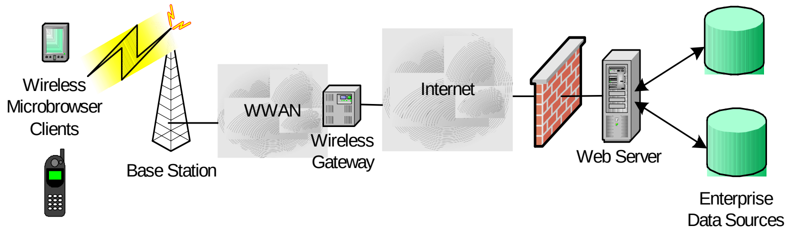
# *Thin Client Development and Wireless Markup Languages*

**David Tipper**
**Associate Professor**
Department of Information Science and Telecommunications
University of Pittsburgh
**tipper@tele.pitt.edu**

http://www.sis.pitt.edu/~dtipper/2727.html
*Slides 10*

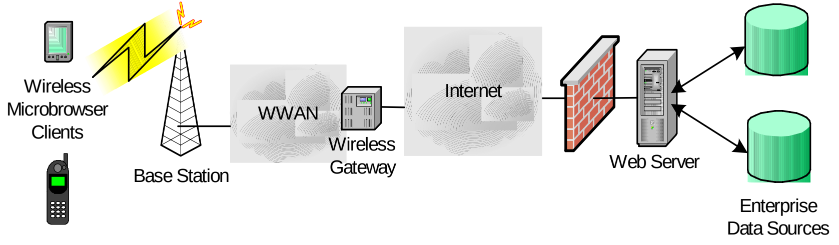
# Thin Client Development

- Adopt architecture similar to wired Internet
- Components
  - microbrowser (thin client), Web Server, content/data source, possibly a proxy gateway
- Advantages
  - Rapid deployment, fresh data, easy to use, broad deployment



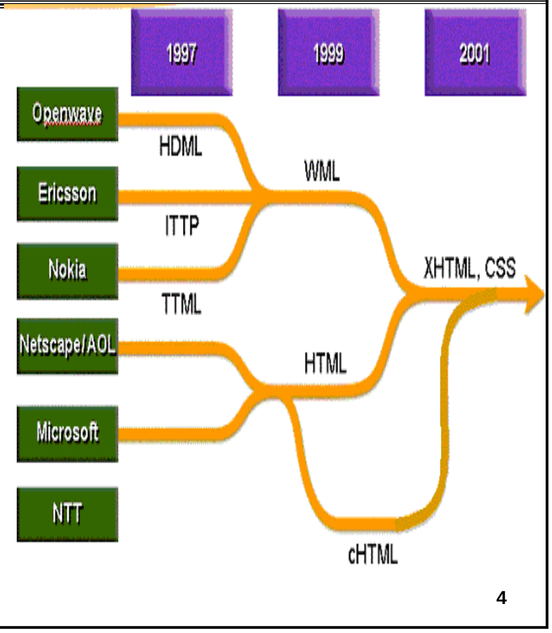
# Content

- Content Creation
  - Make content specifically for wireless handhelds
  - Use wireless markup languages (cHMTL, WML, XHTML-MP)
- Content Adaptation
  - Adapt content for desktops to handheld devices on the fly
    - How to adapt (XLST, web clipping, image mapping, etc.)
    - Where to adapt?
      - Server
      - Portal between server and wireless gateway
      - Gateway to WWAN (WAP approach)
      - Client device



# Wireless Markup Languages

- Markup Languages: ML describe how a document is rendered
- HDML: Handheld ML
- WML : Wireless ML
  - Attempt to get one markup language – part of WAP
- cHTML: compact HTML
  - Developed from i-mode service in Asia
- XHTML-MP: extensible HTML mobile profile
  - Part of WAP 2.0
- Recent survey found that most content in cHTML and WML. WAP 2.0 browsers support WML and XHTML-MP.



## Developing Thin Client Apps

- Stage 1: Development
  - Design User Interface aspects
  - Select server development platform and markup language:
    - Microsoft.NET, ASP, ColdFusion, Java Servlets, Perl, etc.
    - WML, XHTML-MP, cHTML
  - Write Code behind the interface
  - Test application in Emulators

- Stage 2: Testing
  - Test application scalability by simulating multiple clients
  - Test application on a series of ACTUAL devices – look at range of UAProf devices
- Step 3: Deployment
  - Integrate into existing WAP Portal content on site
  - Configure server for access restrictions
  - Check the server log files to see service popularity

## Capabilities/Preferences

- W3C: CC/PP – Composite Capability/Preference Profile
  - Defines device capabilities and user preferences based on resource description framework
  - General for all devices
- Openmobile Alliance
  - User-Agent Profile (UAProf) – adapts CC/PP to handhelds and defines framework for exchange of data
  - HardwarePlatform (memory, screen size, Bluetooth, etc.)
  - SoftwarePlatform (OS, version, JVM CDLC, etc.)
  - Browser
  - Network Characteristics (UMTS, GPRS, etc, WAP versions, etc.))
  - Push Characteristics
  - Set of default profiles defined – terminal increments from a particular default profile
  - http://www.uaprofile.com/html/

## Capabilities/Preferences

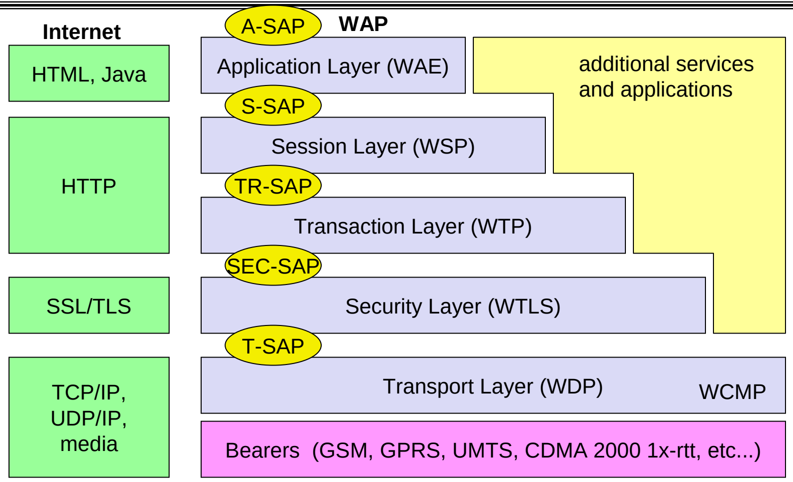
- WURFL: Wireless Universal Resource File
  - XML configuration file  of  subset of UAProf info
  - Open source, contributed descriptions, no guarantee accurate
  - More popular than UAProf
    - can control setup and updating
  - http://wurfl.sourceforge.net/index.php

```
<c:if test="${capabilities.wap_push_support}">
    <a href="subscribepush.jsp">Push Services</a>
</c:if>
```

## Wireless Markup Language WML

- WML Based on OpenWave Handheld Markup Language (HDML) and W3C’s HTML
- Standardized by WAP Forum as part of WAP 1.0
- WAP Now part of open mobile alliance
  - **Open Mobile Alliance** www.openmobilealliance.org
- W3C XML-based language
  - Guarantees well formed document
- WML is part of the Wireless Application Environment of WAP

# WAP 1.x - Reference model and protocols



Internet

HTML, Java

HTTP

SSL/TLS

TCP/IP,
UDP/IP,
media

WAP

A-SAP

Application Layer (WAE)

additional services
and applications

S-SAP

Session Layer (WSP)

TR-SAP

Transaction Layer (WTP)

SEC-SAP

Security Layer (WTLS)

T-SAP

Transport Layer (WDP)

WCMP

Bearers (GSM, GPRS, UMTS, CDMA 2000 1x-rtt, etc...)

WAE comprises WML (Wireless Markup Language), WML Script, WTAI etc.

# WAE - Wireless Application Environment

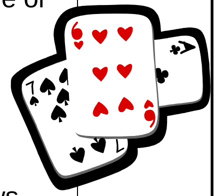
- Goals
  - network independent application environment for wireless mobile devices
  - integrated Internet/WWW programming model with high interoperability
- Requirements
  - device and network independent, international support
  - manufacturers can determine look-and-feel, user interface
  - considerations of slow links, limited memory, low computing power, small display, simple user interface (compared to desktop computers)
- Components
  - Architecture: application model, micro-browser, gateway/proxy, 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, ...
  - Protocal Layers (WAP)

# Wireless Markup Language WML

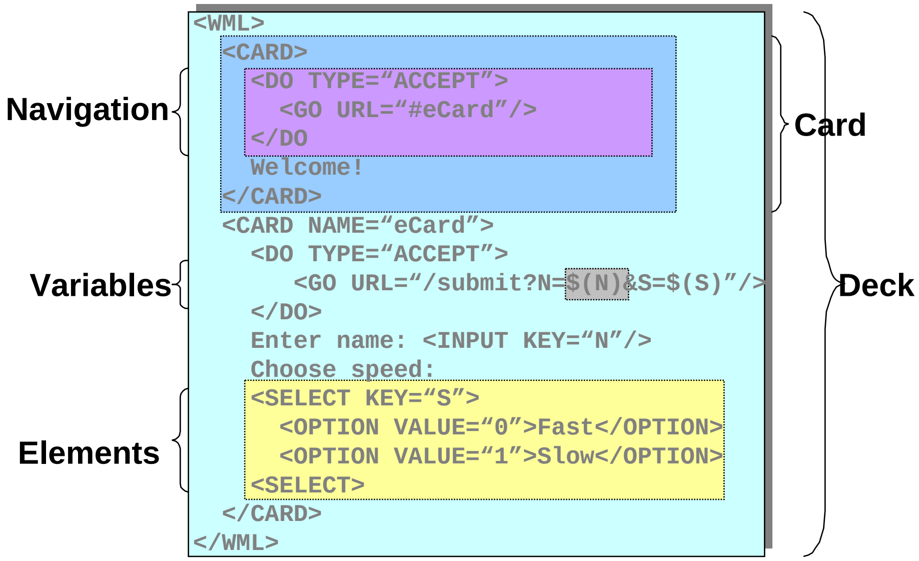
- W3C XML based language – tries to enforce well formed documents
- Tag-based markup language:
  - Screen management (layout, text, images,..)
  - Data input (text, selection lists, etc.)
  - Hyperlinks & navigation support
  - *Presentation depends on device capabilities*
  - Card and Deck approach
- Supports scripting with WMLScript
- Telephone services with WTA

# WML Concepts

- WML follows a Card and Deck metaphor
  - A Deck is a *single WML document* and consists of one or more cards
  - Cards are viewable one at a time and are a single interaction between microbrowser and user
  - User navigation between cards is *local* – a deck allows multiple screens to be downloaded in a single retrieval
  - Movement between decks requires an interaction with a server (fetch deck)
  - Support for navigation among cards and decks – includes provisions for event handling; used for navigation or executing scripts
  - Idea is that since  it takes time to download content should send several pages (i.e., cards ) at once



# A WML Deck Breakdown



```
<WML>
  <CARD>
    <DO TYPE="ACCEPT">
      <GO URL="#eCard"/>
    </DO
    Welcome!
  </CARD>
  <CARD NAME="eCard">
    <DO TYPE="ACCEPT">
       <GO URL="/submit?N=$(N)&S=$(S)"/>
    </DO>
    Enter name: <INPUT KEY="N"/>
    Choose speed:
    <SELECT KEY="S">
      <OPTION VALUE="0">Fast</OPTION>
      <OPTION VALUE="1">Slow</OPTION>
    <SELECT>
  </CARD>
</WML>
```

# WML Concepts (cont.)

- MicroBrowser Related:
  - Special menu options (Options)
  - History of navigation (Back button)
  - Softkeys (special quick action buttons)
  - Bookmarking facilities
  - State management (context) and variables storage facility
  - Caching support for quicker processing
- Card Content:
  - Text rendering and Image layouts
  - Timer and user interaction events
  - Navigation uses hyperlink style URLs

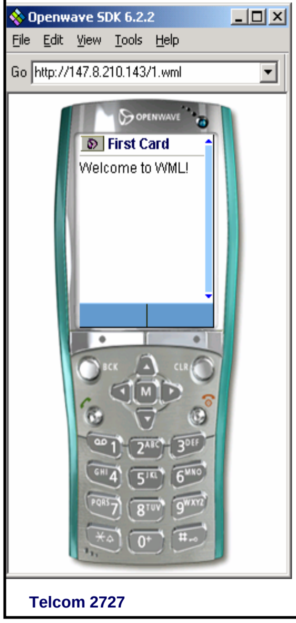
# WML and Deck Format

- **WML Document prologue:**
  - Document type and XML Version
  - Prepares parsing engine to interpret deck according to Document Type Definition (DTD)
  - Markup begins with <wml> tag and concludes with </wml>
  - All elements are bracketed <> and </>

```
<element> value </element>
  may have attributes  (in double quotes)
 <card title="First Card">
          ….
  </card>
```

- Note:
  - WML source must be compiled into binary format by gateway before forwarding to device (phone)
  - Emulators and some PDAs can process WML source without compilation if they have a WML parser

# A Simple WML File



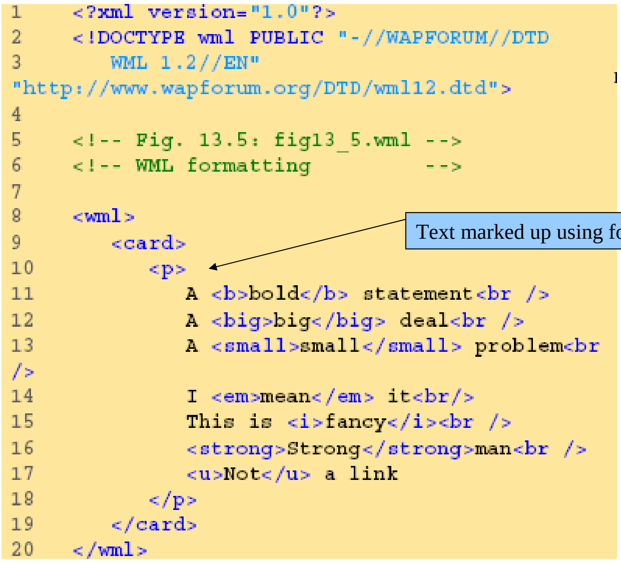
Standard Prolog header for WML 1.1 files

```
<?xml version='1.0'?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card title="First Card">
<p>
Welcome to WML!
</p>
</card>
</wml>
```

# WML Elements

- Predefined WML features for document formatting
- Examples of formatting Elements:
  - Text formatting
    `br p table i b u big small`
  - Variables
    `setvar`
  - User input
    `input select option optgroup fieldset`
  - `Character support`
    &quot;   "   &apos; '  &lt;  <  , etc.
- Comments in source are possible
  <!--     write comment  -->

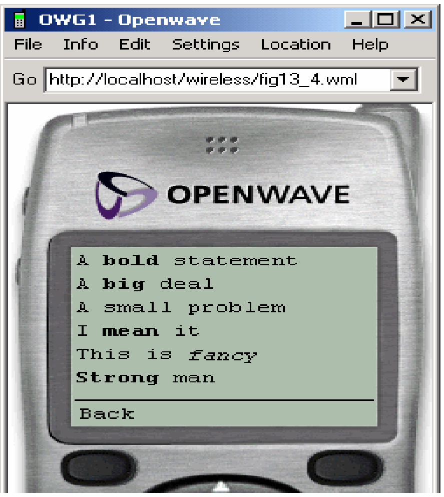
# Text Formatting Example



```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD
   WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">

<!-- Fig. 13.5: fig13_5.wml -->
<!-- WML formatting         -->

<wml>
   <card>
      <p>
         A <b>bold</b> statement<br />
         A <big>big</big> deal<br />
         A <small>small</small> problem<br
/>
         I <em>mean</em> it<br/>
         This is <i>fancy</i><br />
         <strong>Strong</strong>man<br />
         <u>Not</u> a link
      </p>
   </card>
</wml>
```

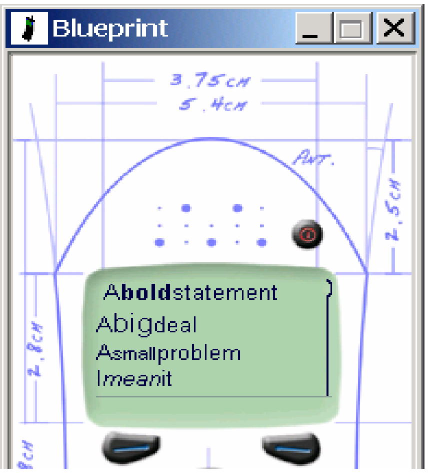
Text marked up using formatting elements.

# Device Dependent Display


OWG1 - Openwave
File Info Edit Settings Location Help
Go http://localhost/wireless/fig13_4.wml
OPENWAVE

A **bold** statement
A **big** deal
A small problem
I **mean** it
This is *fancy*
**Strong** man

Back



A**bold**statement
Abigdeal
Asmallproblem
*Imeanit*

# WML Elements

- Additional WML Elements for document formatting
  - Deck/Card

```
wml – starts/ends deck
card – start/ends a card
head – similar to <HEAD> in html
access –perform access control on a deck
meta – place meta info into a deck (e.g. keywork stuffing)
```

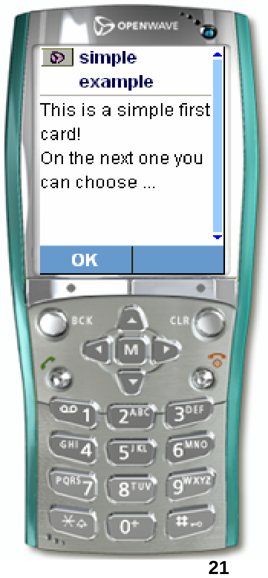
  - Tasks – specify an action to be performed by the browser

```
go – go to card or deck referenced  <go href>
     has get and post options
prev – go to previous card
do  - indicates a control the user can activate
 <do type=“accept”>
     <go href=“card 2”>
     </go>
 </do>
anchor – same as in html
   <a title=“Next”>
       <go href =“page10.wml”/>
        Next Page
    </a>
refresh  noop
```

    - Can move between cards and decks with anchor <a>, go <go href>, prev <pref href>

# WML – example  with cards

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
   1.1//EN"

   "http://www.wapforum.org/DTD/wml_1.1.xml"
   >
<wml>
    <card id="card_one" title="simple
   example">
        <do type="accept">
            <go href="#card_two"/>
        </do>
        <p>
        This is a simple first card!
        <br/>
        On the next one you can choose ...
        </p>
    </card>
```

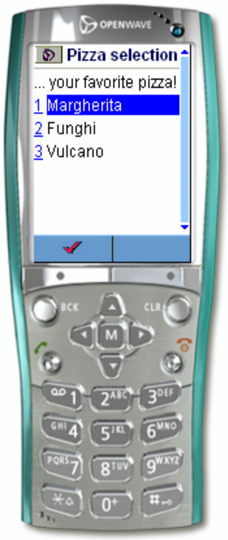

simple example

This is a simple first card!

On the next one you can choose ...

# WML – example with cards

```
<card id="card_two" title="Pizza selection">
        <do type="accept" label="cont">
            <go href="#card_three"/>
        </do>
        <p>
        ... your favorite pizza!
        <select value="Mar" name="PIZZA">
            <option value="Mar">Margherita</option>
            <option value="Fun">Funghi</option>
            <option value="Vul">Vulcano</option>
        </select>
        </p>
    </card>
    <card id="card_three" title="Your Pizza!">
        <p>
        Your personal pizza parameter is
   <b>$(PIZZA)</b>!
        </p>
    </card>
</wml>
```

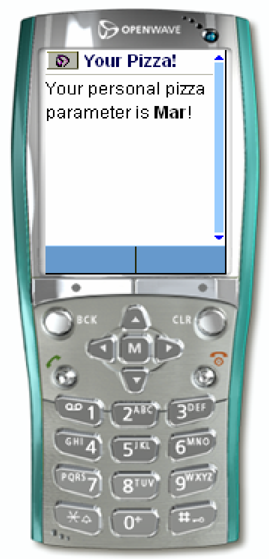

Pizza selection

... your favorite pizza!

1 Margherita
2 Funghi
3 Vulcano

## WML – example with cards

```
<card id="card_three" title="Your Pizza!">
        <p>
        Your personal pizza parameter is
   <b>$(PIZZA)</b>!
        </p>
    </card>
</wml>
```



Your Pizza!

Your personal pizza parameter is **Mar**!

## Advanced WML

- Tables
  - Organize data into rows and columns
- Forms
  - For collecting user input data
- Templates
  - Define common look and feel across cards
  - User interface consistency
- Timers for:
  - Page Refreshes → For example refresh Stock price
  - Animation - display a sequence of cards with a different image per card. Loop this and you have animation

# WML Timer

- <?xml version="1.0"?>
- <!DOCTYPE wml PUBLIC "-//JoeDinner//DTD WML 1.1//EN"
- "http://www.wapforum.org/DTD/wml_1.1.xml">
- <wml>
- <card id="main" ontimer="#card1">
-   <timer value="80"/>
- <p>
- <img id="image27" src="burger.wbmp" alt="JOE'S Diner" hspace="5" vspace="5" />
- <a href="#card1">Enter</a>
- </p>
- </card>
- <card id="card1" title="Joe's Diner">

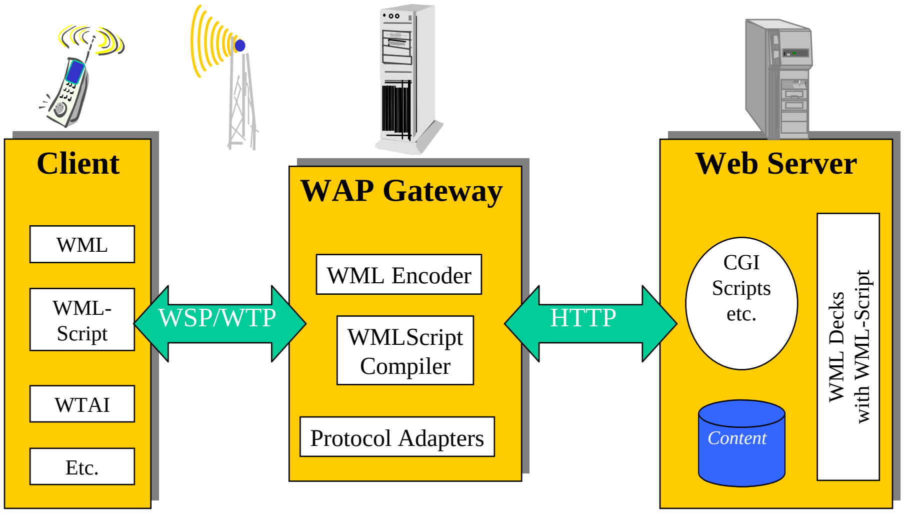
# WML Comments

- Images on monochrome devices in wireless bitmap (WBMP) format.
- For color devices use portable network graphics (PNG) format
- Minimum WAP phone requirements
  - Compressed WML deck must not be larger than 1.4K
  - 4 lines on screen, 12 character per lines
- Rendering WML on some microbrowsers makes navigation difficult
  - Developing WML specific content  for each device may be necessary.
- Cache Problems are common
  - Cached documents do not always expire OR always expire
  - No assumptions can be made about cached documents/images
- Meta tags in header are not always supported
- Some devices do not support POST, only GET operations
- Device capabilities can be established
  - Use UserAgent tag to establish device type and its capabilities (using WURFL or UAProf)

# WMLScript

- Scripting language:
  - Procedural logic, loops, conditionals (if -then -else), etc.
  - String and character processing support
  - Optimized for small-memory, small-cpu devices
- Integrated with WML:
  - Powerful extension mechanism
  - Reduces overall network traffic
  - Mobile Code architecture
- Source Code Compiler in WAP Gateway:
  - Better network bandwidth use
  - Better use of phone memory/cpu

# WAP Architecture



Client

WML

WML-Script

WTAI

Etc.

WSP/WTP

WAP Gateway

WML Encoder

WMLScript Compiler

Protocol Adapters

HTTP

Web Server

CGI Scripts etc.

Content

WML Decks with WML-Script

Source: WAP Forum

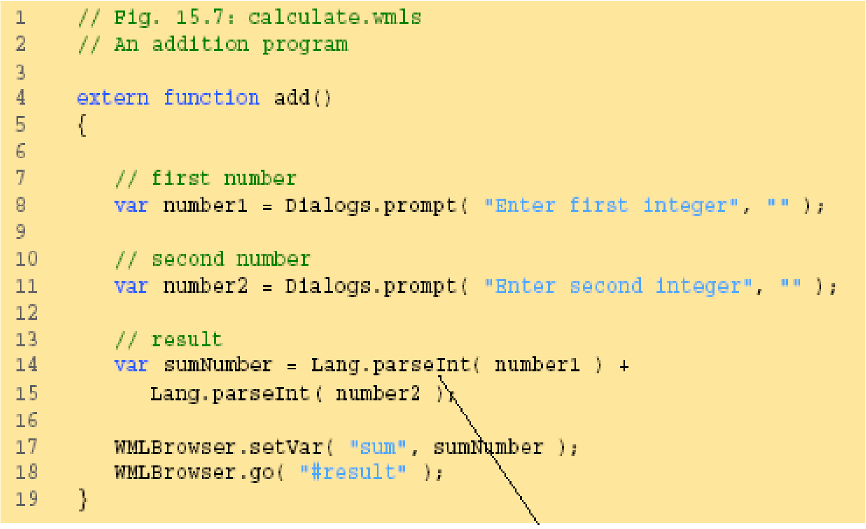
# WMLScript API Libraries

- Available on all WAP compatible devices:
  - Lang - constants, general-purpose math functionality – include logical operations, increment/decrement, etc.
  - String – Character/String processing functions
  - URL - URL processing
  - Browser- WML browser interface
  - Dialog - simple user interface
  - Float - floating point functions
- Other libraries are available as proprietary extensions on device

# Common WMLScript Uses

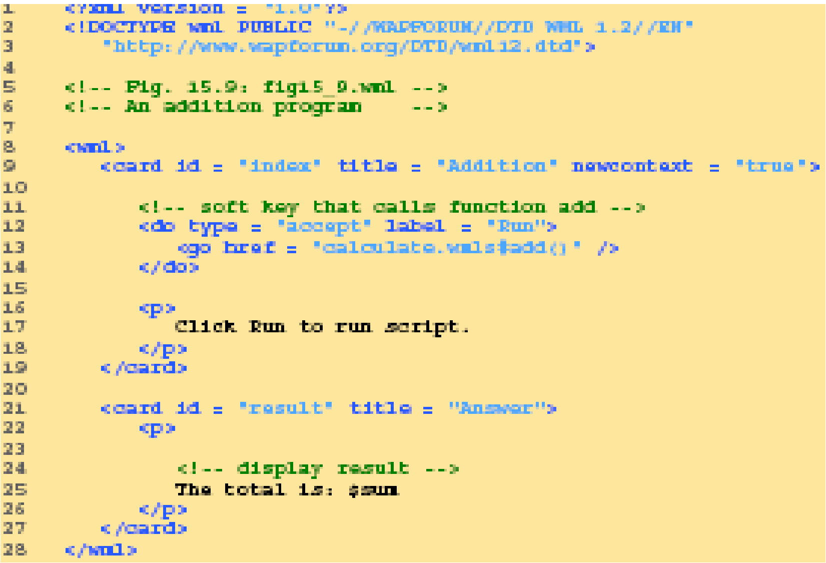
- Reduce network round-trips and enhance functionality
- Field validation
  - Check for formatting, input ranges, for transmitting to server, etc.
- Device extensions
  - Access device or vendor-specific API
    - For example address book for phone numbefr
- Conditional logic
  - Download intelligence into the device as needed
  - For example download new software

# WMLScript Example



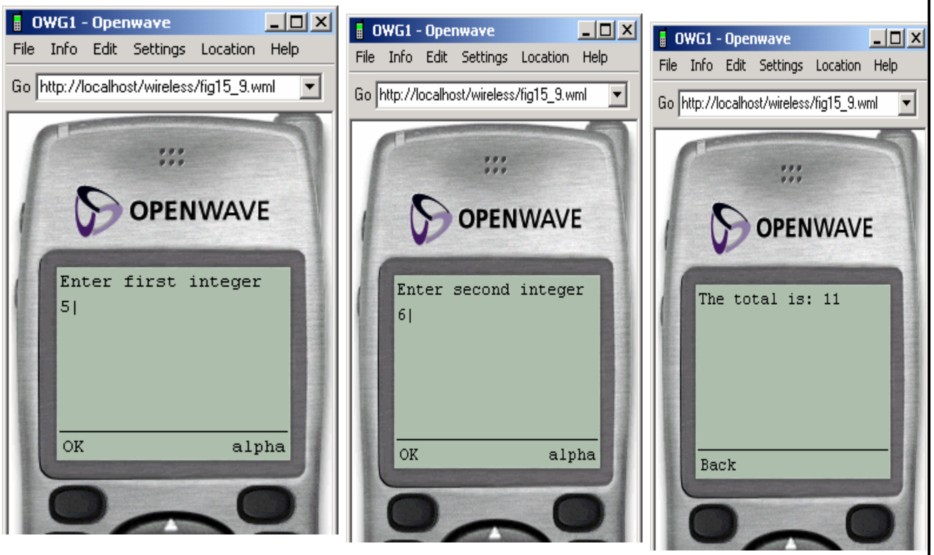
```
// Fig. 15.7: calculate.wmls
// An addition program

extern function add()
{

   // first number
   var number1 = Dialogs.prompt( "Enter first integer", "" );

   // second number
   var number2 = Dialogs.prompt( "Enter second integer", "" );

   // result
   var sumNumber = Lang.parseInt( number1 ) +
      Lang.parseInt( number2 );

   WMLBrowser.setVar( "sum", sumNumber );
   WMLBrowser.go( "#result" );
}
```

# WML Script Example



```
<?xml version = "1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
   "http://www.wapforum.org/DTD/wml12.dtd">

<!-- Fig. 15.9: fig15_9.wml -->
<!-- An addition program    -->

<wml>
   <card id = "index" title = "Addition" newcontext = "true">

      <!-- soft key that calls function add -->
      <do type = "accept" label = "Run">
         <go href = "calculate.wmls#add()" />
      </do>

      <p>
         Click Run to run script.
      </p>
   </card>

   <card id = "result" title = "Answer">
      <p>

         <!-- display result -->
         The total is: $sum
      </p>
   </card>
</wml>
```

# Example



http://localhost/wireless/fig15_9.wml

```
Enter first integer
5|
```

```
Enter second integer
6|
```

```
The total is: 11
```

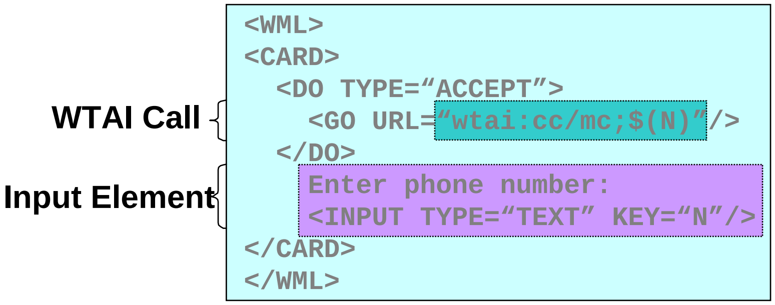
# Wireless Telephony Application (WTA)

- Collection of telephony specific extensions
- Extension of basic WAE application model
  - access to telephony functions
    - any application on the client may access telephony functions (place/answer call, call forwarding, etc.)
  - content push
    - server can push content to the client
  - handling of network events
    - table indicating how to react on certain events from the network
- Example
  - calling a number (WML) `wtai://wp/mc;4126247400`
  - calling a number (WMLScript) `WTAPublic.makeCall(“4126247400");`

# WTA Overview (cont.)

- WTA Browser
  - Extensions added to standard WML/WMLScript browser
  - Exposes additional API (WTAI)
- WTAI includes:
  - Call control
  - Network text messaging
  - Phone book interface
  - Indicator control
  - Event processing
- WTAI access is available from WML & WMLScript.
- Integration of Telephony Application Interface into mobile applications
- Automatic activation of Voice call by user action or WAP site application

# Placing an outgoing call with WTAI: in WML



WTAI Call

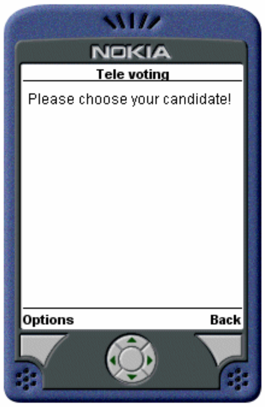
Input Element

```
<WML>
<CARD>
  <DO TYPE=“ACCEPT”>
    <GO URL=“wtai:cc/mc;$(N)”/>
  </DO>
    Enter phone number:
    <INPUT TYPE=“TEXT” KEY=“N”/>
</CARD>
</WML>
```

# WTAI - example with WML

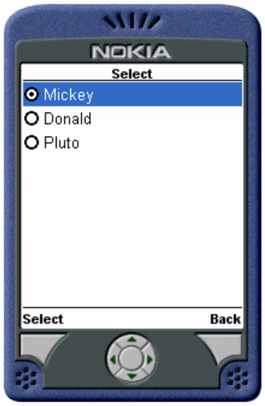
```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
              "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
    <card id="card_one" title="Tele voting">
        <do type="accept">
            <go href="#card_two"/>
        </do>
        <p> Please choose your candidate! </p>
    </card>
    <card id="card_two" title="Your selection">
        <do type="accept">
            <go href="wtai://wp/mc;$dialno"/>
        </do>
        <p> Your selection:
        <select name="dialno">
            <option value="01376685">Mickey</option>
            <option value="01376686">Donald</option>
            <option value="01376687">Pluto</option>
        </select>
        </p>
    </card>
</wml>
```

# WTAI - example with WML



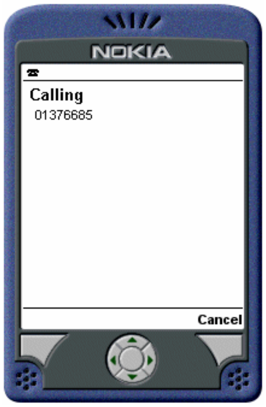
Tele voting

Please choose your candidate!



Select

● Mickey
○ Donald
○ Pluto



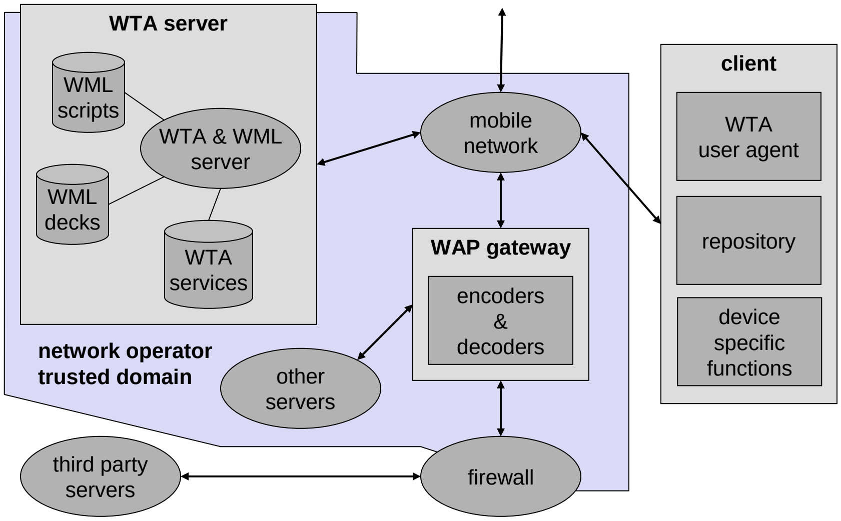
Calling
01376685

# Placing an outgoing call with WTAI: in a WMLScript function

WTAI Call {

```
function checkNumber(N) {
   if (Lang.isInt(N))
      WTAI.makeCall(N);
   else
      Dialog.alert("Bad phone number");
}
```

# WTA logical architecture



## WTAI - example with WML and WMLScript

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
              "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
    <card id="card_one" title="Tele voting">
        <do type="accept"> <go href="#card_two"/> </do>
        <p> Please choose your candidate! </p>
    </card>
    <card id="card_two" title="Your selection">
        <do type="accept">
            <go href="/myscripts#voteCall($dialno)"/> </do>
        <p> Your selection:
        <select name="dialno">
            <option value="01376685">Mickey</option>
            <option value="01376686">Donald</option>
            <option value="01376687">Pluto</option>
        </select> </p>
    </card>
    <card id="showResult" title="Result">
        <p> Status: $Message $No </p>
    </card>
</wml>
```

## WTAI - example with WML and WMLScript I

Script can handle error cases (e.g., phone busy, network unavailable, etc. )

```
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");
}
```

## Summary

- Thin client development
  - Create content vs. adapt content
- Wireless Markup Language (WML)
  - Deck/Card format
  - Text Formating
  - Elements
  - Scripting
  - WTA – not j2me libraries now available to access functions
- Note not as flexible as Smartclient