Chapter 11
Mobile Web

Distributed
Computing
Group

Mobile Computing
Summer 2002

Overview

- Web and Mobility
- HTTP and HTML
- Mobile Web Architectures

- Wireless Application Protocol WAP

- WML
- WMLScript
- WTAI

World Wide Web and Mobility

- Protocol (HTTP, Hypertext Transfer Protocol) and language (HTML, Hypertext Markup Language) of the Web have not been designed for mobile applications and mobile devices.
- Typical transfer sizes
  - HTTP request: 100-350 byte
  - responses avg. <10 kbyte, header 160 byte, GIF 4.1kByte, JPEG 12.8 kbyte, HTML 5.6 kbyte
  - but also many large files that cannot be ignored
- The Web is no file system
  - Web pages are not simple files to download
  - static and dynamic content, interaction with servers via forms, content transformation, push technologies etc.
  - many hyperlinks, automatic loading and reloading, redirecting
  - a single click might have big consequences!

WWW example

- Request to port 80

```
GET/HTTP/1.0
```

- Response from server

```
HTTP/1.1 200 OK
Date: Fri, 20 Jun 2002 14:52:12 GMT
Server: Apache/1.3.26
Connection: close
Content-Type: text/html
<html> <head> <title>Distributed Computing Group</title>
<meta http-equiv="Content-Type" content="text/html;
  charset=iso-8859-1"> <link rel=stylesheet
  href="styles.css" type="text/css"> </head> <body
  bgcolor="#FFFFFF" text="#000000" link="#0000A0"
  vlink="#0000A0" alink="#00A000">
<p align="right" style="margin-bottom:0px"> <a
  href="index.html"><img border="0" height="117"
  width="429" src="pics/dcg.gif" alt="Distributed
  Computing Group"></a> </p>
<hr size="1" noshade>
<h1>Distributed Computing Group</h1>
<p class="topic">
…
```

HTTP 1.0 and mobility

- Characteristics
  - stateless, client/server, request/response
  - needs a connection oriented protocol (TCP), one connection per request (some enhancements in HTTP 1.1)
  - primitive caching and security
- Problems
  - designed for large bandwidth (compared to wireless access) and low delay
  - big and redundant protocol headers (readable for humans, stateless, therefore big headers in ASCII)
  - uncompressed content transfer
  - using TCP (3-way-handshake, slow-start)
  - DNS lookup by client causes additional traffic

HTTP 1.0 and mobility

- Caching
  - quite often disabled by information providers to be able to create user profiles, usage statistics, etc.
  - dynamic objects (counters, time, date, personalization)
  - mobility quite often inhibits caches
  - security problems (how to use SSL/TLS together with proxies?)
  - today: many user customized pages, dynamically generated on request via CGI, ASP, ...
- Sending to a server with POST method
  - can typically not be buffered
  - very problematic if currently disconnected

- Many unsolved problems!

HTML and mobile devices

- HTML
  - designed “high” performance computers: color high-resolution display, mouse, hard disk
  - web pages optimized for design, not for communication
- Mobile devices
  - small, low-resolution displays, very limited input interfaces (small touch-pads, soft-keyboards)
- Many web pages assume existence of additional features
  - animated GIF, Java applets, Frames, ActiveX, Shockwave, movie clips, audio, ...

- Web pages ignore the heterogeneity of end-systems

Approaches toward WWW for mobile devices

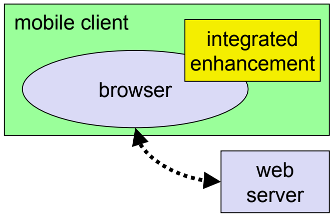
- Application gateways, enhanced servers
  - simple clients, pre-calculations in the fixed network
  - compression, filtering, content extraction
  - automatic adaptation to network characteristics
- Examples
  - picture scaling, color reduction, transformation of the document format (e.g., PS to TXT), detail studies, clipping, zoom
  - headline extraction, automatic abstract generation
  - HDML (handheld device markup language): simple language similar to HTML requiring a special browser
  - HDTP (handheld device transport protocol): transport protocol for HDML, developed by Unwired Planet
- Problems
  - proprietary approaches, require special enhancements for browsers
  - heterogeneous devices make approaches more complicated

Some new issues that might help mobility?

- Push technology
  - real pushing, not a client pull needed, channels etc.
- HTTP/1.1
  - client/server use the same connection for several request/response transactions
  - multiple requests at beginning of session, several responses in same order
  - enhanced caching of responses (useful if equivalent responses)
  - semantic transparency not always achievable: disconnected, performance, availability -> most up-to-date version...
  - several more tags and options for controlling caching (public/private, max-age, no-cache etc.)
  - relaxing of transparency on app. request or with warning to user
  - encoding/compression mechanism, integrity check, security of proxies, authentication, authorization...
- Cookies

WWW in a mobile world: Architectures

- Enhanced browsers
  - Caching, off-line use
  - Examples: Internet Explorer, Netscape



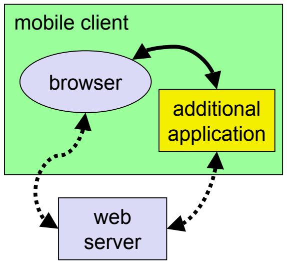
mobile client

integrated enhancement

browser

web server

- Additional, accompanying application
  - Pre-fetching, caching, off-line use
  - Example: original WebWhacker



mobile client

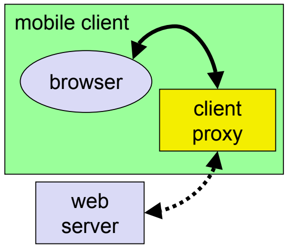
browser

additional application

web server

WWW in a mobile world: Architectures

- Client Proxy
  - Pre-fetching, caching, off-line use
  - Examples: Caubweb, TeleWeb, Weblicator, WebWhacker, WebEx, WebMirror, etc.



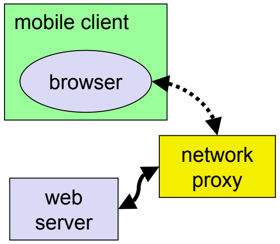
mobile client

browser

client proxy

web server

- Network Proxy
  - adaptive content transformation for bad connections, pre-fetching, caching
  - Examples: TranSend, Digestor



mobile client

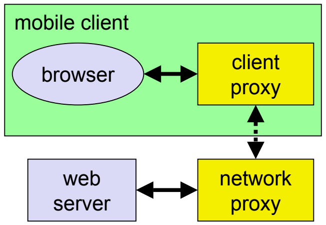
browser

network proxy

web server

WWW in a mobile world: Architectures

- Client and network proxy
  - combination of benefits plus simplified protocols
  - Examples: MobiScape, WebExpress, Mowgli



mobile client

browser

client proxy

web server

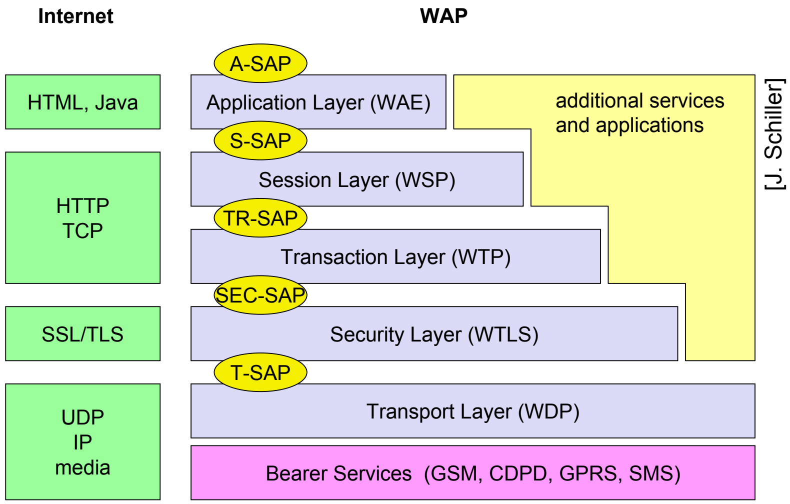
network proxy

- Additionally many proprietary server extensions possible
  - channels
  - content negotiation

Wireless Application Protocol WAP

- Goals
  - deliver Internet content and enhanced services to mobile devices and users (mobile phones, PDAs)
  - independence from wireless network standards
  - open for everyone to participate, protocol specifications will be proposed to standardization bodies
  - applications should scale well beyond current transport media and device types and should also be applicable to future developments
- Platforms
  - e.g., GSM (900, 1800, 1900), CDMA IS-95, TDMA IS-136, 3rd generation systems (IMT-2000, UMTS, W-CDMA)
- Challenger i-mode
  - A big hit in Japan, now coming to the rest of the world
  - Standardized user interface, designed by provider; thus not open
  - “SMS” is seen as (most successful) part of i-mode

WAP reference model and protocols



Internet

WAP

HTML, Java

A-SAP

Application Layer (WAE)

additional services and applications

S-SAP

HTTP
TCP

Session Layer (WSP)

TR-SAP

Transaction Layer (WTP)

SEC-SAP

SSL/TLS

Security Layer (WTLS)

T-SAP

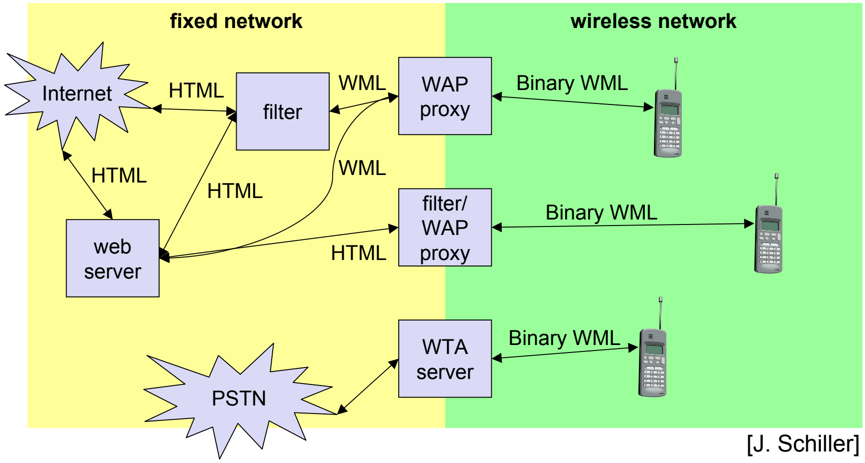
UDP
IP
media

Transport Layer (WDP)

Bearer Services (GSM, CDPD, GPRS, SMS)

[J. Schiller]

WAP network elements

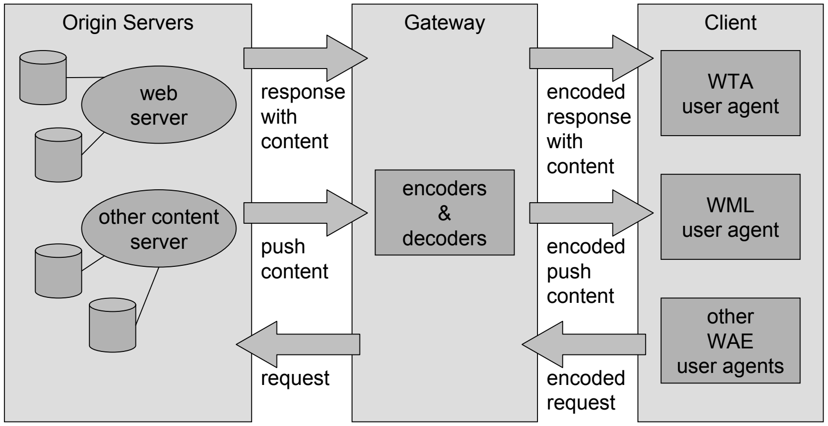


[J. Schiller]

Wireless Application Environment WAE

- Goals
  - network independent application environment for wireless devices
  - integrated Internet/WWW programming model with high interoperability
  - device and network independent, international support
  - manufacturers can determine look-and-feel, user interface
  - considerations of slow links, limited memory, low computing power, small display, simple user interface (compared to desktop computers)
- Components
  - architecture: application model, micro browser, gateway, server
  - WML: XML-Syntax, based on card stacks, variables, ...
  - WMLScript: procedural, loops, conditions, ... (similar to JavaScript)
  - WTA: telephone services, such as call control, text messages, phone book, ... (accessible from WML/WMLScript)
  - content formats: vCard, vCalendar, Wireless Bitmap, WML, ...

WAE logical model



Wireless Markup Language (WML)

- WML follows deck and card metaphor
  - WML document consists of many cards, cards are grouped to decks
  - a deck is similar to an HTML page, unit of content transmission
  - WML describes only intent of interaction in an abstract manner
  - presentation depends on device capabilities
- Features
  - text and images: only limited capabilities, depends on client
  - user interaction: text or password input, options, depends on client
  - Navigation: store already visited sites
  - context management: global variables

WML functionality

- Tags as in HTML
  - <p> <i> <b> <u> <em> <strong> <small> <big> <p align=“center”> &shy; &lt; etc.
- Links as in HTML
  - <a href=“x.html”>link to x</a>
- Supported URL protocols
  - http, https, file, ftp, gopher, mailto, news, telnet
- Other features
  - Tables <table columns=“2” title=“My Table”> …
  - Images <img src=“square.wbmp” alt=“[ ]” align=“top”/>
  - Forms <select> and <option> (see example on next slide)
  - Input <input name=“Number” type=“password”/>
  - Events <do> <onevent type=“onenterforward”> …
  - Variables <setvar> <timer>

WML example

```
<WML>
  <CARD>
      <DO TYPE="ACCEPT">
             <GO URL="#card_two"/>
      </DO>
      This is a simple first card!
      On the next you can choose ...
  </CARD>
  <CARD NAME="card_two">
      ... your favorite pizza:
      <SELECT KEY="PIZZA">
             <OPTION VALUE=”M”>Margherita</OPTION>
             <OPTION VALUE=”F”>Funghi</OPTION>
             <OPTION VALUE=”V”>Vulcano</OPTION>
      </SELECT>
  </CARD>
</WML>
```

WMLScript

- Complement to WML
- Provides general scripting capabilities

- validity check of user input
  - check input before sending it to server
- access to device facilities
  - hardware and software (phone call, address book etc.)
- local user interaction
  - interaction without round-trip delay
- extensions to the device software
  - configure device, download new functionality after deployment

WMLScript functionality

- Data types
  - Boolean, Integer, Real, String, invalid
  - Data types have no fixed type

- Control structures similar to Java (and C, for that matter)
  - if (condition) {…} else {…}
  - while (condition) {…}; (with break/continue, and other features)
  - function f (parameters) {… return result};

- External call
  - use url money “http://wap.money.com/money.wmlsc”;
  - function CHFtoUSD (CHF) {return money#CHFtoUSD(CHF)};

WMLScript main libraries: function examples

- The dialogs library:
  - Dialogs.alert() = create an alert message
  - Dialogs.confirm() = create a confirmation dialog
- The float library:
  - Float.ceil() = return equal or nearest bigger integer
  - Float.int() = return the integer part of the value
- The lang library:
  - Lang.exit() = exit function
  - Lang.float() = test if the device supports floating numbers
- The String library:
  - String.length() = display the length of the string
  - String.trim() = remove extra spaces before and after a string
- The URL library:
  - URL.escapeString() = encode string as URL
  - URL.getScheme() = return the used protocol
- The WMLBrowser library:
  - WMLBrowser.getCurrentCard() = return the address of the current card
  - WMLBrowser.go() = move to another address

WMLScript example

```
function pizza_test(pizza_type) {
  var taste = "unknown";
  if (pizza_type = "Margherita") {
      taste = "well... ";
  }
  else {
      if (pizza_type = "Vulcano") {
             taste = "quite hot";
      };
  };
  return taste;
};
```

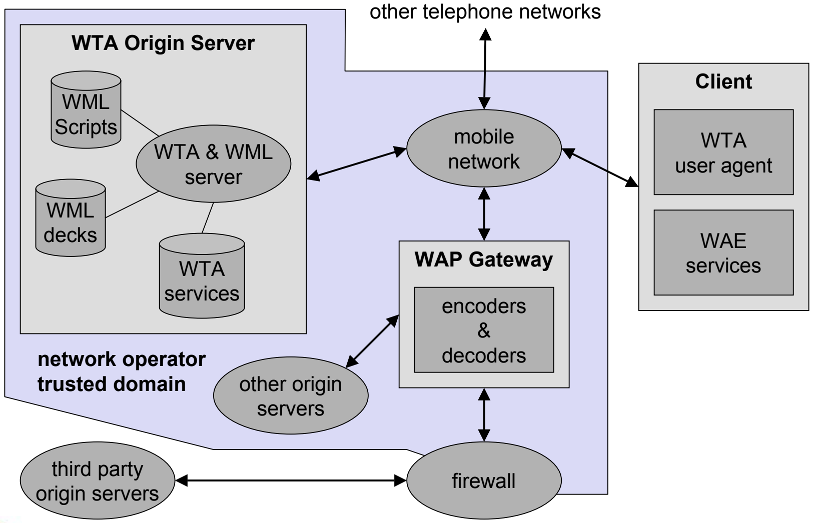
WMLScript is not type-safe

```
extern function allsum(i) {
  var j,sum;
  sum = 0; //attention: if you remove this line, then
           //allsum(5)="12345" :-)
  for (j=1;j<=i;j++) {
   sum = sum + j;
  }
  Dialogs.alert("Summe = "+sum);
  return sum;
}
```

Wireless Telephony Application (WTA)

- Collection of telephony specific extensions
- Extension of basic WAE application model
  - content push
    - server can push content to the client
    - client may now be able to handle unknown events
  - handling of network events
    - table indicating how to react on certain events from the network
  - access to telephony functions
    - any application on the client may access telephony functions
- Example
  - calling a number (WML)
    `wtai://wp/mc;07216086415`
  - calling a number (WMLScript)
    `WTAPublic.makeCall("07216086415");`

WTA logical architecture



WTA Origin Server

WML Scripts

WTA & WML server

WML decks

WTA services

network operator trusted domain

other origin servers

other telephone networks

mobile network

WAP Gateway

encoders & decoders

firewall

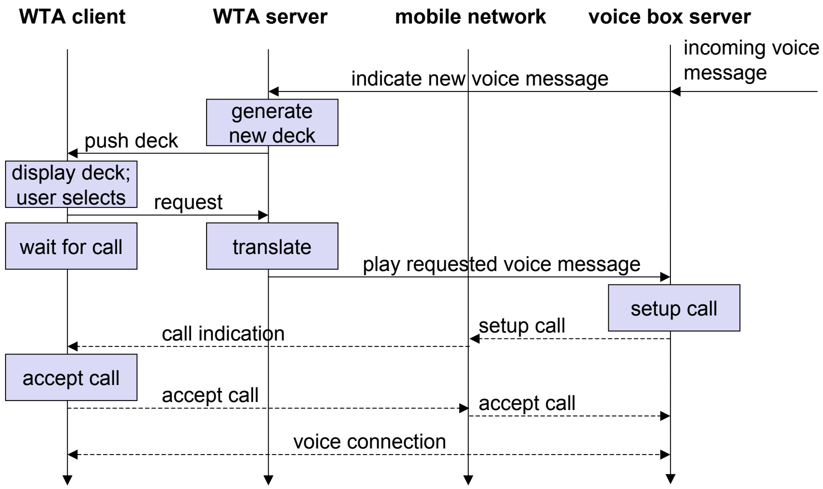
third party origin servers

Client

WTA user agent

WAE services

Voice box example



WTA client

WTA server

mobile network

voice box server

incoming voice message

indicate new voice message

generate new deck

push deck

display deck; user selects

request

wait for call

translate

play requested voice message

setup call

setup call

call indication

accept call

accept call

accept call

voice connection

WTAI example with WML only

```
<WML>
   <CARD>
        <DO TYPE="ACCEPT" TASK="GO" URL="#voteChamp"/>
        Please vote for your champion!
   </CARD>
   <CARD NAME="voteChamp">
        <DO TYPE="ACCEPT" TASK="GO“
                URL="wtai://cc/mc;$voteNo;1"/>
        Please choose:
        <SELECT KEY="voteNo">
                <OPTION VALUE="6086415">Mickey</OPTION>
                <OPTION VALUE="6086416">Donald</OPTION>
                <OPTION VALUE="6086417">Pluto</OPTION>
        </SELECT>
   </CARD>
</WML>
```

WTAI example with WML and WMLScript

```
function voteCall(Nr) {
   var j = WTACallControl.setup(Nr,1);
   if (j>=0) {
        WMLBrowser.setVar("Message", "Called");
        WMLBrowser.setVar("No", Nr);
   }
   else {
        WMLBrowser.setVar("Message", "Error!");
        WMLBrowser.setVar("No", j);
   }
   WMLBrowser.go("showResult");
}
```

WTAI example with WML and WMLScript

```
<WML>
   <CARD>
        <DO TYPE="ACCEPT" TASK="GO" URL="#voteChamp"/>
   Please vote for your champion!
   </CARD>
   <CARD NAME="voteChamp">
        <DO TYPE="ACCEPT" TASK="GO" URL="/script#voteCall($voteNo)"/>
        Please choose:
        <SELECT KEY="voteNo">
                <OPTION VALUE="6086415">Mickey</OPTION>
                <OPTION VALUE="6086416">Donald</OPTION>
                <OPTION VALUE="6086417">Pluto</OPTION>
        </SELECT>
   </CARD>
   <CARD NAME="showResult">
        Status of your call: $Message $No
   </CARD>
</WML>
```

Questions?

Distributed
Computing
Group