
# Mobile Applications: Wireless Application Protocol

**Sridhar Iyer**

**IIT Bombay**

**http://www.it.iitb.ac.in/~sri**

# Outline

- Limitations of mobile environment
- Mobile applications
- WWW and mobility
- Wireless application protocol
- WAP protocol stack
- WAP application environment
- Summary

# Limitations of Mobile Environments

- **Limitations of the Wireless Network**
  - **heterogeneity of fragmented networks**
  - **frequent disconnections**
  - **limited communication bandwidth**

# Limitations of Mobile Environments (contd)

- **Limitations Imposed by Mobility**
  - **lack of mobility awareness by systems, applications**
  - **route breakages**
- **Limitations of the Mobile Computer**
  - **short battery lifetime**
  - **limited capacities**

# Mobile Applications

- Vehicles
  - transmission of news, road condition etc
  - ad-hoc network with near vehicles to prevent accidents
- Emergencies
  - early transmission of patient data to hospital
  - ad-hoc network in case of earthquakes…
  - militarv …

# Mobile Applications

- Traveling salesmen
  - direct access to central customer files
  - consistent databases for all agents
  - mobile office
- Web access
  - outdoor Internet access
  - intelligent travel guide with up-to-date location dependent information

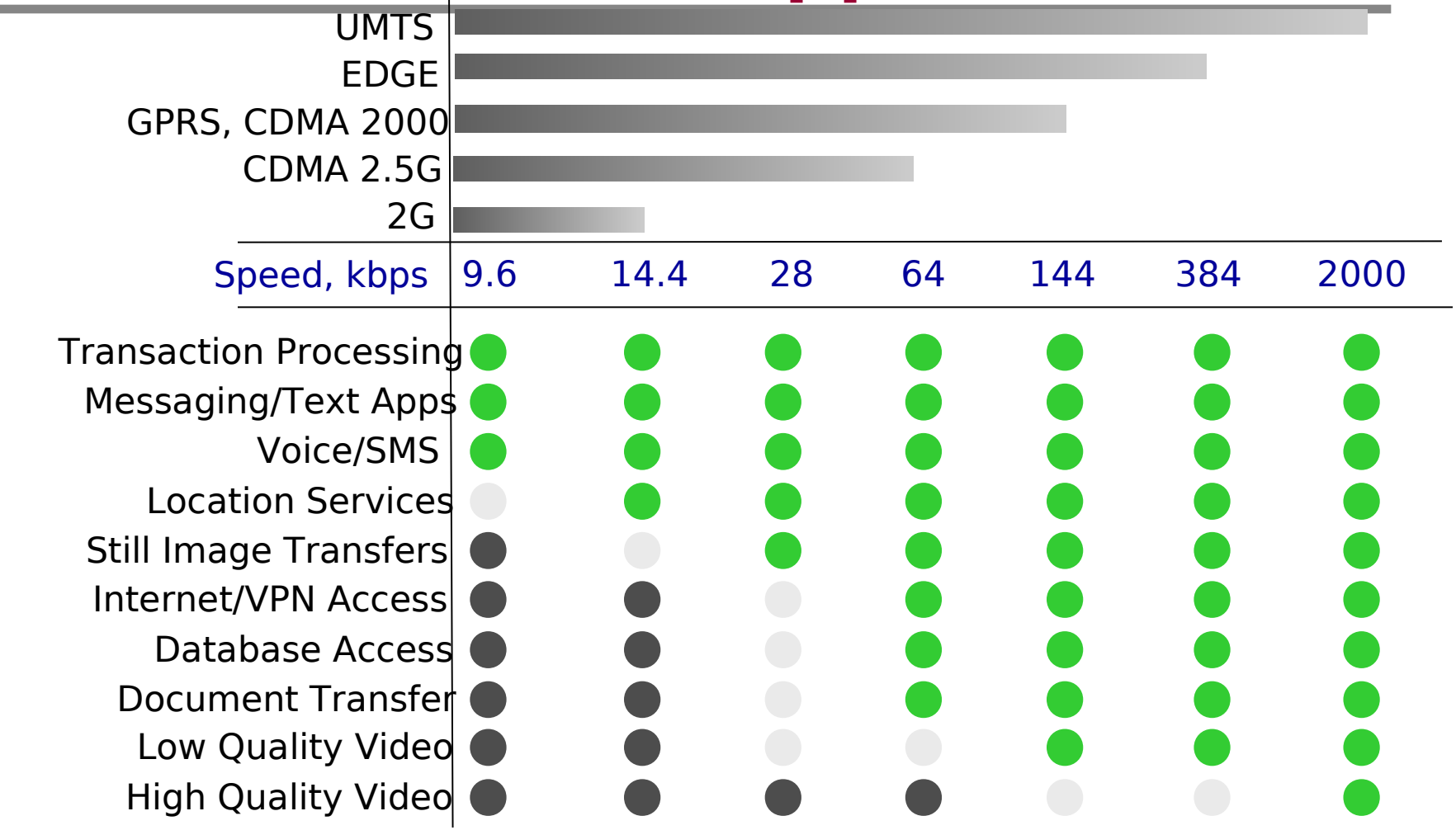
# Mobile Applications

- **Location aware services**
  - **find services in the local environment, e.g. printer**
- **Information services**
  - **push: e.g., stock quotes**
  - **pull: e.g., nearest cash ATM**
- **Disconnected operations**
  - **mobile agents, e.g., shopping**
- **Entertainment**
  - **ad-hoc networks for multi user games**

# Mobile applications in the Industry

- **Wireless access: (phone.com) openwave**
- **Alerting services: myalert.com**
- **Location services: (airflash) webraska.com**
- **Intranet applications: (imedeon) viryanet.com**
- **Banking services: macalla.com**
- **Mobile agents: tryllian.com**
- **....**

# Bandwidth and applications



# World Wide Web and Mobility

- **HTTP characteristics**
  - **designed for large bandwidth, low delay**
  - **stateless, client/server, request/response communication**
  - **connection oriented, one connection per request**
  - **TCP 3-way handshake, DNS lookup overheads**

# WWW and Mobility

- **HTML characteristics**
  - **designed for computers with “high” performance, color high-resolution display, mouse, hard disk**
  - **typically, web pages optimized for design, not for communication; ignore end-system characteristics**

# System Support for Mobile WWW

- **Enhanced browsers**
  - **client-aware support for mobility**
- **Proxies**
  - **Client proxy: pre-fetching, caching, off-line use**
  - **Network proxy: adaptive content transformation for connections**
  - **Client and network proxy**

# System Support for Mobile WWW

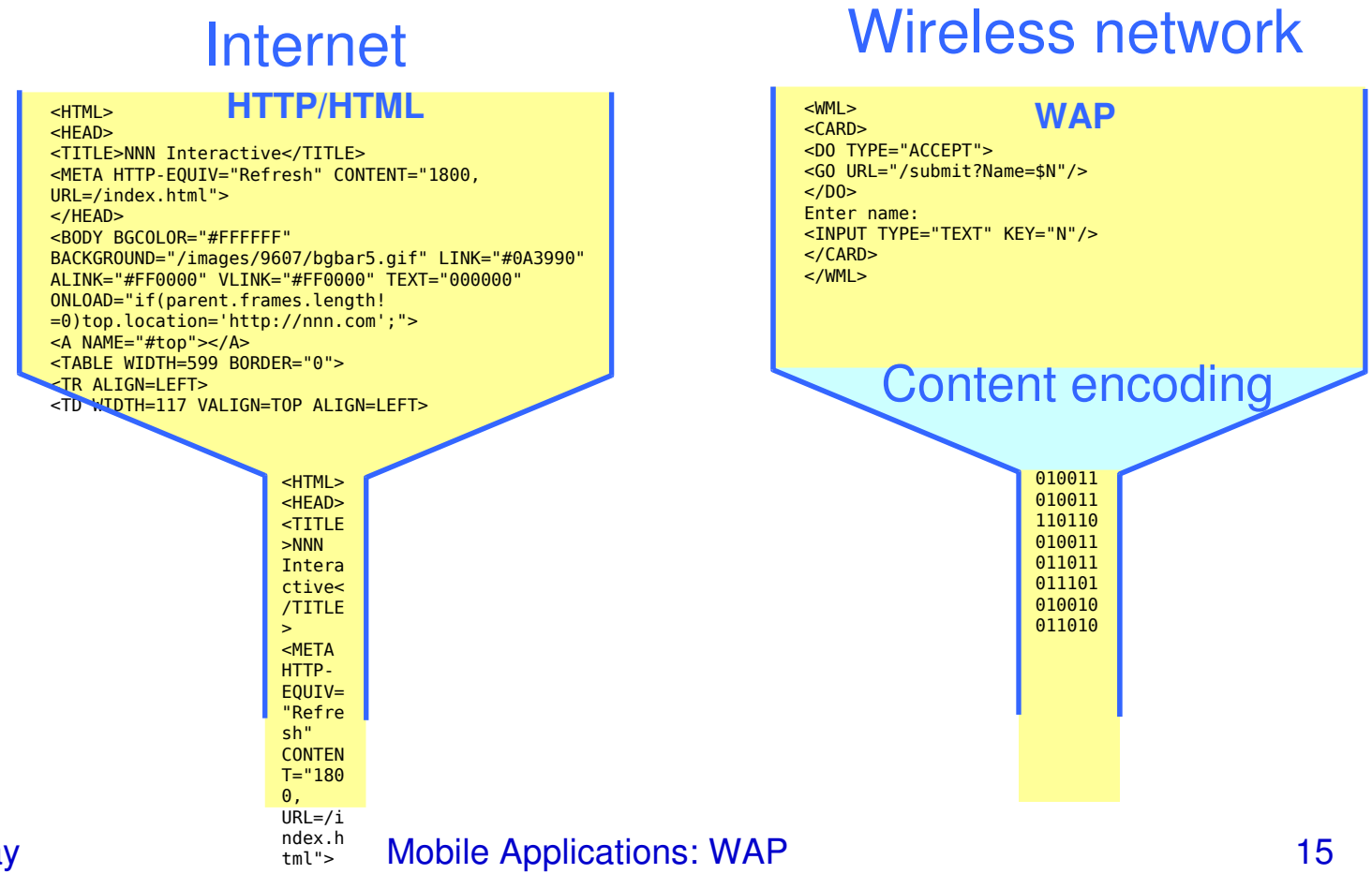
- **Enhanced servers**
  - **server-aware support for mobility**
  - **serve the content in multiple ways, depending on client capabilities**
- **New protocols/languages**
  - **WAP/WML**

# Wireless Application Protocol

- **Goals**
  - **deliver Internet services to mobile devices**
  - **independence from wireless network standards**
  - **GSM, CDMA IS-95, TDMA IS-136, 3G systems (UMTS, W-CDMA)**

# Why is HTTP/HTML not enough?

Big pipe - small pipe syndrome






Source: WAP Forum

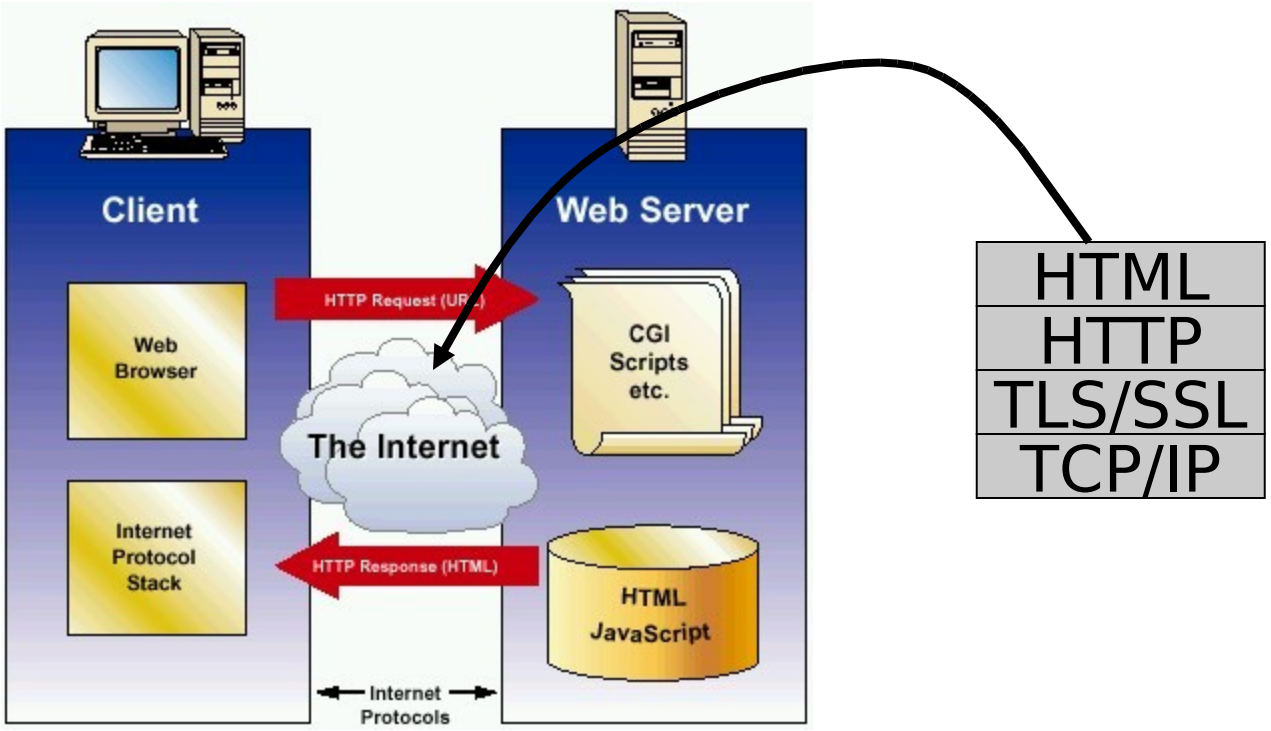
# WHY WAP?

- **Wireless networks and phones**
  - **have specific needs and requirements**
  - **not addressed by existing Internet technologies**
- **WAP**
  - **Enables any data transport**
    - »**TCP/IP, UDP/IP, GUTS (IS-135/6), SMS, or USSD.**
  - **Optimizes the content and air-link protocols**
  - **Utilizes plain Web HTTP 1.1 servers**
    - »**utilizes standard Internet markup language technology (XML)**

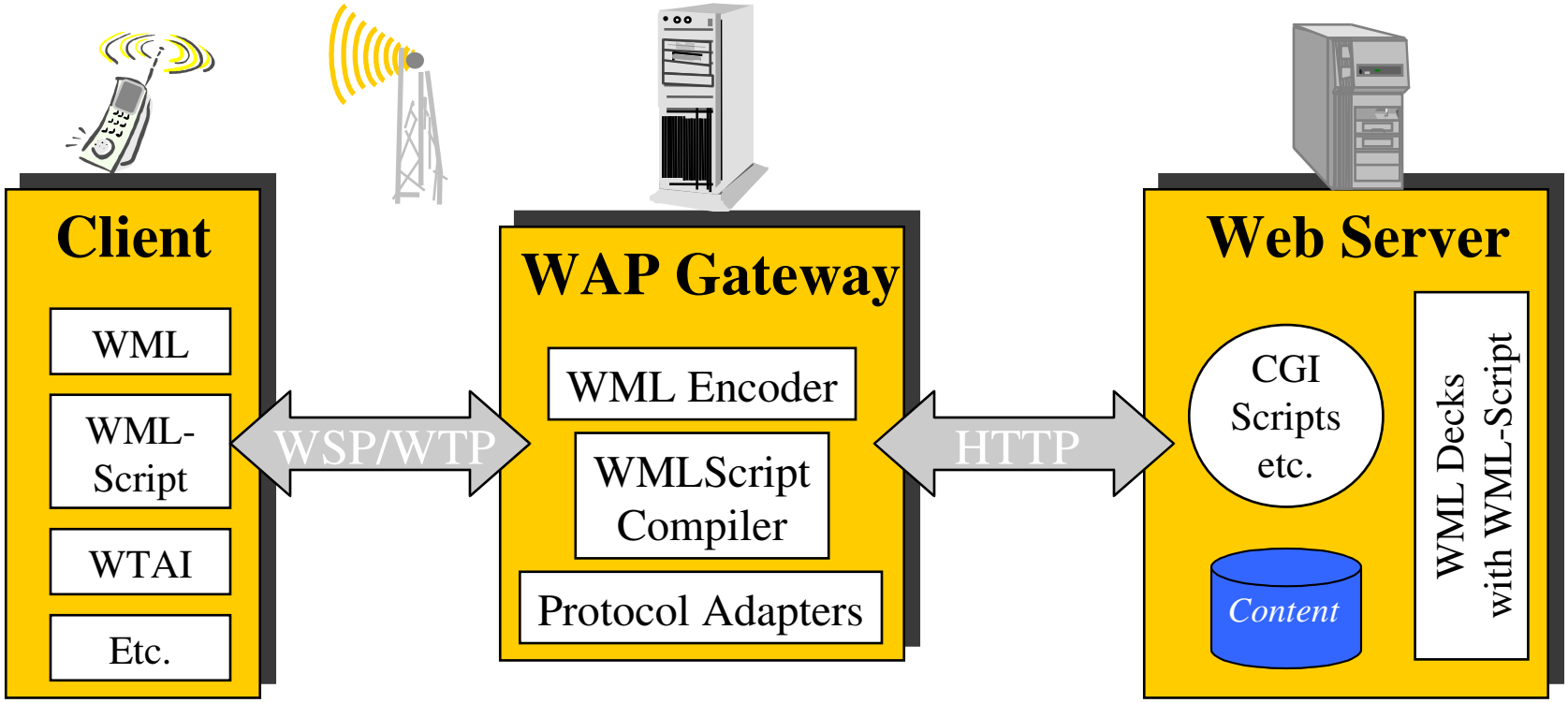
# WAP: main features

- **Browser**
  - **“Micro browser”, similar to existing web browsers**
- **Markup/Script language**
  - **Similar to HTML/Javascript, adapted to mobile devices**
- **Gateway**
  - **Transition from wireless to wired world**
- **Server**
  - **“Wap/Origin server”, similar to existing web servers**
- **Protocol layers**
  - **Transport layer, security layer, session layer etc.**
- **Telephony application interface**
  - **Access to telephony functions**

# Internet model



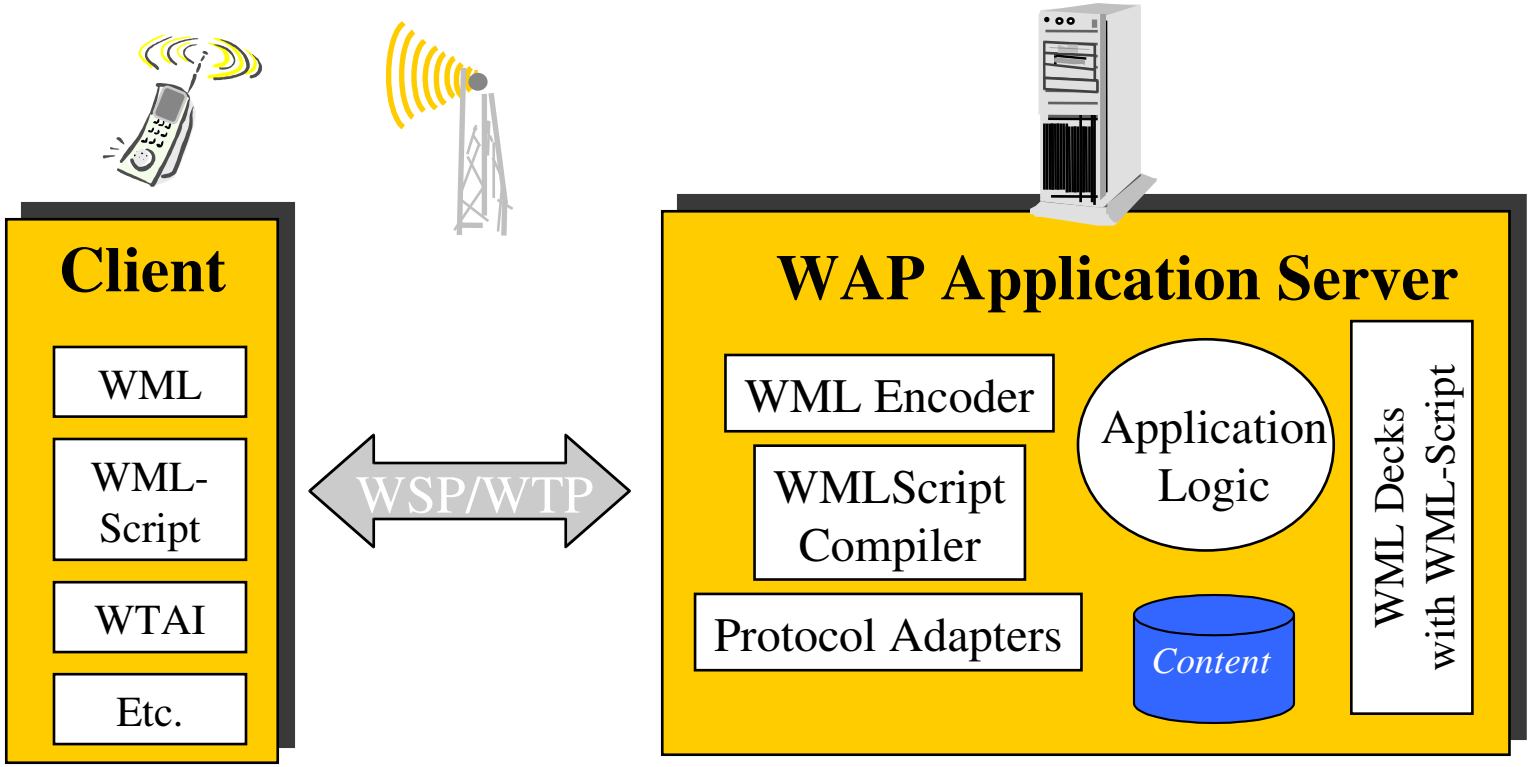
# WAP architecture





Source: WAP Forum

# WAP application server



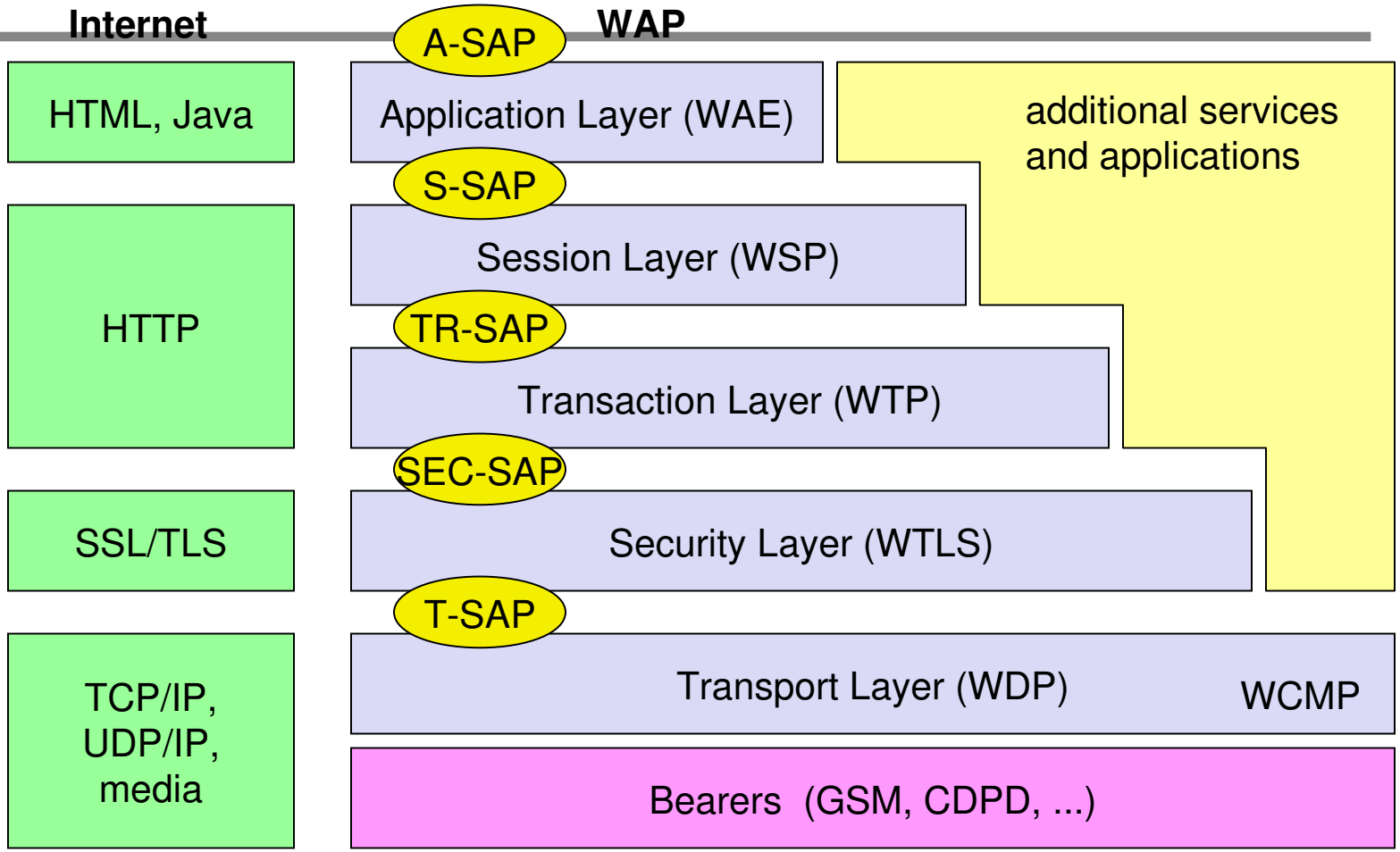



Source: WAP Forum

# WAP specifies

- **Wireless Application Environment**
  - **WML Microbrowser**
  - **WMLScript Virtual Machine**
  - **WMLScript Standard Library**
  - **Wireless Telephony Application Interface (WTAI)**
  - **WAP content types**
- **Wireless Protocol Stack**
  - **Wireless Session Protocol (WSP)**
  - **Wireless Transport Layer Security (WTLS)**
  - **Wireless Transaction Protocol (WTP)**
  - **Wireless Datagram Protocol (WDP)**
  - **Wireless network interface definitions**

# WAP: reference model




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



Source: Schiller

# WAP stack

- **WAE (Wireless Application Environment):**
  - **Architecture: application model, browser, gateway, server**
  - **WML: XML-Syntax, based on card stacks, variables, ...**
  - **WTA: telephone services, such as call control, phone book etc.**
- **WSP (Wireless Session Protocol):**
  - **Provides HTTP 1.1 functionality**
  - **Supports session management, security, etc.**

# WAP stack (contd.)

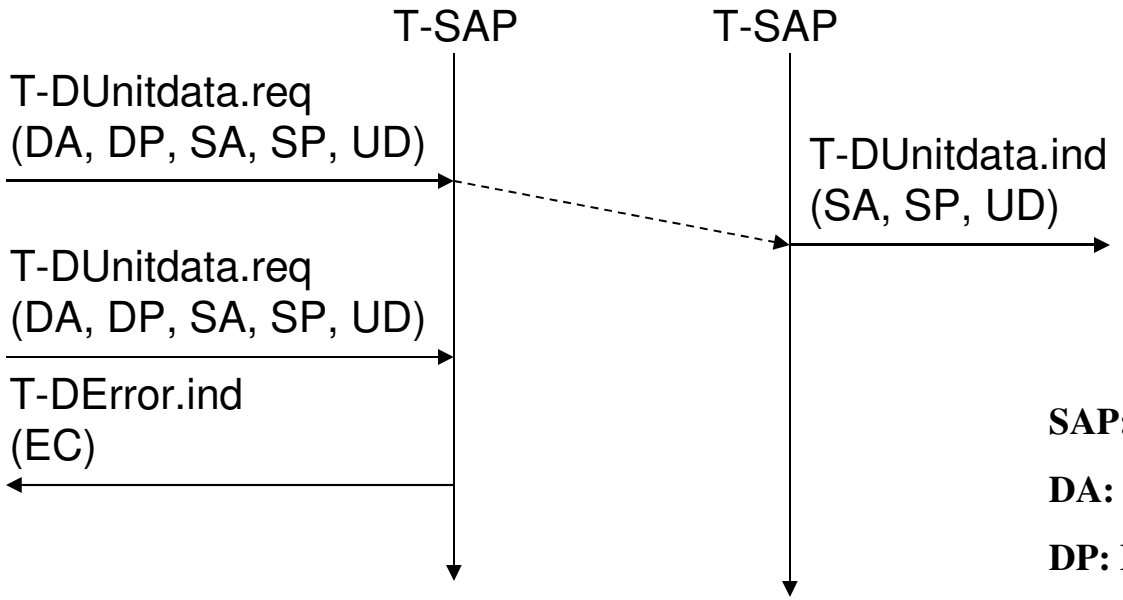
- **WTP (Wireless Transaction Protocol):**
  - **Provides reliable message transfer mechanisms**
  - **Based on ideas from TCP/RPC**
- **WTLS (Wireless Transport Layer Security):**
  - **Provides data integrity, privacy, authentication functions**
  - **Based on ideas from TLS/SSL**
- **WDP (Wireless Datagram Protocol):**
  - **Provides transport layer functions**
  - **Based on ideas from UDP**

**Content encoding, optimized for low-bandwidth channels, simple devices**

# WDP: Wireless Datagram Protocol

- Transport layer protocol within the WAP architecture
- uses the Service Primitive
  - » T-UnitData.req .ind
- uses transport mechanisms of different bearer technologies
- offers a common interface for higher layer protocols
- allows for transparent communication despite different technologies
- addressing uses port numbers
- WDP over IP is UDP/IP

# WDP: service primitives




**SAP: Service Access Point**

**DA: Destination Address**

**DP: Destination Port**

**SA: Source Address**

**SP: Source Port**

**UD: User Data**

**EC: Error Code**

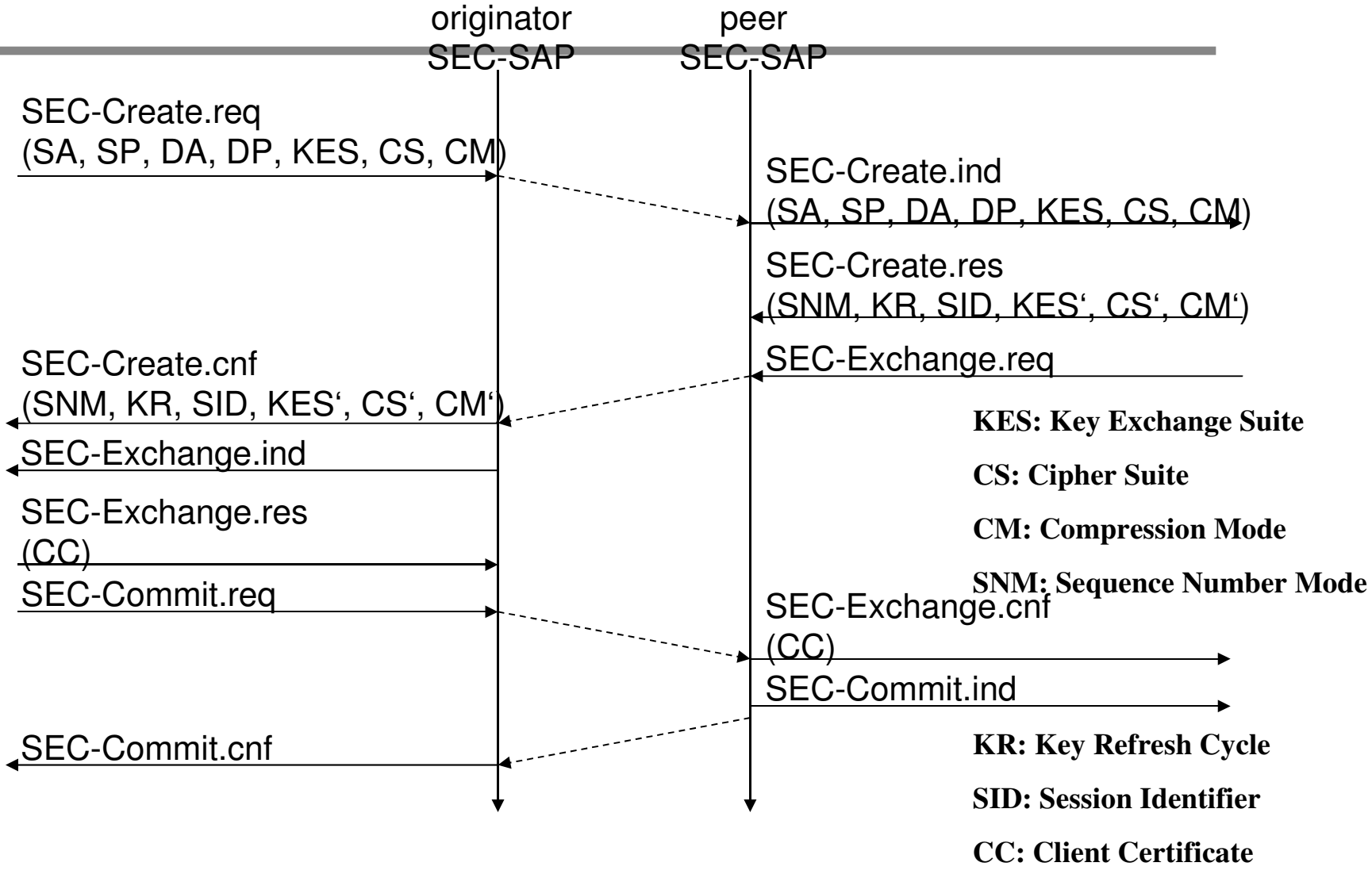


Source: Schiller

# WTLS:Wireless Transport Layer Security

- is based on the TLS/SSL (Transport Layer Security) protocol
- optimized for low-bandwidth communication channels
- provides
  - privacy (encryption)
  - data integrity (MACs)
  - authentication (public-key and symmetric)
- Employs special adapted mechanisms for wireless usage
  - Long lived secure sessions
  - Optimised handshake procedures

# WTLS: secure session, full handshake






Source: Schiller

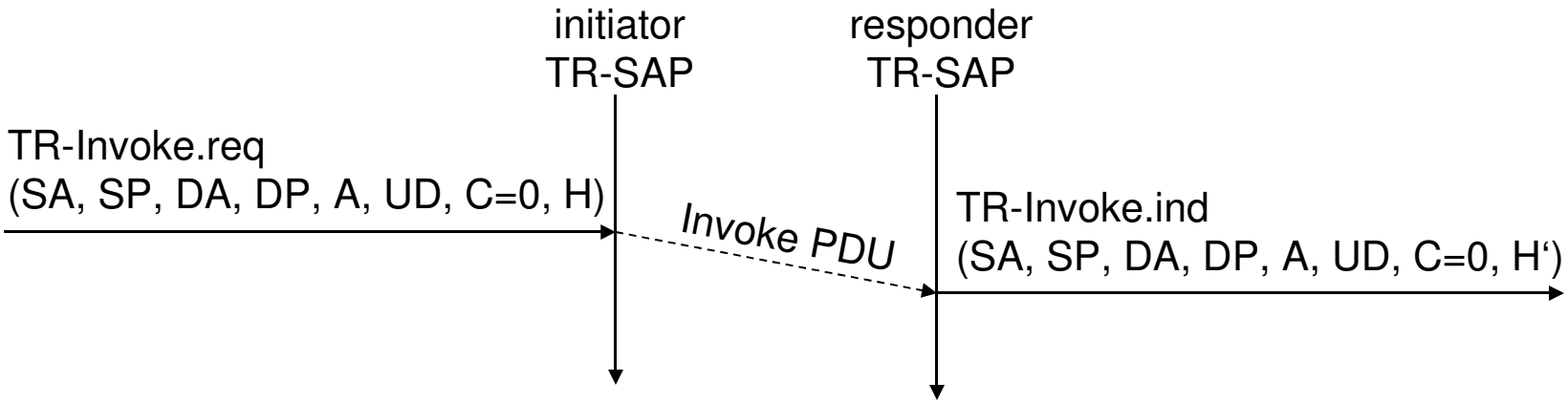
# WTP: Wireless Transaction Protocol

- different transaction services that enable applications to select reliability, efficiency levels
- low memory requirements, suited to simple devices
- efficiency for wireless transmission
- supports peer-to-peer, client/server and multicast applications

# WTP transactions

- **class 0*:* unreliable message transfer**
- **class 1*:* reliable message transfer without result message**
- **class 2*:* reliable message transfer with exactly one reliable result message**

# WTP Class 0 Transaction




**A: Acknowledgement Type (WTP/User)**

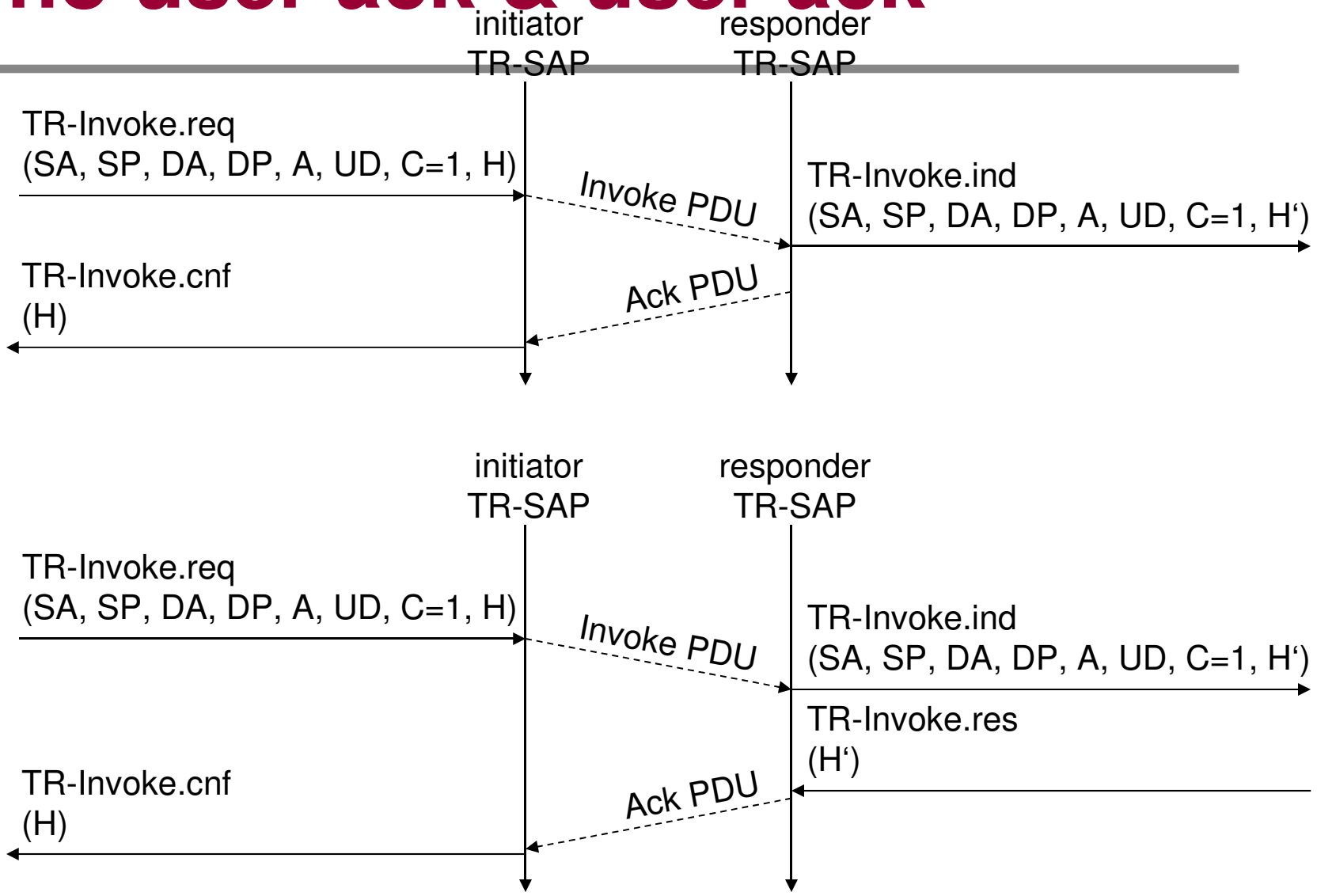
**C: Class (0,1,2)**

**H: Handle (socket alias)**



Source: Schiller

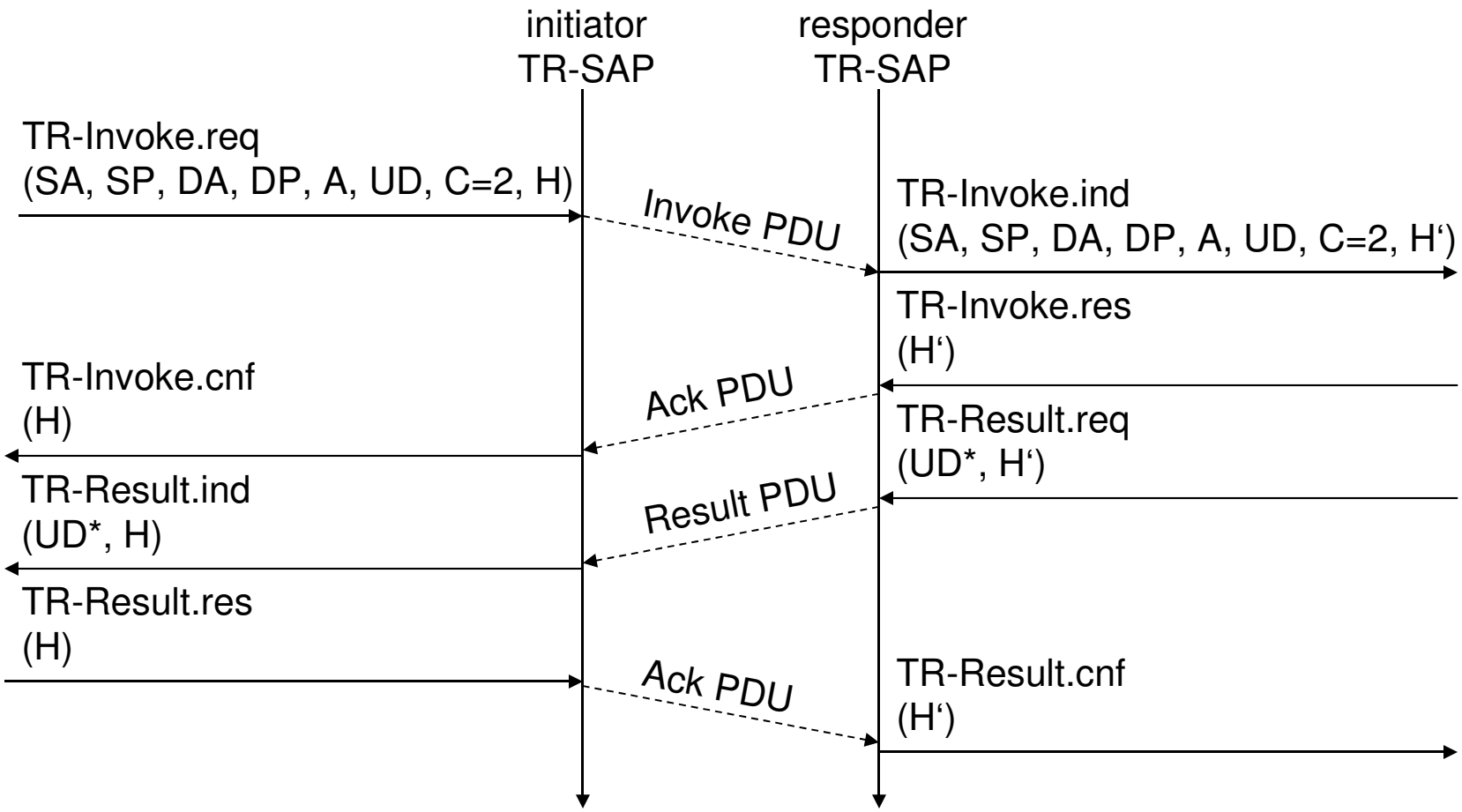
# WTP Class 1 Transaction, no user ack & user ack






Source: Schiller

# WTP Class 2 Transaction



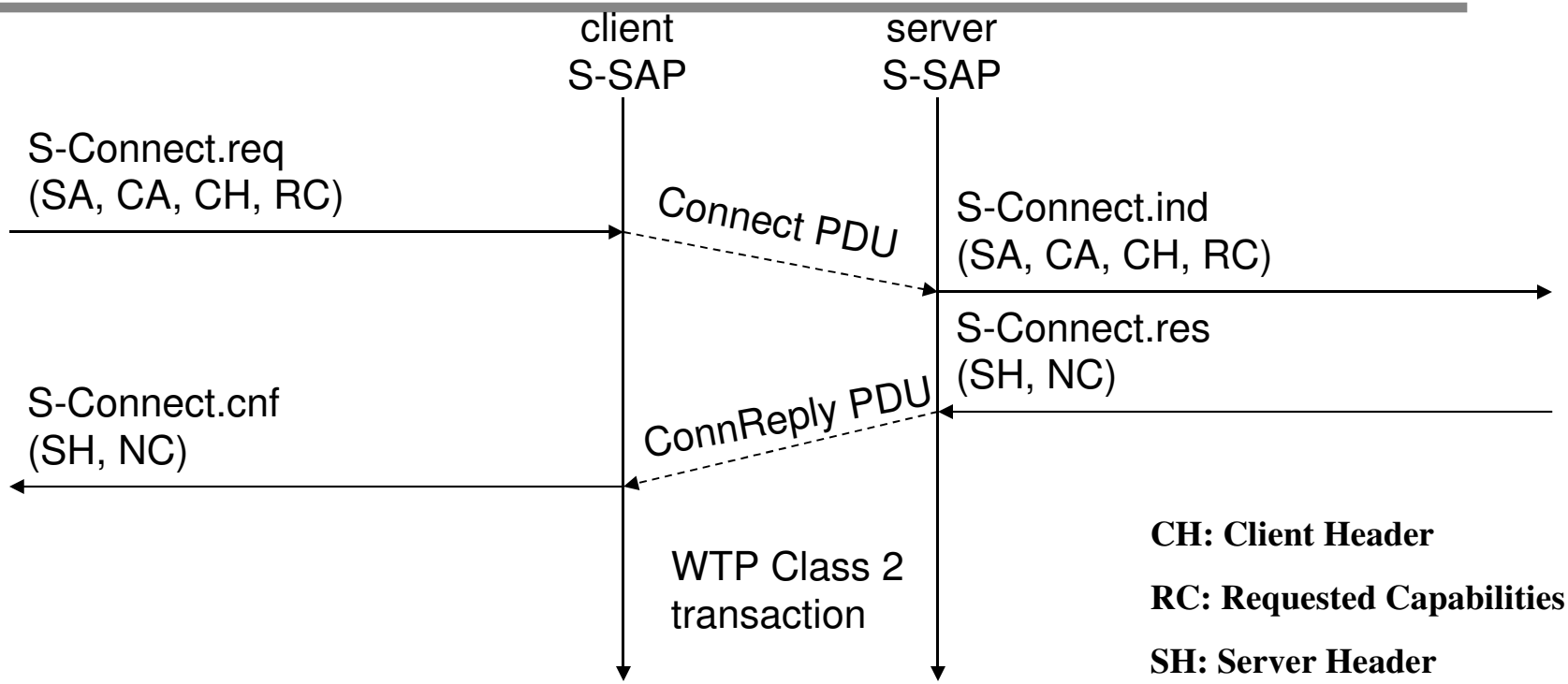



Source: Schiller

# WSP - Wireless Session Protocol

- provides shared state between client and server, optimizes content transfer
- session management (establish, release, suspend, resume)
- efficient capability negotiation
- key management, authentication, Internet security services
- content encoding
- push

# WSP/B session establishment




**CH: Client Header**

**RC: Requested Capabilities**

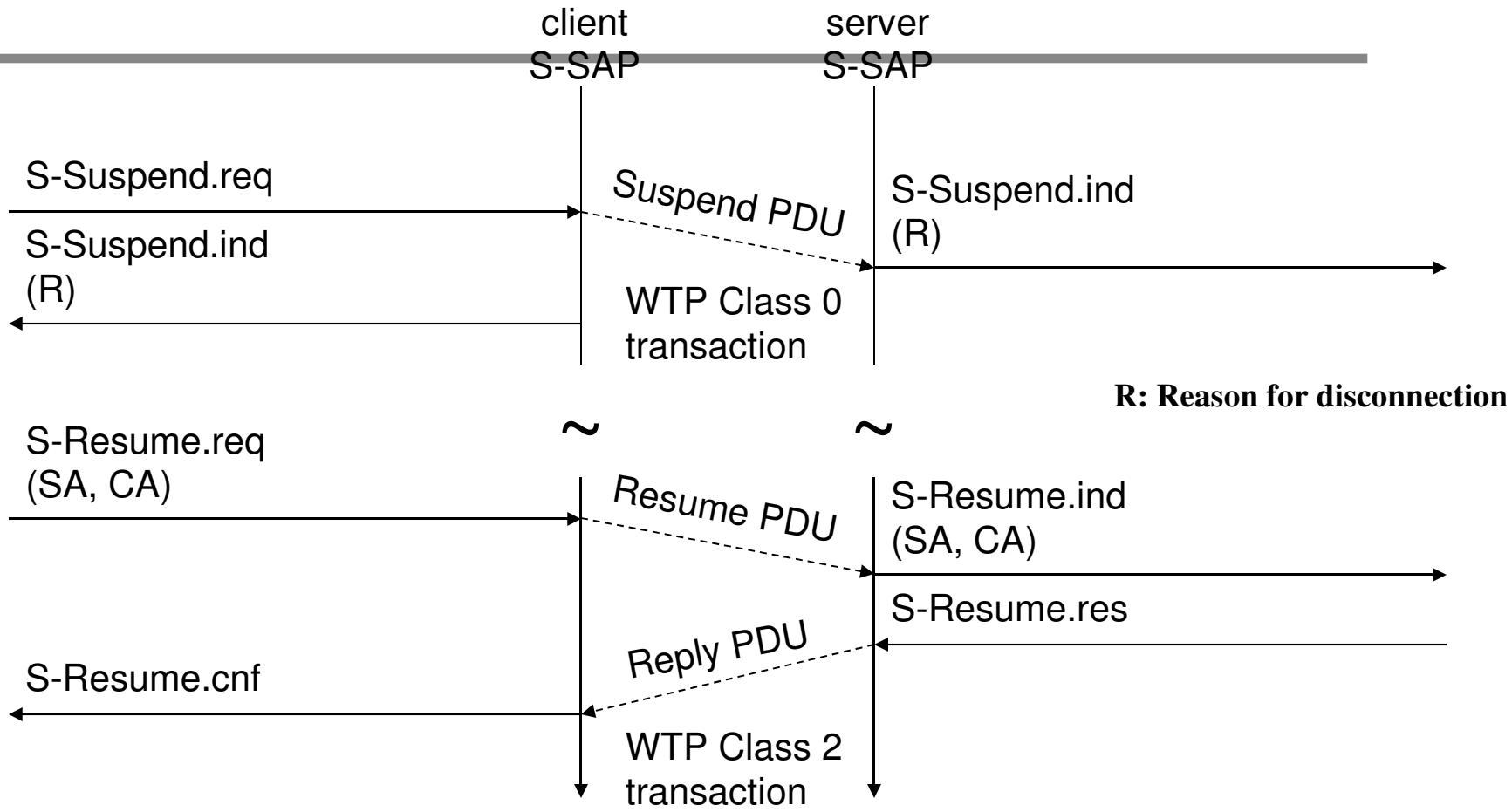
**SH: Server Header**

**NC: Negotiated Capabilities**



Source: Schiller

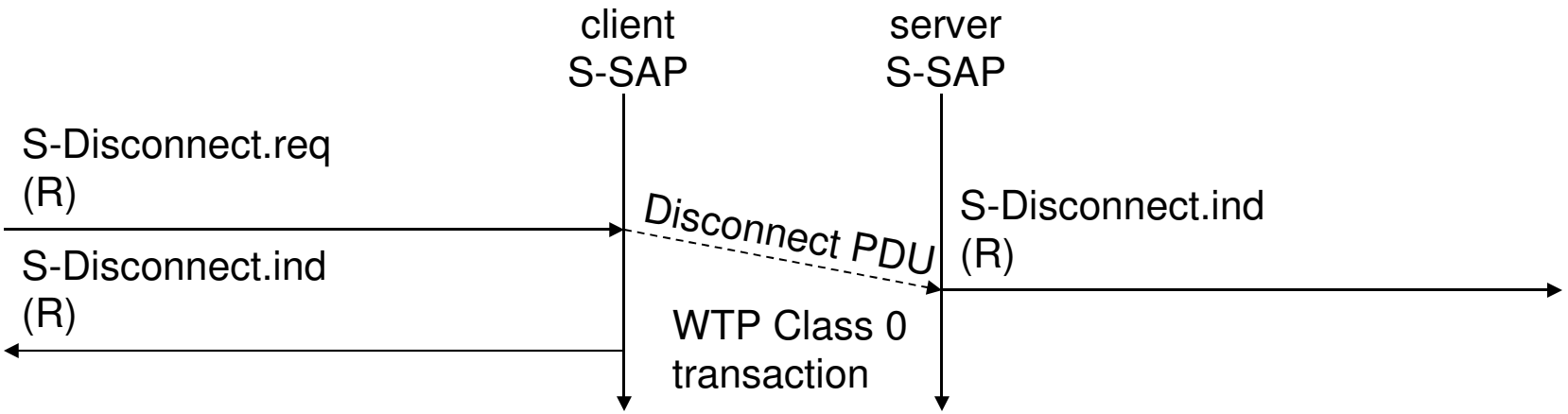
# WSP/B session suspend/resume





Source: Schiller

# WSP/B session termination





Source: Schiller

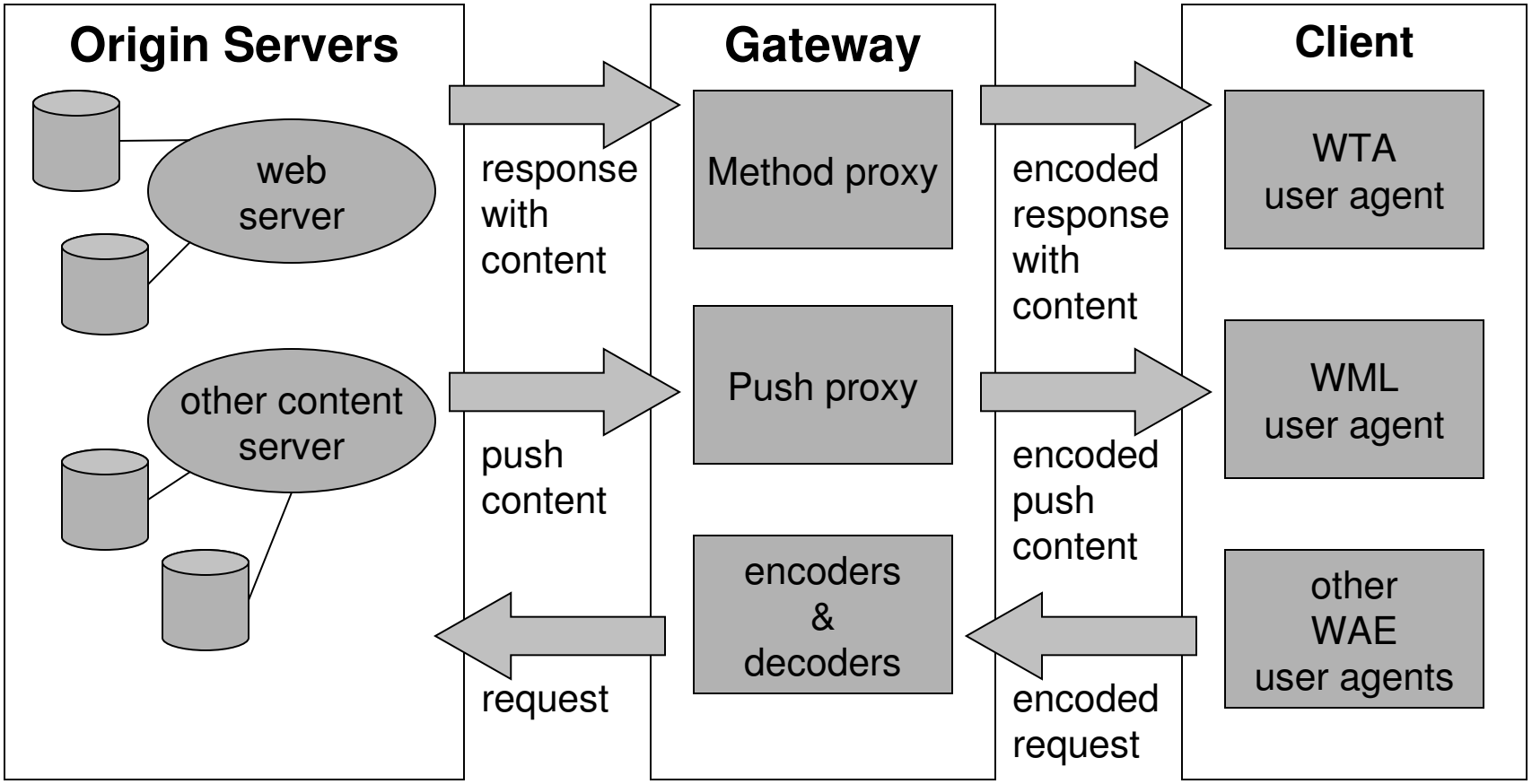
# WAP stack summary

- **WDP**
  - **functionality similar to UDP in IP networks**
- **WTLS**
  - **functionality similar to SSL/TLS (opt. for wireless)**
- **WTP**
  - **Class 0: analogous to UDP**
  - **Class 1: analogous to TCP (without connection setup overheads)**
  - **Class 2: analogous to RPC (optimized for wireless)**
- **WSP**
  - **features of suspend/resume**

# WAE components

- **Architecture**
  - Application model, Microbrowser, Gateway, Server
- **User Agents**
  - WML/WTA/Others
  - content formats: vCard, WML..
- **WML/Script**
  - XML-Syntax, based on card stacks, variables, ...
- **WTA**
  - telephone services, such as call control, text messages, phone book.

# WAE: logical model

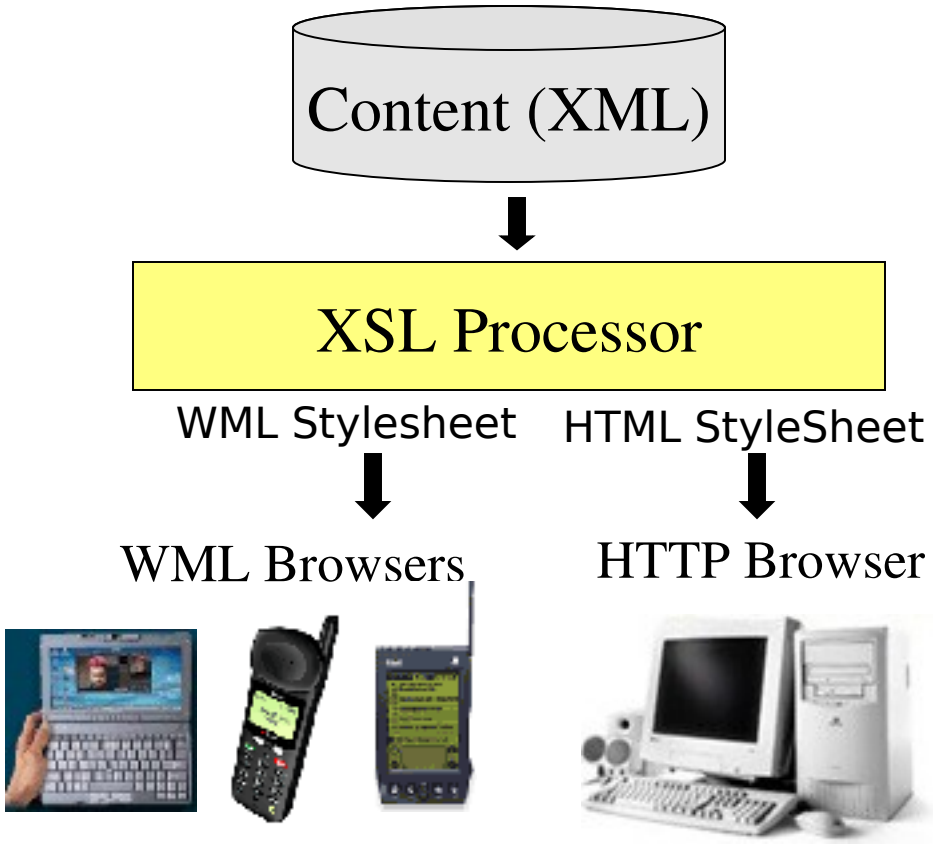


# WAP microbrowser


- **Optimized for wireless devices**
- **Minimal RAM, ROM, Display, CPU and keys**
- **Provides consistent service UI across devices**
- **Provides Internet compatibility**
- **Enables wide array of available content and applications**

# WML: Wireless Markup Language

- **Tag-based browsing language:**
  - **Screen management (text, images)**
  - **Data input (text, selection lists, etc.)**
  - **Hyperlinks & navigation support**
- **Takes into account limited display, navigation capabilities of devices**

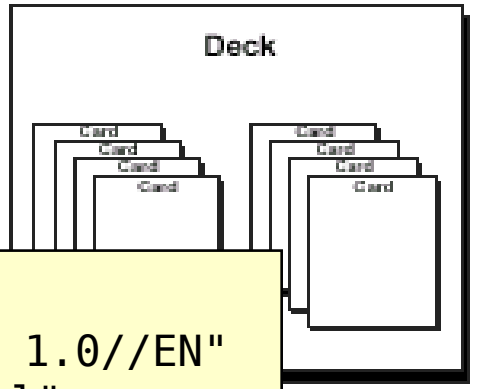


# WML

- XML-based language
  - describes only intent of interaction in an abstract manner
  - presentation depends upon device capabilities
- Cards and Decks
  - document consists of many cards
  - User interactions are split into cards
  - Explicit navigation between cards
  - cards are grouped to decks (unit of content)
- Events, variables and state mgmt

# WML

- All decks must contain
  - Document prologue
    - XML & document type declaration
  - <WML> element
    - Must contain one or more cards

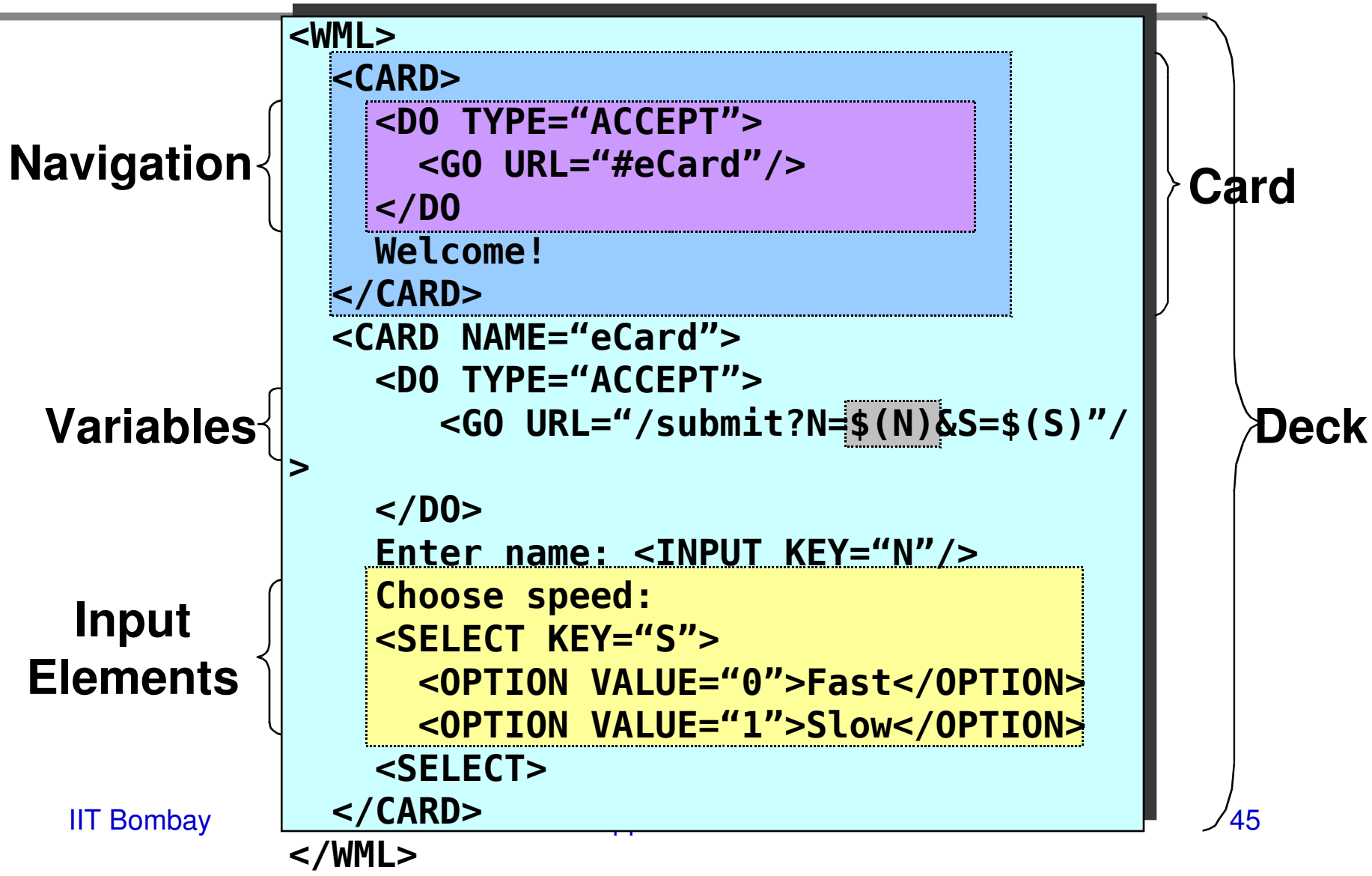



WML File Structure

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

<WML>
    ...
</WML>
```

# WML cards


Navigation
Variables
Input Elements
Card
Deck
<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>

# Wireless Telephony Application

- **Collection of telephony specific extensions**
  - **designed primarily for network operators**
- **Example**
  - **calling a number (WML)**
    ```
    wtai://wp/mc;07216086415
    ```
  - **calling a number (WMLScript)**
    ```
    WTAPublic.makeCall("07216086415");
    ```

# WTA features

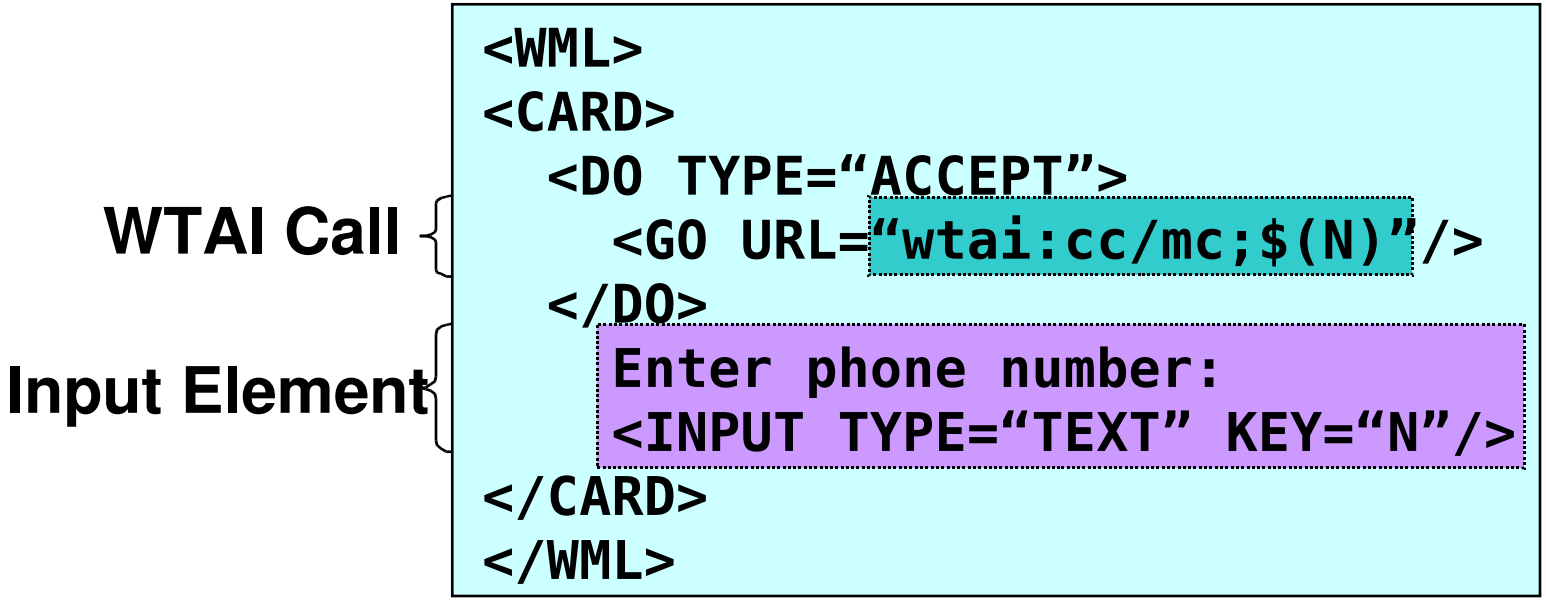
- **Extension of basic WAE application model**
  - **network model for interaction**
    - **client requests to server**
    - **event signaling: server can push content to the client**
  - **event handling**
    - **table indicating how to react on certain events from the network**
    - **client may now be able to handle unknown events**

# WTA Interface

- **WTA API includes**
  - **Call control**
  - **Network text messaging**
  - **Phone book interface**
  - **Event processing**
- **Security model: segregation**
  - **Separate WTA browser**
  - **Separate WTA port**

# WTA Example (WML)

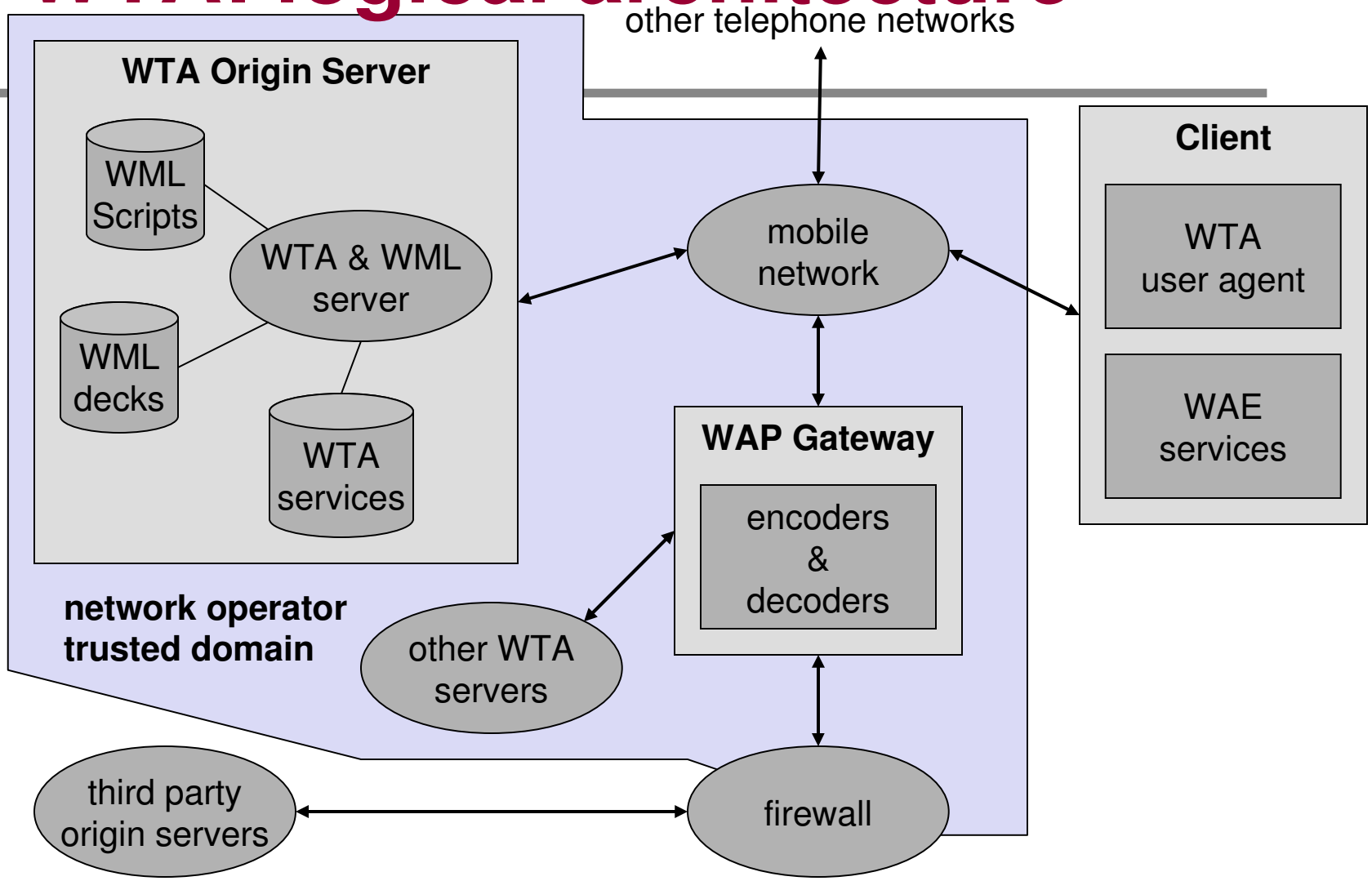
*Placing an outgoing call with WTAI:*





Source: WAP Forum

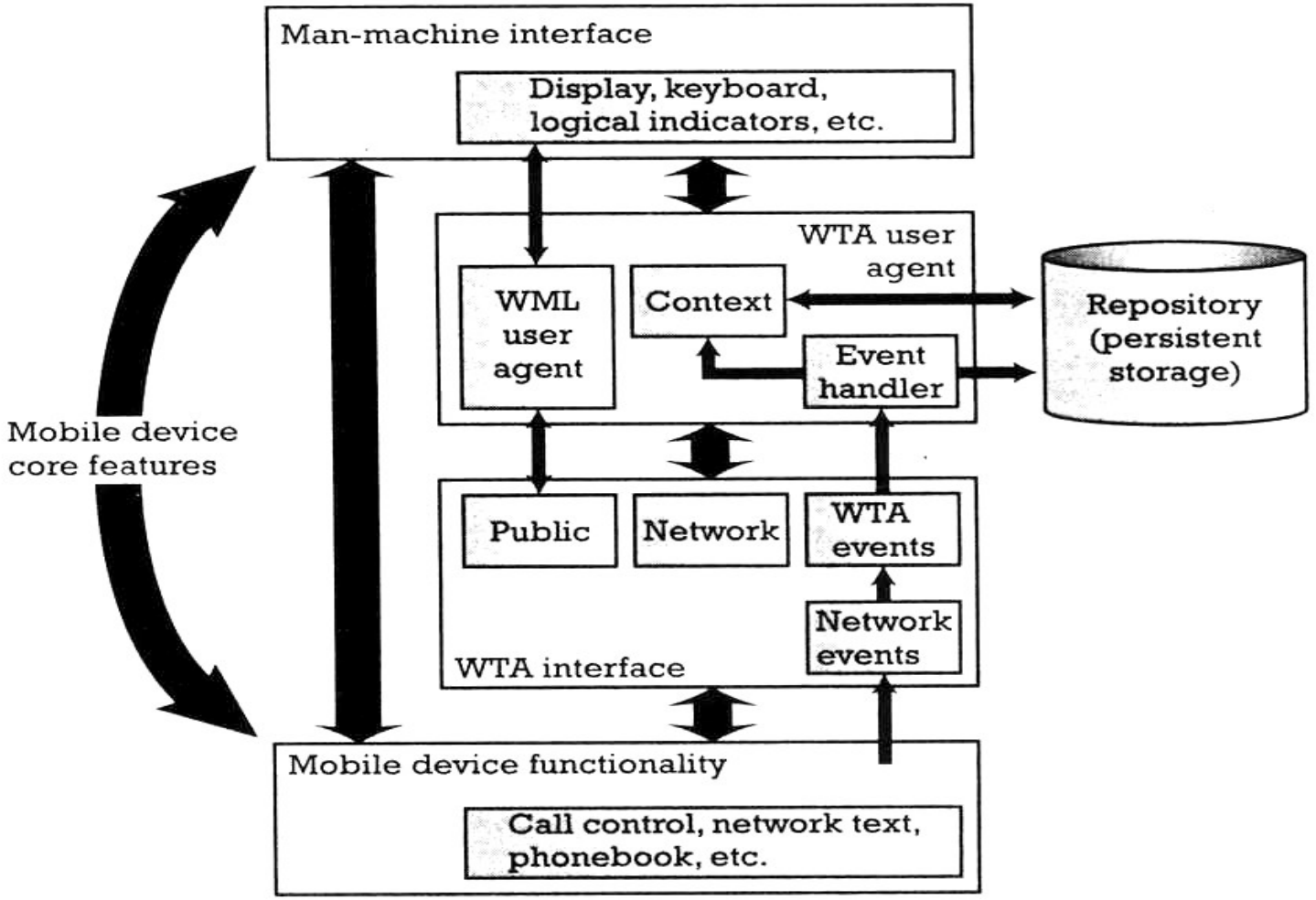
# WTA: logical architecture





Source: Schiller

# WTA: framework components





Source: Heijden

# WTA: user agent

- **WTA User Agent**
  - **WML User agent with extended functionality**
  - **can access mobile device's telephony functions through WTAI**
  - **can store WTA service content persistently in a repository**
  - **handles events originating in the mobile network**

# WTA user agent context

- Abstraction of execution space
- Holds current parameters, navigation history, state of user agent
- Similar to activation record in a OS process

- Uses connection-mode and connectionless services offered by WSP
- Specific, secure WDP ports on the WAP gateway

# WTA: events

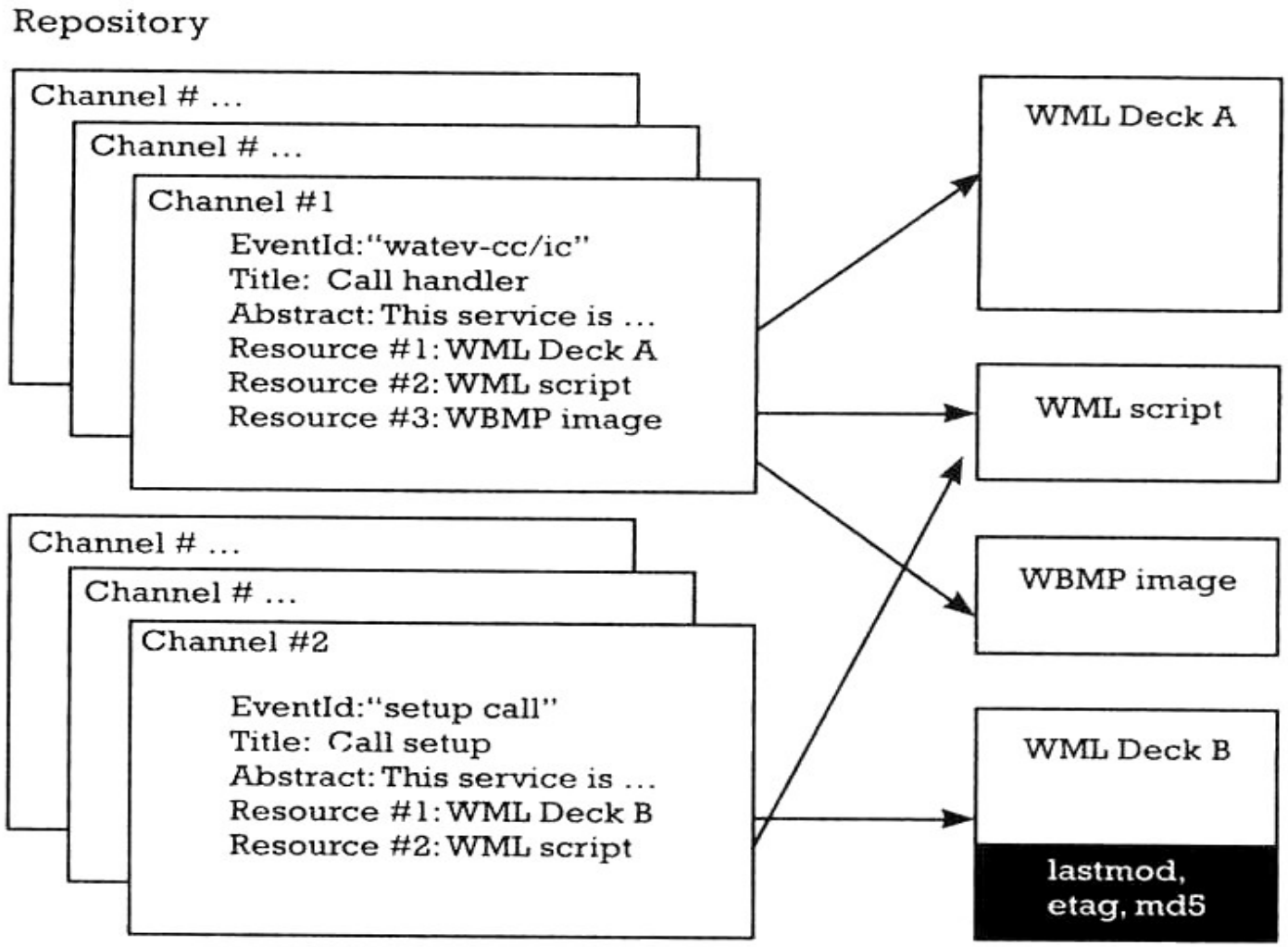
- Network notifies device of event (such as incoming call)
- WTA events map to device's native events
- WTA services are aware of and able to act on these events

- example: incoming call indication, call cleared, call connected

# WTA: Repository

- **local store for content related to WTA services (minimize network traffic)**
- **Channels: define the service**
  - **XML document specifying eventid, title, abstract, and resources that implement a service**
- **Resources: execution scripts for a service**
  - **could be WML decks, WML Scripts, WBMP images..**
  - **downloaded from WTA server and stored in repository before service is referenced**

# WTA: Channels and Resources






Source: Heijden

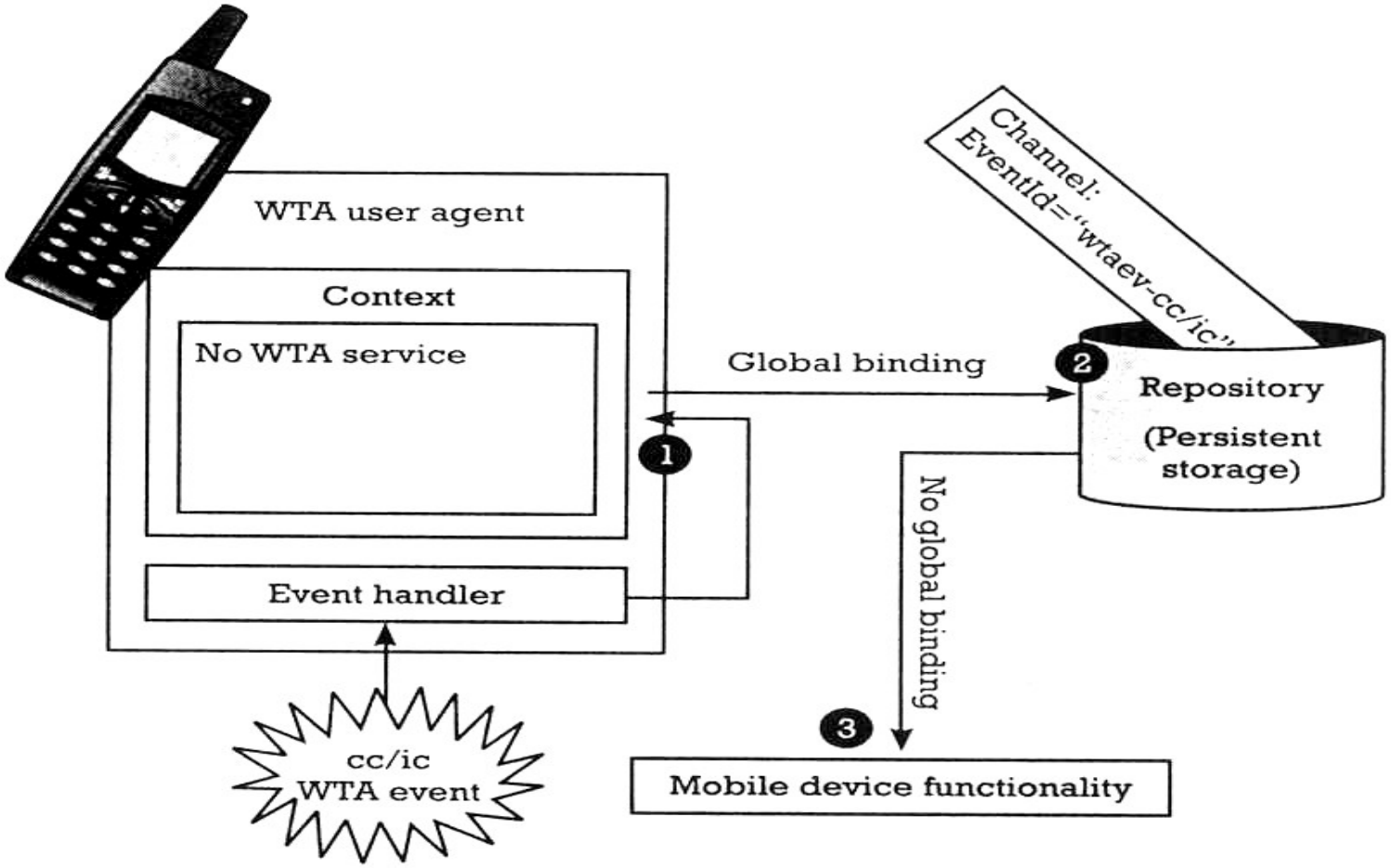
# WTA: event handling

- **Event occurrence**
  - **WTA user agent could be executing and expecting the event**
  - **WTA user agent could be executing and a different event occurs**
  - **No service is executing**
- **Event handling**
  - **channel for each event defines the content to be processed upon reception of that event**

# WTA: event binding

- association of an event with the corresponding handler (channel)
- Global binding:
  - channel corresponding to the event is stored in the repository
  - example: voice mail service
- Temporary binding:
  - resources to be executed are defined by the already executing service
  - example: yellow pages lookup and call establishment

# WTA: event handling



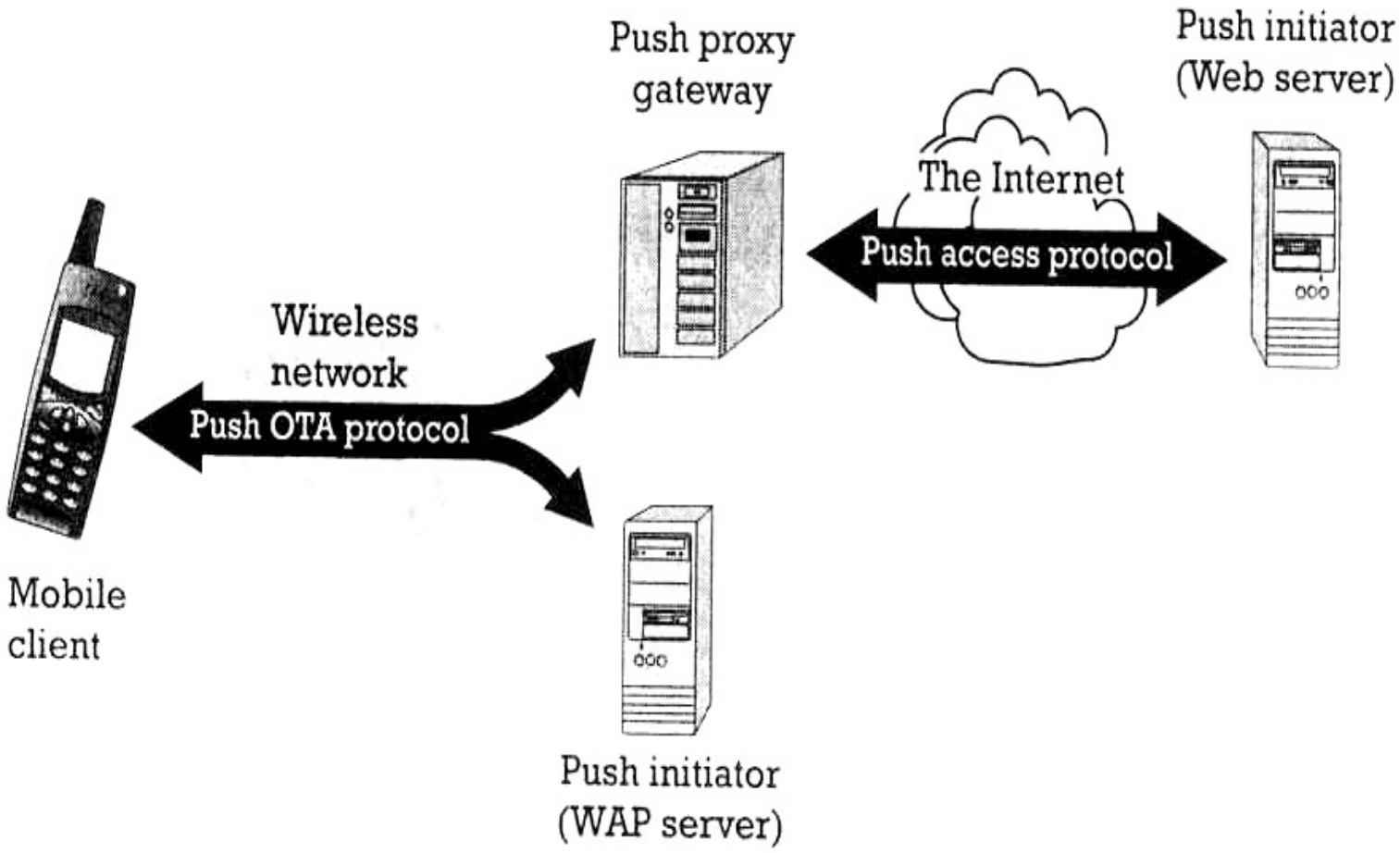


Source: Heijden

# WAP push services

- **Web push**
  - **Scheduled pull by client (browser)**
- **Wireless push**
  - **accomplished by using the network itself**
    - **»example: SMS**
- **WAP push**
  - **Network supported push of WML content**
    - **»example: Alerts or service indications**
  - **Pre-caching of data (channels/resources)**

# WAP push framework





Source: Heijden

# Push Access Protocol

- Based on request/response model
- Push initiator is the client
- Push proxy is the server
- Initiator uses HTTP POST to send push message to proxy
- Initiator sends control information as an XML document, and content for mobile (as WML)
- Proxy sends XML entity in response indicating submission status

# WAE Summary

- **WML and WML Script**
  - analogous to HTML and JavaScript (optimized for wireless)
  - microbrowser user agent; compiler in the network
- **WTA**
  - WTAI: different access rights for different applications/agents
  - WTA User Agent (analogy with operating systems)
    - » Context – Activation Record
    - » Channel – Interrupt Handler
    - » Resource – Shared routines invoked by interrupt handlers
    - » Repository – Library of interrupt handlers
  - feature of dynamically pushing the interrupt handler before the event
- **Push**
  - no analogy in Internet

# References


- J. Schiller, “Mobile Communications”, Addison Wesley, 2000
- M.v.d. Heijden, M. Taylor. “Understanding WAP”, Artech House, 2000


- Websites:
  - www.palowireless.com
  - www.gsmworld.com; www.wapforum.org