a informação do elemento duplicado e o conteúdo associado.

### (1) Ignorar a informação

A solução mais fácil será ignorar, mas esta operação implica a perda de informação, seja ela relevante ou não.

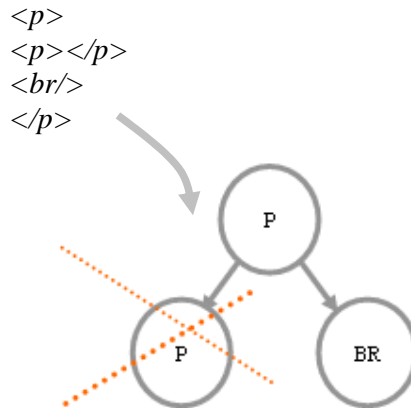


Figura 6.11 - Método 1: Ignorar os elementos

*PARA cada nodo FAZER*  
     *SE existe duplicação nos nodos superiores ENTÃO*  
         *Ignorar informação*  
     *FIMSE*  
*FIMPARA*

### (2) Substituição

A segunda solução pressupõe que existe um erro de formatação do documento, e por conseguinte a tarefa a adoptar deve passar pela substituição de todos os elementos duplicados por outro elemento que não possua um par de *tags* tal como o elemento BR.

*PARA cada nodo FAZER*  
     *SE existe duplicação nos nodos superiores ENTÃO*  
         *Substituir a tag actual pela tag <br/>*  
     *FIMSE*  
*FIMPARA*

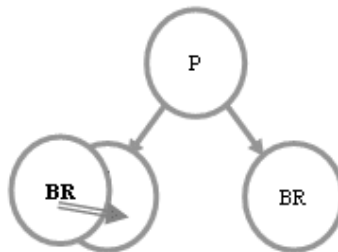


Figura 6.12 - Método 2: Substituir os elementos

Este algoritmo evita a perda de informação; no entanto, a estrutura de apresentação não obedece às regras impostas pela tabela de correspondência.

### (3) Criar novo card

A solução ideal passa por não perder informação e ao mesmo tempo manter as regras da tabela de correspondência directa. Essa solução consiste em agrupar os *sub-nodos* num novo *card*.

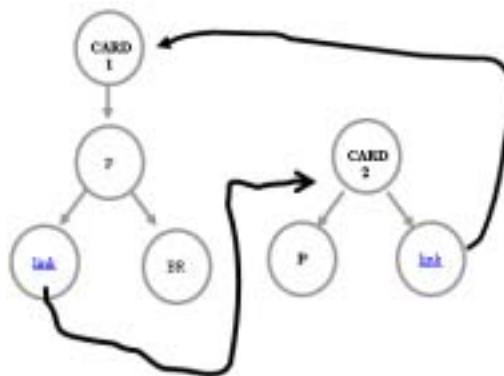


Figura 6.13 – Método 3: Criar novo card

O sistema incorpora um módulo em Perl que verifica se existe duplicação e, em caso afirmativo, o actual *nodo* e *sub-nodos* pertencem a um novo *card*. De referir que deve existir um *hyperlink* do actual *card* para o novo e vice-versa com o propósito de manter uma correcta navegação entre toda a informação.

No entanto este método não se pode adoptar como uma regra geral pois existem elementos com informação agregada que não pode ser segmentada. Por exemplo se o *deck* WML contém um formulário, este deve ser visualizado num único *card*, caso contrário a sua acção é invializada. Neste exemplo em particular o sistema deve extrair toda a informação entre as tags `<form>` e `</form>` e coloca-la num novo *card*, e em seguida substituir todas os elementos duplicados do novo *card* pela tag `<br/>`. Mas pode existir informação que não pode ser dividida, ou cuja incorporação num novo *card* resulta numa divisão excessiva e desaconselhável pois leva a obter *cards* com tamanhos de informação demasiado pequenos.

Desta forma, o módulo permite a configuração dos elementos com informação encapsulada que não pode ser segmentada, assim como configurar se a informação deve ou não se incluída num novo *card*

Tabela 6.5 – Elementos com informação não divisível

Elementos	Novo Card
a (hyperlink)	Não
form (formulário)	Sim

### 6.1.11 Adaptação das listas de itens

As listas numeradas ou não, são características fundamentais para a apresentação de qualquer documento. Para dar resposta a esta necessidade, foi implementado um módulo em Perl que efectua a adaptação de listas de forma correcta, como ilustra o exemplo apresentado na tabela 6.6.

Tabela 6.6 - Exemplo de uma página HTML com listas

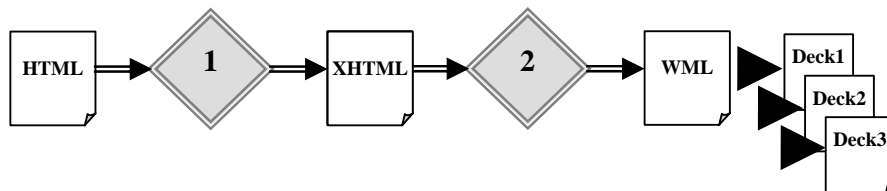
<pre>&lt;ol&gt;   &lt;li&gt;teste1     &lt;ul&gt;       &lt;li&gt;teste11&lt;/li&gt;       &lt;li&gt;teste12&lt;/li&gt;     &lt;/ul&gt;   &lt;/li&gt;   &lt;li&gt;teste 2&lt;/li&gt; &lt;/ol&gt;</pre>	<pre>&lt;p&gt;   1. teste1     &lt;p&gt;       &amp;#149;teste11&lt;br/&gt;       &amp;#149;teste12&lt;br/&gt;     &lt;/p&gt;   &lt;br/&gt;   2. teste 2&lt;br/&gt; &lt;/p&gt;</pre>	<div>1º card</div> <div>1. teste1 ▶ [[+]ver mais...]</div> <div>2. teste 2</div> <div>2º card</div> <div>▪teste11</div> <div>▪teste12</div>
--	--	---

Novo card

Se o documento possui várias listas de itens aninhadas o sistema analisa esta situação e apresenta o resultado tal como ilustra a tabela 6.6, utilizando para o efeito o módulo descrito na secção anterior.

### 6.1.12 Limitação do tamanho do deck

Depois de efectuar a formatação correcta do documento HTML e a respectiva conversão para o documento WML em vários *cards*, ainda resta um problema: a limitação de tamanho do *deck*.

Figura 6.14 - Divisão do documento em vários *decks*

A solução adoptada passa pela segmentação do documento em vários *decks* como ilustra a figura 6.14. Mas a questão que se coloca neste ponto diz respeito à forma como se deve efectuar essa segmentação, sem prejudicar a apresentação do documento final.

A solução passa pela análise de cada elemento dos *nodos* presentes no topo da hierarquia e o tamanho da informação nos *sub-nodos*. Se o tamanho é menor que o valor máximo, a informação é agregada num novo *deck*. Mas se esse tamanho for superior o problema da limitação permanece.



Figura 6.15 – Análise dos elementos do *deck* WML

O módulo implementado em Perl aproveita o facto de a informação estar agrupada em *cards*, analisa o tamanho de cada *card* criado e este é adicionado ao *deck* actual se a operação não provocar um excesso de tamanho do *deck*.

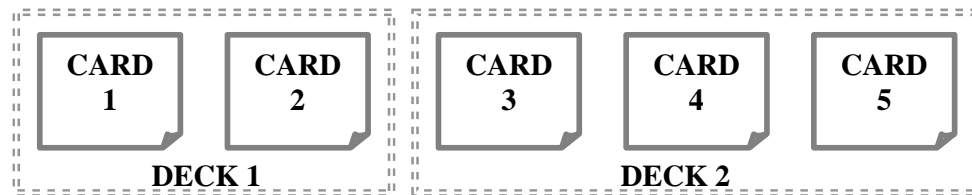


Figura 6.16 - Agrupar os *cards* em *decks*

No entanto a adopção deste algoritmo pode também não ser suficiente quando o tamanho do *card* é superior ao tamanho permitido; nesse caso existe necessidade de o dividir.

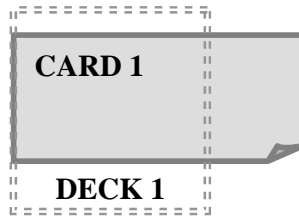


Figura 6.17 – Divisão do card

Verifica-se outro problema quando a página HTML é convertida em  $n$  decks com tamanhos distintos. Por exemplo, o *deck 1* constituído pelo *card 1* e o *card 2* tem um tamanho de 400 *bytes* e o *deck 2* constituído pelo *card 3* com o tamanho de 1900 *bytes*, apresentado na figura 6.16. De forma a minimizar essa diferença, deve-se colocar parte do *card 3* no *deck 1* e outra parte no *deck 2* como ilustra a figura 6.18. Este processo é vantajoso pois permite harmonizar o tempo de *download* de todos os *decks* WML.

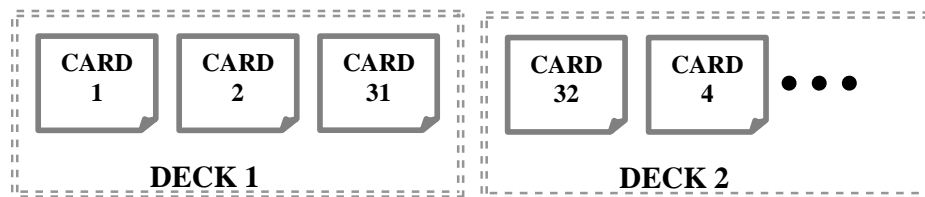


Figura 6.18 – Processo de divisão dos decks.

### 6.1.13 Inserção de parágrafos nos cards

Todos os elementos diferentes do elemento P devem estar encapsulados entre a *tag* de início e de fim do elemento P. Ou seja, qualquer que seja o *card* é imperativo a inclusão do par de *tags* `<p>` e `</p>` para agrupar a informação do *card*.

*SE a tag <p> não existe no início do card ENTÃO*  
*Acréscitar a tag <p> no início do card e a tag </p> no final.*  
*FIMSE*

Este módulo desenvolvido em Perl deve ser o último a utilizar antes de apresentar o resultado final aos dispositivos móveis; caso contrário compromete-se a visualização do conteúdo.

### 6.1.14 Guardar cada deck

Neste momento devem ser criadas condições para que o utilizador aceda ao resultado. O módulo responsável guarda cada *deck* em páginas separadas localizadas no servidor WAP e cria mecanismos de navegação entre os vários *decks* e *cards*.

```

PARA cada nodo FAZER
  SE a tag é link para o próximo card ENTÃO
    incluir o texto:
    <br/><a href="page[deck].wml#card[card]">[+]ver
    mais...</a><br/>
  FIMSE
  SE a tag é link para o card anterior ENTÃO
    incluir o texto:
    <do type="prev" label="[#139;]voltar"><prev/></do>
  FIMSE
FIMPARA

```

## 6.2 Conversão com o documento XSL

O W3C recomenda a utilização do documento XSL para efectuar a apresentação do documento XML. Com base nesta recomendação foi implementado um novo módulo em ASP que inclui a adição do documento XSL ao documento XHTML.

O processo implica a eliminação de alguns módulos desenvolvidos em Perl e apresentados na secção 6.1.1. No entanto, os algoritmos utilizados são aproveitados para projectar o documento XSL que será responsável apenas pela conversão e não pela segmentação em *cards* e *decks*.

Esta abordagem implicou a adição do ActiveX da Microsoft MSXML 4.0 que efectua a transformação XSL (XSLT) com base nos documento XSL e XHTML. O resultado desta operação é guardado num documento WML. De salientar que ainda não estamos perante uma formatação de acordo com a especificação de um documento WML, uma vez que podem ocorrer situações tais como a duplicação de elementos nos *sub-nodos* ou mesmo a falta da inclusão do elemento “P” (parágrafo) a limitar o *card*.

A implementação do documento XSL tornou-se uma tarefa prioritária para responder às exigências do processo de conversão descritas nos módulos eliminados. Assim, são apresentados nos pontos seguintes os módulos a eliminar e, em sua substituição, o documento XSL inclui os respectivos algoritmos:

- Conversão para WML com a utilização da tabela de correspondência;
- Adaptação das listas de indexação;
- Eliminação de conteúdo irrelevante: uma vez implementado o documento XSL, apenas os elementos identificados são tratados e todo o conteúdo que agrega;
- Formatação correcta do documento WML: o documento XSL é responsável pela validação do documento WML no que respeita à inclusão de elementos duplicados. No caso de se verificar a inclusão a duplicação de elementos, o documento XSL não resolve esta situação, uma vez que a resolução do problema será da responsabilidade do módulo que efectua a segmentação em *cards*.

Pelo facto do ActiveX da Microsoft MSXML 4.0 impor que todos os caracteres especiais devem ser tratados, o módulo referenciado na secção 6.1.1.3 efectua esta tarefa, sendo o módulo responsável pela conversão dos caracteres especiais para o código ASCII eliminado.



Os resultados obtidos, como é possível verificar no anexo A, são exactamente os mesmos. No entanto a utilização do documento XSL evita o recurso a linguagens de programação para efectuar tarefas que são especificadas para um documento XSL.

O actual sistema, com a inclusão do processo XSLT, obedece ao fluxograma apresentado na figura 6.19. Note-se que alguns módulos não estão apresentados na figura quando comparada com a figura 6.1 porque os módulos executados à *priori* evitam problemas posteriores, como é o caso da eliminação de conteúdo irrelevante, ou agrupam funcionalidades de outros módulos omitidos, como por exemplo a formatação correcta do documento WML.

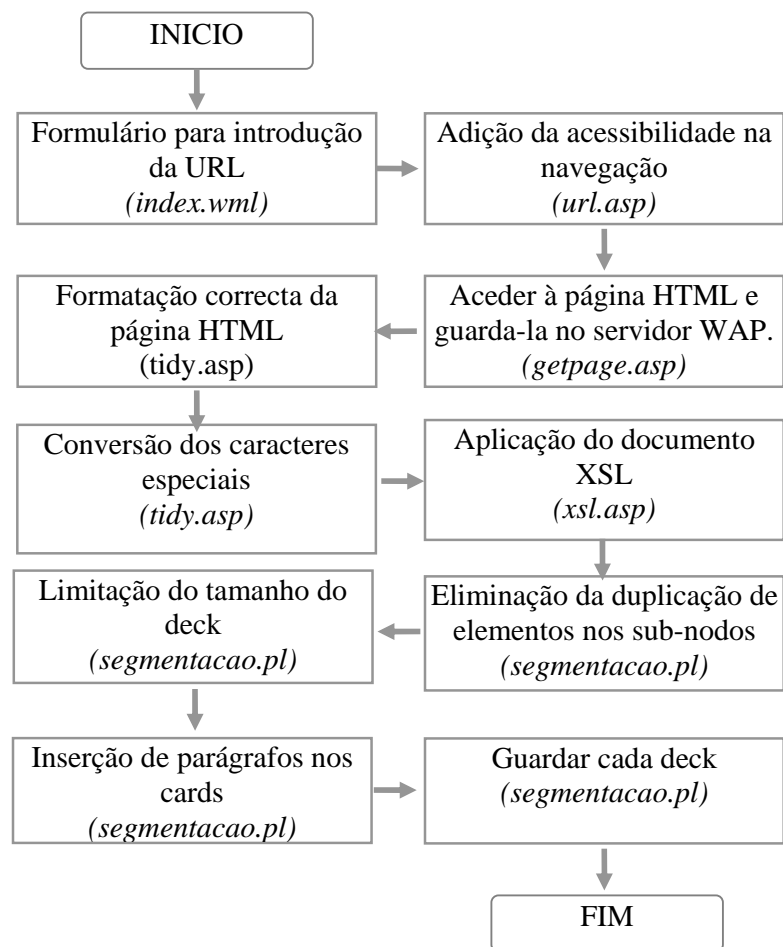


Figura 6.19 – Fluxograma utilizado com a aplicação do documento XSL

## 7 CONCLUSÕES

### 7.1 Síntese

A tecnologia WAP estende o acesso à informação disponível na Internet para além dos computadores pessoais. Os dispositivos móveis equipados com esta tecnologia podem aceder aos escassos conteúdos e às aplicações normalizadas de acordo com uma especificação própria que é diferente daquela que é usada para projectar a informação a disponibilizar através de computadores pessoais.

Afim de apresentar uma solução para dar resposta à diminuta quantidade de informação existente na Internet e adequada ao ambiente WAP, esta tese apresenta uma solução que consiste na adaptação de conteúdos Web para o ambiente dos dispositivos WAP através de uma aplicação. Esta aplicação converte os documentos HTML, sempre que estes são solicitados pelos utilizadores móveis, em documentos formatados de acordo com a especificação da linguagem WML e ignora os elementos multimédia embutidos nas páginas.

A adaptação da informação exigiu um conhecimento prévio das características dos conteúdos presentes na Internet que podem ser úteis aos utilizadores móveis, em seguida foram estudados métodos e técnicas para efectuar a tarefa proposta. Finalmente procedeu-se à implementação da aplicação, de acordo com os requisitos de um sistema conversor.

Para conseguir atingir o objectivo proposto foram também estudadas as tecnologias da rede *wireless* e da Internet, as arquitecturas associadas, assim como as linguagens de *markup* (SGML, HTML, HTML, XML e WML) normalizadas para cada arquitectura.

## 7.2 Discussão dos resultados

Durante a execução foram implementadas duas abordagens para efectuar a conversão dos documentos HTML para documentos WML. Na primeira abordagem foi utilizada uma tabela de correspondências directa entre o XHTML e o WML e, fazendo uso da linguagem Perl implementou-se uma série de algoritmos tendo como objectivo a conversão. Na segunda abordagem o processo de conversão passa pela utilização do documento XSL. Este documento é somado ao documento XHTML e o resultado final é um documento WML. Sendo o XSL uma linguagem própria para a apresentação, possui mecanismos próprios que evitam o recurso a linguagens de programação, como o Perl, utilizado na primeira abordagem.

Na fase posterior, implementou-se a segmentação do documento WML em vários *cards* e *decks* de forma a adaptá-lo às características dos dispositivos WAP.

Cada fase foi projectada com o propósito de manter a sua independência. Ou seja, independentemente da abordagem utilizada na primeira fase, a segunda fase apresenta o resultado final devidamente segmentado. De referir que durante a fase de testes realizada, os resultados são positivos se a tabela de correspondência e outras configurações criadas ou o documento XSL estão escritos de forma correcta.

O tempo de execução do sistema poderia ser melhorado, mas a metodologia utilizada não tinha como objectivo a optimização, mas o estudo e a apresentação de métodos para abordar a conversão através de uma aplicação.

## 7.3 Considerações finais

Actualmente a conversão pode ser facilitada na medida em que a linguagem WML pode desaparecer. No WAP 2.0, o XHTML vem substituir o WML e o trabalho do programador é diminuído pois deixam de ser necessários conhecimentos extras acerca de outra linguagem de *markup*, cingindo-se o trabalho, apenas à

elaboração de soluções para responder aos problemas inerentes aos diferentes tipos de dispositivo de visualização, desde os PCs aos dispositivos WAP.

Um sistema que pretenda ter estes objectivos pode ser aplicada em três localizações distintas: como um serviço WAP, incluído no *Gateway*, ou fazer a parte de uma aplicação que ajude o programador a adaptar *offline* o *web site* desenvolvido em HTML para um novo formato.

Como a utilização da tecnologia WAP ainda está no início, e o interesse é abranger o maior número de utilizadores, o processo de conversão automática cria enormes problemas para atingir esse fim. Actualmente existem dois pontos importantes que devem ser tomados em consideração:

- Os utilizadores que buscam informação através de dispositivos WAP estão à espera de informação diferente daquela que lhes é oferecida a partir de um PC conectado à Internet. Mas aquilo que o conversor automático faz é apresentar tudo;
- A ergonomia dos dispositivos móveis é completamente diferente dos PCs.

Com base nestas considerações, o caminho a seguir será desenvolver conteúdos próprios para o novo ambiente. No entanto este processo acarreta um enorme esforço se se optar pela disponibilização de todos os conteúdos da Web. Mas o desenvolvimento de um conversor semi-automático permite contornar os problemas levantados nos pontos anteriores. A aplicação é responsável pela conversão e pela segmentação e o programador tem a palavra final na medida em que tem liberdade para extrair/modificar/inserir. E finalmente depois de testados os documentos WML podem ser disponibilizados.

Quanto a um possível trabalho futuro, pode-se dizer que os métodos desenvolvidos poderão ser facilmente adaptados para implementar um componente ActiveX, que tenha como objectivo a conversão e segmentação. Ou seja, dada a URL da página HTML, o componente é responsável pela correcta conversão, utilizando um documento XSL externo. Este componente poderia ser então disponibilizado para

o desenvolvimento de qualquer aplicação que tenha como objectivo fornecer a informação Web aos dispositivos WAP, *online* ou *offline*.

## Referências

- [Aether 2001] Aether Systems. <http://www.aethersystems.com/>.
- [ANACOM 02] GSM World. <http://www.anacom.pt>.
- [Apache 01] Apache XML Project. <http://xml.apache.org/>.
- [Aperghis 01] Aperghis-Tramoni, S. – Html2Wml. 2001. <http://maddingue.free.fr/techie/>. Outubro de 2001.
- [Arehart 00] Arehart, C. et al – Professional WAP. 1ª ed. Wrox Press Ltd., 2000. ISBN 1-861004-44. p. 9-41.
- [Argo 01] Argo Group. <http://www.argogroup.com>.
- [Aveaccess 01] Aveaccess. <http://www.aveaccess.com>.
- [Baikov 00] Baikov, M. – Real-time HTML2WAP converter. 2000 <http://www.tourbase.ru/zink>. Outubro de 2001.
- [Berners 89] Berners-Lee, T. - Information Management: A Proposal (1989). <http://www.w3.org/History/1989/porposal.html>. Outubro 2001.
- [Berners 92] Berners-Lee, T. et al - World-Wide Web: An Information Infrastructure for High-Energy Physics (1992). Apresentado na "Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics", em Londe-les-Maures.
- [Bickmore 99] Bickmore, T., Girgensohn, A. e Sullivan, J. W. – Web Page Filtering and Re-Authoring for Mobile Users. The Computer Journal, Vol 42, Nº 6, 1999, p. 534-546.
- [Boutel 94] Boutel, Th. - Frequently asked questions about World Wide Web (1994). <http://www.ibiblio.org/boutell/faq/wwwfaq.txt>. Setembro 2001.
- [Bush 45] Bush, V. - As We May Think (1945). <http://www.w3.org/History/1945/vbush/>. Junho de 2001
- [CISCO 01] CISCO - White Paper GPRS White Paper. <http://www.cisco.com>. Outubro de 2001.

- [Coelho 00] Coelho, P. – XML: A nova linguagem da WEB.FCA. Outubro de 2000. ISBN 972-722-214-5.
- [Cornelius 01] Cornelius, B. – Processing XML. 2001.  
<http://www.dur.ac.uk/barry.cornelius/>. Outubro de 2001.
- [Cummisky 01] Cummisky, J., Purdy, G., Nozick, T. – White Paper: Delivering Internet content to the palm of your hand.  
<http://www.digitalpaths.com/wp/>. Maio de 2001.
- [Forta 01] Forta, B. et al - WAP development with WML and WMLScript. SAMS Publishing. 2001. ISBN: 85-352-9724-4.
- [Gartner 01] Gartner Group. <http://www.gartner.com>.
- [Goldfarb 86] Goldfarb, C. - ISO/IEC JTC1/SC34 Document Description and Processing Languages.1986. <http://www.sgmlsource.com/>.  
Dezembro de 2001.
- [Goldfarb 90] Goldfarb, C. A Brief History of the Development of SGML. 11 de Junho de1990. <http://www.sgmlsource.com/history/sgmlhist.htm>.  
Maio de 2001.
- [Graham 00] Graham, I. – XHTML 1.0: Language and Design Sourcebook. Wiley. 2000. ISBN: 0-471-376485-7.
- [GSM 01] GSM World. <http://www.gsmworld.com>.
- [Hahan 96] Hahn, H. - The Internet: Complete Reference. 2ª ed. Osborne, McGraw-Hill. 1996. ISBN 0-07-882138-X
- [Harm-Jan 01] Harm-Jan, P. – Telematics paper on mobile Internet.  
<http://stuwww.kub.nl/people/p.t.devrieze/telematics/>. Maio de 2001.
- [Harold 99] Harold, E. - XML Bible. Chapter 14: XSL Transformations. 1999. Hungry Minds, Inc. ISBN: 07-645-3236-7.
- [Hjelm 00] Hjelm, J. – Designing Wireless Information Services. 1ª ed. John Wiley & Sons, Inc., 2000. ISBN 0-471-38015-6..
- [Howard 01] Howard, P. – Converting HTML to WML. <http://www.vbxml.com>.  
Outubro de 2001.
- [Hsu 94] Hsu, J., Johnston, W. and McCarthy, J. – Active outlining for HTML documents: An X-mosaic Implementation. In 2ª Conferência Internacional de WWW. Chicago. 1994.  
<http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/HCI/hsu/>.

- [HTML40 97] W3C - HTML 4.0 Specification. 18 de Dezembro de 1997. <http://www.w3.org/TR/REC-html40-971218/>. Maio de 2001.
- [HTML40 99] W3C – HTML 4.0 Guidelines for Mobile Access. 15 de Março de 1999. <http://www.w3.org/TR/NOTE-html40-mobile/>. Maio de 2001.
- [IBM 01] IBM. <http://www.ibm.com>.
- [IDC 01] IDC. <http://www.idc.com>. Outubro de 2001.
- [IPanet 01] Sun Microsystems iPlanet Wireless ServerTM. <http://www.iplanet.com/>.
- [IPlanet 01] IPlanet. <http://www.iplanet.com/>.
- [Jataayu 01] Jataayu Software. <http://www.jataayusoft.com/>.
- [Kaasinen 00] Kaasinen, E. et al. – Two Approaches to Bringing Internet Services to WAP devices. In 2ª Conferência Internacional de WWW. Amesterdão. 2000. <http://www9.org/w9cdrom/228/228.html>. Outubro de 2001.
- [Kapadia 01] Kapadia, D. – CIS-690 WAP/WML project. <http://www.cis.ksu.edu/~deep/>. Outubro de 2001.
- [Lee 01] Lee, W.i M.-Transforming XML into WML. <http://www.wirelessdevnet.com>. Dezembro de 2001.
- [Metawrap 01] Massive Technologies. <http://www.massive.com.au/>
- [Móro 2001] Móro, L. – XML:based content transformation in converged service development. Tese de mestrado. 25 Abril de 2001. Universidade de Lappeenranta, Finlândia. <http://www.lut.fi/~leve/thesis/>. Novembro de 2001.
- [Nielsen 00] Nielsen, J. – WAP Usability Report: Field Study Fall 2000. <http://nngroup.com/reports/wap/>. Abril de 2001.
- [Notess 00] Notess, G. – Reference Librarian. Montana State University. Agosto/Setembro de 2000. Econtent p. 69-72.
- [OLC 01] Online Computer Library Center. <http://www.oclc.org/>. Outubro de 2001.
- [OpenTV 01] OpenTV. <http://www.opentv.com>.



- [Openwave 01] Openwave – Top 10 Usability Guidelines for WAP Applications. 1 de Maio de 2001. <http://www.openwave.com/resources/uiguide.html>. Outubro de 2001.
- [Oracle 01] Oracle. <http://www.oracle.com>.
- [Phone 01] Phone.com <http://developer.openwave.com/>.
- [Raggett 01] Raggett, D. – HTML Tidy. <http://www.w3.org/People/Raggett/tidy/>. Março de 2001.
- [Ressler 97] Ressler, S. - The Era of Electronic Publishing: The Internet and Beyond. Prentice Hall. 1997. ISBN: 0-13-488172-9.
- [RFC1392 01] Malkin, G. e Parker, LaQuey T. – IETF: Internet Users Glossary. RFC1392. <http://www.ietf.org>. Janeiro 2001.
- [Rysavy 01] Rysavy, P. – WAP: Untangling the Wireless Standard. Network Computing, p. 109-113.
- [Sergeant 00] Sergeant, M. – AxKit: XML Web Publishing with Apache and mod\_perl. 2000. <http://www.xml.com/pub/a/2000/05/24/axkit/index.html>. Outubro 2001.
- [Silva 99] Silva M. J. – Mobilidade na Internet. In Alves et al. – O futuro da Internet. 1ª ed. Edições Centro Atlântico, 1999. ISBN 972842608-9. p. 269.
- [Strategis 01] Strategis Group. <http://www.strategisgroup.com>.
- [Teraflops 01] Teraflops. <http://www.teraflops.com/>.
- [Tihel, 2001] Tihel, E. et al - Mastering XHTML. Sybex. 2001. ISBN: 0-7821-2820-3.
- [Valentine 01] Valentine, C. e Minnick, C. – XHTML. New Riders. 2001. ISBN: 07-357-1034-1.
- [W3C 01] World Wide Web Consortium. <http://www.w3.org>.
- [WAP 01] WAP Forum. <http://www.wapforum.org/>.
- [WAP 98] WAP Forum. - Wireless Application Protocol Architecture Specification, 1998. <http://www1.wapforum.org/tech/terms.asp?doc=WAP-100-WAPArch-19980430-a.pdf>. Abril de 2001.

- [Whatis 01] What Is.  
[http://whatis.techtarget.com/definition/0,289893,sid9\\_gci212527,00.html](http://whatis.techtarget.com/definition/0,289893,sid9_gci212527,00.html). Maio de 2001.
- [WML 98] WAP Forum – Wireless Markup Language version 1.0. 1998.  
<http://www.wapforum.org/what/technical.htm>. Maio de 2001.
- [XHTML 00] W3C – XHTML1.0: The Extensible HyperText Markup Language.  
26 de Janeiro de 2000. <http://www.w3.org/TR/xhtml1/>. Junho de 2001.
- [XML10 00] W3C - Extensible Markup Language 1.0. 2ª ed. 2000.  
<http://www.w3.org/TR/REC-xml>. Maio de 2002.

## **Anexos**

# Anexo A – Exemplos de Conversões de páginas HTML

## Exemplo 1

### Página HTML

```
<HTML>
<HEAD>
<TITLE>ESTG Mirandela</TITLE>
</HEAD>

<BODY BGCOLOR="#FFFFFF" LINK="#0000FF" VLINK="#0000FF" ALINK="#FF0000">

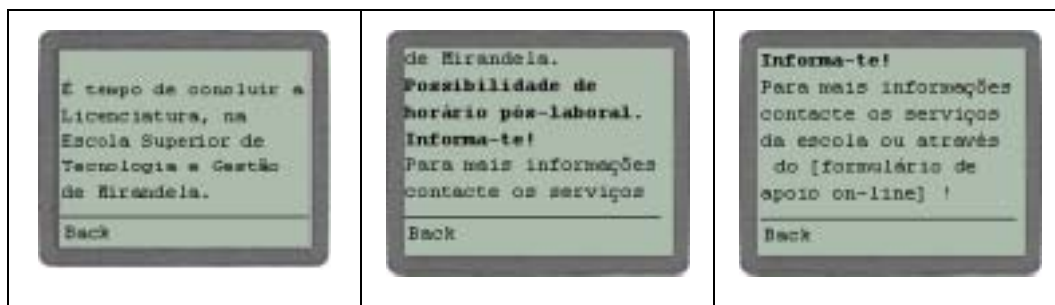
<TABLE BORDER="1" WIDTH="100%">
<TR>
<TD WIDTH="100%"><FONT FACE="verdana, arial" SIZE="2">É tempo de concluir a
Licenciatura, na<BR>
Escola Superior de Tecnologia e Gestão de Mirandela.<BR>
<B><FONT COLOR="#FF0000">Possibilidade de horário pós-
laboral.</FONT></B></FONT></TD>
</TR>
<TR>
<TD WIDTH="100%"><B><FONT FACE="verdana, arial" SIZE="3"
COLOR="#000080">Informa-te!</FONT></B></TD>
</TR>
<TR>
<TD WIDTH="100%"><FONT FACE="verdana, arial" SIZE="1">Para mais informações
contacte os serviços da escola ou através do <A HREF="informações/frmpedido.htm">formulário
de apoio on-line</A>!</FONT></TD>
</TR>
</TABLE>

</BODY>
</HTML>
```



## Deck WML

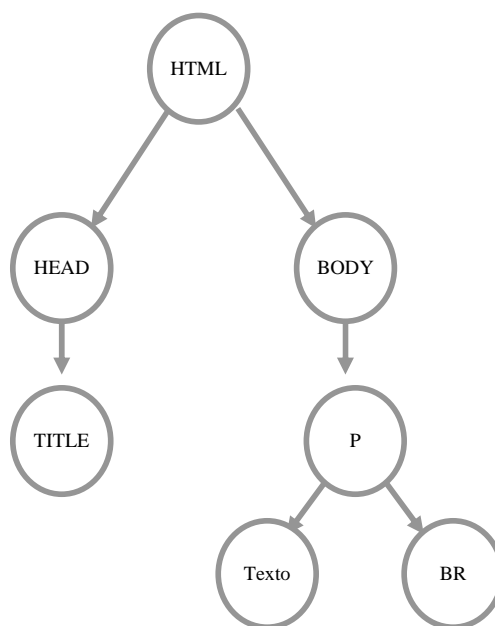
```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<head>
<meta http-equiv="Cache-Control" content="no-cache"/>
</head>
<card id="Principal" title=" ESTG Mirandela ">
<p>
<table columns="1">
<tr><td>
&#201; tempo de concluir a Licenciatura, na<br/>
Escola Superior de Tecnologia e Gest&#227;o de Mirandela.<br/>
<em>Possibilidade de hor&#225;rio p&#243;s-laboral.</em><br/>
</td></tr>
<tr><td>
<em>Informa-te!</em><br/>
</td></tr>
<tr><td>
Para mais informa&#231;&#245;es contacte os servi&#231;os da escola ou atrav&#233;s do
<a href="url.asp?pagina=informacoes/frmpedido.htm">
formul&#225;rio de apoio on-line</a> !<br/>
</td></tr>
</table>
</p>
</card>
</wml>
```



## Exemplo 2

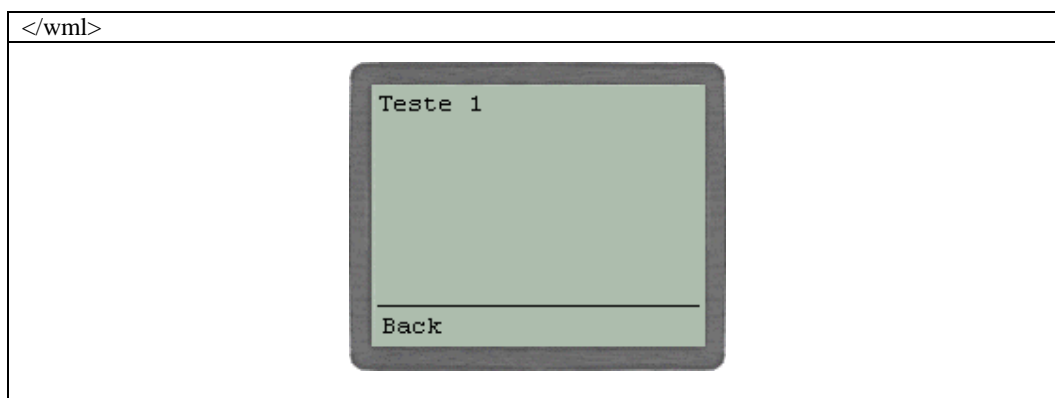
### Página HTML

```
<html>
<head>
  <title>ESTG Mirandela</title>
</head>
<body bgcolor="#FFFFFF"
link="#0000FF" vlink="#0000FF"
alink="#FF0000">
  <p>Teste 1<br/>
  </p>
</body>
</html>
```



### Deck WML

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<head>
<meta http-equiv="Cache-Control" content="no-cache"/>
</head>
<card id="Principal" title="ESTG Mirandela">
<p>Teste 1<br/></p>
</card>
```



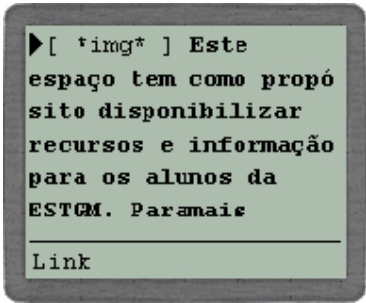
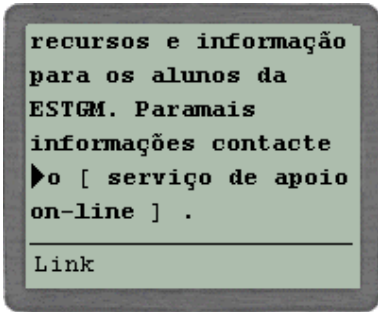
### Exemplo 3

#### Página HTML

```
<HTML>
<HEAD>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=windows-1252">
<META HTTP-EQUIV="Content-Language" CONTENT="en-us">
<TITLE>ESTG Mirandela</TITLE>
<META NAME="GENERATOR" CONTENT="Microsoft FrontPage 4.0">
<META NAME="ProgId" CONTENT="FrontPage.Editor.Document">
<STYLE FPROLLOVERSTYLE>
a:hover { text-decoration: none }
</STYLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF" LINK="#0000FF" VLINK="#0000FF" ALINK="#FF0000">
<P>
<a href="1.html"></a>
<FONT FACE="verdana, arial" SIZE="3">
<B>
Este espaço tem como propósito disponibilizar recursos e informação para os alunos da ESTGM. Para
mais informações contacte o <A HREF="..estgm/informacoes/frmpedido.htm">serviço de apoio on-
line</A>.
</B>
</FONT>
</P>
</BODY>
</HTML>
```

#### Deck WML

```
<?xml version="1.0"?><!DOCTYPE wml PUBLIC "-//WAPFORUM/DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml"><wml>
<head><meta http-equiv="Cache-Control" content="no-cache"/></head>
<card id="card1" title="ESTG Mirandela">
<p>
<a href="/url.asp?pagina=1&#46;html">
```

<pre>*img* &lt;/a&gt; &lt;strong&gt; Este espaço tem como propósito disponibilizar recursos e informação para os alunos da ESTGM. Para mais informações contacte o &lt;a href="/url.asp?pagina=estgm;informacoes;frmpedido.htm"&gt; serviço de apoio on-line &lt;/a&gt; &lt;/strong&gt; &lt;/p&gt; &lt;/card&gt; &lt;/wml&gt;</pre>	
 <p>▶ [ *img* ] Este espaço tem como propósito disponibilizar recursos e informação para os alunos da ESTGM. Para mais informações contacte o [ serviço de apoio on-line ] .</p> <p>Link</p>	 <p>recursos e informação para os alunos da ESTGM. Para mais informações contacte o [ serviço de apoio on-line ] .</p> <p>Link</p>



## Anexo B – Documento XSL

```

<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:fo="http://www.w3.org/1999/XSL/Format">

  <xsl:template match="html">
    <wml>
      <xsl:element name="card">
        <xsl:attribute name="id">Results</xsl:attribute>
        <xsl:attribute name="title"><xsl:value-of
select="head/title"/></xsl:attribute>
        <xsl:apply-templates/>
      </xsl:element>
    </wml>
  </xsl:template>

  <xsl:template match="body">
    <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="frameset">
    <p align="left"><b>Pagina com frames</b></p><p align="left"><b>seleccionar...</b></p>

    <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="frame">
    <xsl:for-each select=".">
      <p>Frame:
        <xsl:element name="a">
          <xsl:attribute
name="href">/html2wml/my/xsl/html2wml.asp?pagina=<xsl:value-of
select="@src"/></xsl:attribute>
          <xsl:value-of select="@name"/>
        </xsl:element>
      </p>
    </xsl:for-each>
  </xsl:template>

  <xsl:template match="noframes">
  </xsl:template>

  <xsl:template match="h1|h2">
    <strong>
      <xsl:apply-templates/>
    </strong>
  </xsl:template>

```

```

</xsl:template>

<xsl:template match="style">
</xsl:template>
<xsl:template match="title">
</xsl:template>

<xsl:template match="h3|h4">
    <em><xsl:apply-templates/>
    </em>
</xsl:template>

<xsl:template match="h5|h6">
    <b><xsl:apply-templates/></b>
</xsl:template>

<xsl:template match="p">
    <xsl:variable name="alignment">
        <xsl:value-of select="@align"/>
    </xsl:variable>
    <xsl:if test="$alignment='center'">
        <p align="center"><xsl:apply-templates/></p>
    </xsl:if>
    <xsl:if test="$alignment='middle'">
        <p align="center"><xsl:apply-templates/></p>
    </xsl:if>
    <xsl:if test="$alignment='right'">
        <p align="right"><xsl:apply-templates/></p>
    </xsl:if>
    <xsl:if test="$alignment='left'">
        <p align="left"><xsl:apply-templates/></p>
    </xsl:if>
    <xsl:if test="$alignment=''">
        <p><xsl:apply-templates/></p>
    </xsl:if>
</xsl:template>

<xsl:template match="center">
    <p align="center"><xsl:apply-templates/></p>
</xsl:template>

<xsl:template match="table">
<table column="1"><xsl:apply-templates/></table>
</xsl:template>

<xsl:template match="tr">
    <tr><td>
    <xsl:apply-templates/>
    </td></tr>
</xsl:template>

<xsl:template match="td">
    <xsl:apply-templates/>
    <br/>
</xsl:template>

```

```

<xsl:template match="ol|ul">
  <p>
    <xsl:apply-templates/>
  </p>
</xsl:template>

<xsl:template match="li">

  <xsl:choose>
    <xsl:when test="parent::ol">
      <xsl:call-template name="numbered-block">
        <xsl:with-param name="format">1. </xsl:with-param>
      </xsl:call-template>
    </xsl:when>
    <xsl:when test="parent::ul">
      <xsl:call-template name="marks-block">
        <xsl:with-param name="format">1. </xsl:with-param>
      </xsl:call-template>
    </xsl:when>
  </xsl:choose>
  <br/>
</xsl:template>

<xsl:template name="numbered-block">
  <xsl:param name="format">1. </xsl:param>
  <xsl:number format="{ $format }"/>
  <xsl:apply-templates/>
</xsl:template>

  <xsl:template name="marks-block">
    -&#160;<xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="blockquote">
    <p><xsl:apply-templates/></p>
  </xsl:template>

  <xsl:template match="a">
    <xsl:element name="a">
      <xsl:attribute name="href">/html2wml/my/xsl/html2wml.asp?pagina=
<xsl:value-of select="@href"/>
      </xsl:attribute>
      <xsl:apply-templates/>
    </xsl:element>
  </xsl:template>

  <xsl:template match="img">
    <xsl:variable name="tooltip">
      <xsl:value-of select="@alt"/>
    </xsl:variable>
    <xsl:if test="parent::a">
      <xsl:choose>
        <xsl:when test="$tooltip="">

```





```
        [Imagem]
      </xsl:when>

      <xsl:otherwise>
        [<xsl:value-of select="@alt"/>]
      </xsl:otherwise>
    </xsl:choose>
  </xsl:if>
</xsl:template>

</xsl:stylesheet>
```

## Anexo C – Exemplos de dispositivos móveis

	<p><b>Ericsson R380</b> Ecrã : 83x28 mm Resolução : 360x120 pixels Imagens : WBMP &amp; GIF (3 cores, limitação na largura ) Browser WML 1.1</p>
	<p><b>Ericsson T20</b> Tamanho : 101x54x28, 128 gr Browser WML 1.1</p>
	<p><b>Alcatel One Touch Pocket</b> Ecrã : 4 x 15 caracteres Imagens: tamanho máximo de 90x42 pixels (WBMP) Browser WML</p>

		<p><b>Motorola V2288</b> Ecrã : 4 x 16 caracteres, 96x64 pixels Browser WML 1.1</p>
		<p><b>Nokia 6210</b> Dimensões : 129.5 x 47.3 x 18.8 mm, 95cc Imagens : WBMP Ecrã : 96 x 60 pixels Browser : WML 1.1</p>
		<p><b>Siemens s35i</b> Dimensões : 118 x 46 x 21 mm Ecrã: 7x16 caracteres Imagens : 101x80 pixels Browser WML 1.1</p>
		<p><b>Siemens ic35</b> Dimensões: 108 x 86.5 x 20.5 mm Imagens : WBMP</p>

	<p><b>Nokia 7110</b> Dimensões: 125 x 53 x 24 mm, 125 cc Imagens : WBMP Ecrã : 96 x 65 pixels Browser : WAP 1.1 tamanho máximo do código binário: 1.397 Kb</p>
	<p><b>Mitsubishi GEO</b> Browser: WML 1.1 Dimensões : 132x49x29, 149gr Ecrã : 4 X 13 caracteres</p>

## Anexo D – Excertos principais do código fonte

### Formulário de introdução da URL da página HTML

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM/DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="Principal" title="Formulario">
<p>
<big>Introduzir a URL completa da p&#225;gina HTML que pretende
visualizar.</big><br/>
URL:<input name="pagina"
value="http://193.136.194.1/html2wml/teste/page0.htm"/>
</p>

<do type="accept" label="Continuar...">
<go href="#confirmar"/>
</do></card>

<card id="confirmar" title="Confirmar URL">
<p>
URL:$pagina
</p>
<do type="accept" label="Confirmar!">
<go href="html2wml.asp?pagina=$pagina"/></do>
<do type="prev" label="voltar"><prev/></do>
</card>
</wml>
```

### Gestão adequada dos *hyperlinks* para evitar a introdução manual de novas URLs

```
<%
...
if(strcomp(request("pagina"), "")<>0) then
url=request("pagina")
...
if(left(url, 7)="http://") then
...
```



```

        session("domain")=domain
        session("path")=path

    else
    ...
        file = getFile(url)
    end if
    url="http://" & session("domain")&"/"
    ...

else
    msgError="URL incorrecta."

end if

function getpath(url)
    ...
    getpath=path
end function

function getdomain(url)
    ...
    getdomain = domain
end function

function getrempath(url, oldPath)
    ...
    getrempath=path
end function

function getAddpath(url, oldPath)
    ...
    getAddpath=path
end function

function getFile(url)
    ...
    getFile=file
end function

Response.ContentType = "text/vnd.wap.wml"
%>
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<head><meta http-equiv="Cache-Control" content="no-cache"/></head>
<card id="Principal" title="Verificar URL"><p>
<%
if(strcomp(msgError, "") = 0) then
%>
1&#170; Etapa conclu&#237;da.

```

<pre>         &lt;br /&gt;         &lt;b&gt;A URL composta &amp;#233; a seguinte:&lt;/b&gt;&lt;br /&gt;         &lt;%=url%&gt;         &lt;do type="accept" label="Continuar..."&gt;         &lt;go href="getpage.php?url=&lt;%=url%&gt;" /&gt;         &lt;/do&gt;      &lt;%     else         response.write msgError     end if     %&gt;     &lt;do type="prev" label="voltar"&gt;&lt;prev/&gt;&lt;/do&gt; &lt;/p&gt; &lt;/card&gt; &lt;/wml&gt; </pre>
<p><b>Acede à página HTML e guarda-a no servidor WAP</b></p>
<pre> &lt;?php header("Content-type: text/vnd.wap.wml"); ... &lt;?php \$filer = fopen (\$url, "w"); if(\$filer) {     \$filer = fopen (\$url, "w");     \$filew = fopen ("d:\\acsantos\\tese\\projecto\\html2wml\\tmp\\html.txt", "w");     while (!feof (\$filer))     {         \$line = fgets (\$filer, 1024);         fputs (\$filew, \$line);     }      fclose(\$filer);     fclose(\$filew); }  ?&gt; ... &lt;/card&gt;&lt;/wml&gt; </pre>
<p><b>Configuração dos parâmetros usados com a aplicação da tabela de correspondência.</b></p>
<pre> &lt;% ... AbsolutePath = Server.MapPath ("\\html2wml\\tmp\\") Set TidyObj = CreateObject("TidyCOM.TidyObject") TidyObj.Options.CharEncoding = 0 TidyObj.Options.TidyMark = false TidyObj.Options.Doctype = "omit" TidyObj.Options.NumericEntities=true </pre>

```

TidyObj.Options.QuoteMarks=true
TidyObj.Options.QuoteNbsp=true
TidyObj.Options.quoteampersand=true
TidyObj.Options.logicalemphasis=true
TidyObj.Options.enclosetext=true
TidyObj.Options.alttext=empty
TidyObj.Options.quiet=true
TidyObj.Options.DropFontTags = true
TidyObj.Options.OutputXml = true
TidyObj.TidyToFile AbsolutePath&"\html.txt", AbsolutePath&"\xhtml.txt"
%>
<wml>
<head>
<meta http-equiv="Cache-Control" content="no-cache"/>
</head>
<card id="Principal" title="Formulario">
...
</card></wml>

```

### Configuração dos parâmetros usados no processo XSLT

```

TidyObj.Options.CharEncoding = 1
TidyObj.Options.TidyMark = false
TidyObj.Options.Doctype = "strict"
TidyObj.Options.NumericEntities=true
TidyObj.Options.QuoteMarks=true
TidyObj.Options.QuoteNbsp=true
TidyObj.Options.quoteampersand=true
TidyObj.Options.logicalemphasis=true
TidyObj.Options.enclosetext=true
TidyObj.Options.alttext=empty
TidyObj.Options.quiet=true
TidyObj.Options.OutputXml = true

```

### Transformação XSL

```

<%
...
AbsolutePath = Server.MapPath ("\\html2wml\\tmp\\")
Set objXML = Server.CreateObject("Msxml2.DOMDocument.4.0")
Set objXSL = Server.CreateObject("Msxml2.DOMDocument.4.0")
objXML.Async = False
objXML.preserveWhiteSpace = False
objXML.validateOnParse = False
objXML.Load(AbsolutePath&"\\xhtml.txt")
...
objXSL.Async = False
AbsolutePath = Server.MapPath ("\\html2wml\\my\\xsl\\")
objXSL.Load(AbsolutePath&"\\wml.xsl")
...

```

```

pageWML=objXML.transformNode(objXSL)
Set fso = CreateObject("Scripting.FileSystemObject")
AbsolutePath = Server.MapPath ("\\html2wml\tmp\")
Set file = fso.CreateTextFile(AbsolutePath&"page.wml", True)
file.Write(pageWML)
file.Close
%>
<?xml version="1.0"?>
<wml>
...
</wml>

```

### Conversão do documento XHTML

```

...
$file="tmp/xhtml.txt";
print "Content-type: text/vnd.wap.wml\n";
if (!$parserHTML = HTML::TokeParser->new($file))
{
...
}
else
{
...
    $parserHTML->get_tag("title");
    $title = $parserHTML->get_text;
    if($title eq "")
    {
        $title="Página sem titulo";
    }

    $result=converter();
}

close(F);
...

sub converter{
    while ($token = $parserHTML->get_token)
    {
        $tagtype=$token->[0];
        $_ = $token->[0];
        $wmlbit="";
        TAGTYPE: {
            /S/ && do { start_tag($token->[1], $token->[2]); $tagtype="S"; last TAGTYPE; };
            /E/ && do { end_tag($token->[1]); $tagtype="E"; last TAGTYPE; };
            /T/ && do { $wmlbit = $token->[1]; $tagtype="T"; last TAGTYPE; };
        }
        $wmlbit = ConvertToAscii($wmlbit);
    }
}
...

```

```
    }  
    ...  
    remove_hidden_tags();  
    ...  
    remove_content_not_show(0, 0);  
    ...  
    create_new_cards(0,1);  
    order_cards();  
    ...  
    split_into_decks();  
    ...  
    remove_content_not_show_final(0,0);  
    ...  
    remove_cards_nulls();  
    ...  
    for($k=1; $k<=$number_total_decks; $k++)  
    {  
        save_deck($k, $str);  
    }  
    return 0;  
}  
...
```