WAP 2.x architecture—Features, services and functions

Peter Arnby, Johan Hjelm and Peter Stark

Web services adapted to the wireless environment have been a tremendous success, especially in Japan, where more than 25 million people use i-mode. However, the architecture of i-mode has some limitations. These will be overcome with WAP 2.x.

In this article, the authors describe WAP 2.x, the universal wireless application environment. They explain the similarities and differences between the WAP 1.x and WAP 2.x architectures, and depict conceptual components and features of the second-generation architecture. These components—the application environment, protocol framework, security services, and service discovery—are key to successful application and service development. Modularity, an important feature of the second-generation architecture, enables developers to select modules from different components and to create user services that fulfill market demands.

Introduction

Most Web services cannot be used directly in a mobile terminal since they have been designed for larger screens and for use that is incompatible with the mobile terminal. Despite these limitations, mobile terminals are positioned to become devices of ubiquitous information access. In Japan, for example, NTT DoCoMo's i-mode service is used by some 25 million subscribers. And this number is growing by approximately 70,000 users per day. Similarly, in Europe, the wireless application protocol (WAP) has been a huge success among developers. In-

BOX A, TERMS AND ABBREVIATIONS

CC/PP	Composite capabilities/ preferences profile	SyncML	Synchronization markup language
CSD	Circuit-switched data	TCP	Transport control protocol
CSS	Cascading style sheet	TLS	Transport layer security (formerly
DNS	Domain name server		secure socket layer)
ECMAscript	Formerly Javascript	UAPROF	User agent profile
EFI	Extended functionality interface	UDP	User datagram protocol
HTML	Hypertext markup language	vCal	Calendar interoperability standard
HTTP	Hypertext transfer protocol		by the Internet Mail Consortium,
iCal	Calendar interoperability standard		superseded by iCal
	established by the Internet Mail	vCard	Business card interoperability
	Consortium		standard by the Internet Mail
IETF	Internet Engineering Task Force		consortium
IP	Internet protocol	W3C	World Wide Web Consortium
IPv4	IP version 4	WAE	WAP application environment
IPv6	IP version 6	WAP	Wireless application protocol
MMS	Multimedia messaging service	WIM	Wireless identity module
OSI	Open systems interconnection	WML	Wireless markup language
ΟΤΑ	Over the air	WSP	Wireless session protocol
PKI	Public key infrastructure	WTA	Wireless telephony application
RDF	Resource description framework	WTLS	Wireless TLS
RSVP	Resource reservation protocol	WTP	Wireless transaction protocol
RTP	Real-time transport protocol	XHTML	Extensible HTML
SMTP	Simple mail transfer protocol	XML	Extensible markup language
	- Provinsi Presenter		· · · · · · · · · · · · · · · · · · ·

deed, in some countries, it is becoming difficult to find major websites that do not contain pages in the wireless markup language (WML), and new services are being launched daily.

Until now, WAP and i-mode have not been compatible. The markup in i-mode is based on an older version of HTML, whereas WAP is based on the new XML data format. The protocols used are totally different—for instance, i-mode uses optimized versions of the protocols used on the Web, and that does not fulfill the design constraints for WAP.

To make life easier for developers, NTT DoCoMo has been working with the WAP Forum to develop the new version of WAP, which will contain the best features from i-mode and WAP, while maintaining compatibility with future standards and the installed WAP base, and exploit features from the next generation of the World Wide Web (WWW). The result, which NTT DoCoMo claims they will use in coming versions of i-mode, is WAP 2.x (WAP Release 2001).

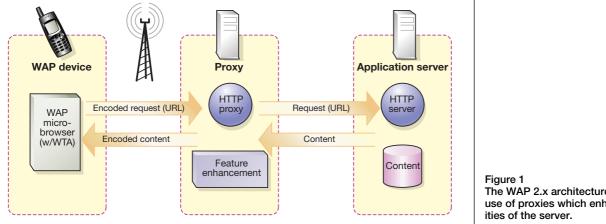
A goal of WAP 2.x initiatives has been to use the wireless application protocol to fully enable the mobile Internet while avoiding problems such as the "worldwide wait." The WAP Forum has become the most important source for collecting feedback on specifications for the mobile Internet and their implementation. As one of the founders, and chairing several important working groups, Ericsson is also playing a key role in the WAP Forum.

Note: Ericsson Review previously discussed hypertext transfer protocol (HTTP) and transport services in the WAP 1.x architecture. However, because the Internet Engineering Task Force (IETF) is still working with the wireless optimization of existing transport protocols, these topics are not discussed in this article.

WAP 2.x

The WAP 1.x architecture consisted of the origin server, gateway, and user-terminal environment. The server could be a WAP or HTTP server; the gateway translated the protocol layer and application information. By contrast, the WAP 2.x architecture consists of four conceptual components, namely the

- application environment;
- protocol framework;
- security services; and
- service discovery.



The WAP 2.x architecture does not have strict divisions between the server, gateway, and user-terminal environment. And there is no longer any intermingling between transport and service. Instead, functionswhich are accessed via the Internet-can be outsourced to capability servers in a WAP network that implements support for, say, a

- wireless telephony application (WTA);
- public-key infrastructure (PKI) portal;
- provisioning server; and

• user-agent profile (UAPROF) repository. Communication from WAP clients can take place directly with the server, but it will most likely take place through a proxy. Proxies are being established as one of the main points of control (for example, through firewalls) and as central points for resource interconnection. WAP clients support a proxy-selection mechanism that allows them to choose the most appropriate proxy for a specific task. This extends the current Internet proxy model.

The WAP 2.x protocol is compatible with WAP 1.x, but it relies more extensively on standard Web protocols (such as HTTP) and formats (such as XHTML). WAP 2.x also clearly separates the

- bearer (CSD, GPRS, IMT-2000);
- transport (WDP, TCP);
- session layer (cookies, CC/PP); and
- applications.

Most protocol services in the WAP 1.x suite are also available in new Web protocols. But the WAP push service cannot be realized through existing Web protocols without significant changes to the current Web architecture. Both the WAP 1.x stack and Internet protocols (such as hypertext and

multimedia transfer services) can provide some services, but only WAP is capable of providing others, such as the WAP push service.

From the start, WAP has been based on the browsing paradigm made popular by the Web: it adapts technologies from the Web, making them work better in wireless networks. In the WAP 2.x architecture, more of the Web technologies are adopted directly (as they are) rather than adapted, and then extended with WAP-specific functions. These functions enable service

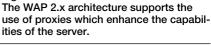
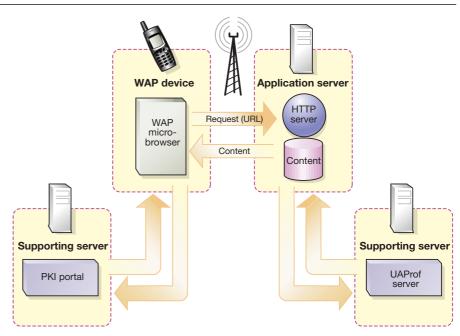


Figure 2

The WAP 2.x architecture also allows for supporting servers which can provide additional functions-for example, public key infrastructure.



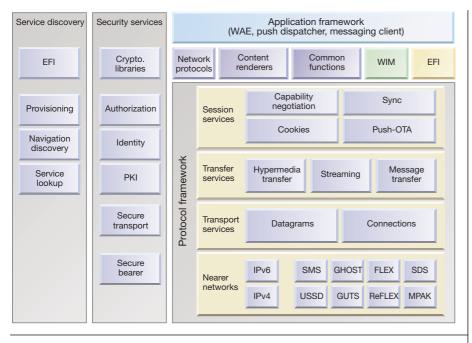


Figure 3

The WAP 2.x architecture is a layered architecture that includes different functions within its framework.

providers and developers of systems, content providers, and devices to provide users with greater added value.

Modularity is one of the main features of the second-generation architecture (modules interact through well-defined interfaces). The security-service and servicediscovery components of the architecture span every layer of the open systems interconnection (OSI) model. The application environment component resides on top of OSI layer 7; the protocol framework comprises everything from OSI layer 2 to 7.

The architecture allows components to interact. Developers can thus select modules from different components and create new user services. Conceivably, a minimal device can be developed by selecting components with the smallest footprints. In practice, devices and proxies will most likely implement either a dual stack or only the Internet stack. Backward compatibility is achieved by providing continued support for WML 1 in the client. The WAP conformance profiles (Box B) determine the configuration of the devices as well as how they work together.

The application environment component

The application-environment component enables the following services:

- WAP application environment (WAE) that is, the browser, calendar agent, and other user agents;
- user-agent profile;
- multimedia messaging and other data formats; and
- push service.

The application environment provides the user interface and other functions that display content. Because it is a flexible environment, modules can be added on an *ad hoc* basis (optional) or through the WAP Forum's specification development process.

BOX B, THE USER-AGENT PROFILE

The WAP Forum defined the term *user-agent* profile (UAPROF) for use within the composite capabilities and preferences profile (CC/PP), which is used to describe the capabilities of the user terminal application environment. The user-agent profile is a data format. A specific set of properties and values describes each terminal. The WAP Forum standardized the property names and values as part of the UAPROF vocabulary.

The CĆ/PP is defined in an XML framework—called the resource description framework (RDF)—which enables users to connect a property to an object (the CC/PP is an application of the RDF). The resource description framework can be used for annotations, meta-

data, and profiles that describe users or their terminals. By knowing the information display capabilities of a terminal, the server can create a display that is optimized for that terminal. Including profile information with the request minimizes the number of transactions needed to optimize the information, and it can be cached in a proxy or retrieved from a repository that the device manufacturer maintains. This minimizes the amount of information transmitted over the air and speeds up information access. Designers can create pages or page templates to be used with database servers (such as the Ericsson WAP application server), displaying them in formats that are adapted to user devices.

WAP application environment

The WAP application environment is in the mobile terminal. It contains a subset of XHTML (for display formatting) and a subset of the cascading style sheet (CSS) language (for content formatting). It also contains user agents for WTA and programming interfaces for use in mobile devices. WML and WMLscript execute in the WAP application environment.

Cascading style sheets

The WAP 2.x architecture contains a subset of the cascading style sheet language, which is the most widely used display language on the Web. Using cascading style sheets, an author can define how each element in a document is to be displayed. This gives authors greater control—compared to when the display is specified inside the markup. A style sheet need only be downloaded once from the network server. After that it can be retrieved from a local cache.

Cascading style sheets can adapt automatically to the capabilities declared by a device's user-agent profile. This is particularly important because display capabilities vary significantly among devices. A format that looks good on one device might be displayed differently on another. The useragent profile ensures that the device gets the most appropriate style sheet. And because style sheets separate display from content, authors can use the same WML document for many different devices with significantly different display capabilities.

Contact and calendar information

WAP 1.x versions contained the vCal and vCard data types, which are not part of the browsing environment. The Internet Mail Consortium standardized vCal and vCard as structured data types for displaying contact and calendar information. iCal, which was developed from vCal, is used in products such as Microsoft Outlook and Lotus Organizer. It is also used in Ericsson's AirCalendar, which allows users to synchronize the electronic calendars they use in the fixed environment with the calendars on their mobile terminals. WAP also accommodates other data types, such as audio and video.

Multimedia messaging

Many of the functions of the new WAP environment are available in existing Internet architectures, but the WAP 2.x application environment contains two modules that were developed in WAP 1.x. These two modules contain functions that are not available in other systems: multimedia messaging service (MMS) was one of the highlights of this year's GSM-UMTS Forum, and push services are not possible on the Web using standard HTTP. MMS is an e-mail-like mechanism for the transmission of multimedia messages (electronic postcards with sound), which are expected to become very popular applications, especially in thirdgeneration mobile systems.

Push services

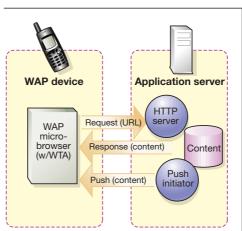
Service providers use push services to send information to users (who need not initiate any action). As simple as it might sound, the push-service architecture has been a major item on the WAP Forum agenda. In the WAP 2.x architecture, the push service has been divided into the user-agent module and the session-layer module.

The protocol framework component

The WAP application environment relies on a protocol framework component that enables the functions needed to provide the services described above. The protocol framework consists of four modular layers, which can be combined:

- the session service layer;
- the transfer service layer;
- the transport service layer; and
- the bearer service layer.

In traditional Internet environments, the protocol framework solely provides transport services for applications, such as hypermedia transport (HTTP), streaming



BOX C, XHTML

XHTML is HTML reformulated as an XML application, defined by the W3C. For WAP 2.0, the WAP Forum has defined a strict subset of XHTML called the *XHTML mobile profile*. The profile is in turn a superset of the W3C *XHTML basic profile*, a W3C recommendation (December 2000) for XHTML in small devices. An XHTML page, written in XHTML mobile profile or XHTML basic, can be viewed in a WAP 2.0 browser or any standard Web browser. All basic XHTML features are supported, including text, images, links, checkboxes, radio buttons, headings, horizontal rules, and lists.

Figure 4

The WAP 2.x architecture allows the application server and the client to connect directly for push and content responses.

(RSVP and RTP Internet protocols), and message transport (standard Internet protocols, such as SMTP). In the WAP architecture, a logical layer has been added: the session service layer.

Session service layer

In the WAP 2.x architecture, the session service layer, which resides between the transport layer and the application environment, brings several new services to applications. Sessions do not exist in HTTP, but cookies can provide sessions. Cookies, which are database markers included in the request and looked up on the server side to identify the user, are part of the WAP 2.x architecture. They enable the reuse of mechanisms that already exist in the Internet and solely give an indication about the relationship between a single server and the user agent. Cookies cannot be used as a general source of information.

The session service layer also includes a technology for reporting to the server. The reports contain information on terminal capabilities and on the terminal application environment. This information is used to optimize the display format.

Data synchronization

Synchronization is another new service in WAP. The WAP Forum has been working with SyncML (another industry consortium) to create a language for data synchronization. Synchronization of data that has been updated in mobile and fixed environments can be a thorny issue. Users retrieve data from the network and store it on a mobile device, which they use to access and manipulate the local copy of the data. Periodically, users reconnect to the network to send changes to the networked data repository. Users also have the opportunity to learn about updates made to the networked data while their terminal was offline. Occasionally, users need to resolve conflicts between their local updates and the networked data. This reconciliation operation (during which updates are exchanged and conflicts are resolved) is known as data synchronization. The data synchronization protocol synchronizes networked data with that on many different devices, including handheld computers, mobile phones, automotive computers, and desktop PCs.

Push sessions

The push solution also contains a session component. The push over-the-air (OTA) session service enables the establishment of push sessions

- across communication links that might not be persistent; and
- in instances when addresses are dynamically assigned.

There is no binding between the transport of data and the session on the Web. The data transport is transparent to the session. Once a hypertext transport transaction is finished, the state it created disappears. In the WAP 1.x stack, the wireless session protocol (WSP) and wireless transaction protocol (WTP) can be used in combination to create and maintain a state, and through it, sessions. This has several advantages (for example, to enable push). By including HTTP as a transport method, the WAP Forum now enables both stateful and stateless transport. Session services provide a "memory" of previous transactions (this feature does not exist in HTTP, since it is a stateless protocol) that enable the retention of terminal characteristics and make for faster initialization of complex transports (such as data streaming).

Apart from the transport of text documents, the WAP architecture has also been prepared for the next generation of messaging. It contains a multimedia transport mechanism for asynchronous message transport (messages are encapsulated for transmission between multimedia and WAP servers in a WAP-specific protocol). The data-transport mechanisms also include IETF data-streaming formats.

Transport service layer

The transport services in WAP 2.x are either datagrams (connectionless service) or connections. Datagrams are more efficient for services that are not dependent on a persistent connection. The datagrams comply with either the user datagram protocol

BOX D, ACQUIRING WAP CERTIFICATION

While some modules are mandatory, a device designer can select and implement desired modules and still satisfy prerequisites for WAP certification. A set of conformance profiles for WAP 2.x determines what a WAP device is, and which components it should make use of. This also helps users to make their choice of devices, since they can easily match the conformance profile with the functions they need.

WAP certification will be increasingly important, since many implementations of technologies developed in the IETF or W3C do not follow the specification, and in practice, implement only a minimal set of functions. This will not happen in a WAP environment whose rigorous certification process determines conformity standards. (UDP), which is used on the Internet, or the wireless datagram protocol, which was defined for the WAP 1.x architecture.

The connection-oriented aspects of the new architecture are handled by the transmission control protocol (TCP). TCP, however, does not work well over mobile networks, so the WAP Forum is discussing an optimized mobile profile, to enable the mobile terminal to function optimally over the mobile network with its special characteristics.

Bearer service layer

In WAP 2.x, the bearer services have been extended considerably. They now include the mobile radio bearers used to transport WAP (such as SMS, FLEX, USSD, and GUTS), as well as IP version 4 (IPv4) and IP version 6 (IPv6).

WAP can be transported over different networks, and mapping can be handled directly from the WAP stack to several bearer services. WAP 1.x contains several modules for bearer networks, some of which (broadcast networks, for example) could not be handled using TCP transport. In WAP 2.x, bearers can be managed by the IP stack or directly by the WAP datagram or connection service, which uses the Internet's transmission control protocol.

Security services component

The security services component is positioned orthogonally to data-transfer and data-use services within the protocol framework component.

Security on the Internet is currently a hot issue, and the WAP 1.x architecture has received a lot of criticism. The telecommunications industry has been a leader in the security area for a long time, and this experience has been transferred into the WAP 2.x architecture.

WAP security services span all layers of the WAP 2.x architecture, thus creating opportunities for users to set up extremely secure environments (in fact, much more so than what is currently possible on the Web). How? By combining application-layer, transfer-layer, transport-layer, and bearerlayer security—the possibilities are endless. Security services include

- mechanisms for signing and encrypting data as a WMLScript crypto library;
- authentication services;
- an identification service that uses the wireless identity module (WIM);
- a PKI system;
- transport layer security (TLS, previously called SSL); and
- WTLS, the WAP 1.x-adapted version of TLS.

Service discovery component

The service discovery component is another orthogonal component in the WAP 2.x architecture that embraces what is available on the Internet and extends it by adding mobile-specific components.

One example, the service lookup protocol, uses the existing domain name server (DNS) from the Internet. Terminal functionality is extended through the extended functionality interface (EFI), which enables a WAP device to have external entities attached to it (thermometers, pressure gauges, and so on).

Provisioning, which is another telecommunications-specific protocol, is translated into WAP. Devices can thus be provided with all the parameters they need to function through the network.

Navigation discovery allows a client to discover services in the network—for example, a client might need to find a proxy in order to download data.

Conclusion

The WAP environment is a consistent architecture composed of standard components taken from the Internet where available, or constructed separately. They enable a consistent application environment for the mobile system. Thanks to the certification process of the WAP Forum, interoperability is also ensured for all the different functions of the WAP 2.x architecture. Using WAP, the developer gets a well-known development environment, and does not have to create several different versions of the application for mutually incompatible environments.

TRADEMARKS

Microsoft and Microsoft Outlook are registered trademarks of Microsoft Corporation. Lotus Organizer is a registered trademark of Lotus Development Corporation.