CS 480: MOBILE NETWORKS

Lecture PowerPoints

Lecture 13

Support for mobility

Topics Covered

- Introduction
- File systems
- World wide web (WWW)
- Wireless application protocol (WAP)
- i-mode
- WAP 2.0

Introduction

Introduction

- To use well-known applications from fixed networks, some additional components are needed in a mobile and wireless communication system
  - Examples are file systems, databases, security, accounting and billing mechanisms

File systems

File systems (1)

- The general goal of a file system is to support efficient, transparent and consistent access to files,
  - no matter where the client requesting files or the server(s) offering files are located
- *Efficiency* is of special importance for wireless systems
  - since bandwidth is low and thus updating operations and others should be kept at a minimum
- *Transparency* addresses the problems of location-dependent views on the file system.
- To support mobility, the file system should provide identical views on directories, file names, access rights etc., independent of current location

File systems (2)

- General problems are limited resources on portable devices and the low bandwidth of wireless access.
- File systems cannot rely on large caches in the end system or perform many updates via a wireless link.
  - Portable devices may be disconnected for a longer period.
- Hardware and software components of portable devices often do not follow standard computer architectures or operating systems.
  - Mobile phones, PDAs have their own operating system, hardware and application software.
- Portable devices are not as reliable as desktop systems or traditional file servers.
- Standard file systems like network file system (NFS) are inefficient and almost unusable in a mobile environment.

File systems (3)

  - Traditional file systems do not expect disconnection, low bandwidth connections, and high latencies
- To support disconnected operation, the portable device may replicate files or single objects.
  - This can be done in advance by prefetching or while fetching data (caching).
- The main problem is consistency of the copy with the original data

World wide web (WWW)

Hypertext transfer protocol (HTTP) (1)

- Web applications encounter problems when used in a mobile and wireless environment.
- HTTP is a stateless, lightweight, application level protocol for data transfer between servers and clients
- HTTP/1.1 is the standard currently used by most implementations.
- An HTTP *transaction* consists of an HTTP *request* issued by a client and an HTTP *response* from a server.
- Stateless means that all HTTP transactions are independent of each other.
  - HTTP does not "remember" any transaction, request, or response.
- HTTP (e.g., version 1.1) causes many problems already in fixed networks but even more in wireless networks

Hypertext transfer protocol (HTTP) (2)

- *Bandwidth and delay*:
  - HTTP has been designed for large bandwidth and low delay.
  - HTTP protocol headers are quite large and redundant.
  - Servers transfer uncompressed content (GIF or JPEG images)
  - As TCP connections are used for each item on a web page, a huge overhead comes with each item
  - The TCP slow-start mechanism can cause additional problems.
  - DNS lookup by client causes additional traffic.
- *Caching*:
  - Although useful in many cases, caching is quite often disabled by content providers to be able to create user profiles, usage statistics etc.
  - With a cache between a server and a client, content providers cannot get realistic feedback

Hypertext transfer protocol (HTTP) (3)

  - Either caches need additional mechanisms to create usage profiles or caching is disabled.
  - Users suffer by downloading the same content repeatedly from the server
  - Many present day pages contain dynamic objects e.g., access counters, time, date that cannot be cached
  - Mobility quite often inhibits caching because the ways of accessing web servers change over time due to changing access points.
  - Many security mechanisms also inhibit caching.
- *POSTing*:
  - Sending content from a client to a server can cause additional problems if the client is currently disconnected
  - The POST request cannot be fulfilled in a disconnected state.

Hypertext markup language (HTML) (1)

- HTML is broadly used to describe the content of web pages in the world wide web.
- HTML was designed for standard desktop computers connected to the internet with a fixed wire.
- These computers share common properties, such as
  - a relatively high performance (compared to handheld devices),
  - a color high-resolution display (24 bit true color, 1,200 × 1,024 pixels is a standard),
  - mouse, sound system, and large hard disks.
- Handheld devices offer
  - only small, low-resolution (320 × 240) displays,
  - very limited input interfaces (small touch-pads, soft-keyboards)
  - and low performance CPUs due to restrictions in power consumption and form factor (still be handheld)

Hypertext markup language (HTML) (2)

- Almost all of today’s web pages are “enriched” with special “features”.
  - Such as animated GIFs, java applets, frames, activeX controls.
- Some can be interpreted directly by the client’s browser, some need a special plug-in.
- Such additional content formats cause several problems
  - Appropriate plug-ins are not often available for handheld devices, each with its own operating system.
- Without additional mechanisms and a more integrated approach,
  - large high resolution pictures would be transferred to a mobile phone with a low resolution display causing high costs.
- Webpages typically ignore the heterogeneity of end systems altogether.

Some approaches that might help wireless access (1)

- *Image scaling*:
  - If a page contains a true color, high-resolution picture, this can be scaled down to fewer colors, lower resolution
- *Content transformation*:
  - Many documents are only available in certain formats e.g., pdf.
  - Before transmitting such documents to a client without appropriate reader, a special converter could translate this document into plain text
- *Content extraction/semantic compression*:
  - Besides transforming the content, e.g., headlines or keywords could be extracted from a document and presented to a user.
  - The user could decide to download more information relating to certain headline or keyword.
  - This semantic compression is quite difficult for arbitrary text

Some approaches that might help wireless access (2)

- *Special languages and protocols*:
  - Other approaches try to replace HTML and HTTP with other languages and protocols better adapted for wireless environment.
- *Push technologies*:
  - Instead of pulling content from a server, the server could also push content from a client.
  - This avoids the overhead of setting up connections for each item, but is only useful for some content e.g., news.

Improvements offered by HTTP version 1.1

- *Connection reuse*: clients and servers can use the same TCP connection for several requests and responses
- *Caching enhancements*: a cache may now also store cacheable response to reduce response time and bandwidth for future, equivalent responses.
- *Bandwidth optimization*: HTTP/1.1 supports not only compression, but also negotiation of compression parameters and different compression styles.
  - It allows for partial transmission of objects. For example, first the initial part of an image is read to determine its geometry.
  - Partial transmissions can also be used to recover from network failure
- *Security*: HTTP/1.1 comprises further mechanisms to check message integrity and to authenticate clients, proxies and servers

System architecture (1)

- The classic underlying system architecture of the www is a client server system.
  - The client, a web browser running as an application on a computer, requests content from a server, the web server running on another computer
- Without any enhancements, each click on a hyperlink initiates transfer of the content that the link points to.
- The browser uses the HTTP protocol for content transfer
- Webpages are described using HTML and other formats
- Caching is a major topic in a web client/server scenario.
  - While caching is also useful for wired computers because it reduces the delay of displaying previously accessed pages, it is the only way of supporting (partially) disconnected web browsers

System architecture (2)

- In mobile wireless clients, network connections can be disrupted or quite often be of bad quality
- The first enhancement was the integration of caching into web browsers.
  - This is standard for all of today's browsers
- Figure 1 shows a mobile client with a web browser running.
  - This browser has an integrated caching mechanism as enhancement
  - This cache does not perform automatic pre-fetching of pages but stores already transferred content up to a certain limit.
- A user can then go "offline" and still browse through the cached content (pages, pictures, etc)

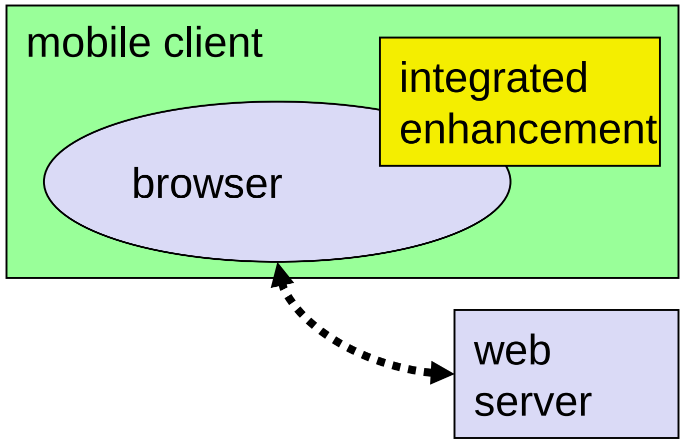

Figure 1: Integrated browser enhancement

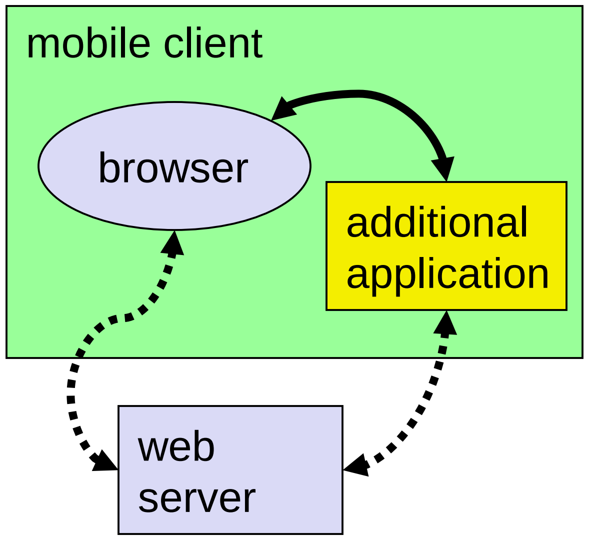

Figure 2: Additional application supporting browsing

System architecture (3)

- Figure 2 shows an architecture for an early approach to enhance web access for mobile clients
- The initial WebWhacker, for example, is a companion application for the browser that supports pre-fetching of content, caching and disconnected service.
  - This approach is not transparent for a browser as there are now two different ways of accessing content (one directly to the web server, one via the additional application)
- The typical enhancements for web browsing act as a transparent proxy as shown in Figure 3.
  - The browser accesses the web server through the client proxy, i.e., the proxy acts as server for the browser and as client for the web server
- The proxy can now pre-fetch and cache content according to many strategies.

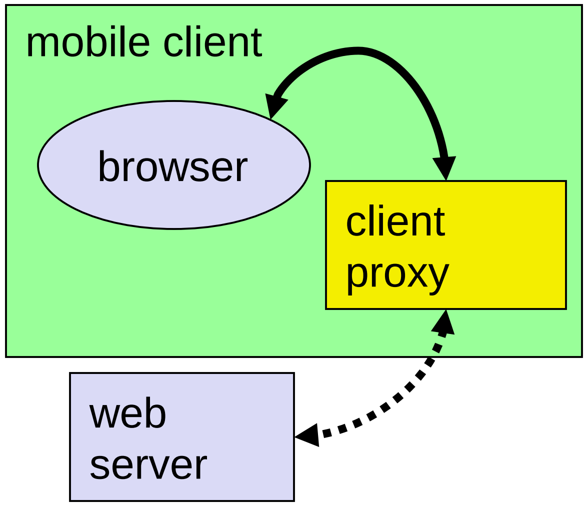

Figure 3: Client proxy as browser support

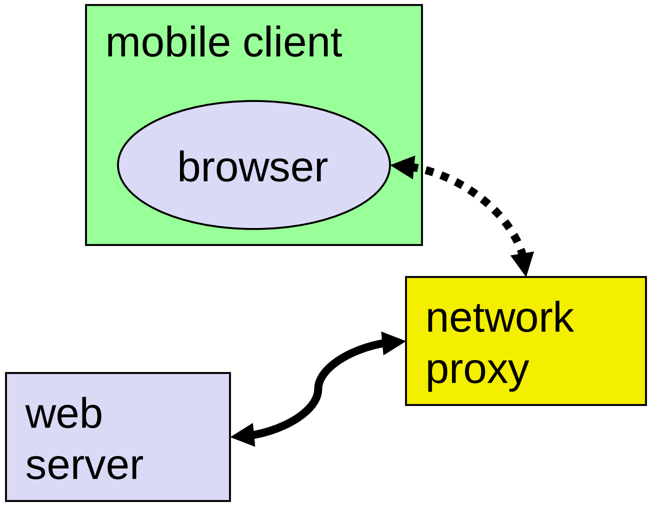

Figure 4: Network proxy as browser support

System architecture (4)

- As soon as the client is disconnected, the proxy serves the content
  - Example strategies for pre-fetching could be:
    - all pages the current pages point to
    - all pages including those the pre-fetched pages point to
    - pages but no pictures
    - all pages with the same keyword on the same server etc.
- A proxy can also support a mobile client on the network side (see Figure 4)
  - This network proxy can perform adaptive content transformation (e.g., semantic compression, headline extraction, etc) or pre-fetch and cache content
- Pre-fetching and caching is useful in a wireless environment with higher error probability.

System architecture (5)

- Similar to these enhancements, for example, as I-TCP achieves, splitting web access into a mobile and fixed part can improve overall system performance
  - For the web server, the network proxy acts like any fixed browser with wired access.
  - Disconnection of the mobile client does not influence the web server. Examples of this approach include TranSend and Digestor
- The benefits of client and network proxies can be combined which results in a system architecture as illustrated in Figure 5
  - An example of this is WebExpress.
- Client proxy and network proxy can now interact better in pre-fetching and caching of data.

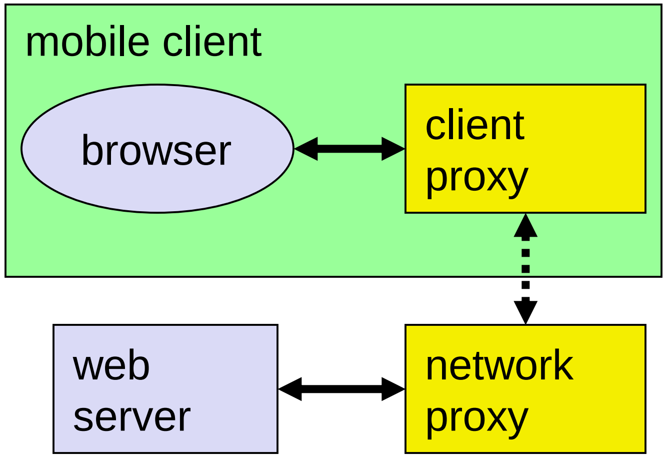

Figure 5: Client and network proxy as browser support

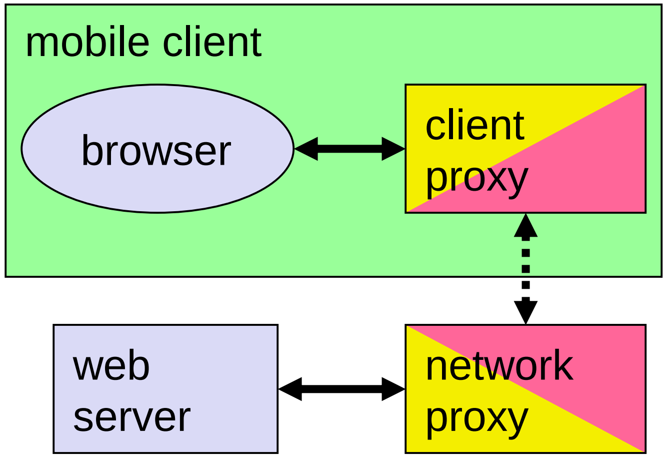

Figure 6: Client and network proxy with special transmission protocol

System architecture (6)

- The client proxy could inform, for example, the network proxy about user behavior, the network proxy can then pre-fetch pages according to this information.
- The whole approach is still transparent to the web server and the client browser
- One can even go one step further and implement a specialized network system as shown in Figure 6
- This solution has the same benefits as the previous one but now, content transfer can be further optimized.
  - Examples are on line compression and replacement of transfer protocols such as HTTP and TCP, with protocols better adapted to the mobility and wireless access of the client
- One example of such a system is Mowgli.

System architecture (7)

- This system supports web access over cellular telephone networks (with low bandwidth and relatively high delay)
  - The system not only replaces transport protocols but also performs additional content transformation needed for mobile phones
  - The browser still uses HTTP to the client proxy. The client proxy then uses a specialized transport service, the Mowgli data channel service, to the network proxy.
  - Standard protocols are used to the web servers. Client and network proxy exchange their messages over long-lived Mowgli connections. This avoids TCP's slow start and the one TCP connection per HTTP request behavior of HTTP/1.0.
- Many other enhancements are possible.
  - Examples are server extensions to provide content especially suited for wireless access and mobile, handheld clients

Wireless application protocol (WAP)

Wireless application protocol (WAP) (1)

- The growth of the internet, internet applications and mobile communications led to many early proprietary solutions providing internet services for mobile, wireless devices
- Some of the problems these partial solutions face were discussed since the world wide web is the most important and fastest growing internet application
- To avoid many islands of incompatible solutions, e.g., special solutions for GSM, IS-136, or certain manufacturers, the *wireless application protocol* forum (*WAP* Forum) was founded in June 1997 by Ericson, Motorola, Nokia and Unwired Planet (now Openwave)

Wireless application protocol (WAP) (2)

- In summer 2002, the WAP forum together with open mobile architecture forum and the SyncML initiative formed the open mobile alliance (OMA)
- OMA cooperates with many other standardization bodies, such as ETSI, IETF, 3GPP (Third Generation Partnership Project)
- The basic objectives of the WAP Forum and now the OMA are to bring diverse internet content (e.g., web pages etc) and other data services (e.g., stock quotes) to digital cellular phones and other wireless, mobile terminals (e.g., PDAs, laptops)

Wireless application protocol (WAP) (3)

- A protocol suite should enable global wireless communication across different wireless network technologies, e.g., GSM, CDPD (Cellular Digital Packet Data), UMTS etc.
- The forum is embracing and extending existing standards and technologies for the internet wherever possible
  - and is creating a framework for the development of contents and applications that scale across a very wide range of wireless bearer networks and wireless device types

Wireless application protocol (WAP) (4)

- All solutions must be:
  - *Interoperable*, i.e., allowing terminals and software from different vendors to communicate with networks from different providers
  - *Scalable*, i.e., protocols and services should scale with customer needs and number of customers
  - *Efficient*, i.e., provision of QoS suited to the characteristics of the wireless and mobile networks
  - *Reliable*, i.e., provision of a consistent and predictable platform for deploying services
  - *Secure*, i.e., preservation of the integrity of user data, protection of devices and services from security problems

Architecture (1)

- Figure 7 gives an overview of the WAP architecture, its protocols and components,
  - and compares this architecture with typical internet architecture when using the world wide web
- The basis for transmission of data is formed by different *bearer services*.
- WAP does not specify bearer services, but uses existing data services and will integrate further services
  - Examples are message services (SMS) of GSM, circuit-switched data, such as high-speed circuit switched data (HSCSD) in GSM, or packet switched data, such as GPRS in GSM. Many other bearers are supported, such as CDPD, IS-136, PHS

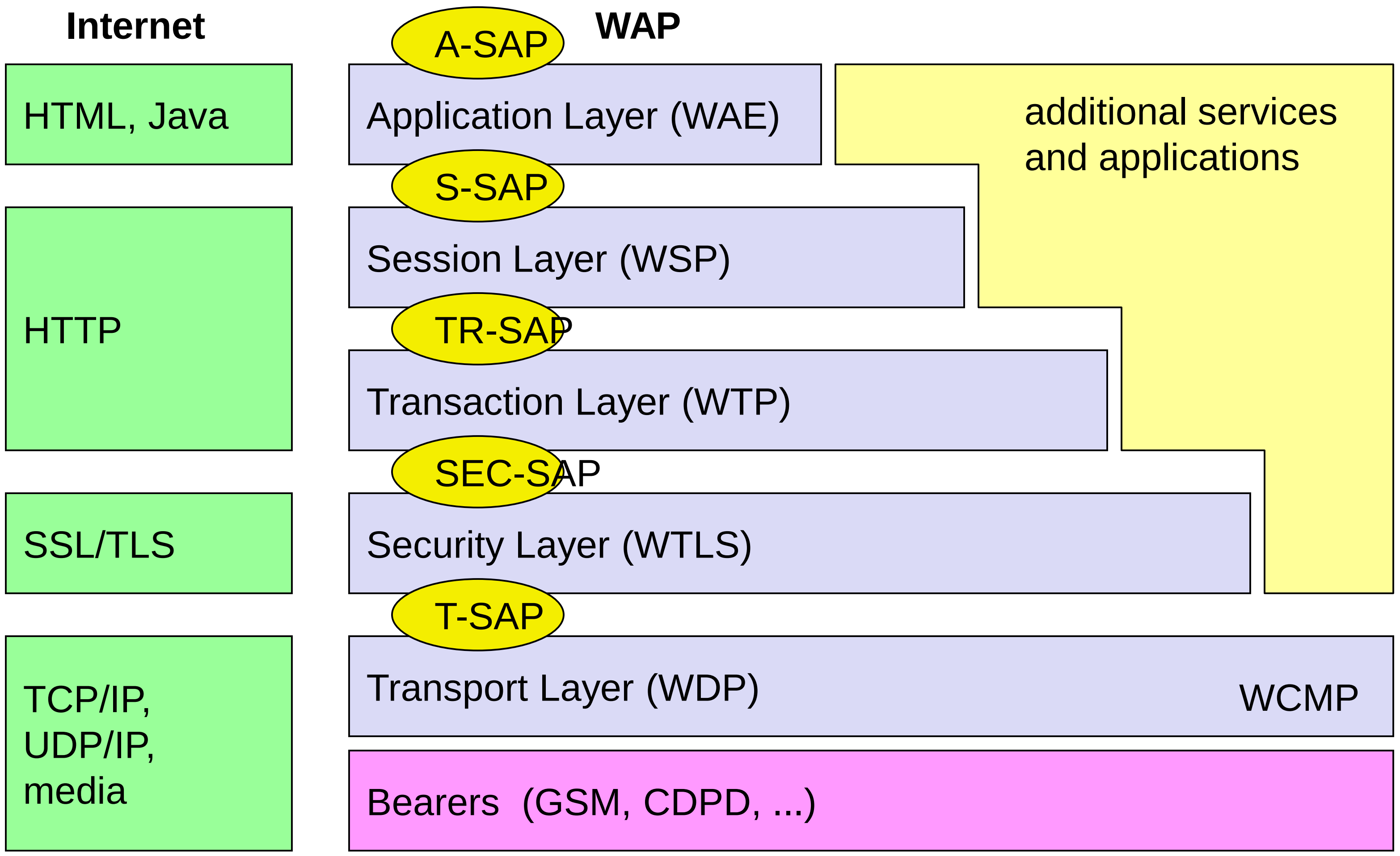

Figure 7: Components and interface of the WAP 1.x architecture

Architecture (2)

- No special interface has been specified between the bearer service and the next higher layer, the *transport layer* with *wireless datagram protocol* (WDP) and the additional *wireless control message protocol* (WCMP)
  - because the adaptation of these protocols are bearer-specific
- The transport layer offers a bearer independent, consistent datagram-oriented service to the higher layers of the WAP architecture
- Communication is done transparently over one of the available bearer services.
- The *transport layer service access point* (TSAP) is the common interface to be used by higher layers independent of the underlying network

Architecture (3)

- The next higher layer, the *security layer* with *its wireless transport layer security* protocol WTLS offers its service at the *security SAP* (SEC-SAP).
- WTLS is based on the transport layer security (TLS, formerly SSL, secure sockets layer).
  - WTLS has been optimized for use in wireless networks with narrow-band channels. It can offer data integrity, privacy, authentication, and (some) denial-of-service protection
- The WAP *transaction layer* with its *wireless transaction protocol* (WTP) offers a lightweight transaction service at the transaction SAP (TR-SAP).
  - This service efficiently provides reliable or unreliable requests and asynchronous transactions

Architecture (4)

- The *session layer* with the *wireless session protocol* (WSP) currently offers two services at the *session-SAP* (S-SAP), one connection-oriented and one connectionless if used directly on top of WDP
  - A special service for browsing the web (WSP/B) has been defined that offers HTTP/1.1 functionality, long-lived session state, session suspend and resume, session migration and other features needed for wireless mobile access to the web
- The *application layer* with the *wireless application environment* (WAE) offers a framework for integration of different www and mobile telephony applications.
  - It offers many protocols with special service access points
  - The main issues here are scripting languages, special markup languages, interfaces to telephony applications and many content formats adapted to the special requirements of small handheld wireless devices

Architecture (5)

- The WAP transport layer together with the bearers can be (roughly) compared to the services offered by TCP or UDP over IP and different media in the internet
  - If a bearer in the WAP architecture already offers IP services (e.g., GPRS, CDPD) then UDP is used as WDP
- The TLS/SSL layer of the internet has also been adopted for the WAP architecture with some optimization
- The functionality of the session and transaction layer can be compared with a role of HTTP in web architecture.
- The application layer offers similar features as HTML and Java
- WAP does not always force all applications to use the whole protocol architecture.

Architecture (6)

- Applications can use only a part of the architecture shown in Figure 7
  - If an application does not require security but needs the reliable transport of data, it can directly use a service of the transport layer.
- Different scenarios are possible for the integration of WAP components into existing wireless and fixed networks (see Figure 8)
- On the left side, different fixed networks, are shown
  - such as the traditional internet and public switched telephone network (PSTN).
- One cannot change protocols and services of these existing networks.

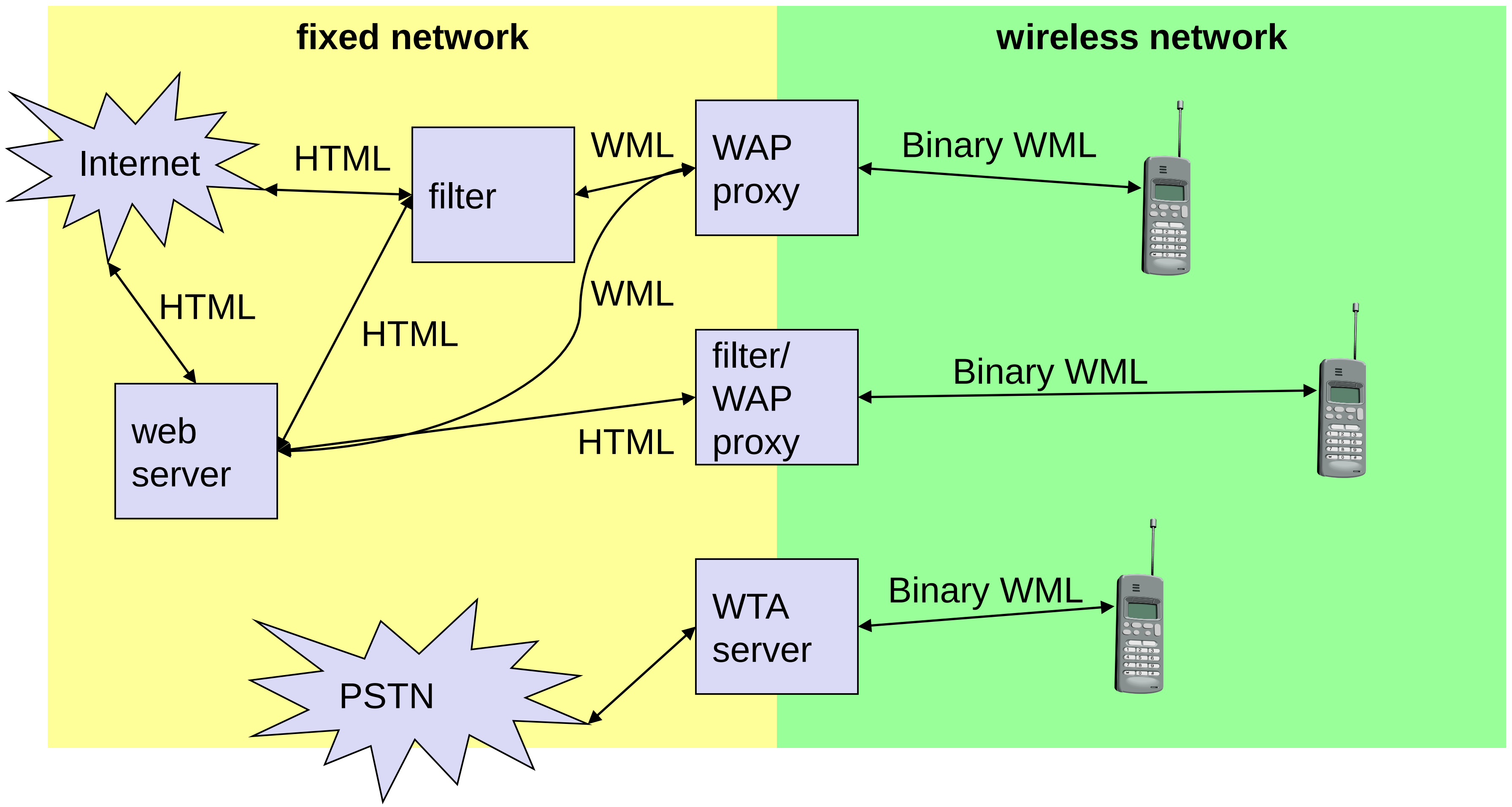

Figure 8: Examples of integration of WAP components

Architecture (8)

- New elements will be implemented between these networks and the WAP-enabled wireless, mobile devices in a wireless network on the right hand side
- The current www in the internet offers web pages with the help of HTML and web servers.
- To be able to browse the pages with handheld devices,
  - a wireless markup language (WML) has been defined in WAP
- Special filters within the fixed network can now translate HTML into WML
- Web servers can already provide pages in WML,
  - or gateways between the fixed and wireless network can translate HTML into WML.

Architecture (9)

- These gateways not only filter pages but also act as proxies for web access.
- WML is additionally converted into binary WML for more efficient transmission
- A special gateway can be implemented to access traditional telephony services via binary WML.
  - This wireless telephony application (WTA) server translates, e.g., signaling of the telephone network (incoming calls etc.) into WML events displayed at the handheld device

Wireless markup language (WML) (1)

- Based on the standard HTML and on HDML
- WML is specified as an XML document type
- WML follows deck and card metaphor
  - WML document consists of cards, cards are grouped into decks
- Figure 9 gives an example of WML.
  - First a reference to XML is given where WML was derived from
  - After the keyword wml, the first card is defined. This first card of the deck “displays” a text after loading
  - As soon as the user activates the do element, the user agent displays the second card
  - On a second card, a user can select one of three pizza options
  - Depending on the choice of the user, PIZZA can have one of the values Mar, Fun, or Vul
  - If the user proceeds to the third card without choosing a pizza, the value Mar is used as default

```
<?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>
```

Figure 9: Example of WML (1)

```
<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>
```

Figure 9: Example of WML (2)

Wireless markup language (WML) (2)

  - WML only describes the intention of a choice
  - The third card finally outputs the value of PIZZA
- WML may be encoded using a compact binary representation to save bandwidth on a wireless link
- The compact representation is based on the binary XML content format as specified in WAP Forum
- The compact format allows for transmission without loss of functionality or of semantic information For example,
  - The URL prefix *href=_http://* will be coded as 4B
  - The code for the *select* key is 37 and *option* is 35
- These single byte codes are much more efficient than the plain ASCII text used in HTML and www

WMLScript (1)

- Complement to WML
- Based on JavaScript, but adapted to the wireless environment
- Provides general scripting capabilities

Features

- validity check of user input
  - check input before sent to server
- access to device facilities
  - access hardware and software (phone call, address book etc.) of the device
- local user interaction
  - direct and local interaction with a user without round-trip delay. Only for example, the result of several interactions could be transmitted to a server

WMLScript (2)

- extensions to the device software
  - configure device, download new functionality after deployment.
  - Users can download new software from vendors and thus upgrade their device easily.
- Figure 10 shows an example of WMLScript
  - The function *pizza_test* accepts one value as input
  - The local variable *taste* is initialized to the string “*unknown*”.
  - Then the script checks if the input parameter *pizza_type* is “*Vul*”. If this is the case, taste is set to “*quite hot*”
  - Finally, the current value of taste is returned as the value of the function *pizza_test*

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

Figure 10: Example of WMLScript

i-mode

i-mode (1)

- The i-mode service was introduced in Japan by the mobile network provider NTT DoCoMo in 1999
- While other network providers in Japan use WAP,
  - NT DoCoMo use its own system which is roughly based on web protocols and content formats from the www
- Example services offered by i-mode are email, short messages, web access and picture exchange
  - Became a big success with more than 30 million users only three years after its introduction.
- In comparison to i-mode, WAP was cited as a failure
- Figure 11 shows the i-mode protocol stack as used in Japan

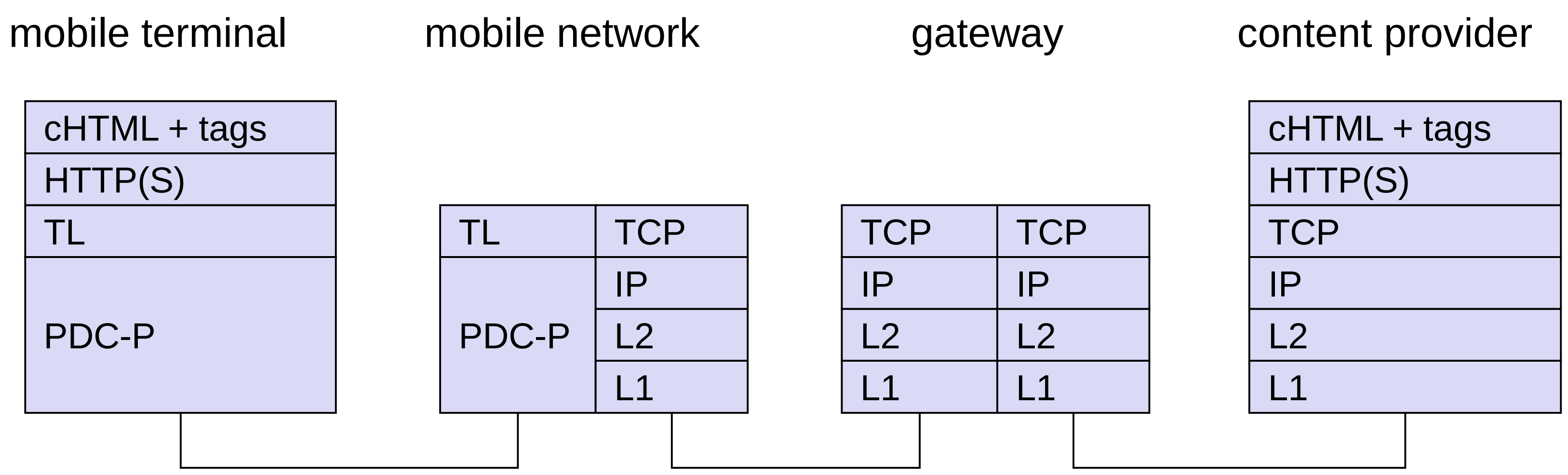

Figure 11: i-mode protocol stacks for Japan

i-mode (2)

- The packet-oriented PDC-P (Packet Data Convergence Protocol) provides bearer service between the mobile terminal and operators’s network
  - Typical data rates are 9.6 kbit/s, while enhanced versions offer 28.8 kbit/s
- On top of the bearer service, i-mode uses a special connection-oriented transport layer protocol
- Within the operator’s network and between the operator’s gateway and a content provider,
  - i-mode uses standard internet protocols (TCP/IP over different layer 1 and layer 2 protocols)
- On top of the transport services, i-mode uses HTTP

i-mode (3)

- i-mode applications can use an email service or display pages described in *compact HTML* (cHTML),
  - which is a subset of HTML, plus some proprietary extra *tags* for emoticons, telephony and email
- Compact HTML supports animated color pictures (GIF format), but no frames, image maps or style sheets.
  - The supported subset of HTML is large enough to display many web pages from the internet.
  - This is a big advantage compared to WAP 1.x, which requires pages written in WML

WAP 2.0

WAP 2.0 (1)

- In July 2001, version two of the wireless application protocol (WAP 2.0) was published
- WAP 2.0 is roughly the sum of WAP 1.x, i-mode, internet protocols, and many mobility specific enhancements
- WAP 2.0 continues to support WAP 1.x protocols,
  - but additionally integrates IP, TCP (with wireless profile), TLS, and HTTP (wireless profiled)
- WAP 2.0 browsers support WML as well as XHTML with a mobile profile (XHTMLMP)
- XHTML is the extensible hypertext mark-up language developed by the W3C to replace and enhance HTML
- WAP 2.0 uses the *composite capabilities/preference profiles* (CC/PP) framework for describing user preferences and device capabilities

WAP 2.0 (2)

- Figure 12 gives an overview of the WAP 2.0 architecture as specified in WAP Forum
- The *protocol framework* consists of the following four components:
  - *Bearer networks*:
    - Similar to WAP 1.x, many bearers are supported. Typical bearers today are GPRS in GSM networks, SMS for push services. Third generation networks will directly offer IP services
  - *Transport services*:
    - These services offer end-to-end abstraction on top of different bearers
    - Transport services can be either connection-oriented or connectionless.
    - For reliable, connection-oriented services TCP with a wireless profile can be used. WDP or UDP (in case of an IP bearer) can be used for unreliable, connectionless (datagram) services

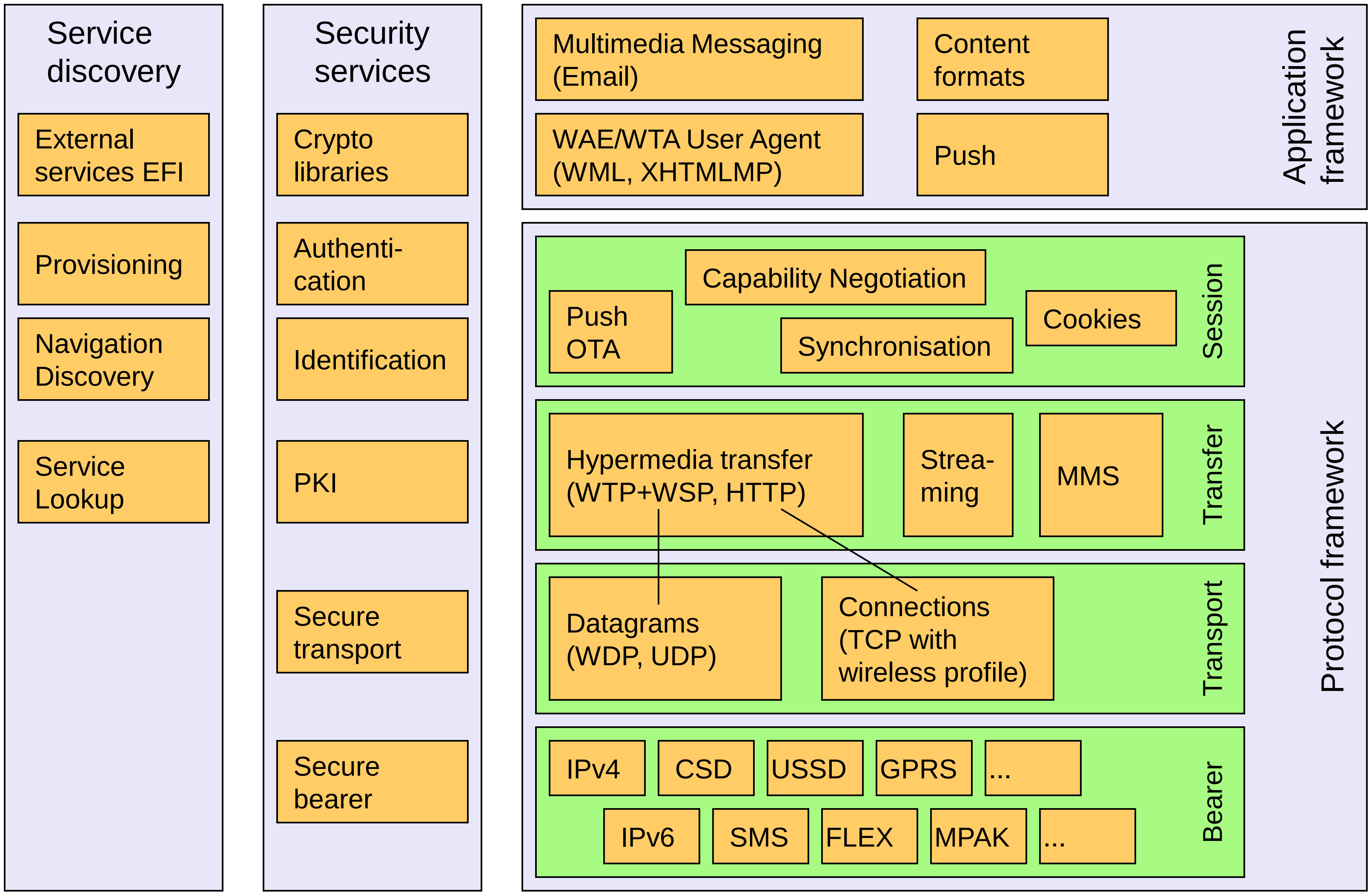

Figure 12: WAP 2.0 architecture

WAP 2.0 (3)

- *Transfer services*:
  - examples for transfer protocols are a wireless profiled HTTP, the combination of WTP/WSP, streaming protocols and message transfer protocols.
  - While the *hypermedia transfer* protocols can be used for web browsing, *streaming* protocols provide tight time-bounds to support isochronous data (audio and video).
  - The *multimedia messaging service* (MMS) transfers asynchronous multi-media content. MMS supports different media types, such as JPEG, GIF, text. There is no fixed upper bound for the message size
- *Session services*:
  - mobile devices need a shared state between network elements to operate efficiently. CC/PP can be used for *capability negotiation*.
  - This includes information about client, server and proxy capabilities and allows customization of content.

WAP 2.0 (4)

    - The *push* OTA service offers reliable and unreliable push services. The *cookie* service has been adopted from the internet.
    - This service can establish state on a client that survives multiple hypermedia transfers.
    - The synchronization service has been defined for synchronizing replicated data

- The *application framework* comprises the basic applications needed for browsing, support of different content formats, email service etc.
  - This framework was developed to establish an interoperable application environment for many different vendors, service providers, and network operators
  - The *WAE/WTA user agent* supports WML known from WAP 1.x as well as XHTML mobile profile.
    - This includes scripting, style-sheets, telephony applications, and programming interfaces adapted to mobile devices

WAP 2.0 (5)

- The applications for *multi-media messaging* and *push* services are also located in this framework
- The many *content formats* have to be supported: color images, audio, video, calendar information, phone book entries etc.

- Mobile devices require extensive *security services*.
  - Just imagine someone pushing many messages on a mobile device that let the device access certain web pages repeatedly.
    - This might cause heavy air traffic that has to be paid for by the user!
    - The security services have to cover the traditional aspects of security, e.g., privacy, authentication, integrity and non-repudiation
  - *Cryptographic libraries* are needed for signing data at the application level
  - *Authentication* services offer several mechanisms to authenticate servers, proxies and clients at different levels
  - The wireless identity module (WIM) provides the functions needed for user *identification* and authentication

WAP 2.0 (6)

  - *Public key infrastructure* (PKI) can be used for the management of public-key cryptography and certificate exchange
  - *WTLS* provides *secure transport* over datagram protocols, while TLS is used for connection-oriented protocols (TCP)
  - Some *bearer* networks already provide *security* functions, e.g., IPSec for IP networks
- *Service discovery* is particularly important for mobile devices.
  - External functions or services on a device (outside WAP specification) can be discovered via *external functionality interface* (EFI)
  - For many network services, a device needs additional parameters (e.g., bootstrap information, smart card specification) to get access.
  - The device can get these parameters via the *provisioning* service

WAP 2.0 (7)

- While surfing though the network using hypermedia documents the device may need to discover new network services.
  - The *navigation discovery* provides a secure way to do this
- The *service lookup* provides for the discovery of parameters needed for a certain service with the help of a directory.
  - The domain name system (DNS) is an example mapping a name onto an IP address

- Components listed for security services and service cannot be assigned to only one layer,
  - they may span several layers in the protocol architecture
- Figure 13 shows four examples of *protocol stacks* using WAP 2.0 components
- The stacks in the upper left corner show the classical WAP 1.x configuration with the WAP gateway translating between internet and WAP protocols

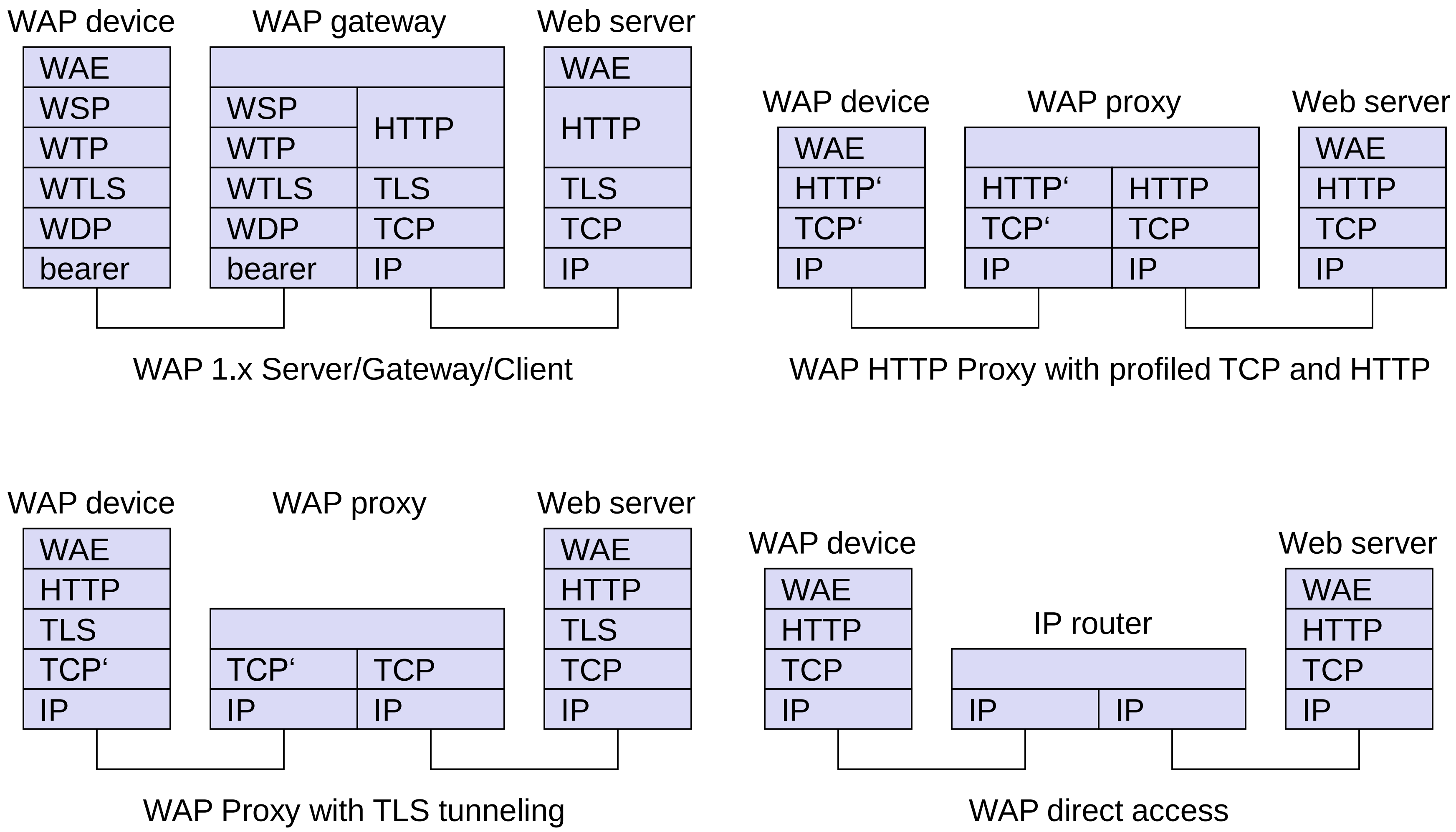

Figure 13: Example protocol stacks according to WAP 2.0

WAP 2.0 (8)

- The upper right corner illustrates protocol stacks similar to those used in i-mode
- The WAP proxy translates the unchanged HTTP/TCP from the internet into profiled versions of these protocols
- The lower left corner shows a WAP proxy tunneling HTTP over TLS.
  - This architecture offers true end-to-end security which contrasts with the example in the upper left corner that breaks the security association in the gateway.
- The lower right corner example shows that WAP 2.0 does not necessarily need a proxy. The WAP device can directly access internet content.
- Although proxies are not required in WAP 2.0 they may enhance the performance.