Bruna Chesye Dias

ANÁLISE DA TECNOLOGIA WAP VIA ESTUDO DE CASO EM JOGOS DISTRIBUÍDOS E INTERATIVOS

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências do curso de Ciência da Computação, para obtenção o título de Bacharel.

Orientador
Prof. Ricardo Martins de Abreu Silva

Lavras
Minas Gerais - Brasil
2003

Bruna Chesye Dias

ANÁLISE DA TECNOLOGIA WAP VIA ESTUDO DE CASO EM JOGOS DISTRIBUÍDOS E INTERATIVOS

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências do curso de Ciência da Computação, para obtenção o título de Bacharel.

Aprovada em *16 de Junho de 2003*

Prof. Guilherme Bastos Alvarenga

Prof. Ricardo Martins de Abreu Silva
(Orientador)

Lavras
Minas Gerais - Brasil

Análise da Tecnologia WAP via Estudo de Caso em Jogos Distribuídos e Interativos

Com o crescimento do mercado das telecomunicações móveis, surgiu o WAP, um protocolo mundial que torna possível o acesso à Internet por meio de dispositivos móveis sem fio, como *Palms* e celulares. Apesar das limitações e do preço, os dispositivos WAP têm obtido sucesso em vários países. Sendo assim, o objetivo deste trabalho consiste em estudar e avaliar esta tecnologia para aplicações distribuídas e com alto índice de concorrência e sincronização, demonstrando ao final deste projeto quais as facilidades e dificuldades encontradas para se implementar jogos interativos utilizando esta tecnologia.

WAP Techonology Analysis by Case Study on Interactive and Distributed Games

The mobile telecommunications market had a great growth, making possible the sprouting of WAP, a global protocol that makes possible the access to Internet through wireless mobile devices, like palms and cell phones. WAP got success in several countries, against the limitations and price of the WAP devices. So, the purpose of this wok is to study and appraise this technology for distributed applications, with high level of concurrence and synchronization, showing the difficulties and easiness of the implementation of interactive games using this technology.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

É fato que os usuários tornam-se cada vez mais dependentes dos serviços oferecidos via Internet. Até pouco tempo,para acessá-los se fazia necessário que eles estivessem conectados à rede através de um fio; no entanto milhões de usuários passam muito tempo em trânsito e o fato de necessitarem de um cabo para a conexão torna-se um empecilho. Para sanar este problema, surge a computação móvel objetivando permitir que usuários tenham acesso à rede, independente de sua localização física.

O avanço das comunicações móveis faz com que, em todo mundo, a transmissão sem fio de conteúdos comece a revolucionar o uso da Web, propagando sua utilização para inúmeras aplicações e tornando-a ainda mais poderosa.

De acordo com um estudo divulgado pela Andersen Consulting, a explosão do *mobile commerce* europeu deverá ocorrer em 2005 quando pelo menos 165 milhões de pessoas poderão estar utilizando o celular para realizar suas compras em transações que podem chegar aos US$ 70 bilhões.

Estima-se que entre 2003 e 2005 o acesso a internet será maior pelos celulares que pelo computador, e um dos principais responsáveis por essa evolução é o *Wireless Application Protocol* (WAP) [1].

O WAP foi desenvolvido através da união de quatro grande empresas: a Nokia, a Phone.com, a Ericsson e a Motorola. O órgão WAP Forum foi estabelecido para administrar e cuidar de sua padronização.

O WAP é uma tecnologia que torna possível o acesso à Internet por meio de dispositivos móveis sem fio, como micros portáteis e celulares.

Para acessar um site WAP, é necessário que o dispositivo seja compatível com a tecnologia WAP e as informações devem estar no formato WML (*Wireless Markup*

Language).

O WAP tende a ser uma nova e poderosa forma de acesso à internet, tanto para usuários da rede como para organizações, já que está posicionado na convergência de duas tecnologias de rede que estão evoluindo muito rapidamente: a transmissão de dados sem fio e a Internet.

De acordo com uma pesquisa da Europe, há uma previsão de que até o ano 2003, haverá mais de meio bilhão de usuários com equipamentos WAP [11].

Esta tecnologia pode ser descrita como sendo um protocolo largamente aceito, com capacidade de reconhecer os serviços WWW, além de possibilitar às operadoras de telefonia celular e, em breve, aos provedores de serviços de telecomunicações oferecer para seus usuários acesso a várias aplicações da Internet a partir de qualquer aparelho móvel.

O objetivo deste trabalho consiste em estudar e avaliar a tecnologia WAP para aplicações distribuídas e com alto índice de concorrência e sincronização. Uma das possibilidades de aplicação seria a implementação de jogos interativos distribuídos, onde dois ou mais jogadores poderiam jogar ao mesmo tempo.

De acordo com uma pesquisa realizada pela Datamonitor em setembro 2000, em 2005 haverá 200 milhões de jogadores através de celular, quatro de cada cinco usuários de celulares [11].

Esta monografia apresenta a seguinte estrutura:

Capítulo 1 - Introdução: descreve de forma objetiva, o assunto abordado neste projeto;

Capítulo 2 - Um estudo da tecnologia WAP: capítulo responsável por fornecer todo conhecimento teórico necessário para se trabalhar com a tecnologia WAP.

Capítulo 3 - Um protótipo do jogo de Batalha Naval utilizando WAP: este capítulo descreve as dificuldades e facilidades encontradas para se desenvolver um jogo interativo utilizando a tecnologia WAP.

Capítulo 4 - Conclusões: conclusões finais a respeito da utilização da tecnologia WAP no desenvolvimento de jogos interativos e distribuídos.

Capítulo 2

Um estudo da tecnologia WAP

O objetivo principal deste capítulo consiste em fornecer todo o conhecimento necessário para se desenvolver um aplicativo utilizando a tecnologia WAP. Para facilitar seu entendimento, foi dividido em oito seções, a saber: seção 2.1 *wireless* - breve descrição do conceito de redes *wireless*; além de seus principais benefícios e aplicações; seção 2.2 WAP - nesta seção, estarão as informações mais importantes sobre esta tecnologia, como histórico, definição, principais características, limitações, e especificações, além de um comparativo entre o modelo WWW e o modelo WAP e uma descrição das camadas que a compõem este protocolo; seção 2.3 WML - descrição desta linguagem englobando suas principais características; seção 2.4 *WMLScript* - descrição desta linguagem ressaltando sua importância para o desenvolvimento de um aplicativo WAP; seção 2.5 Elaboração de Aplicativos WAP - requisitos necessários para se escrever e testar os aplicativos WAP, citando alguns dos principais editores, *browsers* e emuladores para se desenvolver um programa em WML; seção 2.6 PHP - breve descrição da linguagem de programação PHP e suas principais características; seção 2.7 *Sockets* - descreve a definição de *sockets* e os passos para se estabelecer uma conexão; seção 2.8 Java - definição da linguagem Java e suas principais características.

2.1 *Wireless*

2.1.1 O que é *wireless*?

Wireless[1] é uma tecnologia que permite a conexão entre diferentes equipamentos sem a necessidade de nenhum tipo de conexão física, ou seja, sem a necessidade do uso de cabos (nem de telefonia, nem de TV a cabo, nem de fibra ótica), através da instalação de uma antena e de um rádio de transmissão.

Muitas das redes sem fio comerciais são baseada em redes celulares existentes, que dividem uma região geográfica em células. Este caminho da rede sem fio para a Internet com fio envolve saltos da estação rádio base de uma célula, por meio de um ou mais centros de comutação, para um *gateway* e, finalmente, para um destino na Internet.

A figura 2.1 mostra um exemplo de uma simples rede *wireless* LAN.

Principais benefícios das redes sem fio:

- Mobilidade: permite que os utilizadores tenham acesso a informações em tempo real em qualquer lugar dentro da sua organização.
- Instalação rápida e simples por evitar a necessidade de passar cabos por paredes e tetos.
- Flexibilidade: permite que a rede chegue onde os cabos não permitem.
- Custo reduzido: embora o investimento em equipamento de rede *wireless* seja superior ao das redes que utilizam cabos, o custo global da instalação e as despesas do tempo de vida da rede podem ser significativamente mais baixos. Estas redes trazem grandes benefícios no longo prazo em ambientes dinâmicos e com grande mobilidade.
- Escalabilidade: podem evoluir de redes *peer-to-peer* para poucos utilizadores até redes com infra-estruturas complexas para centenas de utilizadores que cobrem áreas mais vastas.

Apesar destes benefícios, as redes de dados *wireless* apresentam um ambiente de comunicação mais limitado, comparado com as redes comuns, pois possuem [3]:

- menor largura de banda;

[1] Todo o conteúdo desta seção foi extraído de ZANETTI [2], RISCHPATER [14] e Redes *wireless* [16].

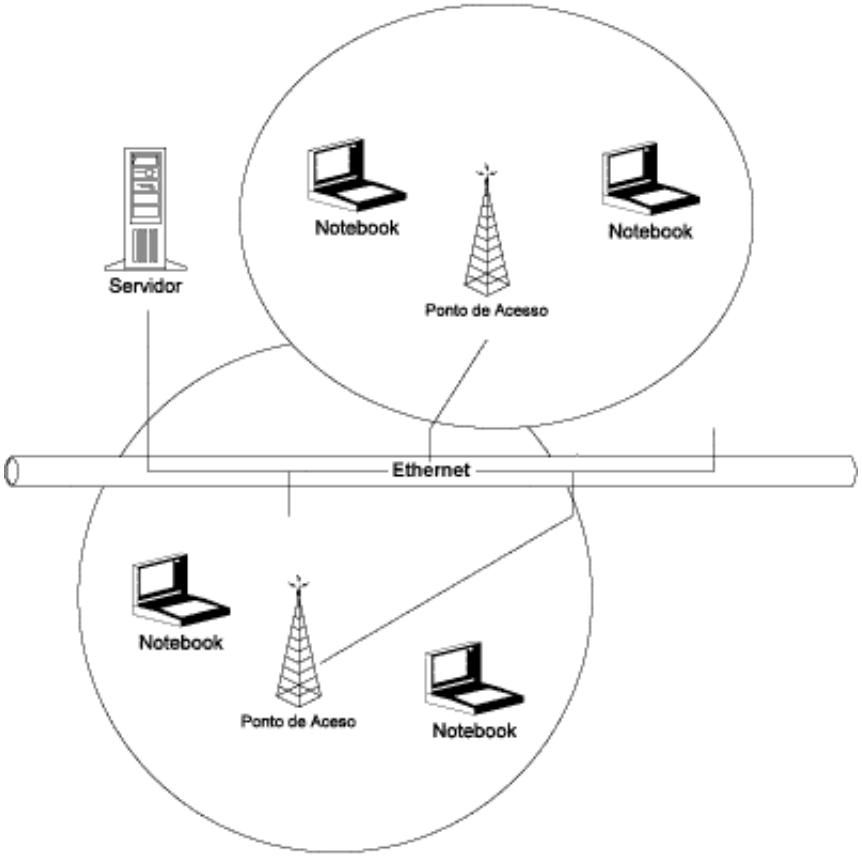

Figura 2.1: Rede *wireless* LAN típica

- mais latência;
- bateria limitada;
- menos estabilidade de conexão.

Além disso, o aumento de largura de banda, impõe um aumento de consumo na já limitada carga da bateria de um dispositivo móvel. Assim, até mesmo com as melhorias das redes *wireless*, a disponibilidade de potência ao dispositivo portátil ainda limitará o processamento efetivo dos dados.

2.1.2 Aplicações

O poder e a flexibilidade das redes sem fio, permitem uma série de aplicações práticas:

- doutores e enfermeiras com *notebooks* obtendo instantaneamente informações sobre pacientes;
- estudantes, durante a aula, acessando a Internet para consultar o catálogo da Biblioteca;
- gerentes de rede em ambientes dinâmicos, podendo realizar mudanças e extensões com muito menos preocupações;
- gerentes de rede instalando redes locais sem fio em prédios velhos por preços muito mais acessíveis;
- redes locais sem fio ligando as estações de trabalho nos andares de uma fábrica e auxiliando na coleta de dados de máquinas;
- através de dispositivos móveis, em qualquer lugar a qualquer hora, pode-se consultar qualquer tipo de notícia, verificar condições de tempo e trânsito, verificar *e-mails*, pagar hospedagens e outras compras.

2.1.3 Opções de Plataforma sem Fio

Existem 3 plataformas que podem ser utilizadas para o desenvolvimento de aplicativos para redes sem fio [14]:

WEB - *World Wide Web***:** Rede mundial baseada em padrões de dados abertos que alimentou uma explosão de intercâmbio de informações.

WAP - *Wireless Application Protocol***:** Baseia-se no modelo adotado pela WEB e será o objeto de estudo desta monografia.

HDML - *Handheld Device Markup Language***:** Precendo o WAP, contribuiu para a criação do WML e é atualmente mantida pela Phone.com como um padrão aberto separado, competindo com o padrão WAP.

2.1.4 Opções de *Hardware*

Os principais terminais sem fio com acesso à WEB, são agrupados em 4 categorias onde o tamanho e a faixa de preço determinam quais são os recursos que cada dispositivo pode oferecer [14]:

Supertelefones: Oferecem uma plataforma de computação baseada em *Personal Digital Assistant*(PDA) combinada com *hardware* de telefone celular.

Telefones Inteligentes: São uma extensão do conceito de produto de telefone celular existente, combinando as melhores características do telefone digital com a capacidade de obter dados da Internet usando WAP ou HDML.

PDAs: Tabletes eletrônicos do tamanho da palma da mão que aceitam módulos encaixáveis ou cartões de PC que permitem aos dispositivos acessar serviços sem fio de redes digitais.

***Laptops*:** Embora comparáveis aos seus correspondentes de mesa, são limitados principalmente pela rede sem fio e normalmente são operados quando o usuário está parado.

2.2 WAP (*Wireless Application Protocol*)

2.2.1 Histórico

Em 1995 [2], a *Telefonaktiebolaget LM Ericsson* iniciou um ambicioso projeto que visava o desenvolvimento de um protocolo genérico para serviços que agregassem valor às redes móveis. Este novo protocolo foi denominado ITTP e gerenciava a comunicação entre o nó de serviço, onde a aplicação estava instalada, e um telefone móvel inteligente. Este protocolo visava melhorar a performance do HTTP.

Em julho de 1996, a *Unwired Planet*, por sua vez, apresentou o HDML semelhante ao HTML usado na WWW, como uma linguagem de descrição de conteúdo e interface, porém otimizada para permitir o acesso à Internet a partir de dispositivos portáveis com pequenos *displays* e capacidade de entrada limitada.

Em março de 1997, a *Nokia* apresentou oficialmente os conceitos de *Smart Messaging* ou Serviço de Mensagem, uma tecnologia de serviço para acesso à Internet especialmente projetada para os dispositivos GSM. A comunicação entre

[2] O conteúdo desta sub-seção foi extraído de SILVA JÚNIOR [4] e Phone.com [5].

o usuário móvel e o servidor contendo informações da Internet usava o SMS e uma linguagem de marcação chamada TTML, que, assim como o HDML, foi otimizada para a comunicação sem fio.

Em 26 de junho de 1997, a *Ericsson*, a *Motorola*, a *Nokia* e a *Unwired Planet* (hoje *Phone.com*) se uniram para desenvolver um protocolo de aplicações sem fio em comum. Inicialmente chamado de MDI *Mobile Data Initiative*, herdou as características principais dos protocolos e linguagens já criados por estas empresas.

Ainda neste mesmo ano, passou a ser chamado de WAP e o o *WAPForum*, um orgão independente com a finalidade de desenvolver e padronizar este protocolo, foi oficialmente criado.

Em 15 de Setembro de 1997 foi publicada a versão 1.0 do WAP. Desde então o número de sócios do *WAPForum* cresceu muito, reunindo provedores de serviço *wireless*, fabricantes de equipamento portáteis, provedores de infra-estrutura e pesquisadores de *software*.

2.2.2 O que é WAP?

O WAP[3] é uma especificação aberta e global, que visa permitir que usuários de dispositivos móveis, sem fio, acessem facilmente informações e serviços de forma instantânea. A comunicação pode ser feita através de ondas de rádio via satélite ou antenas.

Embora possa ser utilizado em *palmtops*, *handhelds* e *notebooks* equipados com *modem* sem fio e ainda aparelhos de rádio usados em sistema de *trunking* [4], o telefone celular é o dispositivo móvel mais usado, com larga vantagem, para acessar a Internet através de WAP.

O WAP adotou as melhores características de infra-estrutura WEB e aperfeiçoou sua operação permitindo a clientes e servidores intercambiar a maioria dos dados usando a mínima quantidade de largura de banda. Por ter sido criado especificamente para ser utilizado em dispositivos sem fio tornou-se mais eficiente que os protocolos tradicionais para este tipo de aplicação [14].

O protocolo WAP é basicamente uma pilha de protocolos de comunicação que tem como meta unir um servidor de aplicação à um dispositivo sem fio. Este conjunto de protocolos especifica dois elementos essenciais para a comunicação

[3] Todo o conteúdo desta seção foi extraído de SILVA JÚNIOR [4] e SPOSITO [8].

[4] É um moderno sistema de comunicação que permite a transmissão de voz e de dados de forma segura e eficiente, onde um certo número de canais é disponibilizado para os usuários, e todos estes canais podem ser usados a qualquer instante.

sem fio: um protocolo de comunicação fim-a-fim e um ambiente de aplicação baseado em visualizadores (*browsers*).

Quando o *browser* WAP é usado para solicitar uma informação, ocorrem os seguintes passos:

- o pedido da *Universal Resource Locator* (URL) é enviado através da rede *wireless* para o *gateway* WAP, usando os protocolos WAP;

- o *gateway* WAP decodifica a solicitação recebida do protocolo *Wireless Session Protocol* (WSP) (no padrão WAP) para o protocolo da Internet *HyperText Transfer Protocol* (HTTP) e envia essa requisição para o *web server* através da internet;

- ao ler a requisição, o *web server* retorna uma resposta com conteúdo *Wireless Markup Language* (WML) para o *gateway*WAP que o converte para o padrão *bytecode* do WAP, codificando a informação para um formato binário de forma que possa ser usada menos largura de banda. O conteúdo codificado é encriptado e enviado através da rede *wireless* para o *browser* WAP.

Em suma, o *gateway* WAP recebe o conteúdo WML do servidor WEB e o *browser* recebe a resposta do *gateway* WAP e a exibe no *display* do dispositivo WAP. Este conteúdo interpretado e exibido no *display* é criado através da linguagem WML, abordada com mais detalhes na seção 2.3.

2.2.3 Características

Principais características do protocolo WAP [5]:

- Um modelo de programação baseado fortemente no modelo de programação WEB existente.

- Independência dos dispositivos: uma linguagem de marcação que obedece os padrões *Extensible Markup Language* (XML), projetada para criar aplicativos WAP independentes de qualquer dispositivo particular. Especifica a funcionalidade mínima que um dispositivo deve ter e foi projetado para aceitar funcionalidade extra sobre este mínimo.

- A linguagem de *script WMLScript*, baseada na *European Computer Manufacturers Association* (ECMA), ECMAScript, para estender a capacidade da WML.

[5] O conteúdo desta sessão foi extraído de CABRAL & LEITE [6] e MANN [13].

- A especificação de um micro-navegador que define como a WML e a *WMLScript* devem ser interpretadas no equipamento portátil e apresentadas ao usuário.

- Interoperabilidade: qualquer componente construído para ser compatível com a especificação WAP pode interoperar com qualquer outro componente WAP compatível.

- Um sistema adequado para os aplicativos de telefonia sem fio *Wirelers Telephony Applications* (WTA), propiciando funções que as operadores de telefonia possam usar para integrar as funções de telefonia e micronavegação de um equipamento WAP.

- Otimização de Transferência de Dados: houve interesse em otimizar os protocolos existentes, tais como o HTTP e o TCP, para maximizar a transferência de dados.

- Independência de Sistema Operacional: como este protocolo é dependente de padrões de comunicação ao invés de estarem baseados nas plataformas, qualquer plataforma capaz de implementar esses padrões de comunicação será compatível com WAP.

- Uma pilha de protocolos leve, projetada para minimizar a exigência de largura de banda, trabalhar com uma diversidade de portadoras sem fio e propiciar conexões seguras.

2.2.4 Aplicações do Protocolo WAP

Alguns *sites* WAP oferecem uma grande quantidade de serviços, chegando a compararem-se com os *sites* WEB baseados no modelo WWW, eis aqui alguns exemplos de aplicações do protocolo WAP [6]:

Transações Bancárias

Tendo um celular WAP, com acesso á Internet, e considerando que sua operadora ofereça os serviços WAP em parceria com o Banco, pode-se fazer transações bancárias como as disponíveis no auto-atendimento do Banco do Brasil, que já possui *site* WAP `http://wap.bb.com.br`.

[6] O conteúdo desta seção foi extraído de [15].

Programação dos Cinemas

Uma outra aplicação do Protocolo WAP é o acesso à programação dos cinemas. *Sites* WAP como o `http://wap.cineguia.com.br` apresentam os filmes em cartaz no Brasil inteiro.

Ferramentas de Auxílio para a Polícia Militar

A Polícia Militar de Minas Gerais possui um sistema que permite que os policiais façam consultas em uma base de dados, podendo, assim em mais ou menos 18 segundos, obter dados sobre carros suspeitos e até antecedentes criminais do motorista.

Serviços de Agenda *On-Line*

Sendo o usuário cadastrado no sistema, ele pode dispor dos serviços de agenda *on-line*. Este serviço é muito utilizado para lembrar o usuário de compromissos, datas importantes, horóscopo e notícias.

Indicadores Financeiros

Sites WAP oferecem, diariamente, informações do Sistema Financeiro Nacional com a cotação do dólar e o valor do grama do ouro, além de informações sobre as principais bolsas de valores.

Situação de Aeroportos

Através do endereço `http://www.folhawap.com.br/` pode-se obter informações e saber como está a situação de aeroportos como Congonhas, Santos Dumont e Cumbica.

Previsão do Tempo

O site `http://www.climawap.com.br/index.wml` oferece a previsão do tempo para várias cidades do Brasil divididas por regiões.

Conversão de Moeda

Havendo necessidade de fazer conversões de moeda pode-se também utilizar o WAP. Este serviço pode ser conferido em `http://wap.nodel.com.br`.

Comércio Eletrônico

Compras também podem ser feitas pelo celular WAP, nesta tecnologia chamado de *m-comerce* (comércio móvel). O *site* `http://www.superlojas.com.br/wap/index.wml` hospeda cerca de 500 lojas e pode-se fazer compras como se estivesse navegando em um *site WEB* normal.

Serviços de *E-mail*

Mensagens não muito longas e sem aquelas formatações e quantidades de figuras como as enviadas pelos programas de *e-mail* da WEB tradicional, podem ser enviadas e recebidas através do celular WAP.

Jogos

E, é claro, não se poderia deixar de descrever a aplicação que será apresentada nesta monografia: jogos. Uma série de jogos já foram implementados utilizando esta tecnologia, a maioria deles no entanto para um único jogador, alguns exemplos: *Memory, Battleships, Russian Roulette, kniffel* [28]. Nesta monografia serão descritos os passos para se implementar um Jogo de Batalha Naval interativo e distribuído.

2.2.5 Limitações

Apesar de fornecer várias vantagens, o protocolo WAP possui uma série de limitações [7]:

- Tamanho do ecrã (tela): seu tamanho reduzido limita seriamente as capacidades do sistema.
- Teclado: por ser pequeno e numérico, o teclado se torna um empecilho para digitar palavras, reduzindo a capacidade de entrada de dados.
- Velocidade: a atual velocidade pode variar de 9,6 a 14,4kbs, assim, além de ainda ser muito lenta, há a necessidade de fazer uma nova chamada sempre que houver necessidade de uma nova conexão.
- *Micro-browser*: por apresentar apenas informações escritas em WML, as operadoras têm que desenvolver conteúdos próprios para WAP.

[7] Conteúdo extraído de CABRAL & LEITE [6] e MANN [13].

- Possui CPU, RAM, ROM e velocidade de processamento limitados e poucas opções de fornecimento de energia (baterias, pilhas, etc);
- Redes de baixa capacidade com largura de banda modestas.
- As redes de voz e dados sem fio são inerentemente pouco confiáveis, instáveis e imprevisíveis.

2.2.6 O Modelo WWW

A arquitetura Internet [8] provê um modelo de programação flexível e poderoso, onde aplicações e conteúdo são apresentados em formatos de dados padrões e são mostrados por aplicações conhecidas como *web browsers* ou navegadores *web*.

O *web browser* é uma aplicação para rede, o que significa que ele envia requisições para objetos de dados nomeados para um servidor de rede e este responde com o dado codificado, usando os formatos padrões. Este processo é ilustrado na Figura 2.2.

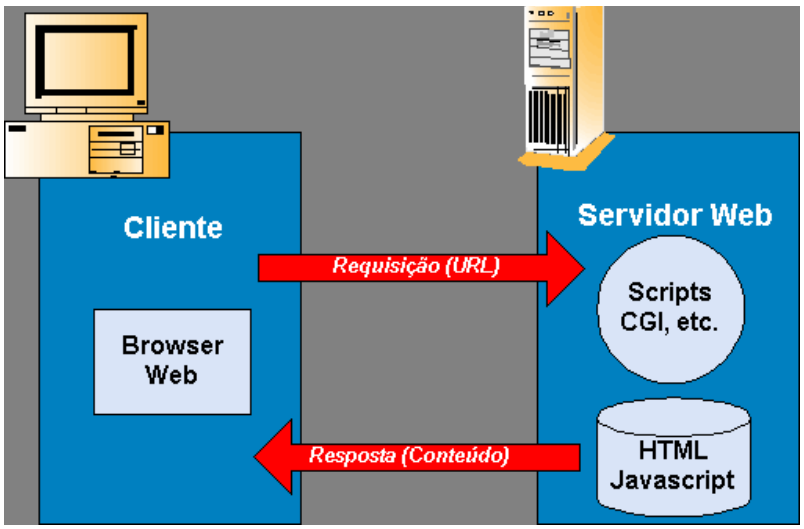

Figura 2.2: Modelo WWW

Os padrões WWW especificam a maioria dos mecanismos necessários para a construção de um ambiente de aplicação de propósito geral, incluindo:

[8] Todas as informações desta seção foram extraídas de SILVA JÚNIOR [4].

- Modelo de nomeação padrão: todos os servidores e conteúdo na *Web* são nomeados de acordo com o padrão da Internet, conhecido como URL.

- Tipificação de conteúdo: todo conteúdo na internet é produzido de maneira que os navegadores *web* o processem da maneira correta, baseado em seu tipo.

- Formatos de conteúdo padrão: todos os *web browsers* suportam um pacote de formatos de conteúdo padrões, que incluem o HTML e o *JavaScript*, entre outros.

- Protocolos de comunicação padrões: os protocolos padrões de rede, tais como o HTTP, permitem a qualquer navegador comunicar-se com qualquer servidor WEB.

Essa infraestrutura permite aos desenvolvedores criarem aplicações e conteúdo para um grande número de clientes.

Os protocolos da WWW são definidos em três classes de servidores :

- Servidor de origem: servidor onde um dado recurso reside ou será criado.

- *Proxy*: um programa intermediário que age tanto como cliente quanto como servidor. Este tipo de servidor reside entre os clientes e os servidores que não têm meios de comunicação direta.

- *Gateway*: um servidor que age como intermediário para algum outro servidor. Diferente do *proxy*, um *gateway* recebe as requisições como se ele fosse o servidor de origem para o recurso solicitado.

2.2.7 O Modelo WAP

O modelo de programação WAP é muito similar ao modelo de programação WWW, visando justamente se agregar a ela da melhor maneira possível. Esta tentativa de produzir um ambiente coerente com a estrutura já existente, provê vários benefícios à comunidade de desenvolvimento, como a familiarização com o modelo existente, a confiabilidade de uma arquitetura testada e a possibilidade de utilização de ferramentas existentes (como servidores WEB, editores XML, etc). A figura 2.3 demonstra o modelo WAP.

No entanto, apesar de muito similar o modelo de programação WAP possui duas diferenças cruciais:

- Há sempre pelo menos um servidor *proxy* WAP entre o usuário e o servidor de conteúdo.
- A comunicação entre o agente de usuário e o servidor *proxy* é feita com protocolos WAP [13].

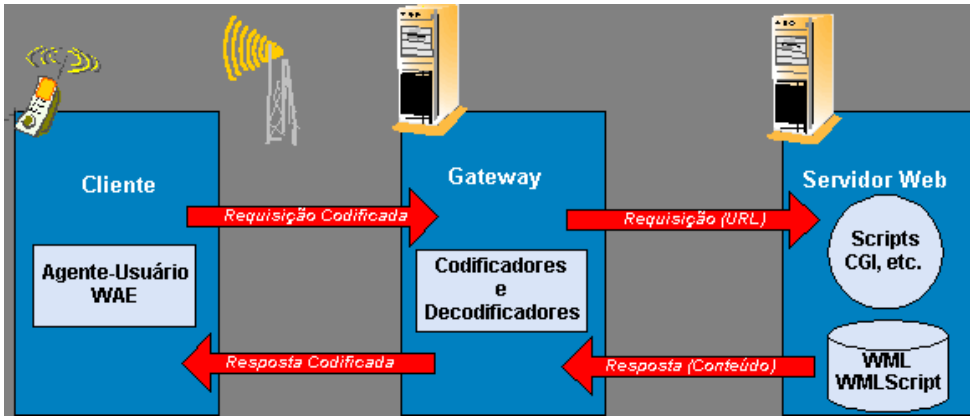

Figura 2.3: Modelo WAP

Assim como o WWW, o WAP define um conjunto de componentes padrões para permitir a comunicação entre terminais móveis e servidores de rede:

- Modelo de nomeação padrão: o padrão WWW de *Universal Resource Locator* (URL) é usado para identificar o conteúdo WAP nos servidores originais, enquanto o padrão WWW de *Universal Resource Identifier* (URI) é usado para identificar recursos locais em um dispositivo.
- Tipificação de conteúdo: todo o conteúdo WAP tem um tipo específico consistente com a tipificação da WWW, o que permite aos agentes-usuários WAP processarem corretamente um conteúdo baseado em seu tipo.
- Formatos de conteúdo padrões: todos os formatos de conteúdo WAP são baseados na tecnologia WWW e inclui marcação para *display*, informações de calendário, objetos de cartões de negócios eletrônicos, imagens e linguagem de *script*.
- Protocolos de comunicação padrões: o protocolo de comunicação WAP possibilita a comunicação das requisições do *browser* do terminal móvel com o servidor *web*.

Os tipos de conteúdo e protocolos WAP foram otimizados para micro portáteis sem fio. O WAP utiliza a tecnologia de *proxy* para conectar um domínio sem fio à Internet.

Principais funções do *proxy* WAP:

- *Gateway* de protocolo: o *gateway* de protocolo traduz as requisições das camadas da pilha do Protocolo WAP. Essas camadas (WAE, WSP, WTP, WTLS e WDP) serão explicadas em mais detalhes na subseção Pilha de Protocolos.

- Codificador e decodificador de conteúdo: os codificadores de conteúdo traduzem o conteúdo WAP em um formato codificado compacto para reduzir o tamanho dos dados que trafegam pela rede. Os decodificadores traduzem o conteúdo codificado compacto para o conteúdo WAP.

A partir desses funções, o *proxy* WAP permite que conteúdo e aplicações sejam hospedados em servidores WWW padrões e sejam desenvolvidos usando tecnologias WWW como, por exemplo, *scripts* CGI. A figura 2.4 ilustra um exemplo de uma rede WAP.

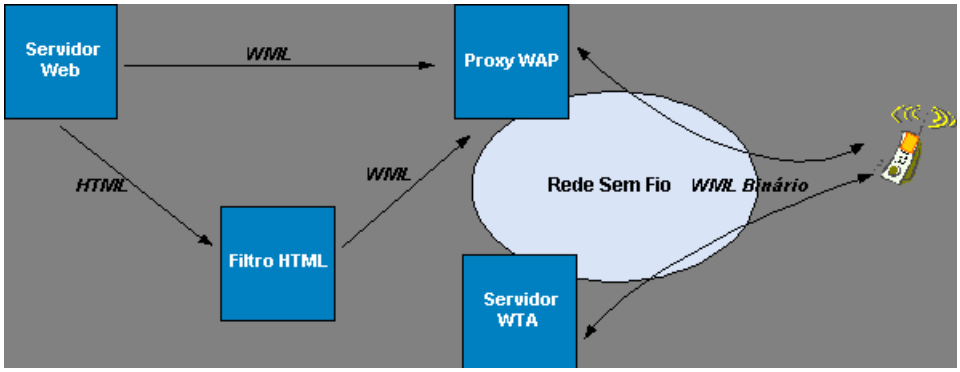

Figura 2.4: Exemplo de uma rede WAP

O WAP *gateway* diminui o tempo de resposta do dispositivo portátil, agregando dados de diferentes servidores na WWW e guardando informações freqüentemente acessadas. O WAP *gateway* também pode se conectar a bancos de dados de assinantes e utilizar informações da rede *wireless*, tais como informação de localização, para personalizar as páginas WML de acordo com o usuário.

2.2.8 Arquitetura - A Pilha de Protocolos

Apesar de ser visto como um protocolo [9], na verdade o WAP é uma pilha composta por cinco protocolos independentes, organizados em um ambiente escalável e extensível para o desenvolvimento de aplicações direcionadas à utilização em dispositivos móveis. Isto foi alcançado através da construção de uma pilha de protocolos dividida em camadas, conforme Figura 2.5.

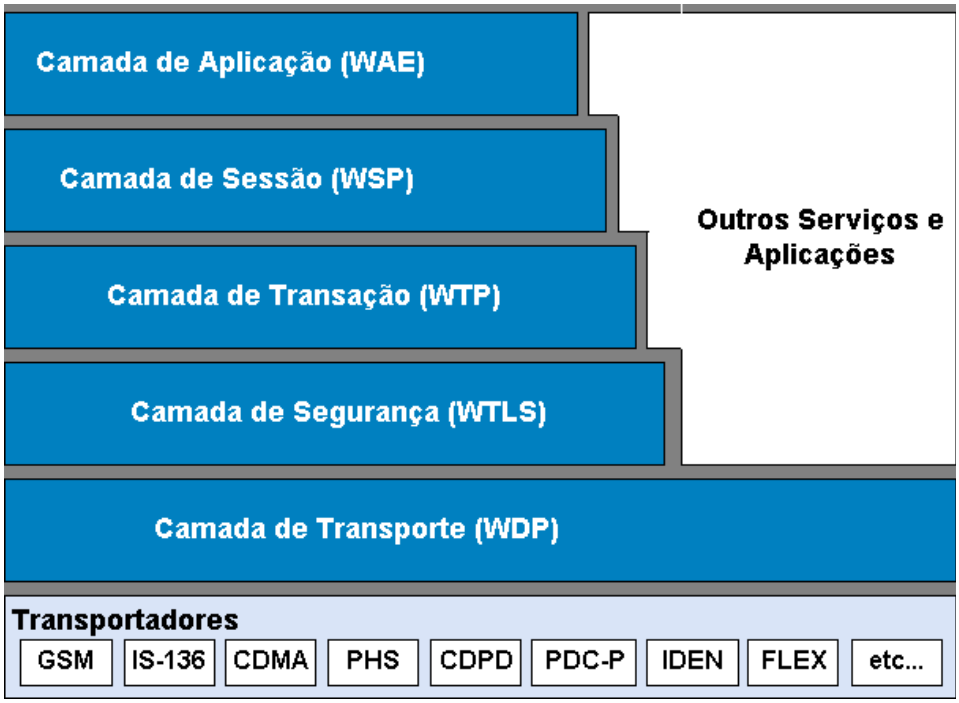

Figura 2.5: Camadas do Protocolo Wap

WAE (*Wireless Application Environment*)

O WAE é um ambiente de aplicação de propósito geral baseado na combinação das tecnologias da WEB e de telefonia móvel.

Essa camada é responsável por estabelecer um ambiente interativo onde operadoras e prestadores de serviços poderão construir aplicações que alcançarão um

[9] O conteúdo desta seção foi extraído de CABRAL & LEITE [6], MANN [13] e SILVA [15].

grande número de plataformas sem fio de uma maneira útil e eficiente.

O WAE inclui um ambiente de *micro-browser* contendo os seguintes componentes:

- uma especificação que define como o WML e o *WMLScript* devem ser interpretados pelo aparelho portátil e apresentado ao usuário [13];

- *Wireless Markup Language* (WML), uma linguagem de marcação leve, similar ao HTML, mas otimizada para utilização em dispositivos móveis;

- *WMLScript*, uma linguagem de script leve, similar ao *JavaScript*;

- serviços de telefonia e interfaces de programação: *Wireless Telephony Application*

- um conjunto de formatos de dados bem definidos, incluindo imagens, registros de agenda de telefones e de compromissos.

O WAE assume a existência de funcionalidades de *gateway* responsáveis pela codificação dos dados transferidos para o cliente móvel. O objetivo desta codificação dos dados é minimizar o tamanho desses enviados por ar, e a utilização de recursos necessário para o cliente processar esses dados. A funcionalidade do *gateway* pode ser adicionada a servidores já existentes ou colocada em *gateways* dedicados.

O modelo WAE oferece as seguintes vantagens:

- disponibiliza serviços de rede móvel avançados ao usuário, através de serviços de telefonia controlados pelo operador da rede;
- tira vantagem dos padrões, tecnologia e infra-estrutura desenvolvidas para a internet;
- fornece uma área de trabalho extensa para a construção de serviços *wireless*.

WSP (*Wireless Session Protocol*)

WSP disponibiliza para a camada de aplicação uma interface consistente para dois serviços de sessão. O primeiro é o serviço orientado a conexão, que opera sobre o protocolo de transação WTP e o segundo é o serviço não orientado a conexão, que opera sobre o serviço de datagrama seguro ou não (WDP).

Esta camada fornece às aplicações meios para estabelecer uma sessão confiável do cliente ao servidor e liberar esta sessão de maneira ordenada, bem como concordarem com um nível comum da funcionalidade do protocolo usando capacidade de negociação, trocarem conteúdo entre cliente e servidor usando codificação compacta, e fornecerem meios para suspender e continuar a sessão.

O WSP atualmente oferece as seguintes funcionalidades:

- funcionalidades e semântica do HTTP/1.1 através de uma codificação compacta;
- estado de sessão de longa vida;
- suspensão e retomada de sessões;
- facilidades na utilização de tecnologia *push*; [10]
- protocolo para negociação de capacidades.

Os protocolos da família WSP são otimizados para uma banda de conexão baixa, com latência relativamente longa. O WSP/B (WSP *browsing*) por exemplo foi projetado para permitir que um *proxy* WAP conecte um cliente WSP/B a um servidor HTTP comum. Opcionalmente, este protocolo também pode suportar requisições assíncronas, ou seja, um cliente pode enviar múltiplas requisições para o servidor simultaneamente, o que melhora a utilização de transmissão, uma vez que estas múltiplas requisições e respostas podem ser combinadas em algumas poucas mensagens.

WTP (*Wireless Transaction Protocol***)**

WTP é executado no topo de um serviço de datagramas, provendo um protocolo orientado a conexão leve, otimizado para implementação em dispositivos móveis, como telefones celulares.

O WTP opera eficientemente sobre datagramas, disponibilizando os seguintes serviços:

- três classes de serviços de transação:
  - requisições *one-way* (pedidos) não confiáveis;
  - requisições *one-way* (pedidos) confiáveis;
  - requisições *two-way* (pedidos e respostas) confiáveis.
- Confiabilidade opcional *user-to-user* - O usuário ativa a confirmação a cada mensagem recebida.

[10] Tecnologia *push*: permite a atualização no computador, de notícias e informações, através de informações periódicas pela Internet.

- Concatenação de *Protocol Data Unit*(PDU) e reconhecimento do atraso para reduzir o número de mensagens enviadas.

- Transações assíncronas.

Esta camada possui uma **Entidade de Gerenciamento** que além de monitorar o estado do ambiente móvel, também pode ser usada como interface para o usuário ajustar uma série de parâmetros de configuração relativos a inicialização, configuração, reconfiguração dinâmica e recursos usados pelo WTP, como por exemplo o endereço do dispositivo.

WTLS (*Wireless Transport Layer Security***)**

WTLS é uma camada opcional da pilha WAP que fornece às camadas superiores um serviço de interface de transporte seguro. É um protocolo de segurança baseado no protocolo padrão da indústria, o *Transport Layer Security* (TLS), formalmente conhecido como *Secure Socket Layer* (SSL).

O WTLS disponibiliza os seguintes serviços:

- integridade de dados: contém dispositivos que asseguram que os dados transmitidos entre o terminal e um servidor de aplicações não foram modificados ou corrompidos;

- privacidade: possui dispositivos que garantem que os dados transmitidos entre o terminal e o servidor de aplicações não podem ser compreendidos por alguém que intercepte o fluxo de dados;

- autenticação: permite facilidades que estabelecem a autenticidade do terminal e do servidor de aplicações;

- detecção e rejeição de dados incorretamente enviados.

Este protocolo pode ser divido em dois sub-protocolos [3]:

- Protocolo de *Handshake*: responsável pela negociação da conexão segura, permite que uma das partes envolvidas na conexão concorde com os parâmetros de segurança definidos, isto pode ser feito iniciando estes parâmetros e informando aos outros participantes sobre as condições em que serão emitidos os alertas de erro.

- Protocolo de Gravação: recebe as mensagens a serem transmitidas, comprime os dados (opcional), aplica o código *Message Authentication Code* (MAC) [11], criptografa e transmite o resultado. Os dados recebidos são descriptografados, verificados e descomprimidos, passando então para o nível de apresentação do cliente.

WDP (*Wireless Datagram Protocol*)

WDP é a camada de transporte da arquitetura WAP e opera sobre os serviços portadores de dados suportados por vários tipos de redes. Esta camada oferece uma forma consistente de transmissão de pacotes, permitindo às camadas de Segurança, Sessão e Aplicação funcionarem independentemente da rede sem fio utilizada.

Dentre os serviços oferecidos pelo WDP estão: endereçamento de aplicação por número de porta, segmentação opcional, e reunião e detecção de erros opcionais.

Existe uma **Entidade de Gerenciamento** na camada WDP que é usada como uma interface entre esta camada e o ambiente do dispositivo, tratando de toda a circulação de dados relacionada a inicialização, configuração e reconfiguração dinâmica de recursos.

Para reportar os possíveis erros encontrados no processamento dos datagramas WDP, o *WAPForum* especificou o *Wireless Control Message Protocol* (WCMP), muito semelhante ao *Internet Control Message Protocol* (ICMP), ele é utilizado também pelos *gateways* de dados sem fio e provê um mecanismo eficiente de tratamento de erros, resultando em ganhos de desempenho para os protocolos WAP e para as aplicações que os utilizam.

Camada de Rede

A camada de protocolos WAP foi projetada para operar sobre uma variedade de transportadores de serviços. Estes oferecem níveis diferentes de qualidade no que diz respeito a rendimento, taxa de erro e atraso, que devem ser compensados ou tolerados pelo WAP.

Tipos de redes telefônicas que atualmente suportam a utilização de WAP [7]:

- CDMA (*Code Division Multiple Access*): tecnologia que utiliza sistema exclusivo, permitindo enviar a ligação codificada e resgatá-la de outra porta de forma clara e sem erros;

[11] MAC é um código calculado a partir de funções de *hash* seguras, para garantir a integridade das mensagens trocadas.

- FDMA (*Frequency Division Multiple Access*): é uma das principais técnicas de múltiplo acesso utilizada em sistemas de comunicação móvel. Nesta, a multiplexação é feita por divisão de freqüência;

- GSM (*Global System for Mobile Communication*): sistema criado para padronizar as comunicações celulares, permitindo ao usuário utilizar seu telefone em qualquer país;

- EDGE (*Enhanced Data Rates for Global Evolution*): trata-se de uma tecnologia que permite às redes GSM suportar e oferecer serviços de terceira geração à telefonia móvel;

- W-CDMA (*Wireless Code Division Multiple Access*): é um padrão norte-americano que possui um importante papel na realização das comunicações sem fio, porque tem quase a mesma performance na transmissão quanto as comunicações com fio;

- TDMA (*Time Division Multiple Access*): tecnologia para serviços digitais sem fio baseada na divisão de uma freqüência de rádio em frações de tempo que serão usadas em diferentes conexões.

Além dos transportadores de serviços, também existe a rede portadora que é responsável pelo encaminhamento de datagramas até ao dispositivo destino. O endereçamento desta varia conforme o tipo de rede (endereços IP ou números de telefone), no entanto, o protocolo WAP foi projetado para compensar ou tolerar estas diferenças, permitindo que seus endereços de rede incluam o tipo de portadora e o endereço (ex.: IP; 123.456.789.123).

Exemplos de portadores:

- SMS (*Smart Messaging Service*): é um serviço que permite mandar ou receber mensagens curtas de/para telefones celulares sem que se tenha de fazer uma ligação, economizando desta forma tempo e dinheiro.

- CSD (*Circuit Switched Cellular Data*): uma tecnologia que permite a transferência de dados via celular através de um canal de tráfego digital, por exemplo: com um *laptop*, um aparelho celular com esta tecnologia e um cabo, o cliente pode acessar a internet independente do horário e local a uma velocidade de 9.600 bps, além de poder enviar e receber fax.

- CDPD (*Cellular Digital Packet Data*): é uma especificação que suporta o acesso a internet e outras redes públicas através de comutação de pacotes. Um telefone celular e um modem CDPD possibilitam velocidades até 19,2Kbps.

- GPRS (*General Packet Radio Services*): serviço não baseado em voz que permite o envio e recepção de informações através de uma rede telefônica móvel. Ele utiliza as tecnologias de CSD e SMS.

Outros Serviços e Aplicações

A arquitetura baseada em camadas do WAP possibilita que outros serviços e aplicações utilizem as capacidades desta pilha através de um conjunto de interfaces bem definidas. Aplicações externas, por exemplo, podem acessar as camadas de sessão, transação, segurança e transporte diretamente, permitindo desta forma que a pilha WAP seja usada por aplicações e serviços para os quais ele não foi inicialmente projetado, mas que são demandados pelo mercado sem fio.

Cache na Arquitetura WAP

O modelo de *cache* do WAP é baseado no protocolo HTTP/1.1. Este modelo é bastante sensível à perda de tempo de sincronização, no entanto, para melhor adaptá-la a dispositivos de funções limitadas, como os terminais sem fio, algumas alterações foram feitas. Neste caso por exemplo uma solução seria manter no *gateway* WAP um relógio com a hora do dia, sendo recomendável para implementação deste que seja usado o *Network Time Protocol* (NTP) a fim de se manter uma base de tempo real confiável.

A interação entre o *cache* do usuário e o histórico do WML é controlada pelo cabeçalho *Cache-control*.

O armazenamento de informações sensíveis em meios de armazenamento não-voláteis levanta uma série de considerações de segurança e os implementadores devem assegurar-se de que informações privadas no usuário são protegidas de acesso acidental ou para fins maliciosos.

2.2.9 Segurança em WAP

Para que uma transação realizada em WAP possa ser considerada segura, deverão ser utilizados:

- criptografia de 128 Bits, a nível de aplicação e transporte;
- *gateway* com implementação de segurança;
- autenticação digital através de PKI (*Public Key Infrastructure*);
- certificação digital;
- *firewall*.

Além do sistema WTLS especialmente desenvolvido para ser utilizado em ambiente *wireless*, pode-se citar outros elementos de segurança utilizados na Internet e no ambiente WAP: o SSL e o TLS.

SSL (Secure Socket Layer)

Além de ser bastante utilizado na WWW para codificar o fluxo de dados entre o *browser* e o servidor WEB, é também utilizado para o ambiente WAP, entre o servidor WEB e o WAP *gateway*. Sua principal função é fornecer de um nível seguro de transporte entre o serviço de transporte utilizado na internet (*Transmission Control Protocol* - TCP) e as aplicações que se comunicam através dele, como comércio eletrônico e transações bancárias.

O SSL é composto de duas partes diferenciadas:

- *Handshake Protocol*: responsável por estabelecer a conexão, verificando a identidade das partes e determinando os parâmetros que serão utilizados posteriormente.
- *Record Protocol*: Comprime, criptografa, descriptografa e verifica a informação que é transmitida desde o início da conexão.

TLS (Transport Layer Security)

Seu principal objetivo é oferecer privacidade e integridade dos dados, na comunicação entre duas aplicações. É composto de duas camadas:

- Protocolo de gravação TLS: utilizado para encapsular protocolos situados nos níveis mais altos da pilha.
- Protocolo TLS *Handshake*: permite ao servidor e ao cliente autenticar um ao outro e negociar um algoritmo e uma chave de criptografia antes que o protocolo de aplicação transmita ou receba seu primeiro *byte* de dados.

2.3 WML - *Wireless Markup Language*

2.3.1 O que é WML?

A WML [12] herda a maior parte de suas construções sintáticas da XML - *Extensible Markup Language*, um subconjunto limitado da SGML - *Standardized Generalized Markup Language* e um padrão WWW para linguagens de marcação na Internet. A WML é definida formalmente como sendo uma aplicação do XML. Sua especificação oficial foi desenvolvida e é mantida pelo *WAPForum*.

Embora seja semelhante à HTML que é usada na criação de páginas da WWW, a linguagem WML foi criada para atender às necessidades dos dispositivos e redes *wireless*, descrevendo como o conteúdo WAP será mostrado ao usuário. Com WML pode-se por exemplo: mostrar informações num telefone celular, fornecer ao usuário opções de entrada de dados, e especificar como o dispositivo responderá quando o usuário ativar funções da interface ou pressionar botões.

A WML foi especificamente projetada para gerar conteúdo e interface de usuário para dispositivos pequenos com baixa banda de transmissão tais como celulares, *pages* e *palms*.

Para acrescentar recursos dinâmicos às aplicações WML, utiliza-se a linguagem *WMLScript*, uma linguagem semelhante ao *JavaScript*. Tanto WML como *WMLScript* são adaptadas e otimizadas para utilização em ambiente *wireless*.

Na WWW, os *sites* são constituídos de páginas que são carregadas em um *browser*. Várias páginas podem ser mostradas com o uso de *frames*, sendo a primeira página de um site chamada de *Home page*.

Um dispositivo WAP não carrega uma página, mas sim um *deck* de *cards*. Este mostra apenas um *card* de cada vez, mas pode guardar muitos *cards* diferentes num mesmo *deck*. Seu conteúdo é exibido num *micro-browser*. A tabela 2.1 elucida estas principais diferenças entre WAP e WEB neste aspecto.

A WML foi projetada para ser uma linguagem de marcação interativa, ou seja, possui vários conceitos permitindo que operações simples executadas inteiramente no cliente, possam ser realizadas sem a necessidade de consultas sem fio.

Além de eventos e tarefas, a WML torna disponíveis objetos de seleção e campos de texto para permitir que os *decks* aceitem entrada de usuário. Seu *deck* pode usar essa entrada localmente para criar *scripts* dinâmicos ou para interagir com serviços remotos, como os de cotações de ações, relatórios climáticos e assim por diante.

[12] Todas as informações desta seção foram extraídas de SPOSITO [9], DEITEL & DEITEL [12], SILVA [15] e MANN [13].

Tabela 2.1: Quadro comparativo: WAP X WEB

	Linguagem	Conteúdos dispostos	Página
WEB	Html (*Hypertext Markup Language*)	*Site*	*Home page*
WAP	Wml (*Wireless Markup Language*)	*Deck*	*Card*

2.3.2 Principais Características

Principais características da linguagem WML:

- WML oferece suporte a texto e a imagem através de uma variedade de comandos de formatação e *layout*. Embora seja necessário que o autor especifique a apresentação em termos gerais, o agente-usuário possui uma grande liberdade para determinar como ele deve apresentar a informação ao usuário final. Os dispositivos compatíveis com WAP não são obrigados a suportar imagens.

- Suporte à entrada de dados do usuário, incluindo entrada de texto, listas de opções e controles que chamam tarefas.

- As unidades básicas mostradas (uma por vez) no *display* do dispositivo semfio, os *cards* são agrupados, numa estrutura chamada *deck*. Um *deck* é similar á uma página HTML identificado por uma URL e é a unidade de transmissão de conteúdo.

- WML oferece suporte para gerenciamento da navegação entre *cards* e *decks*, e inclui comandos para manipulação de eventos que podem ser usados para navegação ou execução de *scripts*; também suporta *hiperlinks*, similar ao que é utilizado atualmente no HTML versão 4.0.

- Variáveis podem ser usadas em lugar de *strings* e são substituídas em tempo de execução. A configuração de parâmetros permite que recursos de redes sejam utilizados de maneira mais eficiente.

- Todo o conteúdo WML é transmitido de forma compilada e codificada nas redes sem fio, essa compilação, é feita pelo WAP *Gateway* após o recebimento dos dados de uma requisição HTTP e da verificação da sintaxe do mesmo.

- O conjunto de caracteres utilizados em WML é o conjunto de Caracteres Universais do ISO/IEC (*International Standards Organization/International Engineering Consortium*), definido no documento ISO/IEC-10646, o qual, atualmente, é idêntico ao *Unicode 2.0*, mas não é obrigatório que os *decks* WML sejam codificados usando todo o *Unicode* e sim que o conjunto de caracteres de codificação seja um subconjunto do *Unicode*, isto permite suporte de múltiplos idiomas e dialetos.

- A especificação abstrata do *layout* e da apresentação do WML permite aos fabricantes controlarem o projeto da MMI *Man-Machine Interface* para seus dispositivos.

2.3.3 *Decks e Cards*

Um *card* é um agrupamento de um ou mais elementos de interface do usuário, como texto, listas em que o usuário pode selecionar um item ou linhas para entrada de dados do usuário. Normalmente representa uma única tela em um terminal WAP, embora não haja garantia de que um determinado *card*, da forma em que é construído, caberá na tela de um determinado terminal. O *deck* é um único documento WML, que é a unidade de requisição HTTP, ou seja, cada vez que se consulta uma URL, um *deck* WML é baixado do *site* para o *WapGateway*, o equivalente a uma página HTML na WEB.

Um *deck* é composto de um ou mais *cards*, que constituem a unidade de interação com o usuário. Assim, o usuário navega acessando um, e somente um, *card* de cada vez. À medida em que são percorridos, é que ocorre a transmissão do *WapGateway* para o dispositivo sem fio.

A vantagem desta arquitetura é que múltiplas telas podem ser baixadas para o cliente numa única requisição. Usando *WMLScript*, as seleções e entradas do usuário podem ser retidas e distribuídas entre os cartões já carregados, eliminando então, transações excessivas com os servidores remotos, mantendo a maior parte da navegação entre o cliente e o *WapGateway*.

A metáfora *card/deck* é um lembrete constante para uma das diferenças significativas entre WML e HTML: as restrições do dispositivo a que se destina. Assim, dependendo da capacidade de memória do cliente, pode ser necessário dividir vários *cards* e vários *decks* para evitar que um *deck* fique demasiadamente grande.

2.3.4 Sintaxe WML

Documentos WML consistem em dados de caracteres entremeados por *tags*, que são cercados pelos caracteres < e >. Essas *tags* têm duas partes - uma *tag* de abertura e uma de fechamento - que cercam a informação afetada e definem a estrutura de um documento. As *tags* <wml> e </wml> de abertura e fechamento por exemplo contêm todo o conteúdo do *deck*. É importante destacar que também podem existir *tags* vazias como a de quebra de linha:
.

Exemplo de código escrito em wml:

```
1<?xml version="1.0"?>
2<!DOCTYPE wml PUBLIC "-//PHONE.COM//DTD WML 1.1//EN"
3   "http://www.phone.com/dtd/wml11.dtd" >
4 <wml>
5   <card id="calc" title="Conversao">
6     <p>
7       <b>Digite o valor a ser convertido:</b>
8       <input type="text" name="valor" />
9       <b>Opcao:</b>
10      <select name="opcao" value="reais">
11        <option value="reais">real->dolar</option>
12        <option value="dolares">dolar->real</option>
13      </select>
14    </p>
15    <do type="accept" label="Calcular">
16      <go href="programa.wmls#submit()"/>
17    </do>
18  </card>
19
20  <card id="Resultado" title="Resultado">
21    <p>
22      <b>Valor convertido:</b> = $(resultado)
23    </p>
24    <do type="accept" label="Voltar">
25      <go href="#calc"/>
26    </do>
27  </card>
28</wml>
```

A figura 2.6 mostra como ficaria este programa emulado no *Nokia WAP Toolkit*[13]:

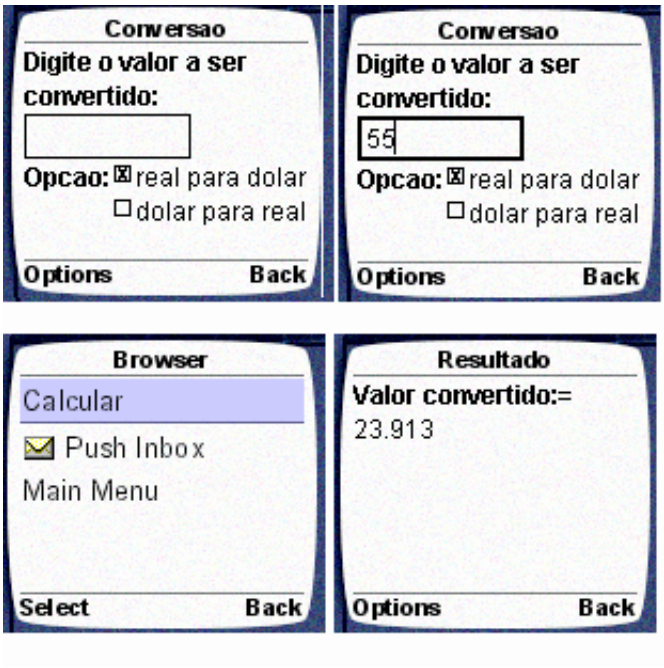

Figura 2.6: Telas exibidas na emulação do exemplo acima.

No código do exemplo acima, todo o texto contido entre as linhas 1 e 28 constituem um *deck*, sendo que as linhas 1, 2 e 3 descrevem o prólogo de XML. Este *deck* é constituído de dois *cards*, o primeiro chamado de ”Conversão”, está descrito entre as linhas 5 e 18 e é identificado como ”calc”e o segundo descrito entre as linhas 20 e 27, é chamado e identificado pelo nome ”Resultado”.

Organização do Documento

Um *deck* em WML consiste em um prólogo de XML,um cabeçalho e um ou mais *cards*, todos contidos na *tag* <wml>.

As informações contidas no prólogo são empregadas por *gateways* em conformidade com a XML e outras aplicações para determinar como um certo documento

[13]Para se executar a emulação, necessita-se também do código em *WMLScript*. Este será descrito na seção 2.4.

deve ser interpretado.

Atributos

As *tags* podem conter atributos que descrevem suas características e podem ser escritos como pares nome/valor, separados por um sinal de igual(=).

Por exemplo: <option value=”reais”/>

Pode-se fazer comentários em WML utilizando os delimitadores <!– e –>.

Variáveis

As variáveis são sempre avaliadas, ou seja, seu nome é substituído por seu valor em WML antes que outra marcação seja executada. Elas podem ser definidas por elementos do usuário usando comandos de formulário da WML ou pela *tag* <SETVAR>. A abrangência de uma variável é geralmente global em todo o navegador; não há meios de ocultar variáveis.

Eventos e Tarefas

A WML define vários eventos que são acionados quando o navegador muda de estado, como quando ele carrega um novo *deck*. Os *decks* podem usar esses eventos para acionar ações denominadas tarefas.

Essas tarefas, podem pedir ao navegador para renovar uma página, pular para um URL específico, pular para o URL anterior e assim por diante.

Caracteres Reservados

Os caracteres <, >, ’, “, & e $ são reservados para serem usados em qualquer texto contido em um *deck*.

2.4 *WMLScript*

2.4.1 O que é *WMLScript*

O *WMLScript* [14] é uma linguagem de *script* leve e procedural, projetada para sofisticar e realçar as facilidades de navegação e apresentação do WML, bem como

[14] O conteúdo desta seção foi extraído de SILVA JÚNIOR [4], DEITEL & DEITEL[12], SILVA [15] e MANN [13].

proporcionar aos terminais de acesso sem fio a capacidade de servir conteúdo dinâmico aos assinantes com um mínimo de interação sem fio.

O WML lida com entrada e saída de dados, exibição de conteúdo e processamento de eventos, mas não tem recursos computacionais elaborados. O *WMLScript* preenche este vazio. Ele pode definir funções que podem ser chamadas a partir de programas WML. Dentro destas funções pode-se manipular instruções de atribuição, chamadas de funções, construções de laços, tipos de dados básicos fracamente tipados e um conjunto completo e poderoso de operadores lógicos, aritméticos e de comparação.

Esta linguagem é capaz de:

- dar suporte a comportamento da *Standardized Generalized Markup Language* (UI) de modo mais avançado;
- adicionar inteligência ao cliente;
- prover um mecanismo conveniente de acesso aos dispositivos e seus periféricos;
- verificar a validade dos dados de entrada por parte do usuário;
- manejar mediante código funções próprias do terminal, como realizar chamadas deste telefone, enviar mensagens e agregar um número de telefone a uma lista de endereços;
- realizar alertas, mensagens de erro, confirmações, etc;
- reduzir a necessidade de idas e voltas ao servidor.

A linguagem é fracamente baseada na linguagem de *script* da Internet o *JavaScript*, pois foi refinada para dispositivos de banda estreita.

Os tipos de dados suportados pelo *WMLScript* são lógico, inteiro, ponto flutuante, cadeia de caracteres e inválido, e a própria linguagem tenta fazer a conversão automaticamente entre os tipos.

A linguagem suporta também várias categorias de operações, como: atribuições, aritméticas, lógicas e comparações, e vários tipos de funções, como locais, externas e de bibliotecas, além de definir uma série de bibliotecas incluindo uma para linguagem, uma para cadeia de caracteres, uma para navegador, uma para ponto flutuante e uma para diálogo.

2.4.2 Principais Características

Principais características da linguagem *WMLScript* [14]:

- O *WMLScript* foi criado para ser semelhante ao ECMAScript (desenvolvido pela *European Computer Manufacturers Association*) que foi a formalização do *JavaScript*, o que o torna bastante fácil de aprender e usar.
- Lógica Procedural: o WML proporciona comportamento dinâmico simples usando eventos, tarefas e substituição de variáveis, mas suas ações são limitadas por falta de lógica procedural.
- Implementação compilada pelos *gateways* em uma representação binária compacta, apropriada para redes sem fio e eficiente para ser interpretada por terminais sem fio.
- A linguagem é totalmente integrada com o navegador WML, o que permite ao desenvolvedor construir serviços usando ambas as tecnologias, escolhendo a mais apropriada para a tarefa.
- Suporte internacional.
- A linguagem pode ser usada para expor e estender a funcionalidade do dispositivo sem mudanças no *software* deste.

Exemplo de programa escrito em *WMLScript*:

```
1extern function submit()
2{
3   var a = Lang.parseFloat(WMLBrowser.getVar("valor"));
4   var op = WMLBrowser.getVar("opcao");
5   var c = invalid;
6
7   if (op == "reais")
8   {
9      c = a / 3;
10  }
11  else if (op == "dolares")
12  {
13     c = a * 3;
14  }
```

```
15  WMLBrowser.setVar("resultado", String.toString(c));
16  WMLBrowser.go("exercicio4.wml#Resultado");
17}
```

Este programa é constituído de apenas uma função responsável por fazer a conversão dólar/reais. As variáveis ”a” e ”op” declaradas nas linhas 3 e 4, são capazes de capturar os valores das variáveis ”valor” e ”opcao” utilizadas pelo documento WML. Entre as linhas 7 e 14 são feitos os cálculos propriamente ditos da conversão. Na linha 15 a variável ”resultado” recebe o valor encontrado após a conversão e na última linha de código existe uma instrução para voltar ao programa WML(no caso o ”exercicio4.wml”).

2.5 Elaboração de Aplicativos WAP

Para escrever e testar os aplicativos WAP são necessários [15]:

- Um SDK - *Software Development Kit* compatível com WAP1.1
- Um servidor WEB.

A maior parte dos SDKs usam um simulador que se parece e opera igual a um dispositivo WAP típico. Integrado ao simulador, porém, estão os codificadores de WML e *WMLScript*.

Usando um editor de texto padrão pode-se criar arquivos-fontes de WML e *WMLScript* e colocá-los nos lugares apropriados do servidor WEB.

Uma vez alcançado o ponto em que se deseja criar e testar conteúdo WEB dinâmico, deve-se decidir como criar este conteúdo. A maioria dos servidores WEB suportam *scripts Perl* e programas em C para geração de conteúdo CGI. Outros suportam *servlets Java, ASP* e outros mecanismos.

Finalmente para testar os aplicativos no mundo real, precisa-se de um equipamento compatível com WAP, uma cobertura de rede sem fio para esse equipamento, uma maneira para esse dispositivo acessar um servidor *proxy* compatível com WAP e uma maneira para o servidor encontrar seu aplicativo.

Exemplos de editores WAP que podem ser utilizados: *EasyPad WAPtor, CardONE, Waptor 2.2, WmlEdit, Mpresence*.

A visualização pode ser feita por um *browser* especializado em ler páginas WML. Exemplos de *browsers*: *WinWap, Klondike, M3Gate, WAPman, 4thpass, KBrowser, ccWAP Browser* e *Waptor*.

[15] O conteúdo desta seção foi extraído de DEITEL & DEITEL [12] e MANN [13].

Pode-se testar a aplicação através de emuladores situados na WEB que mostram como as páginas WAP seriam vistas num modelo específico de telefone celular. Para telefones Nokia e Ericsson por exemplo o *Gelon.net* seria um boa opção. Outros exemplos de emuladores: *Klondike, YourWap, NZ Phone, Opera, WAPsilon*. Todos estes programas se encontram disponíveis para *download* em Telemóveis.com [10], *WirelessBr*Br [20] e WML Club [17].

É possível também encontrar disponível para *download*, *toolkits* como: o *Nokia WAP Toolkit* [18] e o *Openwave Software Development Kit* [19], que englobam num mesmo aplicativo as funções de editor, *browser* e emulador.

2.6 PHP

2.6.1 Conceito de PHP

PHP [16] é uma linguagem que permite criar *scripts* ou páginas dinâmicas [17], possibilitando uma interação com o usúario através de formulários, parâmetros da URL e *links*.

O código PHP é executado no servidor, sendo enviado para o cliente apenas o código da linguagem de marcação pura (HTML ou WML por exemplo). Assim, é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente.

O protocolo HTTP, provê dois principais métodos para enviar informações para o servidor WEB, além da URL referente ao arquivo solicitado. Esses métodos são o GET e o POST:

Método GET: apesar de ser possível passar parâmetros utilizando este método, ele tem pelo menos dois problemas que podem ser considerados sérios: - permite uma quantidade de dados passados limitada a 1024 caracteres e - permite que todos os dados enviados, possam ser vistos pelo usuário já que as informações enviadas fazem parte da URL.

Método POST: a partir da versão 1.0 do protocolo HTTP, apareceram os *headers* nas mensagens de requisição e de resposta. Os headers são informações trocadas entre o navegador e o servidor de maneira transparente ao usuário,

[16] O conteúdo desta seção foi extraído de CONVERSE & PARK [21] e DIAS [22].

[17] Páginas dinâmicas são aquelas geradas a partir da interação com Banco de Dados ou outras fontes, permitindo que o conteúdo da página seja montado durante a execução da aplicação, podendo ser diferente a cada montagem, dependo de opções definidas pelo usuário.

e podem conter dados sobre o tipo e a versão do navegador, a página de onde partiu a requisição (*link*), os tipos de arquivos aceitos como resposta, e uma série de outras informações. Assim, a utilização de *headers* tornou possível através do método POST, enviar os parâmetros da URL solicitada sem expor esses dados ao usuário, e também sem haver um limite de tamanho.

2.6.2 Características

Principais características da linguagem PHP:

- Código aberto: todo o código fonte do PHP está disponível.
- Custo zero: não possui nenhum tipo de custo; basta fazer *download* no site oficial.
- Multiplataforma: pode rodar sobre o *Unix, Linux* ou *Windows*.
- Eficiente: consome poucos recursos do servidor, permitindo que programas complexos sejam desenvolvidos, sem que isto implique em grande demora na sua execução.
- Acesso a bancos de dados: pode-se acessar diretamente qualquer banco de dados do mercado, como *dBase, Interbase, mSQL, MySQL, Oracle, SyBase, PostgreSQL*, além disso, tem suporte a outros serviços através de protocolos como IMAP, SNMP, NNTP, POP3 e, HTTP, também é possível abrir *sockets* e interagir com outros protocolos.
- Processamento de imagens: permite criar imagens dinamicamente e enviá-las ao *browser* do usuário.
- Orientado a objetos: pode-se modelar e projetar um sistema utilizando os conceitos de Orientação a Objetos e implementá-lo em PHP.

2.7 *Sockets*

2.7.1 O que são *Sockets*?

Um *socket* [18] é uma extremidade de um canal de comunicação que permite a um processo executando num computador enviar/receber mensagens para outro

[18] O conteúdo desta seção foi extraído de FERREIRA [26] e Sockets: apresentação e comentários [27].

processo executando neste mesmo computador ou num remoto. Em outras palavras, *sockets* são usados para representar a conexão entre um programa cliente e um programa servidor.

Etapas para se estabelecer um conexão via *sockets* no lado do cliente:

1. criar um *socket*;
2. ligar o *socket* ao endereço do servidor;
3. emitir e receber dados.

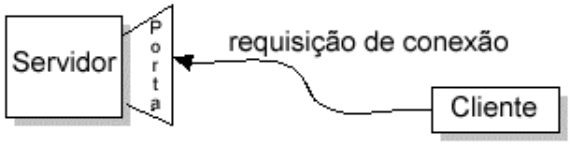

Figura 2.7: *Sockets*: Cliente - Servidor

Etapas para se estabelecer uma conexão via *sockets* no lado do servidor:

1. criar um *socket*;
2. ligar o *socket* a um endereço que normalmente é um número de porta da máquina anfitriã;
3. aguardar até escutar conexões;
4. aceitar uma conexão;
5. enviar e receber dados.

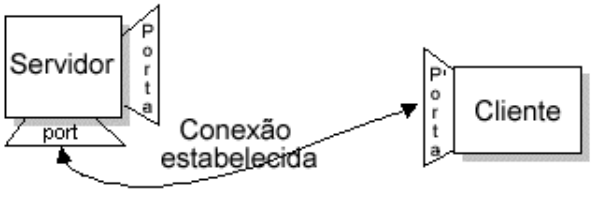

Figura 2.8: *Socktes*: Servidor - Cliente

Quando um *socket* é criado, o programa tem que especificar o endereço do domínio e o tipo de *socket* para que a conexão possa ser estabelecida. Dois processos podem se comunicar somente se seus *sockets* são do mesmo tipo e no mesmo domínio.

Há dois tipos de *sockets*:

- *stream sockets:* tratam comunicações como uma sequência contínua de caracteres e utilizam o protocolo de comunicação TCP, que é confiável;
- *datagram sockets:* precisam ler mensagens inteiras de uma vez e utilizam o UDP(Unix Datagram Protocol), que é pouco confiável e orientado para mensagens.

A utilização de *sockets* representa uma boa solução para a escrita de programas cliente-servidor em ambientes que suportem a arquitetura TCP/IP. Com a utilização destes, a rede é vista como se fosse um arquivo, permitindo que se utilize, para a comunicação, primitivas simples como *read/write* ou *sendto/recvfrom*.

2.8 Java

2.8.1 A linguagem Java

Java [19] é uma linguagem computacional completa, adequada para o desenvolvimento de aplicações baseadas na rede Internet, redes fechadas ou ainda programas *stand-alone*.

Foi desenvolvida na década de 90 nos laboratórios da *Sun Microsystems* com o objetivo de ser mais simples e eficiente do que suas predecessoras. O alvo inicial era a produção de *software* para produtos eletrônicos de consumo (fornos de microondas, agendas eletrônicas, etc.).

A linguagem obteve sucesso em cumprir os requisitos de sua especificação,mas apesar de sua eficiência não conseguiu sucesso comercial, obteve, no entanto, grande sucesso quando passou a ser utilizado pela comunidade da Internet através de pequenos aplicativos embutidos *applets* em páginas da WEB.

Atualmente, a linguagem Java é a força propulsora por trás de alguns dos maiores avanços da computação mundial, como:

- acesso remoto a bancos de dados;

[19] O conteúdo desta seção foi extraído de DEITEL & DEITEL [24] e INDRUSIAK [25].

- bancos de dados distribuídos;
- comércio eletrônico no WWW;
- interatividade em páginas WWW;
- interatividade em ambientes de Realidade Virtual distribuídos;
- gerência de Documentos;
- integração entre dados e forma de visualização;
- *Network Computer*;
- ensino à distância;
- jogos e entretenimento.

2.8.2 Características

Principais características da linguagem Java:

- é uma linguagem simples, de fácil aprendizado ou migração, pois possui um reduzido número de construções;
- suporte a orientação a objetos permitindo a modularização das aplicações, reuso e manutenção simples do código já implementado;
- robustez;
- segurança: enfatiza a verificação em tempo de compilação de possíveis problemas e em tempo de execução realiza checagem dinâmica e eliminação de situações que podem gerar erros;
- traz classes para o suporte a vários níveis de conectividade: acesso a URLs (padrão Internet), uso de conexões em *sockets*, criação de protocolos, criação de clientes e servidores;
- independência da plataforma;
- execução automática de coletor de lixo(*garbage colector*);
- presença de recursos para tratamento de *threads* possibilitando programação concorrente.

Capítulo 3

Um protótipo do jogo de Batalha Naval utilizando WAP

Para que o objetivo deste trabalho pudesse ser alcançado, inicialmente foi necessário realizar uma pesquisa bibliográfica aprofundada para que se pudesse compreender o funcionamento da tecnologia WAP, bem como suas restrições, vantagens e desvantagens.

Foi também necessário estudar a linguagem de programação WML, pois é esta a linguagem reconhecida pelo protocolo WAP, analisando entre outros aspectos, quais suas principais características, vantagens e desvantagens.

Para se conhecer um pouco melhor as funcionalidades do WML, foi implementado um protótipo do jogo de Batalha Naval. Neste, o jogador poderia jogar apenas contra o computador e, não ainda contra um adversário real. Embora esta implementação não seja muito complexa, já que não precisa fazer a interação entre cliente e servidor, foram encontradas diversas dificuldades para desenvolvê-la.

Após a implementação deste prótotipo, foi feita uma análise de requisitos para se implementar jogos interativos e distribuídos nesta linguagem. Nesta fase do projeto, começaram a surgir as maiores dificuldades pois descobriu-se que implementar um jogo em tempo real entre dois ou mais jogadores sobre o protocolo HTTP é algo não trivial. Isto ocorre porque tanto a linguagem *HyperText Markup Language*(HTML) quanto o WML, são linguagens de marcação, ou seja, servem simplesmente para se indicar formatações para textos, inserir imagens e ligações de hipertexto, e não para se programar.

Para resolver este problema, a solução encontrada foi a de interagir o WML com um programa servidor escrito em PHP, esta conexão é estabelecia através dos

métodos *get* e *post*.

A linguagem PHP, é capaz de trabalhar com diferentes sessões para os usuários, mas não permite que haja interação entre elas, sendo assim, utilizando apenas esta linguagem não é possível verificar se um jogador atingiu outro, de quem é a vez de jogar ou mesmo mandar mensagens diferentes para cada um deles.

É importante ressaltar que estas mesmas dificuldades também são encontradas quando se interage WML com ASP ou com *Servlets Java*. Sendo assim, para possibilitar a implementação de jogos interativos em WAP, uma alternativa seria fazer uma interação através de *sockets* do servidor PHP com outro servidor escrito na linguagem Java, este sim, capaz de gerenciar sessões e usuários. Esta interação se encontra descrita na figura 3.1.

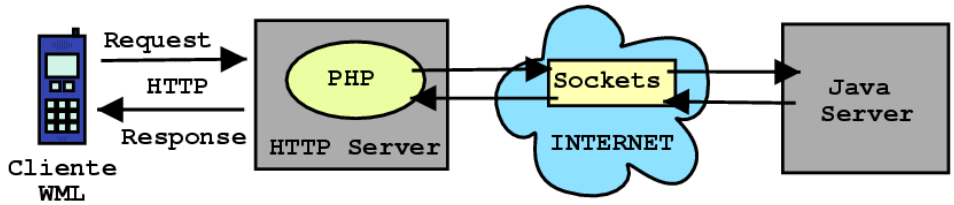

Figura 3.1: Interação entre os servidores

A partir destas considerações, é possível se discutir e demonstrar todas as dificuldades e facilidades encontradas para se implementar um jogo interativo utilizando a tecnologia WAP.

Para facilitar seu entendimento, foi dividido nas seguintes seções: seção 3.1 Ferramentas necessárias - demonstra quais as ferramentas necessárias para se desenvolver um aplicativo em WAP, justificando o motivo da escolha dos aplicativos adotados; seção 3.2 O Jogo - descreve o funcionamento do jogo de uma maneira geral; seção 3.3 Interface - mostra as duas possibilidades encontradas para se fazer a interface com o jogador, elucidando suas principais vantagens e desvantagens; seção 3.4 Dinâmica do Jogo - descreve o funcionamento do jogo fazendo um comparativo entre a perspectiva do jogador e a perspectiva tecnológica; seção 3.5 Comunicação Cliente-Servidor - mostra como é feita a comunicação entre o cliente e o servidor, descrevendo de forma detalhada as funções de cada um dos servidores.

3.1 Ferramentas necessárias

Para se escrever um código em WML, pode-se utilizar qualquer editor de texto, no entanto para o desenvolvimento deste projeto utilizou-se o *WAP Nokia Toolkit* (figura 3.2), constiuído ao mesmo tempo de um editor de textos e imagens (figura 3.3) e um emulador (figura 3.4). Apesar de haver muitas opções de emuladores e editores, este *toolkit* foi escolhido por ser o mais recomendado em praticamente todas a referências.

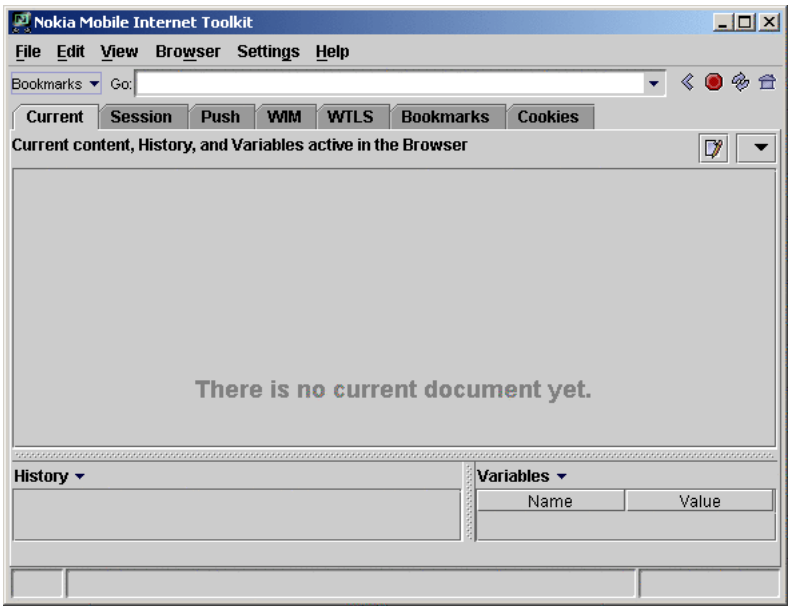

Figura 3.2: *WAP Nokia Toolkit*

As aplicações desenvolvidas em PHP e Java, também podem ser escritas em qualquer editor de texto, mas os mais usados foram o *JFE* e o *Anjuta*. Além disso, também há a necessidade de se utilizar um servidor *Apache* para interpretar a linguagem PHP e um compilador java, para executar a compilação desta.

3.2 O Jogo

O jogo escolhido para demonstrar uma aplicação do protocolo WAP foi o de Batalha Naval, pois é um jogo simples mas não trivial.

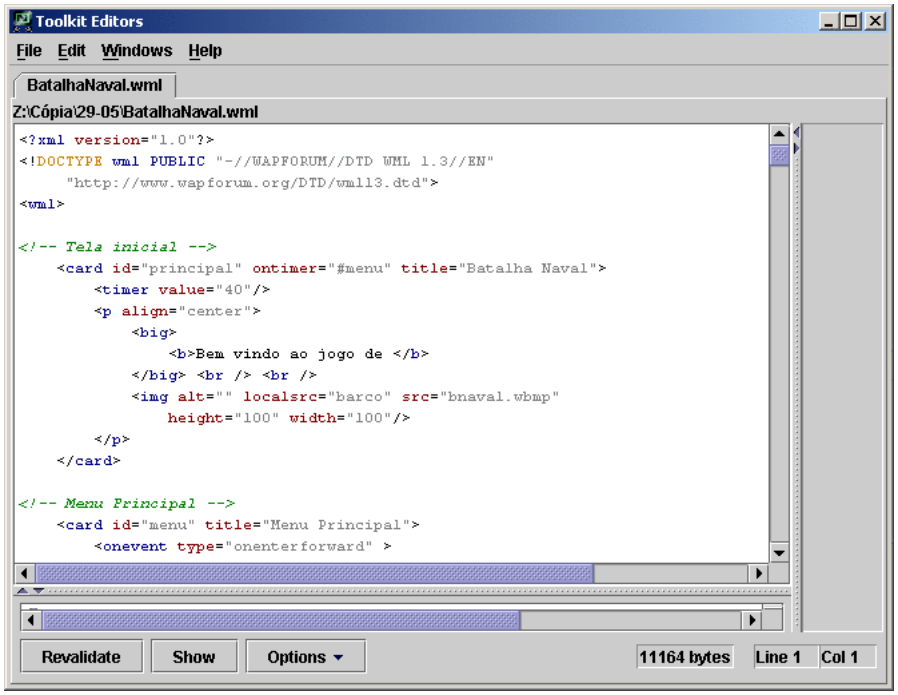

Figura 3.3: Editor do *Toolkit*

3.2.1 Regras do Jogo

Armas disponíveis:

1 Porta-aviões
1 Encouraçado
1 Cruzador
1 Submarino

As armas descritas acima ocupam respectivamente quatro, três, dois e dois quadros consecutivos no tabuleiro.

Preparação do jogo:

1. Cada jogador distribui suas armas pelo tabuleiro. Neste jogo isto é feito de forma aleatória pelo próprio dispositivo, quando o usuário clicar no botão ”Novo Jogo”.

Figura 3.4: Emulador do *Toolkit*

2. Não é permitido que 2 armas se toquem.

3. O jogador não deve revelar ao oponente as localizações de suas armas.

4. Para cada usuário serão exibidos, dois tabuleiros, onde o primeiro demonstra a posição dos seus próprios navios e o segundo do seu oponente, onde lhe será permitido clicar para disparar os tiros.

5. Os tabuleiros são constituídos de 50 pequenos quadros, distribuídos numa estrutura de matriz com 5 linhas e 10 colunas;

6. O jogador poderá escolher entre jogar contra o computador ou contra outro jogador.

Jogando

Cada jogador, na sua vez de jogar, seguirá o seguinte procedimento:

1. Disparará 1 tiro, clicando num dos quadros do tabuleiro adversário onde deseja atirar.

2. Após cada tiro, o tabuleiro dos dois jogadores deverá ser atualizado para que o jogador tenha controle dos tiros disparados, e caso o tiro tenha acertado algum deles, ambos deverão receber uma notificação informando qual a arma foi atingida. Se ela for afundada, esse fato também deverá ser informado. Além disso aparecerá uma outra figura no local em que ele atirou, esta poderá ser um pedaço de navio ou um outro símbolo para mostrar que o tiro não acertou nenhum dos alvos.

3. Uma arma é afundada quando todas as casas que formam essa arma forem atingidas.

4. Os jogadores podem jogar simultaneamente, e seus tabuleiros deverão ser atualizados após cada disparo.

5. O jogo termina quando um dos jogadores afundar todas as armas do seu oponente.

3.3 Interface

Foram encontradas duas possibilidades para fazer a interface do jogo com o usuário:

- através de caracteres de texto, por exemplo: como '~' para simbolizar o lugar que ainda não recebeu nenhum tiro, '_' para mostrar o local que já recebeu um tiro mas que não acertou nenhum navio e 'x' para mostrar que um navio atingido. Partindo do princípio que a cada tiro todo tabuleiro deverá ser atualizado, e que neste caso as figuras são apenas simples caracteres de texto este método é vantajoso pois recarregará a tela rapidamente, no entanto, dificulta um pouco para o jogador, pois para atirar este deverá digitar as coordenadas (linha e coluna) e clicar num botão "atirar".

- por meio de figuras .wbmp: neste caso, a interface com o usuário fica bem melhor, pois pode-se demonstrar de forma clara quando e onde os navios foram atingidos, além de facilitar o disparo do tiro, pois cada quadro pode ser tratado como um link, ou seja, para atirar, basta clicar no quadro correspondente. No entanto, apesar de melhorar de forma significativa a interface com o usuário, este método tem uma grande desvantagem: o processo de carregar/recarregar figuras é muito lento.

A tabela 3.1 demonstra de modo resumido as principais vantagens e desvantagens dos dois tipos de interface e a figura 3.5 demonstra um exemplo de cada uma.

Tabela 3.1: Vantagens/Desvantagens dos modelos de interface

Interface	Vantagens	Desvantagens
Caracteres de texto	Monta o tabuleiro rapidamente	Possui uma interface ruim e dificulta o disparo de tiros
Figuras .wbmp	Interface amigável com o usuário	Demora para remontar os quadros

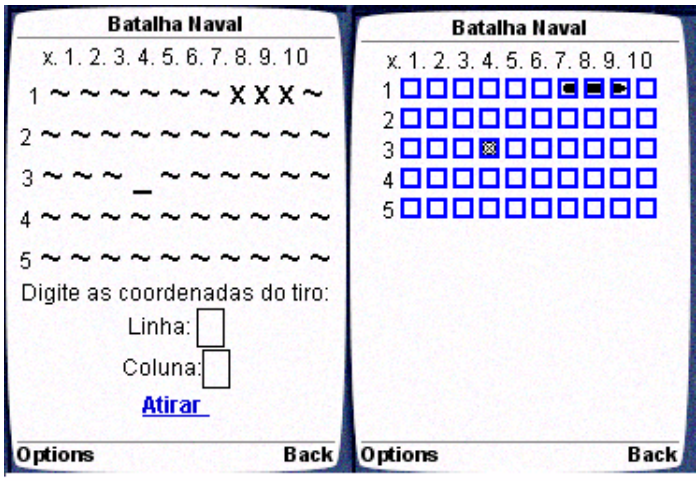

Figura 3.5: Comparação entre as duas possibilidades de interface para este jogo. A primeira utiliza caracteres de texto e a segunda figuras .wbmp.

Todas estas características das duas interfaces foram analisadas antes de se definir qual seria adotada para este projeto. Visando proporcionar uma interface mais amigável com o jogador, a segunda opção foi escolhida.

3.4 Dinâmica do Jogo

3.4.1 Perspectiva do Jogador

Todas as possíveis telas do jogo estão descritas na figura 3.6 e ilustradas nas figura 3.7 e 3.8.

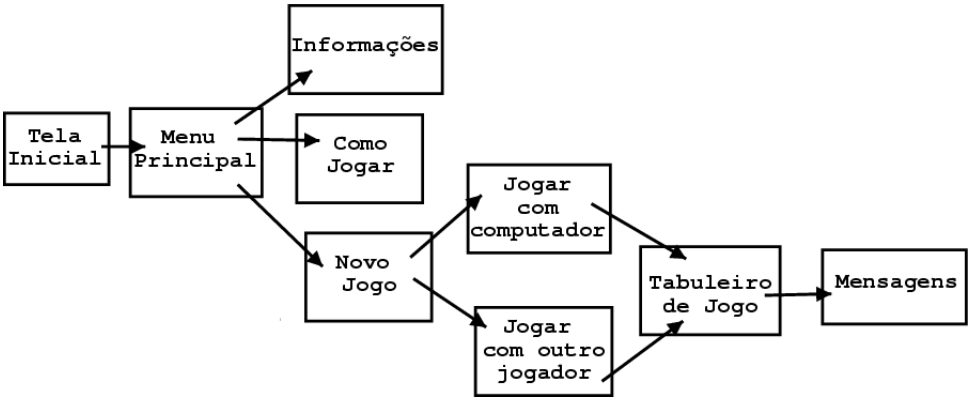

Figura 3.6: Diagrama das possíveis telas do jogo

A primeira tela é apenas uma apresentação enquanto a segunda permite ao usuário escolher entre visualizar informações, acessar a ajuda ou iniciar um novo jogo.

A tela de informações descreve o funcionamento do jogo de um modo geral, enquanto a ajuda visa explicar o que precisa ser feito para se jogar.

O jogo começa quando o usuário clica em ”Novo Jogo”. A partir deste momento o usuário poderá escolher entre jogar com o computador ou com outro jogador.

Caso ele resolva jogar com o computador, as posições dos seus navios e as do computador serão geradas aleatoriamente. E a cada tiro disparado, receberá um outro que neste caso também será calculado aleatoriamente pelo dispositivo.

No seu tabuleiro, ele poderá verificar onde estão os seus navios, em que posição recebeu um tiro e visualizar se algum deles foi atingido. Já no do adversário, lhe será possível disparar tiros, bastando para isso clicar no quadro em que deseja atirar. A cada disparo toda a tela deverá ser atualizada.

O jogador é notificado toda vez que um de seus navios for atingido e toda vez que atingir algum navio adversário, não há contagem de pontos e será o ganhador o que primeiro destruir toda a frota inimiga.

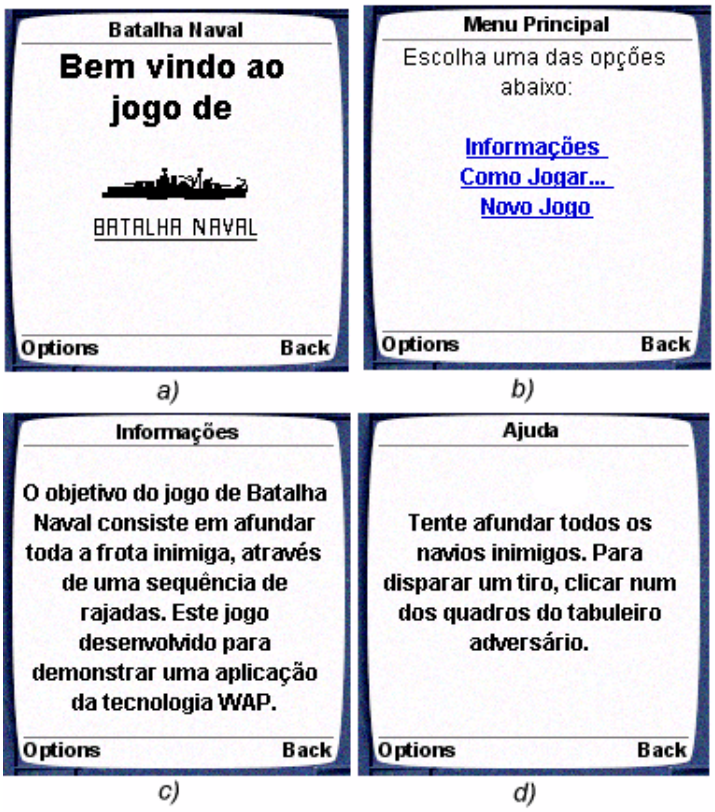

Figura 3.7: Telas do Jogo: a)Tela inicial do jogo b)Menu Principal c)Informações d)Ajuda

3.4.2 Perspectiva Tecnológica

O programa que fica no cliente é constituído de dois arquivos, um escrito em WML e outro em *WMLScript*. O primeiro é responsável principalmente pela exibição das telas ao usuário e pelo envio de mensagens para o servidor PHP, e o segundo complementar ao primeiro, possibilita a manipulação de funções como por exemplo a que calcula a posição em que os navios ficarão no tabuleiro.

Ao clicar no botão “Novo Jogo”, o usuário poderá escolher entre jogar contra o computador ou contra um outro jogador.

Caso ele escolha jogar contra o computador, será executada a função para encontrar aleatoriamente as posições dos navios do jogador e do computador. Após

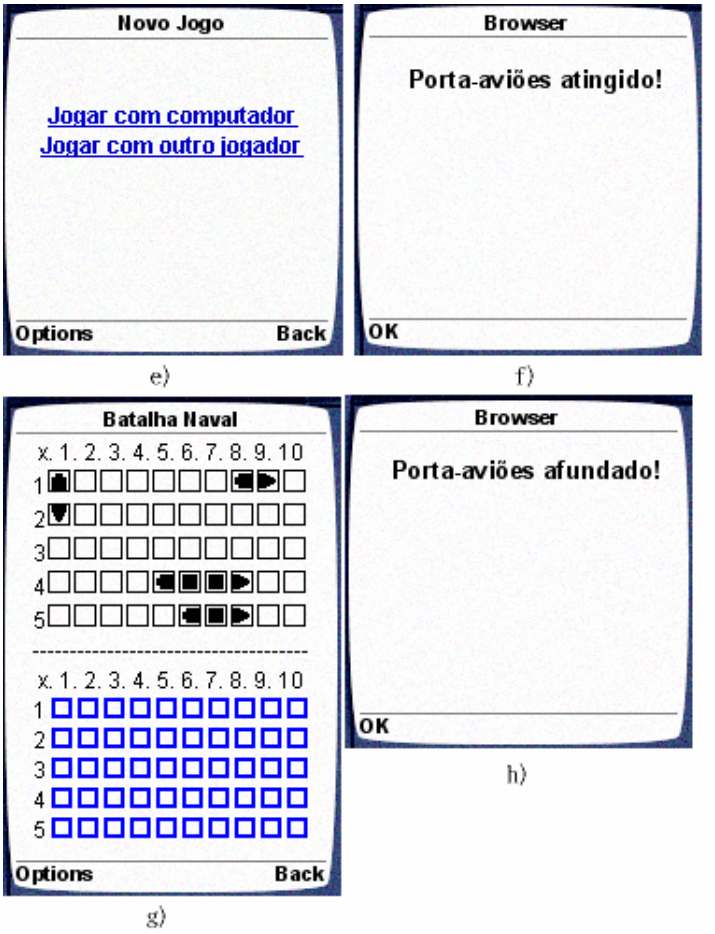

Figura 3.8: Telas do Jogo: e) Opções de novo jogo f) Mensagem navio atingido g) Tabuleiro do jogo h) Mensagem navio afundado

este processo, o tabuleiro será atualizado e exibido.

Ao disparar um tiro, uma função no arquivo *WMLScript* verificará se algum navio inimigo foi atingido e enviar uma mensagem de notificação caso isto tenha ocorrido. A coordenada do tiro do computador também será encontrada aleatoriamente e o processo de verificação será o mesmo. Após cada tiro todo o tabuleiro deverá ser atualizado.

Para o usuário, visualmente, não faz muita diferença se ele está jogando con-

tra o computador ou contra um jogador real, a diferença é que no segundo caso o dispositivo irá gerar apenas um conjunto de coordenadas para as posições dos navios, as do próprio jogador e quando um tiro for disparado, ao invés do próprio dispositivo verificar se acertou algum navio ou não, haverá uma troca de mensagens entre os programas WML-PHP-Java, e esta verificação será feita apenas no servidor Java.

3.5 Comunicação Cliente-Servidor

Como já foi dito anteriormente, para viabilizar a construção de um jogo interativo utilizando a linguagem WML, foi necessário trabalhar também com PHP e Java. A figura 3.9 ilustra as trocas de mensagens executadas entre estes aplicativos.

Em seguida são descritas as principais funções de cada um destes programas.

O Cliente WML

O código implementado no cliente, será responsável principalmente pelo que será exibido ao jogador e por enviar mensagens para o servidor PHP, utilizando para isso os métodos *get* e *post*. Uma desvantagem deste processo é que utilizando este tipo de transmissão de dados, o servidor só envia alguma informação para o cliente quando este solicita, ou seja, o cliente deverá sempre ”perguntar” qual é a sua situação atual para poder assim atualizar sua tela.

Quando o usuário clica no botão ”novo jogo”, deverá ser enviado ao servidor PHP, um número que irá identificar o usuário e a posição dos seus navios numa mensagem do tipo: JOGAR_idJog_posNavios. Ele poderá receber como resposta uma das duas mensagens: TIRO_idJog ou NÃOJOG_idJog. A primeira significa que o usuário foi aceito para jogar e deverá enviar a coordenada do tiro e a segunda que houve algum problema, e neste caso o usuário deverá continuar tentando jogar.

Depois disto, cada vez que o jogador clicar num quadro adversário, ou seja, cada vez que ele atirar, novos dados serão enviados ao servidor, neste caso porém deve-se enviar o seu identificador e a coordenada do tiro(TIRO_idJog_torpedo).

Exemplo de código em WML utilizado para enviar o identificador e a posição dos navios, através do método POST:

```
<onevent type="onenterforward">
   <go href="http://www.comp.ufla.br/~chesye
    /servPHP.php" method = "post">
```

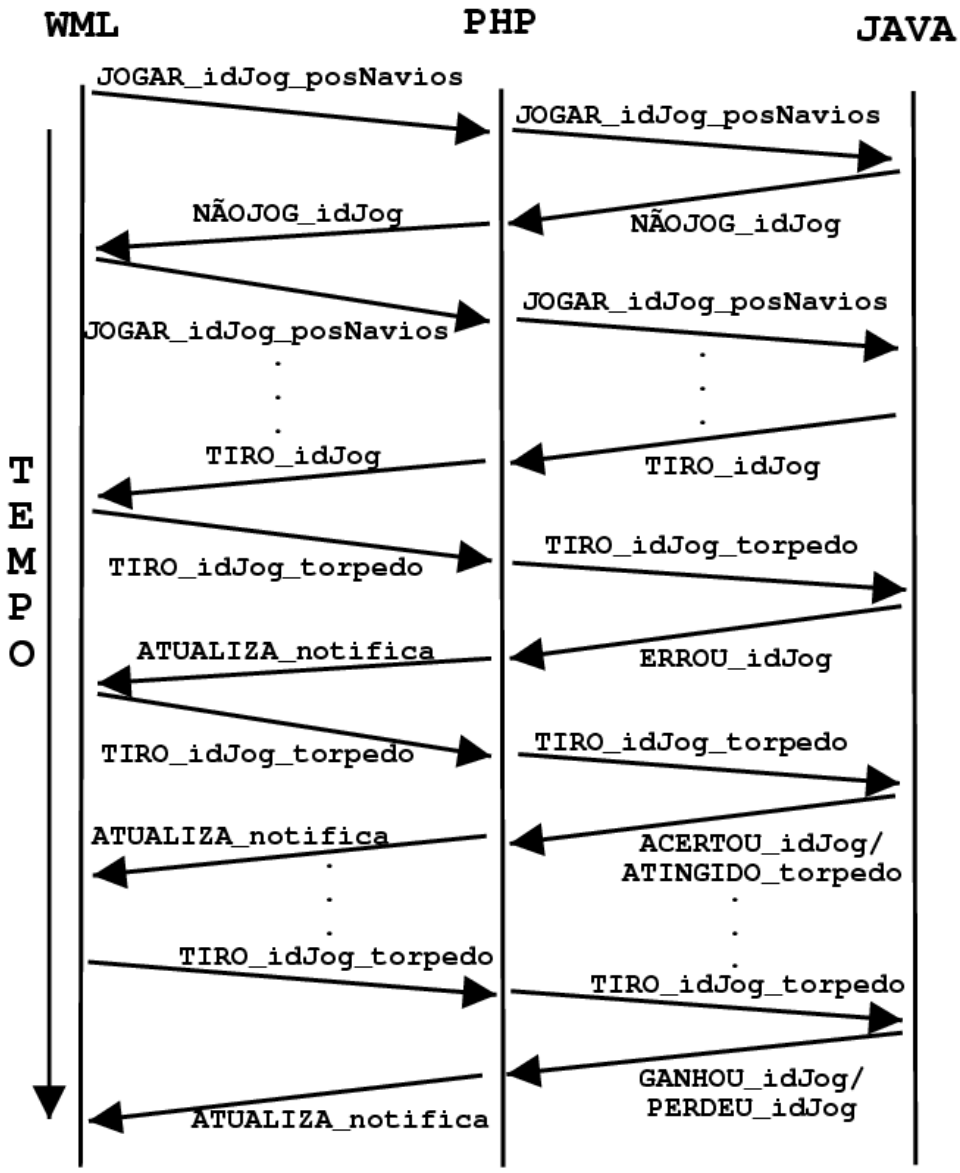

Figura 3.9: Exemplo de possíveis trocas de mensagens entre WML - PHP - Java.

```
<postfield name="idJog" value= "$(idJog)"/>
<postfield name="meusNavios"
```

```
            value="$(meusNavios)"/>
        </go>
    </onevent>
```

Após ocorrer toda a interpretação destes dados pelos servidores PHP e *Java*, os tabuleiros deverão ser reconstruídos e enviados novamente ao cliente.

Servidor PHP

Este programa será o responsável por fazer a interação entre o cliente WML e o servidor *Java*. Para que isto seja possível ele deverá criar uma conexão com o servidor através de *sockets*.

Depois de estabelecida esta conexão, ao receber informações do cliente, deverá reenviá-las ao servidor em *Java*. Na maioria dos casos, as mensagens serão apenas repassadas, isto só não ocorre quando a mensagem enviada for um tiro, pois nesta situação as respostas deverão ser analisadas e os tabuleiros atualizados e enviados ao jogador. Além disto, também deverá notificar o cliente com a resposta mais adequada, esta resposta poderá ser algo como ”envie a coordenada do tiro”(TIRO_idJog) ou ”tente começar um jogo novamente” (NÃOJOG_idJog).

Servidor Java

Neste programa estará a parte mais importante da interação cliente-servidor, pois é nele que ocorre todo o processamento de verificação se o jogador já está cadastrado ou não, com quem ele está jogando, de quem é a vez e se o tiro disparado acertou ou não a frota inimiga.

A primeira mensagem recebida por este consiste em duas informações: um identificador do jogador e a posição de seus navios. Para armazenar estas informações, este servidor possuirá um vetor para registrar todos os usuários que estão jogando, assim toda vez que receber este tipo de mensagem, deverá verificar se o id já pertence ao vetor, se sim, isto significa que ele já havia feito uma requisição para jogar anteriormente, se não, suas informações deverão ser adicionados a ele.

Como apenas dois usuários poderão jogar ao mesmo tempo, deverá haver um modo de definir quem irá jogar com quem. Para facilitar esta decisão, pode-se utilizar as posições do vetor de jogadores, assim, o jogador da posição 0 (zero) jogaria com o da posição 1 (um), o da posição 2 (dois) com o da 3 (três) e assim por diante. Talvez este não seja o melhor método para definir os adversários, mas certamente será simples e justo, pois leva em consideração a ordem de requisição para jogar dos usuários.

Caso um jogador envie uma requisição para jogar e não haja um adversário para ele (isto pode ocorrer se este for o único usuário querendo jogar, ou se houver um número ímpar de usuários cadastrados), este deverá ser notificado.

Um outro tipo de mensagem, que ele poderá receber, será aquela enviada quando um tiro for disparado. Esta mensagem, trará consigo o identificador do jogador e a coordenada do tiro. Neste caso, o programa deverá verificar a posição deste jogador no vetor e a partir desta encontrar seu adversário.

Após esta etapa, deverá verificar se a coordenada do tiro coincide com alguma das posições dos navios do inimigo, caso isto aconteça, deverá ser enviada uma notificação para os dois jogadores informando qual navio foi atingido.

Uma observação importante a ser feita é que nem sempre as mensagens enviadas para os jogadores será a mesma, e sendo assim, será responsabilidade deste programa definir qual mensagem deverá ser enviada para qual jogador.

Capítulo 4

Conclusões

Este capítulo visa mostrar quais foram as conclusões obtidas após o estudo da tecnologia WAP em jogos interativos e distribuídos. Demonstrando quais as suas vantagens e desvantagens e algumas sugestões para trabalhos futuros. Para melhor compreensão, está dividido em duas seções: seção 4.1 - Analisando o WAP em jogos interativos e distribuídos e seção 4.2 - Sugestões para trabalhos futuros.

4.1 Analisando o WAP em jogos interativos e distribuídos

Pode-se concluir, através dos estudos realizados nesta monografia, que implementar um jogo em tempo real entre dois ou mais jogadores utilizando a tecnologia WAP é algo não trivial. Isto ocorre, principalmente, porque até o presente momento, a única linguagem suportada por esta tecnologia é o WML, uma linguagem de marcação que serve simplesmente para se indicar formatações para textos, inserir imagens e ligações de hipertexto, e não para se fazer programas mais complexos.

O WML também possui muitas restrições com relação ao HTML, já que foi criado especialmente para atender às necessidades dos dispositivos e redes *wireless*. Devido a estas restrições, um novo conceito de interface homem-máquina deve ser adotado para se desenvolver aplicações para estes dispositivos.

Jogos simples, com apenas um jogador por exemplo, são simples de serem implementados pois a programação neste caso ficaria restrita ao cliente, deve-se ressaltar no entanto, que isto não significa que não seja possível criar jogos *multiplayers* em WML, mas apenas que como esta linguagem não foi criada com esta finalidade, talvez não seja capaz de produzir resultados satisfatórios para este tipo

de aplicativo.

Apesar de todas estas considerações, pode-se dizer que a linguagem WML além de ser muito simples e de fácil compreensão, possui os todos os pré-requisitos necessários para se construir uma boa estrutura de navegação para Internet em dispositivos móveis.

4.2 Sugestões para trabalhos futuros

É importante deixar claro, que até o presente momento, apenas parte do Jogo de Batalha Naval interativo foi implementado. Entretanto, como todas as suas especificações já foram definidas, pôde-se realizar a análise proposta da tecnologia WAP. Desta forma, a principal sugestão para trabalhos futuros seria continuar a implementação e melhorá-la nos seguintes aspectos:

- ampliar o jogo para mais de 2 jogadores;
- deixar que o jogador decida onde ficarão os seus navios;
- verificar a real viabilidade de se trabalhar com figuras .wbmp ao invés de simples caracteres, objetivando otimizar o tempo para exibi-las no celular.

Referências Bibliográficas

[1] AGUSTINI, A. M. V.; Novos Papéis da Tecnologia para a Internet: Wireless Application Protocol. Disponível por www em `http://www.revista.unicamp.br/infotec/artigos/anapatr2.html`. Pesquisado em Novembro de 2002.

[2] ZANETTI, A. R.; Gonçalves, L. C. Redes Locais sem Fio. Disponível por www em `http://www.dc.ufscar.br/~carvalho/WLAN/index.html`. Pesquisado em Novembro de 2002.

[3] Wap Forum, Wireless Application Protocol Architecture Specification, 1998. Disponível por www em `http://www1.wapforum.org/tech/terms.asp?doc=WAP-100`. Pesquisado em Novembro de 2002.

[4] SILVA JÚNIOR, G. F., WAP - Wireless Application Protocol, 2000. 94 f. Monografia apresentada ao CCFT, Curso de Tecnologia em Processamento de Dados, Aracaju/SE. Disponível por www em `http://sites.uol.com.br/helyrosa/resumo.html`. Pesquisado em Novembro de 2002.

[5] Phone.com. The Wireless Application Protocol. Wireless Internet Today, Redwood City: 1999. Disponível por www em `http://www.phone.com/pub/feb99WAPWP.pdf`. Pesquisado em Novembro de 2002.

[6] CABRAL, J. L. M.; LEITE, L. M., Segurança em Transações e Aplicações WAP. Monografia apresentada ao Curso de Informática do CCEI - Centro de Ciências de Economia e Informática da URCAMP - Universidade da Região da Campanha. Disponível por www em `http://sites.uol.com.br/wirelessbr/campanha1.html`. Pesquisado em Novembro de 2002.

[7] HENKEL, C. A., WAP - Wireless Application Protocol, 2001. 108 f. Monografia apresentada à UNISINOS, Curso Especialização em Redes de Computadores e Internet, São Leopoldo/RS. Disponível por www em http://www.wapbr.org/forum/artigos/monografia3.asp. Pesquisado em Novembro de 2002.

[8] SPOSITO, G., WirelessBR. Desenvolvimento de Aplicações WAP. Brasil, Agosto de 2000. Disponível por www em http://sites.uol.com.br/helyr/sposito01. Pesquisado em Novembro de 2002.

[9] SPOSITO, G., WirelessBR. Desenvolvimento de Aplicações WAP: Parte II - *Wireless Markup Language*. Brasil, Agosto de 2000. Disponível por www em http://sites.uol.com.br/helyr/sposito02. Pesquisado em Novembro de 2002.

[10] Telemóveis.com Disponível por www em http://www.telemoveis.com/articles/item.asp?ID=59. Pesquisado em Novembro de 2002.

[11] WML Club, Dados sobre o mercado da tecnologia sem fio. Disponível por www em http://br.wmlclub.com/docs/mercamundo.htm. Pesquisado em Junho de 2003.

[12] DEITEL,H.M., DEITEL,P.J., NIETO,T.R., STEINBUHLER,K., Wireless Internet and Mobile Business How to Program. Prentice Hall, 1a Edição, 2002.

[13] MANN, S., Programando Aplicativos WAP. São Paulo, Makron Books, 1a Edição, 2000.

[14] RISCHPATER, R., Desenvolvendo Wireless para WEB. São Paulo, Makron Books, 1a Edição, 2000.

[15] SILVA, J., Um Estudo da Tecnologia WAP, 2001. 142 f. Trabalho apresentado à Universidade Regional Integrada do Alto Uruguai e das Missões, Curso de Informática, Frederico Westphalen/RS. Disponível por www em http://www.bibliovirtual.fw.uri.br/bvinf/tc2001/tcjoel.pdf. Pesquisado em Abril de 2003.

[16] Universidade de Aveiro, Redes Wireless. Disponível por www em http://www.wireless.ua.pt/funciona.asp. Pesquisado em Maio de 2003.

[17] WML Club. Disponível por www em http://br.wmlclub.com/programas/. Pesquisado em Maio de 2003.

[18] Nokia - Connecting People. Disponível por www em http://www.forum.nokia.com/main.html. Pesquisado em Maio de 2003.

[19] Openwave. Disponível por www em http://developer.openwave.com/download/. Pesquisado em Maio de 2003.

[20] WirelessBr. Disponível por www em http://helyr.sites.uol.com.br/simuladores.html. Pesquisado em Maio de 2003.

[21] CONVERSE, T., PARK, J., PHP 4 - A Bíblia. Rio de Janeiro, Editora Campus, 2001.

[22] DIAS, S. P., Criando Aplicações Web Dinâmicas Utilizando PHP e MySQL. Lavras, 2002.

[23] VIVAS, M., Curso de Aplicações WEB em PHP, Comite de Incentivo a Produção do Software Gratuito e Alternativo CIPSGA. Junho de 2000.

[24] DEITEL,H.M., DEITEL,P.J., Java - Como Programar. Bookman, 4a Edição, 2002.

[25] INDRUSIAK, L. S., Linguagem Java. Grupo JavaRS, Rio Grande do Sul, 1996. Disponível por www em http://www.inf.ufrgs.br/tools/java/introjava.pdf. Pesquisado em Maio de 2003.

[26] FERREIRA, P. O., Sockets. Instituto Politécnico da Guarda - ESTG, 2002. Disponível por www em http://www.ipg.pt/user/~sduarte/rc/trabalhos/Sockets/. Pesquisado em Maio de 2003.

[27] Sockets: apresentação e comentários. Disponível por www em http://magnum.ime.uerj.br/~alexszt/cursos/topesp_inter/trabs/992/g4/. Pesquisado em Maio de 2003.

[28] WML Spiele. Disponível por www em http://www.siemensinfo.de/games.htm. Pesquisado em Junho de 2003.

Apêndice A

Lista de Abreviaturas

API *Application Programming Interface*

CDMA *Code Division Multiple Access*

CDPD *Cellular Digital Packet Data*

CGI *Common Gateway Interface*

CSD *Circuit Switched Cellular Data*

DSSS *Direct Sequence Spread Spectrum*

ECMA *European Computer Manufacturers Association*

EDGE *Enhanced Data Rates for Global Evolution*

FDMA *Frequency Division Multiple Access*

GPRS *General Packet Radio Services*

GSM *Global System for Mobile Communication*

HDML *Handheld Device Markup Language*

HTML *HyperText Markup Language*

HTTP *HyperText Transfer Protocol*

ICMP *Internet Control Message Protocol*

ITTP *Intelligent Terminal Transfer Protocol*

MAC *Message Authentication Code*

MDI *Mobile Data Initiative*

MMI *Man-Machine Interface*

NTP *Network Time Protocol*

PDA *Personal Digital Assistant*

PDU *Protocol Data Unit*

SDK *Software Development Kit*

SGML *Standardized Generalized Markup Language*

SMS *Smart Messaging Service*

SSL *Secure Socket Layer*

TCP *Transmission Control Protocol*

TDMA *Time Division Multiple Access*

TLS *Transport Layer Security*

TTML *Tagged Text Markup Language*

UI *User Interface*

URI *Universal Resource Identifier*

URL *Universal Resource Locator*

XML *Extensible Markup Language*

WAE *Wireless Application Environment*

WAP *Wireless Application Protocol*

WTAI *Wireless Telephony Application Interface*

W-CDMA *Wireless Code Division Multiple Access*

WCMP *Wireless Control Message Protocol*

WEB *Handheld Device Markup Language*

WDP *Wireless Datagram Protocol*

WML *Wireless Markup Language*

WSP *Wireless Session Protocol*

WTA *Wirelers Telephony Applications*

WTLS *Wireless Transport Layer Security*

WTP *Wireless Transaction Protocol*

WWW *World Wide Web*

Resumo

Dias, Bruna Chesye. ANÁLISE DA TECNOLOGIA WAP VIA ESTUDO DE CASO EM JOGOS DISTRIBUÍDOS E INTERATIVOS. 73p. 2003.

Com o crescimento do mercado das telecomunicações móveis, surgiu o WAP, um protocolo mundial que torna possível o acesso à Internet por meio de dispositivos móveis sem fio, como Palms e celulares. Apesar das limitações e do preço, os dispositivos WAP têm obtido sucesso em vários países. Sendo assim, o objetivo deste trabalho consiste em estudar e avaliar esta tecnologia para aplicações distribuídas e com alto índice de concorrência e sincronização, demonstrando ao final deste projeto quais as facilidades e dificuldades encontradas para se implementar jogos interativos utilizando esta tecnologia. O jogo escolhido para fazer esta demonstração foi o de Batalha Naval, por ser um jogo simples porém não trivial. Através dos estudos realizados, pôde-se concluir que implementar um jogo em tempo real entre dois ou mais jogadores utilizando a tecnologia WAP é algo não trivial. Isto porque o WML é uma simples linguagem de marcação, não permitindo assim a construção de programas mais complexos. Uma solução encontrada foi a de interagir o WML com um programa servidor escrito em PHP (através dos métodos get e post), e este com outro servidor escrito em Java (através de sockets).

Palavras-chave: WAP, tecnologia, jogos, wml, análise, interação cliente-servidor.