Thin Client Content Options

- **Content Creation**
  - Make content specifically for wireless handhelds
    - Use wireless markup languages (cHTML, WML, XHTML-MP)
  - **Content Adaptation**
    - Adapt content for desktops to handheld devices on the fly
      - What and how to adapt?
      - Where to adapt?
Content Adaptation

• Content Types delivered to Mobile Users
  – Text
  – Audio
  – Visual
  – Application Data
  – Code (native or machine independent)

• Multimedia Data Unit (MMU)
  • Unit of data transmitted over a network containing one or more content elements (MMS, SMS, web page, code, etc.)

• Issues
  – Terminal diversity, network environment (data rate, power level, etc.), cost, time, user preferences.

Content Adaptation

• What can one Adapt in Content?

• Types of MMU Content Adaptation
  – Format
  – Characteristics
  – Appearance
  – Size
  – Encapsulation
Content Adaptation

• Format
  – Convert MMU format from one standard to another that matches the capabilities of the device
    • Wav to MP3,
    • JPEG to gif or PNG
    • XML to WML
    • Etc.

• Characteristics
  – Modify MMU parameters within a given format
    • For example within MPEG 3 stream modify
      – Number of colors
      – Resolution
      – Frame rate
      – Etc.
Content Adaptation

• Appearance
  - Modify content of MMU to fit device and improve usability
    • Portrait to landscape, remove MMU object, add/remove links/features
    • Break up web page to multiple smaller linked pages
    • Fonts
    • Etc.

Content Adaptation

• Size
  - Adjust content to fit screen
    • Resize image, convert formats for size reasons (JPEG to JPEG 2000)

• Encapsulation
  - Reformat MMU from one application protocol to another
    • Convert email to multiple SMS messages
Architectures/Techniques for Adaptation

• Technique employed for content adaptation tied to location of adaptation and overall architecture
• Options for location of adaptation
  • Server/Source
  • Intermediate Point
    – Portal between server and wireless gateway
    – Gateway to wireless networks (WAP approach)
  • Client device

Architectures for Adaptation

• Architectures for Content Adaptation
  – Multimedia Transcoding (Intermediary or Server)
  – Content Selection (Server)
  – Rendering at the Client (client)
  – Hybrid Techniques (combination)
• The first two and hybrid technique require capability negotiation between device and server.
Multimedia Transcoding

- Modify the properties of MMU in real time to fit the mobile terminals capabilities
  - Capability Negotiation/Preference Profiles of terminals and users
  - Adaptation Policies based on profiles
  - Media Transcoding Adaptation Engine to modify content

Multimedia Transcoding

- Media Adaptation Engine is core of the approach
  - Audio/Visual Content
    - Decode object -> modify uncompressed object -> encode to desired format
    - For example convert compressed streaming MP3 audio to analog then convert to vocoder of terminal
  - Text/Web Document content
    - Store document and presentation separate (e.g., XML, CSS)
    - Convert with style sheet transformation (e.g., XSLT)

- Advantages
  - Automatic process, content developed once

- Drawbacks/Hurdles
  - Complexity, Scalability, Determining capabilities/preferences, Processing Resources, poor model for code
  - Adapted results may not be usable
  - Copyright issues?
Content Selection

• Multiple versions of MMUs stored on server
• Server selects version that best fits terminal capabilities/preference profile
• Infopyramid concept
  – Modality axis (various media modes)
  – Resolution axis (quality vs. bit rate)

Content Selection

• Need Capabilities/Preferences of terminals/users
• Customizer
  – Selects best content representation supported by capabilities/preferences/environment
• Advantages
  – Selection faster than transcoding
  – Fixed quality/usability, suited for code distribution
  – Solves legal issues
• Disadvantages
  – Storage
  – Creation and management of content
Rendering at Client

• Send full binary content to device as one would for desktop machine then adapt to terminal

• Browser/Render handles content adaptation
  – For example show text page if `show images" disabled, web clipping
  – Desktop approach

Rendering at Client

• Advantages
  – Client knows capabilities and has current status
  – Content can be readapted if necessary

• Drawbacks
  – Bandwidth inefficient
  – Requires processing resources on terminal (not suitable for many handhelds)

• Wave of the future?
  – As handhelds get more powerful CPUs – larger memory and better screens
Hybrid Methods?

- Multimedia transcoding
  - works well for automatic adaptation of simple media content
  - difficult when rich content
- Content selection
  - gives more control on adapted versions of rich content
  - Difficult when content changes often
- Rendering
  - requires high end processing capabilities and lots of bandwidth
- Hybrid methods combine adaptation techniques
  - For example Opera Mobile Browser and Proxy Gateway combines transcoding and rendering
  - Project Consensus IBM, Nokia, Fujitsu – combines content selection and transcoding
- Usability and Quality still a big concern

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
### UAProf

```xml
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/"
   xmlns:mms="http://www.wapforum.org/"
   xmlns:prf="http://www.openmobile.org/"

   <rdf:Description rdf:ID="Profile">
     <prf:component>
       <rdf:Description rdf:ID="HardwarePlatform">
         <rdf:type rdf:resource="http://www.openmobi.org/>
         <prf:BitsPerPixel>16</prf:BitsPerPixel>
         <prf:ColorCapable>Yes</prf:ColorCapable>
         <prf:ScreenSize>120x160</prf:ScreenSize>
         <prf:ImageCapable>Yes</prf:ImageCapable>
       </rdf:Description>
     </prf:component>
   </rdf:Description>
</rdf:RDF>
```

### 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

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

• Thus far considered how to develop mobile applications and their architecture (messaging, smart client, thin client)

• Content Provisioning
  – How to deliver the application to the end user?
  – How to make prevent application from being copied.
  – How does one charge for application?

Delivery of Applications

• Can the phone handle the content?
• Did the content reach the terminal
• Where should it be stored?
• Was it properly installed – under what name?
• Can the user be redirected to a new page for confirmation/billing/etc
• Can the user be prevented from sharing application with friends?
• OpenMobile Alliance has developed
  – Download Over the Air architecture/standard
  – Digital Rights Management architecture/standard
Download Over the Air (OTA)

- Download OTA Components shown below
- Core of Download OTA is download descriptor document
  - separate from the content/application being downloaded

Download Descriptor

- Download Descriptor - document that contains
  - Information about the application
    - Size, type, name, vendor, description, etc...
  - Where to find content/application
    - ObjectURI
  - Status Report
    - Download NotifyURI
    - InstallNotify URI
  - Additional items
    - License, delete, next URL to go to after installation, etc..
Download Descriptor

- XML format for DD (.dd file)

```xml
<?xml version="1.0" ?>

<media xmlns="http://www.openmobilealliance.org/xmlns/dd">
  <description>KTH logo</description>
  <name>KTH</name>
  <type>image/jpeg</type>
  <objectURI>
    http://www.iot.kth.se/courses/2g1722/kth.jpg
  </objectURI>
  <size>4980</size>
</media>
```

Download OTA

- After getting the download descriptor file (.dd)
- The browser passes it on to the download agent
- Download agent checks if device had the required capabilities to run the application/content
- The user is prompted for download confirmation (optional if trusted server)
Download

• Content is downloaded using HTTP, WSP or other protocol as specified in the URI
• Time
  – The download descriptor can specify at what time the download should take place
• Update
  – Content can replace previous downloaded content
  – Version handling
• Pause, Resume, Chunks (V2.0)

Download OTA

• Installation
  – The dd contains info on where to store content/application

• Completion
  – When finished or if an error the can send status report or a complete notification to the status server

POST status.php HTTP/1.1
Host: status.foo.com
Content-Length: 23

981 Download Completion
Additional Notes

- Can co-deliver .dd and content/application using WSP or W-HTTP (WAP 1.x, 2.0)
  - Use multi-part encoding
  - Downloads descriptor and content in one get
  - Quicker download
  - No user confirmation – may waste download on device that can’t support application

Digital Rights Management

- How can one control access to content?
- OpenMobile Alliance (OMA) has a DRM working group for mobile applications with proposed standard
  - centralized approach
- Other methods have been proposed (e.g., peer-to-peer)
- Basic OMA approach is to handle content separate from access rights
  - DRM Content Format (DCF)
    - Content encrypted using shared secret
  - Rights Object, RO
    - Specifications of rights and key to decrypt content
  - DCF is not locked to device, you need a RO to use it.
DRM Architecture

Content Issuer

Rights Issuer

DCF

RO

DCF

DRM and Download OTA

• If download descriptor has a license element then the Download Agent must contact the license server (Rights Issuer) or a license agent) to get Rights Object (license) for the content

• RO can be accessed separately as a result of following a next URL

• The RO grants Permission to access content/application on a device or group of devices
  – RO can be limited in time, number of sessions, etc
  – RO should not be copyable
Billing

- The network service provider can easily charge for content and add to the user’s monthly bill
- If not an operator – options
  - Collaborate with operator and provide content and use their billing infrastructure
    - Content provider get ~50-80% of charge
    - Content is accessed through service providers WAP gateway
  - Build your own billing relationship with user
  - Have third party bill user – Mobile Portal/Store

Summary

- Content Adaptation
  - Multimedia Transcoding
  - Content Selection
  - Client Rendering
  - No one approach dominant
- Content Provisioning
  - Download OTA
  - DRM
  - Billing