

**Mobile Services and Radio Systems Division (DMR)**

**Mobile Service Laboratory (SMO)**

**T&I/FTR&D/DMR/SMO/03.047/TV**

Version 1.2 – 25 May 2004

# **MIDP 2.0 versus WAP 2.0**

*Diffusion of this document needs to be submitted to France Telecom R&D for approval*

*The information in this document is based on information available in January 2003 and has only been updated with information available on Orange World in October 2003.*

## Author

Thomas Vergouwen	TI/FTR&D/ DMR/SMO/TOM	<a href="mailto:thomas.vergouwen@francetelecom.com">thomas.vergouwen@francetelecom.com</a>	+33 1 45 29 47 72
------------------	--------------------------	--	-------------------

## Document history


	§	Modifications
--	---	---------------


<b>Date</b>	28/01/03	all	Creation
<b>Version</b>	1.0		
<b>Author</b>	TV		

<b>Date</b>	29/01/03	4.1	Correction of security-related information after feedback from P. Calvet
<b>Version</b>	1.1		
<b>Author</b>	TV		

<b>Date</b>	25/05/04	all	Removal of all confidential information (without modification of contents) Addition of information on Orange World offer
<b>Version</b>	1.2		
<b>Author</b>	TV		

## Contents

<b>1</b>	<b>Introduction .....</b>	<b>5</b>
1.1	Background of this document.....	5
1.2	MIDP 2.0, a short introduction.....	5
1.3	WAP 2.0, a short introduction .....	7
<b>2</b>	<b>What is the use of mobile services? .....</b>	<b>8</b>
2.1	What is a service?.....	8
2.2	Usage of mobile services .....	8
2.3	Use cases of mobile services.....	9
<b>3</b>	<b>Two technologies enabling two services.....</b>	<b>10</b>
3.1	Distance of the content to the end user .....	10
3.2	Placement of the content .....	14
3.3	User interface level .....	14
<b>4</b>	<b>Two technologies enabling content .....</b>	<b>15</b>
4.1	New functionalities in MIDP 2.0.....	15
4.1.1	New classes for networking support (io) .....	16
4.1.2	New classes for the user interface .....	18
4.1.3	New packages for media support (media and media.control) .....	19
4.1.4	New package for secure connections using Public Key Infrastructure (pki) .....	20
4.1.5	Other new functionalities .....	20
4.2	New functionalities in WAP 2.0.....	23
4.2.1	Navigation .....	24
4.2.2	Configuration and adaptation .....	25
4.2.3	Push and messaging .....	26
4.2.4	Security .....	27
4.2.5	Miscellaneous.....	28
<b>5</b>	<b>Mapping content on technologies.....</b>	<b>30</b>
5.1	Different types of content .....	30
5.2	Types of content mapped on technologies.....	30
5.2.1	Informational content .....	30
5.2.2	Entertainment related content .....	31
5.2.3	Conclusion .....	31
<b>6</b>	<b>Use cases .....</b>	<b>33</b>
6.1	Orange World  .....	33
6.1.1	Description .....	33
6.1.2	Pricing .....	33
6.1.3	Orange World services .....	34
6.1.4	MMS .....	36
6.1.5	Technical implementation.....	36
6.1.6	Analysis .....	36

6.2	Vodafone live! 	38
6.2.1	Description .....	38
6.2.2	Pricing .....	38
6.2.3	Vodafone live! services.....	39
6.2.4	Picture messaging.....	42
6.2.5	Technical implementation.....	42
6.2.6	Analysis.....	44
6.3	Japan.....	45
6.3.1	Operators and handset manufacturers.....	45
6.3.2	Content providers .....	46
6.3.3	Available services.....	46
6.3.4	Java introduction .....	47
6.3.5	Java standards .....	49
6.3.6	Access to Java services & contents .....	49
<b>7</b>	<b>Conclusion .....</b>	<b>51</b>
	<b>References.....</b>	<b>52</b>

## 1 Introduction

### 1.1 Background of this document

The initial question presented in this document is the question for the differences and similarities between MIDP 2.0 and WAP 2.0. That's why this document first presents a short introduction of the two technologies, followed by an analysis of the two different services that can be provided by using these two technologies from an operator's point of view (i.e. access services and not content services). The chapter that analyses the content that can be created by using these two technologies and their corresponding access services is followed by a chapter which tries to map different types of content on the two different technologies from a global point of view. This chapter, in combination with the use cases of Orange World, Vodafone live! and the Japanese situation, shows that the question behind the initial question described above is not the most basic question to be studied regarding these two technologies. The basic question is in fact whether the two technologies exclude each other or whether they are supplementary technologies which together form a stronger combination than the addition of the two separate technologies. As this document will show, MIDP 2.0 and WAP 2.0 do not compete but reinforce each other.

### 1.2 MIDP 2.0, a short introduction

**MIDP 2.0 is the latest version of the MIDP profile, formerly known as MIDP NG** (NG for next generation). MIDP 2.0 was created by a working group including the major manufacturers (Nokia, Motorola, Siemens, ...), network operators (Vodafone, Orange, ...) and some platform vendors (Aplix, 4<sup>th</sup> Path, ...). The working group is known as JSR 118 and is part of the Java Community Process (JCP), the forum in charge of defining Java standards.

MIDP 1.0 established a standard Java environment for small devices with wireless network connectivity. MIDP 2.0 defines the core of the next generation of MIDP. MIDP 2.0 expands considerably on the original specification with far-reaching support for advanced user interfaces, multimedia, secure HTTP networking, and many other useful features.

MIDP 2.0 allows the addition of extensions, such as the wireless messaging package, which offers SMS functionality. Some people may therefore refer to MIDP NG as the next generation of MIDP including those packages.

The figure below shows the standardisation work on MIDP. At the moment the scope of the MIDP 2.0 group was defined, functionalities like wireless messaging (SMS) and multimedia were not fully covered. Therefore, new working groups were necessary to create those extensions.

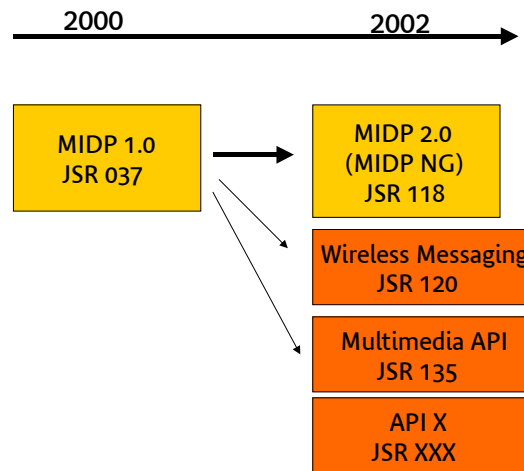


Figure 1: Standardisation work on MIDP

A typical MIDP NG phone could thus have the following software structure regarding Java. The phone is composed of 4 layers in figure 2.

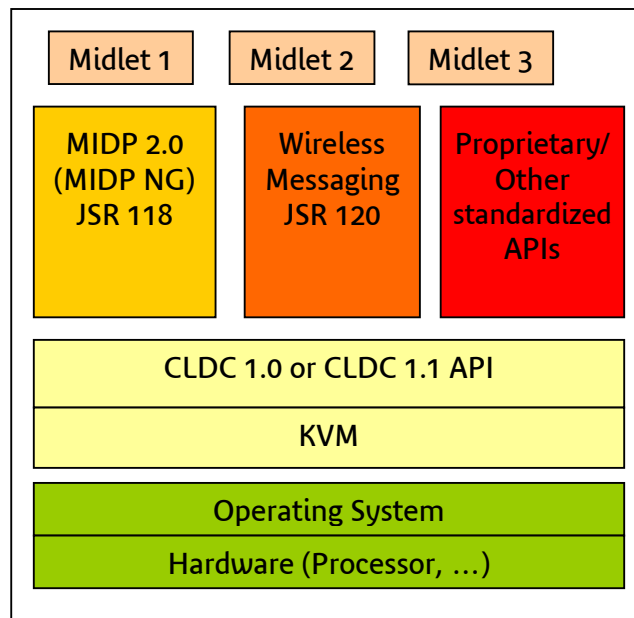


Figure 2: Software structure typical MIDP NG telephone

The first layer is the phone hardware including the processor and the operating system. The operating system is in charge of managing the relationship with the different peripherals: screen, processor or memory.

The second layer is composed of the virtual machine itself and the core Java libraries. The Connected Limited Device Configuration (CLDC) defines the Java configuration of the device: which part of the Java language is used, the features of the virtual machine, the core platform libraries, security and networking features.

MIDP NG phones can either be based on CLDC 1.0 or on CLDC 1.1. The virtual machine (KVM) is a software program running on top of the hardware processor.

The third layer is composed of the MIDP profile. The profile defines the additional set of APIs and features for the mobile phone market. It includes application installation and life cycle management, user interface functionality, event handling and a high-level application model. The MIDP profile offers the necessary libraries to write a number of applications, while other libraries can be added for any missing features.

The last layer is the application layer. Applications are written using the different libraries: MIDP 2.0, wireless messaging (SMS), and proprietary or other standardized extensions.

### **1.3 WAP 2.0, a short introduction**

The Wireless Application Protocol (WAP) is an open, global specification that empowers mobile users with wireless devices to easily access to and interact with information and services instantly. WAP is a protocol defined by the WAP Forum, which was originally created in 1998 by Nokia, Ericsson, Motorola and Unwired Planet (Openwave). The first version of the WAP protocol (1.0) was released in April 1998. The last version (2.0), which follows three intermediate 1.x versions, was published in January 2002.

The main differences between WAP 2.0 and the last 1.x version (1.2.1) are the introduction of a new language (XHTML with CSS support), the support of cookies, pictograms, WAP provisioning, MMS, end to end security, support for certificates for access to secure sites (PKI), an interface to external devices and the support of SyncML. This document mainly focuses on the new language, because the impact of this element is the largest to the user.

## 2 What is the use of mobile services?

In order to be able to compare the two different technologies, WAP 2.0 and MIDP 2.0, which can be used to offer mobile services, we should verify what is the need of mobile services: why are we using mobile services and in which situation?

### 2.1 What is a service?

First of all, the abstract term *service* should be defined in order to be able to do any statements on the use of mobile services. In this document we will use the following definition of a mobile service: *a mobile service to an end user is a service offered by the operator; this comprises any service offered by a carrier like voice, WAP, i-mode, i-appli, ...* This must not be confused with a service offered by a *service provider* such as a service offering the programme of the nearest cinema. This kind of service is called *content* in this document.

### 2.2 Usage of mobile services

Three major factors drive the usage of mobile services: *personal relevance*, *time* and *place*. People want to have access to information with a personal relevance, for example the share price of *my* investments, the result of the match of *my* favourite team, et cetera. This information should be up-to-date and available on the users' demand, the factor time is therefore extremely important: 'I want to know when the *next* bus is coming' (time) or 'I wonder where is the nearest McDonald's' (time and place). Finally, a factor driving the use of a mobile service is place: the information the user can retrieve by using the service should be adapted to his current location. For example: 'I wonder where is the *nearest* McDonald's' or 'I wonder *where* I can get *local* currency'.

These three factors can be summarised by expressing the need of the user in ***I want it here and now*** and become evident by looking at the following picture, which defines the need of a mobile service.





Figure 3: Three factors driving the need of mobile services

### 2.3 Use cases of mobile services

By looking at the three factors driving the need for mobile services we can identify two basic types of services demanded by users:

- ⌚ Services for *time to kill*: when a user has some spare time, he almost always has his mobile phone with him and wants to use this mobile phone (time: time to kill, place: anywhere, personal interest: entertainment)
- 📄 Services for *information needs*: when a user is looking for information on a specific subject, he wants to have access to this information (personal interest) directly (time), at the place where he is (place)

### **3 Two technologies enabling two services**

Both WAP 2.0 and MIDP 2.0 are technologies enabling an access service provided by a mobile operator. However, these two technologies offer a different level of end user experience. This chapter discusses several issues regarding the level of end user experience.

#### **3.1 Distance of the content to the end user**

The first difference between the two access services that can be offered by using these two technologies is the distance of the content offered by the service to the end user. In the context of this document the distance to the end user can be defined as the number of actions the user has to perform on his mobile terminal to access the content using the access service.

When using WAP 2.0, the user has to select the WAP browser in the menu of his mobile terminal (first action) and then launch the browser by selecting a specific address or using the homepage (second action). When the user arrives at the page that corresponds to the address he has chosen, this page normally shows a list of content services available. From this list, the user clicks (arborescence structure) until he can choose the content service he wants (third action + n actions) ...

Using MIDP 2.0, the user once downloads the Java application he wants to use and from that moment on the access to the application consists of two steps: he opens the applications menu on his mobile terminal (first action) and then selects the application he wants to use (second action).

The difference in the distance of the content to the end user for NTT DoCoMo's browsing (i-mode) and active services (i-appli) is shown in figure 2. The same situation exists for browsing (WAP 1.x/2.0) and active services (MIDP 1.0/2.0) in Europe.

### Browsing services



Standby screen

### Active services



Standby screen



Access to the menu



Access to the menu



Choice of the access service  
(i-mode)



Choice of the access service  
(i-appli)



Choice of the content to see  
(bookmarks / homepage)  
(here i-menu = portal)



Choice of the application  
(list of downloaded applications)



The portal  
 (list of available content)



The requested content  
 (the application)



A section of the portal  
 (informational content)

Choice of the specific content in  
 arborescence (n steps)

*Figure 4: Distance of the content to the end user (browsing / active services)*

### **3.2 Placement of the content**

When using a service that can be provided using the WAP 2.0 technology, the content will (almost) always be online on a WAP server and it can be accessed using a WAP browser.

A service that can be provided using the MIDP 2.0 technology can bring the content to the user: after download, the application will be available on the user's mobile device, making it accessible anytime, anywhere. One of the advantages of the placement on the user's side is for example that an application can run as a background task and perform regular tasks using timers or that information can be pushed to the user's application.

### **3.3 User interface level**

In WAP 2.0 the user interface level depends heavily on two aspects: the capacities of the mobile terminal and the design used by the developer of the content. Basically the design has to be adapted to the capacities and capabilities of the mobile terminal for every terminal being used to access the content. Using WAP 2.0, it's hardly impossible to access user input keys directly.

In MIDP 2.0 two application programming interfaces (API) are available offering two different levels of user interfaces:

- High-level API
- Low-level API

The high-level API requires the developer to use task-oriented abstractions to define what the user interface does and gives the developer no real control over what gets drawn on the screen: the implementation selects the best approach for the device. The low-level API, which is mainly aimed at game developers, gives the developer complete access to the screen and to input events. The drawback for the developer is that he is then responsible for drawing everything that is shown on the screen.

The two API's can be used in the same application, but not at the same time.

When using the high-level API, the content will be shown correctly on any device supporting MIDP 2.0. When using the low-level API, the application should in most cases specifically be adapted to the mobile device on which it's being used in order to be shown correctly. An advantage of the low-level API is that it gives the developer the possibility to precisely place and control graphic elements, to access low-level input events and possibly even to access special, device-specific features. However, using the low-level API, applications are – in contrast to applications using the high-level API – not guaranteed to be portable.



## 4 Two technologies enabling content

Both technologies, MIDP 2.0 and WAP 2.0, offer several functionalities, which enable the creation of content, such as the storage of information, interaction functionalities, et cetera. The functionalities that are new in MIDP 2.0 (compared to MIDP 1.0) and the functionalities that are new in WAP 2.0 (compared to WAP 1.2) and the possibilities they offer for enabling the access to content are discussed here, but this chapter does not offer a complete list of functionalities offered by both technologies.

### 4.1 New functionalities in MIDP 2.0

MIDP 2.0 offers its functionalities – as MIDP 1.0 – through an *Application Programming Interface* or *API*. The MIDP 2.0 specification is based on the MIDP 1.0 specification and provides backward compatibility with MIDP 1.0 so that MIDlets written for MIDP 1.0 can execute in MIDP 2.0 environments. In the API for MIDP 2.0 the different functionalities are divided in packages (packages which are completely new in MIDP 2.0 are marked in red; packages marked in orange contain classes which are new in MIDP 2.0; packages marked in green have not changed since MIDP 1.0):

- ⊙ The MID Profile includes networking support ([javax.microedition.io](http://javax.microedition.io)) based on the Generic Connection framework from the *Connected, Limited Device Configuration* (CLDC)
- ⊙ The UI API ([javax.microedition.lcdui](http://javax.microedition.lcdui)) provides a set of features for implementation of user interfaces for MIDP applications
- ⊙ The Game API packages ([javax.microedition.lcdui.game](http://javax.microedition.lcdui.game)) provides a series of classes that enable the development of rich gaming content for wireless devices
- ⊙ The MIDP 2.0 Media API ([javax.microedition.media](http://javax.microedition.media)) is a directly compatible building block of the Mobile Media API specification
- ⊙ The Media Control package ([javax.microedition.media.control](http://javax.microedition.media.control)) defines the specific Control types that can be used with a Player
- ⊙ The MIDlet package ([javax.microedition.midlet](http://javax.microedition.midlet)) defines Mobile Information Device Profile applications and the interactions between the application and the environment in which the application runs
- ⊙ Certificates are used to authenticate information for secure Connections: Public Key Infrastructure (PKI; [javax.microedition.pki](http://javax.microedition.pki))
- ⊙ The Mobile Information Device Profile provides a mechanism for persistently store data and later retrieve it: the RMS ([javax.microedition.rms](http://javax.microedition.rms))
- ⊙ The two core packages:
  - ⊙ MID Profile Language Classes ([java.lang](http://java.lang)) included from *Java 2 Standard Edition* (J2SE)
  - ⊙ MID Profile Utility Classes ([java.util](http://java.util)) included from J2SE

*In the following sections, only new classes (or completely new packages) and the possibilities they offer are discussed. When new methods have been added in MIDP 2.0 to a class that has already been introduced in MIDP 1.0, these methods are not mentioned here.*

#### 4.1.1 New classes for networking support (io)

In MIDP 2.0 several new interface classes have been introduced for networking support. These interface classes and possible use cases of these classes are discussed here.

Network services offered by the MIDP 2.0 platform are:

- ⊗ Local network connections
- ⊗ Remote network connections
  - ⊗ http
  - ⊗ tcp, udp
- ⊗ Secure remote network connections
  - ⊗ https
  - ⊗ ssl
- ⊗ Wireless telephony
  - ⊗ Voice call initiation
  - ⊗ WAP browser session initiation.

Finally, a push mechanism that can be used with network interfaces as well as other uses like alarm-based push completes the network services of MIDP 2.0.

##### 4.1.1.1 Local network connections

The first new class, *CommConnection*, defines a logical serial port connection, which is defined as *a logical connection through which bytes are transferring serially*. It can therefore correspond to a classic RS-232 serial port, but it can also be an infrared port, which has been configured as a logical serial port.

Possible applications of this new class are for example the interaction of a MIDlet with a digital camera attached to the phone or with another device connected to the phone via a serial cable.

##### 4.1.1.2 Remote network connections

MIDP 2.0 adds socket and datagram support to the high-level network layer HTTP already defined in MIDP 1.0. Three pretty open remote networking classes have been defined in the MIDP 2.0 networking support package: *ServerSocketConnection*, which defines the inbound server socket stream connection, *SocketConnection*, which defines the outbound socket stream connection and *UDPDatagramConnection*, which defines a datagram connection which knows its local end point address (transaction oriented without delivery and duplicate protection guarantees). Applications requiring ordered reliable delivery of streams of data should use the *SocketConnection*.

Due to the openness of these networking classes, these can be used for all kinds of networking needs. An example could be an application that uses a proprietary networking protocol that does not work on HTTP.

##### 4.1.1.3 Secure remote network connections

The first new class for secure remote networking support in MIDP 2.0 is the *HttpsConnection* interface, which defines the necessary methods and constants to establish a secure network connection (HTTP over TLS, SSL V3, WTLS or WAP TLS Profile and Tunneling Specification).

Another new class related to secure networking is *SecureConnection*, which defines the secure socket stream connection (implemented by TLS Protocol Version 1.0, SSL V3 and / or WAP TLS Profile and Tunneling Specification).



The third new networking class related to secure networking is *SecurityInfo*: this interface defines methods to access information about a secure network connection. Protocols that implement secure connections may use this interface to report the security parameters of the connection (for example certificate, protocol, version and cipher suite, etc.).

Possible uses of these secured networking support classes include all cases wherein an application has to send for example personal or confidential information to a remote device (for example transactional information for payments).

#### **4.1.1.4 Wireless telephony**

MIDP 2.0 covers some phone specific functions. An application can open a WAP page in a WAP browser or initiate a voice call. The approach is very similar to the WAP WTA, Wireless Telephony Application.

For this purpose, a new method was added to the MIDlet class in the Application Lifecycle Package. The method is called `MIDlet.platformRequest(String URL)`. The method requests that the device handles the indicated URL, Uniform Resource Locator. The URL can either be a WAP address, a telephone number or another resource.

##### **🕒 WAP browser**

If the URL string is a WAP address, then the WAP browser will be launched to open the page available at this address. For example, the execution of the code:  
`MIDlet.platformRequest("http://wap.orange.fr");` starts the browser on the Orange France WAP portal.

Depending on the phone capabilities, the application remains in the background or will be shutdown and the browser takes over the display.

##### **🕒 Voice call**

If the URL specified is of the form `tel:<number>`, as specified in RFC2806, a voice call will be initiated. For example, the execution of the code:  
`MIDlet.platformRequest("tel:+33145294444");` will make a telephone call to FTR&D.

##### **🕒 Other uses: content/application installation or update**

If the URL specified refers to a MIDlet suite (either an Application Descriptor or a JAR file), the application handling the request will install the named package. In this case, the platform's normal MIDlet suite installation process should be used and the user must be allowed to control the process (including cancellation of download and/or installation). If the MIDlet suite being installed is an *update* of the currently running MIDlet suite, the platform must first stop the currently running MIDlet suite before performing the update.

Devices may choose to support additional URL schemes beyond those explained above.

#### **4.1.1.5 Push services**

Finally, another new class for I/O operations is the *PushRegistry*, which maintains a list of inbound connections. This registry allows an application to declare a listener for an incoming connection on which information can be pushed to the application.

A possible use of this class in an application can be the automatic update of information used in the application (for example an update of the train schedule), even when the application is not running at the moment the update is delivered (the MIDlet will then be started automatically by the Application Management Software [AMS] of the device) or a direct push of information to a user by opening the application automatically.

#### 4.1.2 New classes for the user interface

##### 4.1.2.1 General purpose user interface (lcdui)

Four new classes related to the user interface have been introduced in MIDP 2.0 in *javax.microedition.lcdui*. The first one, *CustomItem*, is a class that defines a subclass of the class *Item* that already existed in MIDP 1.0. The class *CustomItem* is customizable and can therefore be used to introduce new visual and interactive elements into *Forms*.

With this class application developers can introduce *Items* with a new visual representation and interactive elements in a user interaction form in an application.

Another new class for the user interface is the class *ItemCommandListener*, a listener type for receiving notification of commands that have been invoked on *Item* objects. An *Item* can have *Commands* associated with it. When such a command is invoked, the application is notified by having the *commandAction()* method called on the *ItemCommandListener* that has been set on the *Item* with a call set to *setItemCommandListener*.

The third new class, *Spacer*, is a blank, non-interactive item that has a settable minimum size. The minimum width is useful for allocating flexible amounts of space between *Items* within the same row of a *Form*. The minimum height is useful for enforcing a particular minimum height of a row. The application can set the minimum width or height to any non-negative value. The implementation may enforce implementation-defined maximum values for the minimum width and height. Since a *Spacer*'s primary purpose is to position other items, it is restricted to be non-interactive, and the application is not allowed to add *Commands* to a *Spacer*.

Finally, the class *StringItem* is an item that can contain a string. A *StringItem* is display-only; the user cannot edit the contents. A possible application of this new class can be to display the status of the application at a given location.

The new classes for the user interface offer a more sophisticated layout of forms used in MIDP applications.

##### 4.1.2.2 Game package (lcdui.game)

The game API package (*javax.microedition.lcdui.game*) is a completely new package in MIDP 2.0. It provides a series of classes that enable the development of rich gaming content for wireless devices. This API contains five classes:

- *GameCanvas* – this is a subclass of the class *Canvas* defined in *javax.microedition.lcdui* and it provides the basic 'screen' functionality for a game. In addition to the methods inherited from *Canvas*, this class specifically provides game-centric features such as the ability to retrieve the current state of the game keys and synchronous

graphic flushing. These features simplify game development and improve performance.

- ⌚ *Layer* – A class that represents a visual element in a game such as a *Sprite* or a *TiledLayer*. This abstract class forms the basis for the Layer framework and provides basic attributes such as location, size and visibility.
- ⌚ *LayerManager* – For games that employ several *Layers*, the *LayerManager* simplifies game development by automating the rendering process. It allows the developer to set a 'view window' that represents the user's view of the game. The *LayerManager* automatically renders the game's *Layers* to implement the desired view.
- ⌚ *Sprite* – A *Sprite* is a basic animated *Layer* that can display one of several graphical frames. The frames are all of equal size and are provided by a single *Image* object. In addition to animating the frames sequentially, a custom sequence can also be set to animate the frames in an arbitrary manner. Furthermore, the *Sprite* class provides various transformations (flip and rotation) and collision detection methods that simplify the implementation of a game's logic.
- ⌚ *TiledLayer* – This class enables a developer to create large areas of graphical content without the resource usage that a large *Image* object would require. It is comprised of a grid of cells and each cell can display one of several tiles that are provided by a single *Image* object. Cells can also be filled with animated tiles whose corresponding pixel data can be changed very rapidly; this feature is very useful for animating large groups of cells such as areas of water in a game.

The Game API uses the standard low-level graphics classes from MIDP (*Graphics*, *Image*, etc.) so that the high-level Game API classes can be used in conjunction with graphics primitives. For example, it would be possible to render a complex background using the Game API and then render something on top of it using graphics primitives such as *drawLine*, etc.

#### 4.1.3 New packages for media support (media and media.control)

The *media* package is a completely new package introduced with MIDP 2.0. It contains two sets of APIs: J2ME devices range from cell phones with simple tone generation to PDAs and Web tablets with advanced audio and video rendering capabilities. To accommodate diverse configurations and multimedia processing capabilities, an API with a high level of abstraction is needed. Two API sets have been defined:

- ⌚ Mobile Media API – intended for J2ME devices with advanced sound and multimedia capabilities, including powerful mobile phones, PDAs and set-top boxes, et cetera.
- ⌚ MIDP 2.0 Media API – intended for resource-constrained devices, such as mass-market mobile devices.

In MIDP 2.0, only the audio-only media package (MIDP 2.0 Media API) is supported.

The proposed audio building block (MIDP 2.0 Media API) consists of three main parts:

- ⌚ A *Manager*: the top level controller for audio resources. Applications use the *Manager* to request *Players* and to retrieve properties, supported content types and supported controls. The *Manager* also includes a method to play simple tones.
- ⌚ A *Player* that plays the multimedia content. An application obtains a *Player* by giving the locator string to the *Manager*.

- ② *Control* is an interface that is used to implement all different controls a *Player* might have. An application can query a *Player* of controls it supports and then ask for a specific *Control* (for example *VolumeControl* to control the volume).

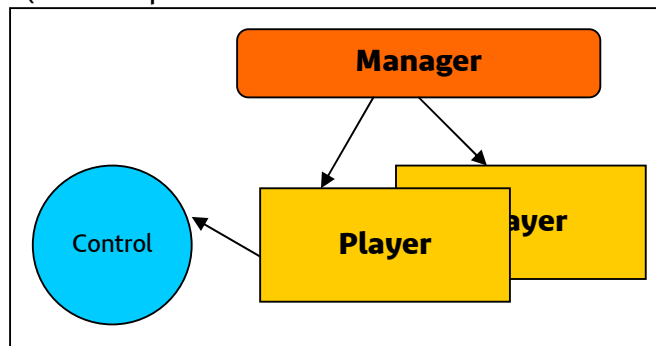


Figure 5: Interaction of the different elements of the audio building block

The possibilities offered by these two new packages (media and media control) are the use of sound in any application and on any device. So far, sound support depended heavily on the device used, but with these packages this interoperability problem has been solved. The MIDP 2.0 Media API supports playing simple tones, sequences of tones and the playback of sampled audio (at least WAV format).

#### 4.1.4 New package for secure connections using Public Key Infrastructure (pki)

In MIDP 2.0 the support for a Public Key Infrastructure (PKI) has been added in order to support the authentication of information for secure connections. The certificate format that must be supported by MIDP 2.0 compatible devices is the X.509 Certificate defined by the Open Mobile Alliance (WAP Forum)

The PKI package of MIDP 2.0 (*javax.microedition.pki*) contains two classes:

- ② The *Certificate* interface, with features such as subject, issuer, type, version, serial number, signing algorithm and dates of valid use for Certificates.
- ② The *CertificateException*: a class that encapsulates any error that occurs while using a *Certificate*.

#### 4.1.5 Other new functionalities

##### 4.1.5.1 Over The Air User Initiated Provisioning

Besides the new packages and classes that are introduced for MIDP 2.0, the MIDP 2.0 specification now integrated the *Over The Air User Initiated Provision Recommended Practice*, which allows the download of MIDlets over-the-air (for example using a WAP browser for application discovery, followed by download and installation of a MIDlet found). The changes made to this recommended practice are very limited and do not really change the result for the user of the technology, so it's not discussed in detail here.

##### 4.1.5.2 Security for MIDP applications

Using PKI, MIDlet Suites can be identified as *trusted* or *untrusted*. An *untrusted* MIDlet Suite is a MIDlet Suite for which the origin and the integrity of the JAR file can NOT be trusted by the device. Untrusted MIDlet Suites execute in the untrusted domain using a restricted environment where access to protected APIs or functions either is not allowed or is allowed with explicit user permission.

The untrusted domain for untrusted MIDlet Suites must allow, without explicit confirmation by the user, access to:

API	Description
<code>javax.microedition.lcdui</code>	User Interface APIs
<code>javax.microedition.lcdui.game</code>	Game APIs
<code>javax.microedition.media</code> <code>javax.microedition.media.control</code>	Multi-media APIs for playback of sound
<code>javax.microedition.midlet</code>	MIDlet Lifecycle APIs
<code>javax.microedition.rms</code>	RMS APIs

Table 1: APIs directly accessible for untrusted MIDlet Suites

Furthermore, the untrusted domain for untrusted MIDlet Suites must allow, with explicit confirmation by the user, access to protected APIs or functions:

Function	Protocol
<code>javax.microedition.io.HttpConnection</code>	http
<code>javax.microedition.io.HttpsConnection</code>	HTTPS

Table 2: Additional APIs accessible after user confirmation for untrusted MIDlet Suites

Security for *trusted* MIDlet Suites is based on protection domains. Each protection domain defines the permissions that may be granted to a MIDlet Suite in that domain. The protection domain owner specifies how the device identifies and verifies that it can trust a MIDlet Suite and bind it to a protection domain with the permissions that authorize access to protected APIs or functions.

A protection domain defines a set of permissions and related interaction modes. A protection domain consists of:

- ② A set of permissions that should be allowed (*Allowed*) – any permissions which explicitly allow access to a given protected API or function on the basis of the MIDlet Suite being associated with the protection domain. These permissions do not require any user interaction.
- ② A set of permissions that the user may authorize (*User*), each with its user interaction mode (*blanket* – valid for every invocation of the API/function by the MIDlet Suite until it is uninstalled or the permission is changed by the user, *session* – valid from the invocation of the MIDlet Suite until it terminates or *oneshot* – the user must be prompted on each invocation of the protected API/function) – any permissions for a protected API or function on the basis of the MIDlet Suite being associated with protection domain: access to the protected API or function will be allowed after the prompt given to the user and explicit user permission being granted.

A MIDlet Suite that requires access to protected APIs or functions must request the corresponding permissions. Permissions requested can be required by listing them in the attribute *MIDlet-Permissions*. These permissions are critical to the function of the MIDlet Suite and it will not operate correctly without them. If the MIDlet Suite can function correctly with or without particular permissions it should request them using the *MIDlet-Permissions-Opt* attribute. The MIDlet Suite is able to run with reduced functionality (for example as a single player game instead of an online multi-player

game) without these non-critical permissions and can be installed and run even when these permissions are not granted.

## 4.2 New functionalities in WAP 2.0

The table below shows the functionalities offered by the three latest versions of WAP: 1.1, 1.2 and 2.0. The new functionalities of WAP 2.0 will be discussed in detail in this paragraph.

	Associated technologies	WAP 1.1	WAP 1.2	WAP 2.0
<b>Navigation</b>				
Navigation in a WML / WMLScript site	Protocol stack (WDP, WSP, WTP) + WML	x	x	x
Navigation in an XHTML site	TCP/IP + HTTP + WML2			x
Unified worldwide representation of pictograms	Pictogram			x
<b>Configuration and adaptation</b>				
Content adaptation based on terminal and network used	UAProf		x	x
Automatic terminal configuration by a server	WAP Provisioning			x
<b>Push and messaging</b>				
Asynchronous notification (e.g. arrival of an e-mail)	WAP push SI		x	x
Sending a WAP page for future execution	WAP push SL		x	x
Multimedia messaging	MMS			x
<b>Security</b>				
Authentication for a secure access to certain pages with a login / password	WTLS	x	x	x
Separate memory card for the storage of certificates	WIM		x	x
End to end security without an intermediate gateway	End to end security			x
Usage of certificates for the access to certain secured sites	WAP PKI			x
<b>Miscellaneous</b>				
Access to telephony functions from within a WAP page	WTA		x	x
Usage of external devices (camera, GPS device, card reader, etc.) from within a WAP page	External Functional Interface			x
Data synchronisation	SyncML			x

Table 3: Functionalities available in WAP 1.1, 1.2 and 2.0

For the end user the most visible additional functionalities in WAP 2.0 are the multimedia messaging (MMS), the provisioning which should ease the configuration of



the terminal, WAP push which allows the notification of the user in order to invite him to visit a specific site and the applications which use external devices (photo camera, GPS device, card reader, MP3 reader, etc.). The other new functionalities mainly make the life of a developer easier, but will not directly be visible for the user.

In this paragraph, only the new functionalities that change the way the content can be shown will be discussed. Modifications such as the use of a direct connection without using a WAP gateway / proxy will therefore not be discussed in detail here.

#### 4.2.1 Navigation

The Wireless Application Environment (WAE) basically defines the specifications of the WAP browser. The most important modifications to these specifications in WAP 2.0 are:

##### ② Language

The evolution of the presentation language from WML to XHTML, the new presentation standard defined by the W3C that should follow up HTML 4.0. The WAP Forum continues to use the name WML by calling the new language WML2.

WML2 is the addition of the XHTML Mobile Profile (XHTML MP) and some WAP extensions. The XHTML MP is XHTML Basic plus some additional XHTML modules (Style Sheet/Attribute, Forms, Legacy and Presentation). The WAP extensions in WML2 are elements that existed in WML, but don't exist in XHTML MP (for example card, setvar, timer, etc) and are used for backwards compatibility only. These elements are represented using a wml namespace (i.e. wml:card, wml:setvar, wml:timer, etc). The support of WML2 in WAP 2.0 is optional and it should not be used to create new applications (XHTML MP should be used). The language structure in WAP 2.0 is depicted in figure 6 below.

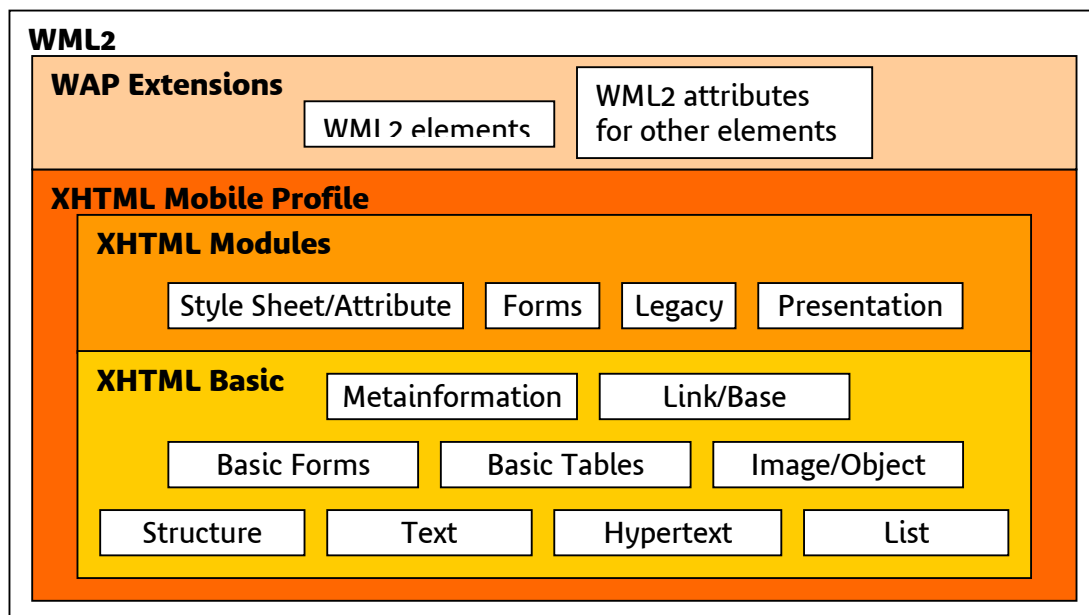


Figure 6: Language structure in WAP 2.0

This new language structure allows, together with the style sheets discussed later, the convergence to web standards (the language for WAP becomes true XML), to use



XHTML-compatible editing tools and a clear distinction between the content (in XHTML) and the presentation of this content (in the CSS).

#### ➤ **Style sheets**

WAP 2.0 introduces cascading style sheets (CSS) for the presentation of the content, which allows a clear distinction between the content and its presentation. The style sheets defined in WAP 2.0 are a subset of the CSS2 specification defined by the W3C. The style of an XHTML document can be defined in three ways:

- External: the XHTML document and the style sheet are two distinct files; the style sheet is *included* in the XHTML document by using the tag `<?xml-stylesheet ?>` or the instruction `<link>`.
- Internal: the style sheet is embedded in the headers of the document using the instruction `style`.
- Inline: style information (a complete style sheet is not used in this case) is embedded in the XHTML document using the instruction `style` attached to any other WML2 instruction.

The introduction of style sheets is very important for the success of WAP, by providing the possibility to distinct contents and its presentation, especially with the extended graphical capabilities, larger screens and greater processing capacity for graphic enrichment and animation in new terminals.

#### ➤ **Cookies**

WAP 2.0 introduces session support by using cookies:

- Locally: at least 4 cookies must be stored locally on the telephone with a maximum size of 125 bytes each
- Remote storage: during a session with a server, the user agent sends management commands in the request header, which allows to use either local management or remote management of cookies or both (or neither).

#### ➤ **Pictograms**

In WAP 2.0 the notion of a worldwide unified representation of pictograms is introduced. These pictograms can represent functions or objects on the small screen of a telephone and reside in the telephone, thereby reducing download time for common pictograms. The pictogram specification allows pictograms to be stored temporary or definitively and to download new pictograms. Pictograms are divided in classes and all available classes have been divided in two categories: Core (arrow, button, action, message, state, media and info) and Dictionary (animal, appliance, emotion, entertainment, music, sport, weather, etc).

The advantages of the use of pictograms are a reduced download time, an increased visual aspect for the user and the internationalization of the user interface (pictograms recognized in any country with any operator; possible use of Orange look and feel in pictograms).

### 4.2.2 Configuration and adaptation

#### ➤ **Automatic terminal configuration by a server (provisioning)**

WAP 2.0 introduces a transparent and secured mechanism, which allows the configuration of the terminal (gateways, homepage, ...). The automatic configuration can either be performed using the push mechanism (OTA) or via a SIM card (which can contain WAP configuration information).

The network elements for the configuration of a terminal are depicted in figure 7.

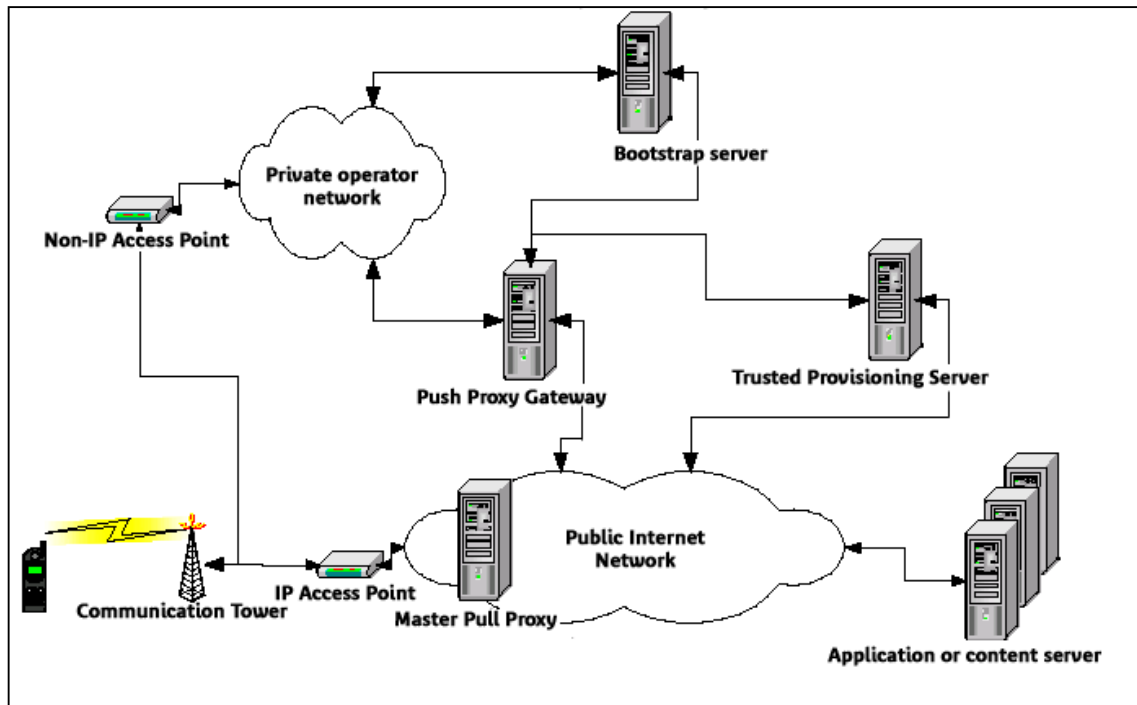


Figure 7: WAP provisioning structure

The configuration mechanism is very important for the ease of use of WAP. The configuration process becomes transparent to the user with this mechanism and the access to the WAP service will therefore be much easier.

#### 4.2.3 Push and messaging

##### ➤ Push

The push mechanism has been introduced in WAP 1.2 and completed in WAP 2.0. This mechanism will therefore not be discussed in detail here, because it's not completely new in WAP 2.0. The push mechanism allows the notification of a terminal in order to let it automatically load a specific page, security information, configuration information or a multimedia message. This mechanism is used for distant configuration of the terminal (provisioning), the multimedia messaging service (MMS; see messaging) and the services for notification of the user of a new service (service indication) and for automatically loading pages (service loading).

In addition to WAP 1.2, push is supported over HTTP in WAP 2.0 (WAP terminal is considered as an HTTP server and the WAP proxy as an HTTP client).

The push services allow a transparent service to the user, wherein the difference between content stored locally (in the terminal) and content accessed on a server is less visible for the user. In this way, the user might not even notice he's using WAP, because it's completely transparent.

##### ➤ Multimedia Messaging Service (MMS)

The messaging service introduced in WAP 2.0, MMS, allows to compose, send and receive messages with rich content (images, videos, text and sound). Depending on the service model, MMS permits a quick delivery (like SMS) or a store-and-forward approach (like e-mail) or both.

MMS provides the framework to develop applications that support feature-rich messaging solutions, permitting delivery of various types of content in order to improve the user experience.

#### 4.2.4 Security

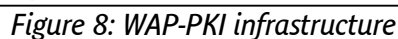
In WAP several security related technologies are available: WAP 1.x already supported WTLS, which is a specific version of SSL using WDP datagrams. WAP 2.0 introduces the protocol TLS (Transport Layer Security), which allows the establishment of secured end-to-end sessions in connected mode. The implementation of TLS in WAP allows the establishment of tunnels through HTTP proxies (the so-called TLS Tunneling), which allows the establishment of a secure session from the terminal to the secured content server, even when there are intermediate proxies. The use of this functionality will probably be limited (because WTLS is better adapted to the use in wireless networks), especially for banking sites or secured Intranet applications.

The highest level of security in WAP 2.0 is offered by the support of a Public Key Infrastructure (PKI), which allows the exchange of user and server certificates before the access to a service. The WAP-PKI infrastructure is depicted in figure 6 on the next page. In this picture the following consequent steps in the WAP-PKI procedure are depicted:

1. A gateway asks a server certificate to a PKI portal
2. The portal asks the certificate to the certification authority which should be able to deliver this certificate
3. The certification authority send the certificate to the gateway

After this step, two possible cases exist:

- ⊙ Security in two phases
  4. A WTLS session is established between the telephone and the gateway
  5. A SSL/TLS session is established between the gateway and the contents server
- ⊙ End-to-end security
  6. The server asks a certificate to the certification authority via the PKI portal
  7. The PKI portal sends the request to the certification authority
  8. The certificate is sent to the contents server
  9. A secured end-to-end session is established between the telephone and the contents server



In both scenarios, two new entities / actors are introduced in the WAP infrastructure:

- ② The certification authority, which is a company that has been approved for its capacity to deliver certificates. For example, Verisign is such a company.
- ② The PKI portal, which is a server that is allowed to send requests for the activation or cancellation of certificates. This intermediate entity is necessary to guarantee a sufficient level of trust that a service provider can attribute to a specific terminal. Service providers establish a trust relation with a PKI portal, which manages the requests for certificates from users.

The PKI infrastructure seems to be essential to create a mass-market use of the so-called m-commerce (mobile e-commerce), but it can also be of use for applications such as an authenticated and confidential access to internal data of a company (intranet applications).

#### 4.2.5 Miscellaneous

### ④ External Functional Interface (EFI)

The External Functional Interface standardizes the relations between a WAP application and external devices connected to the terminal in order to implement the access to functions available outside the WAP browser (for example taking a picture with a camera attached to or integrated in the telephone). The EFI specification defines the functional framework to access functions provided by

external devices regrouped in Classes and explains how to access to this framework from within WML pages or WMLscript.

This interface might be of use for interactive WAP applications with the need to communicate for example with a digital camera attached to the telephone, but the specification still seems to be too incomplete to be of use in short- or middle term applications.

➤ **Data synchronization**

The data synchronization specification in WAP 2.0 just states that if a terminal implements the synchronization of data, it should do this using the SyncML protocol. In the SyncML protocol a sync client is defined, which is independent from the WAP browser. The SyncML messages used for the synchronization can be supported over several different protocols; one of the protocols is WSP. The specification of SyncML is out of the scope of the WAP 2.0 specifications and will therefore not be discussed in detail here.

Data synchronization is important for reducing communication costs and to compensate the weaknesses of mobile networks.

## 5 Mapping content on technologies

The two technologies discussed in this document, MIDP 2.0 and WAP 2.0, both allow the creation and consultation of content. This chapter tries to map several kinds of content on one or both of the technologies. This can be used as a guideline to decide which technology to use for which service, but this information will probably be too general to decide to use a specific technology in a specific case, so a case per case study might be necessary to decide which of the two technologies (or both) to use.

### 5.1 Different types of content

Numerous different types of content exist. This paragraph tries to give a general overview of the different types of content. This overview will be used further on to map the different types on one of the two (or both) technologies. An identifier that will be used to reference the type of content further on follows the description of the type of content.

We identify the following different types of content:

- ② Informational content (information needs) (A)
  - ③ Simple textual informational content, possibly with b/w images (A1)
  - ③ Rich-media informational content (A2)
    - ④ Simple textual with still color images (A2.1)
    - ④ Simple textual with still color images, video, audio (A2.2)
- ② Entertainment related content (time to kill) (B)
  - ③ Simple game (for example question-answer game) (B1)
  - ③ Simple game with images (for example chess, poker, solitaire) (B2)
  - ③ Complex game with (moving) images and/or sound (B3)

### 5.2 Types of content mapped on technologies

#### 5.2.1 Informational content

Informational content, which is used by users to fulfill their informational needs, can be any content that supplies general information, information on a specific item or any other information (for example actualities). Two basic subtypes of the informational content have been identified:

- ② A1: Simple textual informational content, possibly with black and white (b/w) images. This concerns specifically actualities and static or dynamic textual information. This type of content can best be provided using WAP 2.0, because of its simplicity. However, if the level of personalization is high, MIDP 2.0 might be a better solution.
- ② A2.1: Simple textual information with still color images. This type of content can be provided in WAP 2.0 and in MIDP 2.0; because of its simplicity the best / simplest solution will probably be WAP 2.0. However, when a high level of personalization of the content is required, MIDP 2.0 might better meet the demands.
- ② A2.2: Simple textual information with still color images, video and audio. For this type of content MIDP 2.0 will probably be the best solution because of its capability of managing color graphics and audio. However, depending on the capabilities of the terminal some of these functionalities (especially color images) might as well be

available in WAP 2.0). Especially when the content has to be personalized at a high level, MIDP 2.0 is the best solution.

### 5.2.2 Entertainment related content

Entertainment related content, which is used by users during their *time to kill*, can be any content that does not supply any important information, but is just used to pass some spare time (specifically games). Three subtypes of games have been identified:

- ② B1: Simple game. Simple games (like question-answer games) are currently available on WAP. These games can easily be implemented on WAP 2.0 as well. The same simple games can also be implemented in MIDP 2.0, but it seems to be a bit useless to download an application to access to such a game, because it cannot really be played offline, the questions could change pretty often, et cetera.
- ② B2: Simple game with images (chess, poker, solitaire, etc.). For this kind of games a complicated implementation in WAP is possible, but it seems to be easier to implement this kind of games in MIDP 2.0, because of its support for graphics, the fact that the game can be played offline when downloaded, et cetera. However, in some cases an implementation in WAP 2.0 can be useful as well.
- ② B3: Complex game with (moving) images and/or sound. Complex games using moving or still images and sound cannot be implemented in WAP 2.0. MIDP 2.0 offers the necessary functionalities to implement these kind of games and another advantage of the implementation in MIDP 2.0 is that the game can be played offline (possibly with additional online functionalities such as high score lists or even a community around the game) once downloaded.

### 5.2.3 Conclusion

Based on the short analysis of different kinds of content in the previous two paragraphs, we can conclude that most content can be provided using either WAP 2.0 or MIDP 2.0. Some types of content can only be provided using MIDP 2.0 and some other can better be provided using WAP 2.0, because of their simplicity.

We can clearly state that the two technologies do not exclude each other for providing several types of content. For example, we describe a specific content service and a way of implementing it in WAP 2.0 and a way of implementing it in MIDP 2.0: a train schedule service. In this example, the train schedule service contains two functionalities:

- ② Complete train schedule offering access to all departure and arrival data of all trains in the region (for example France) concerned
- ② Personalised train schedule offering access to departure and arrival data of a frequently used train

These functionalities can for example be used in the following way: someone working in Paris and living in Champagne sur Oise everyday takes the train from Champagne sur Oise to Paris to go to work. Sometimes he takes the train to visit other places in France (during holidays for example) and now and then he visits his grandma in Vitry sur Seine. This person will use both functionalities:

- ② The personalized train schedule wherein he entered two regular used routes (home to work and home to grandma; both configured with their start and end point). When he wants to take the train from home to work (or from work to home) or from home to his grandma (or from his grandma to go home) he uses the personalized

train schedule to retrieve the information concerning the next train he can take or to retrieve the information for this train at a specific departure or arrival time.

- ⊙ The complete train schedule for his irregular train trips: he can enter the departure and arrival station and possibly a departure or arrival time and retrieve the train information for these trips.

An implementation of these functionalities in WAP can consist of several dynamic WAP pages that allow access to the complete train schedule and the personalized train schedules by using the MSISDN or some logon information. All information can be made available in this way and the only drawback for the user is that he always has to be online to access to the train schedules, even for his previous personalized train trips. In MIDP the same functionalities can be implemented with a high-level user interface allowing access to the complete train schedule and the personalization of train schedules. The complete train schedule can either be downloaded with the application (and therefore be in the memory of the phone) or can be on a server with which the application communicates when a non-personalised train schedule is requested or to download the data for a personalized train schedule now and then. The personalized train schedules will always be available, even if no network connection is available and the application only needs a connection to download new schedules in the case of changes (the application, and therefore the user, can be notified when such an update is available by using the push functionalities in MIDP 2.0) or when adding a new personalized trip.

The implementation of this content service is therefore possible both with WAP 2.0 as with MIDP 2.0, however the use of MIDP 2.0 has some advantages.



## 6 Use cases

### 6.1 Orange World



Remark: data and description used in this paragraph are derived from the French, UK and Dutch Orange World offer. The Orange World offer differs slightly per country (such as additional or other content services), but the main idea is always the same.

#### 6.1.1 Description

Orange introduced *Orange World* in October 2003 first in France, UK and Switzerland and later on in other countries such as The Netherlands.

Orange World is Orange's new personal, simple compelling experience for Orange customers worldwide containing:

- New and relevant personalised services
- New easy-to-understand tariffs (Orange World Bundles)
- Simple access to services with Orange Signature phones
- Orange phone trainers deployed across Europe
- Global advertising campaign

The technology is based upon the Wireless Application Protocol (WAP versions 1.2 and 2.0) and MIDP 1.0 and runs over GPRS networks.

The service has been launched on several different handsets wherein some handsets are recommended for the access to Orange World services like the Motorola V500, LG 7100, Alcatel OT735, Nokia 6600, Sony Ericsson T610, Sagem MyX-6, Treo 600 in France at launch time.

Orange World is available internationally with roaming; Orange offers GPRS coverage with roaming in several countries.

#### 6.1.2 Pricing

Orange World has a fairly straightforward set of charges proposing two packages containing a certain amount of data traffic and a certain number of SMS and/or MMS messages to be sent and a tariff for occasional users:

- For regular users: the package Orange World 6 euros containing 5 MB of data traffic for GPRS phones or 5 hours of communication for non GPRS phones and 5 MMS or 15 SMS messages
- For intensive users: the package Orange World 10 euros containing 10 MB of data traffic for GPRS phones or 10 hours of communication for non GPRS phones and 10 MMS or 30 SMS messages
- A per usage tariff for occasional users also accessible for prepaid users

### 6.1.3 Orange World services






The services offered in the Orange World offer will be discussed shortly here. To access the Orange World menu directly several devices have direct access button which allows a one-click access from the standby screen.

Figure 9: Access to the Orange World menu



In the Orange World menu the user can choose from different services which can be regrouped in three categories:

- ① Exchange 
  - ⓧ Agenda
  - ⓧ Chat
  - ⓧ Electronic cards
  - ⓧ Forums
  - ⓧ MMS Images
  - ⓧ E-mail
  - ⓧ Voice mail
  - ⓧ Phonebook (personal contacts)
  - ⓧ Online SMS storage
  - ⓧ Personal site
- ② Consult/Information 
  - ⓧ News
  - ⓧ Phonebooks (Yellow pages)
  - ⓧ Auto/Moto
  - ⓧ Banks

- ⊗ Stock exchange
  - ⊗ Cinema
  - ⊗ Restaurants
  - ⊗ Hotels
  - ⊗ Travel planning
  - ⊗ Weather
  - ⊗ TV programme
  - ⊗ Sports
  - ⊗ Etc.
- ⊗ Download 
- ⊗ Personal voice mail message
  - ⊗ Games
  - ⊗ Logos
  - ⊗ Ringtones

The access to one of the newest games available on the Orange World portal (Ancient Empires) is shown in the figure below.

From the Orange World menu the user accesses to the Games channel which always shows the newest games (illustrated) and then allows the user to choose to access the Latest arrivals, the Top ten games, the Game of the week (illustrated) and the Full catalogue.

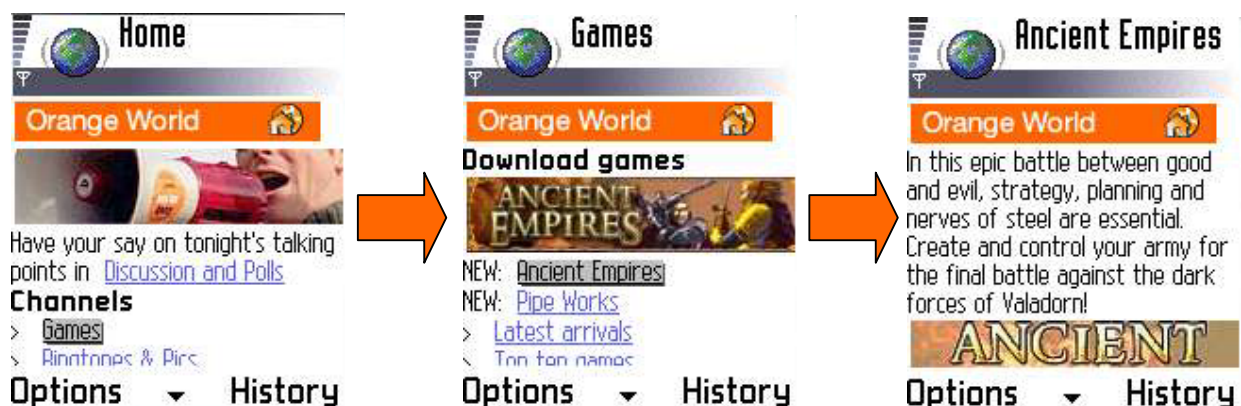


Figure 10: Games download via the Orange World menu

Once the user has selected a game to download he sees a description of the game and an image which shows a preview of the game (right picture in figure 10). He is then informed of the price of the game and can start downloading it using the normal download process (depends on the device being used and is therefore not depicted here).

The contents of the Orange World portals in the different countries where Orange is present are localized for the specific country with important contracts with the important information providers in the countries where the offer is available (for example Mappy, AlloCiné and MétéoConsult in France, Voetbal International, TMF, The Box and Belbios in The Netherlands).

#### 6.1.4 MMS

The access to the MMS functionalities uses the default access method of the phone and has not been modified in the Orange World service. The Orange World packages contain a certain number of MMS or SMS messages (the number of messages is a total number of messages where 1 MMS has the same price as 3 SMS messages). The Orange World portal contains images which can be downloaded and which can then be used in MMS messages to be sent. The web portal even allows the sending of an MMS message (one message for free per day).

#### 6.1.5 Technical implementation

The technical implementation of Orange World is completely based on WAP 1.x or 2.0 and Java MIDP 1.0 or 2.0. Orange has worked together with several device manufacturers in order to allow an easy access to the Orange World services (on several levels such as a one-click access for the Sony-Ericsson T610 for example or even more integration on the Orange Treo 600).

The Java specification used in the Orange World compatible handsets is based on J2ME MIDP 1.0 or MIDP 2.0.

An Orange World demo is available on Orange UK's web portal at the following address:  
<http://www.orange.co.uk/multimedia/wapdemo.html?phone=n7650>.

#### 6.1.6 Analysis

##### **6.1.6.1 Personalised mobile services**

Personalisation of the mobile services is one of the core advantages of the Orange World offer: everyone is different. To ensure a truly relevant experience for its customers, Orange has created the Orange World portal, allowing customers to personalise their phone. Also within Orange World are a series of services, making it easy to personalise and share content. "Add this to your page" makes personalisation easy; "Send to a Friend" enables customers to share their favourite content with friends; and "Get a Map" allows customers to find out exactly where they need to go, using the Orange intelligent network. All of these services appear below the relevant content and are accessible with just one click. The content and services are all available through the Orange World icon which is accessible on all WAP and GPRS handsets - not just new phones. This personalisation initiative is being rolled out across the Group following the success of Orange UK's Your Page which, in just six months, has seen over half-a-million customers personalise their content on their Orange phone.

##### **6.1.6.2 Consistent use of Orange World icons**

The new icons used for the Orange World offer are used consistently in all Orange communications: on TV, in print, on the web and on the Orange World portal on the mobile phone.

##### **6.1.6.3 Learning users to use mobile services**

With the introduction of the Orange World offer and portal, Orange also introduced the Orange Phone Trainers programme rolled out across the different Orange countries after a successful introduction in the UK. The Orange Phone Trainers programme is focused on driving both new and existing customers in-store to ensure they get the

most out of their phone and build a deeper relationship with Orange. In the UK, the programme turned a 5% year-on-year decline of footfall in Orange Shops into a 20% year-on-year increase. Following this success, Orange Phone Trainers are now being rolled out across other Orange territories. France firstly introduced mobile coaches with Switzerland and other Orange countries launching similar programmes early in 2004.

#### **6.1.6.4 Combination of J2ME and WAP**

In the Orange World offer, Orange uses both WAP (WAP 1.x and WAP 2.0 on any device, even old ones) and J2ME technology (MIDP 1.0 and MIDP 2.0) where available as main building blocks. These two technologies are clearly used as complementary technologies and Orange currently focuses on games with the J2ME technology. Although Orange supports all (including previous versions (MIDP 1.0 and WAP 1.x)) of these technologies, it rolls out Orange World on the new versions of these technologies as well (MIDP 2.0 and WAP 2.0) and Orange is currently testing Orange World services in its 3G trial in several trial cities in France and the UK.

## 6.2 Vodafone live!

Remark: data and description used in this paragraph are derived from the UK Vodafone live! offer. The Vodafone live! offer differs slightly per country (such as additional or other content services), but the main idea is always the same.

### 6.2.1 Description

On Thursday, October 24<sup>th</sup>, 2002, Vodafone introduced *Vodafone live!* in England, Germany, Italy, The Netherlands, Spain and Ireland. The service is now also available in Greece, Portugal and Sweden, Australia and New-Zealand will probably follow in 2003. Vodafone live! is Vodafone's new consumer mobile data platform composed of new mobile functionalities such as picture messaging (MMS), polyphonic ringtones, games (using Java), e-mail (*Vodafone mail*) and instant messaging (*Vodafone messenger*). The technology is based upon the Wireless Application Protocol (WAP versions 1.2 and 2.0) and MIDP 1.0 and runs over GPRS networks.

The service is being launched on three handsets:

- Sharp GX10, which comes with a 'hot button' leading directly to the Vodafone live! service, a built-in camera and 4 pre-loaded games  
This is a Vodafone exclusive handset with direct access to Vodafone live! with a specific button.
- Nokia 7650 with a built-in camera. The Vodafone live! service is accessed through a colour menu pre-configured on the phone
- Panasonic GD87



Figure 11: Sharp GX10

Vodafone live! is supposed to be available internationally with roaming: currently, Vodafone offers GPRS coverage with roaming to 15 countries.

### 6.2.2 Pricing

Vodafone Live has a fairly straightforward set of charges:

- Per-bit usage charge (depending on GPRS plan chosen by users)
- Event charges for specific content

The table below shows the headline prices of Vodafone live! services. There is free picture messaging and free access to 'many' of the third-party services on the Vodafone live! menu until 31 January 2003. There is no initial or subscription charge.

Service	Price
<i>Examples of browsing charges on starter tariff</i>	
Access Vodafone live! menu, browse news for 5 minutes then check the weather: 65 Kbytes	47.5p
Access menu, download a ring tone, a wallpaper and find a near-by restaurant: 76 Kbytes	55.5p
Access menu, check horoscope. Download a Java arcade game and a screen saver: 91 Kbytes	66.0p



<i>Per item charges</i>	
Send a picture message (up to 30KB)	£0.36
Polyphonic ring tones	£1.50
Games Arcade – Java games	£1, £3 or £5
Wallpaper	£2.40
Screensaver	£2.50

Handsets Price	Contract <sup>1</sup>	Prepaid
Sharp GX10	£ 199.99	£349.99
Panasonic GD87	£ 229.99	N/A
Nokia 7650	£ 229.99	N/A

#### GPRS

Before using Vodafone Live! services, users must choose a GPRS price plan in addition to their normal voice price plan (see section 3.1. service set-up). [10]

Table 4: Vodafone live! Pricing

#### 6.2.3 Vodafone live! services

The services offered in the Vodafone live! services will be discussed shortly here. To access the Vodafone live! menu directly on the Sharp handset the user can click on the right button from the standby screen.

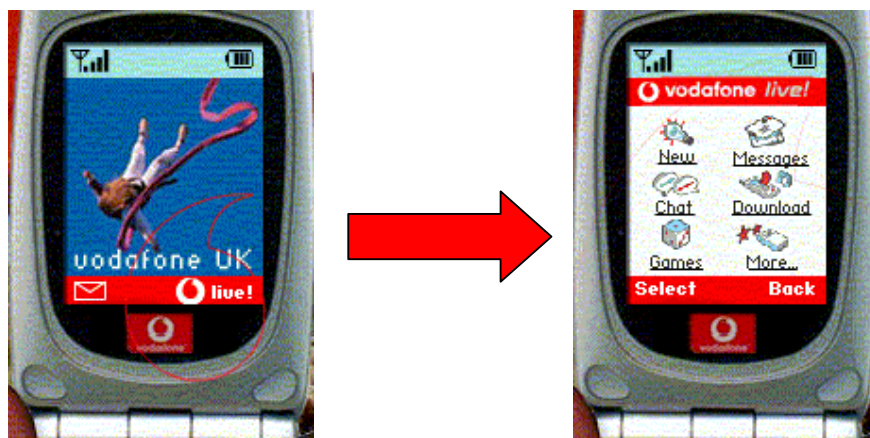


Figure 12: Access to the Vodafone live! menu

In the Vodafone live! menu the user can choose from six icons: new, messages, chat, download, games and more.... The applications behind these icons will be discussed shortly here:

##### ➤ **New**

Here, the user can access the new services. At the time of the launch of Vodafone live!, this area contained for example (in the UK) a column of The Sun (paid service £ 1,00 per week), the portal Handbag.com, horoscopes by Russell Grant (paid service

<sup>1</sup> Pay monthly customers are able to spread handset payments over 10 months. For example buying the Sharp GX10, customers will make a first payment of £19.99, then 9 instalments of £20.00.

£ 2,00 per month), a section More fun and info containing among others Find and seek (location based service), News and weather, Sports, Entertainment, Travel, etc.

### ➤ **Messages**

The section Messages lets the user access:

- Free picture gallery (download of 120 pictures on various themes)
- Vodafone Messenger
- My mail (Vizzavi mail, Vodafone mail, messages). Users are invited to set up a single mailbox address to which is sent a text message whenever they receive any type of message.


### ➤ **Chat**

The Chat section is divided in:

- WAP chat, a mobile chat room
- Text friends, which is a service for people looking for other people to chat with
- Text flirt, a service which acts as a match maker for people who want to text flirt with other people.
- Vodafone Messenger, see 'Messages' above

### ➤ **Download**

Available sections in 'Download' are:

- Ringtones, which allows the download of 1300 polyphonic ringtones from a range of suppliers 
- Logos and more, which allows access to 'Logos' (not for the Sharp) and 'Wallpaper', 'Screensaver' (not for the Sharp), 'Wicked welcome' (pre-recorded greeting messages), 'Mobile backgrounds' (many free backgrounds), '8888 logos' (logos for Nokia phones only)
- Games arcade, see 'Games' below
- Free pictures
- 8888 tech (paid text alert service with daily technology news)

### ➤ **Games**

Vodafone has put a lot of effort into searching for and testing the best games for Vodafone live! services. It selected 19 classic and contemporary arcade titles for the launch.

An agreement with Gameloft, a leading publisher of wireless games, allows Vodafone customers to discover Prince of Persia: Harem Adventures, Block Breaker, Siberian Strike and Lock'em up on their Vodafone live! Java enabled handsets by downloading them directly over-the-air.

The 'games' section of the Vodafone live! menu gives access to:

- Games Arcade
- Text games
- 8888 fun
- Goddess Stones (Russel Grant horoscope forecasts)
- Compatibility (Russel Grant)
- Tarot reading (Russel Grant)



The access to Games Arcade is shown in the figure on the next page (the steps between the Vodafone live! menu and the access to the list of games (the user always sees the 'Game of the week' and can then choose to access the Top 10 games for example) are not shown).



Figure 13: Games Arcade via the Vodafone live! menu

④ **More...**

At this moment the contents of the 'More...' section are exactly identical as the contents of the 'New' section.

#### 6.2.4 Picture messaging

The access to the picture messaging (i.e. MMS) functionalities is very easy: the user chooses the camera on his mobile phone, takes a picture and chooses the option Save and Send. The user can then type the text which should accompany the picture and chooses the receiver of the message. The message is then sent to the receiver.

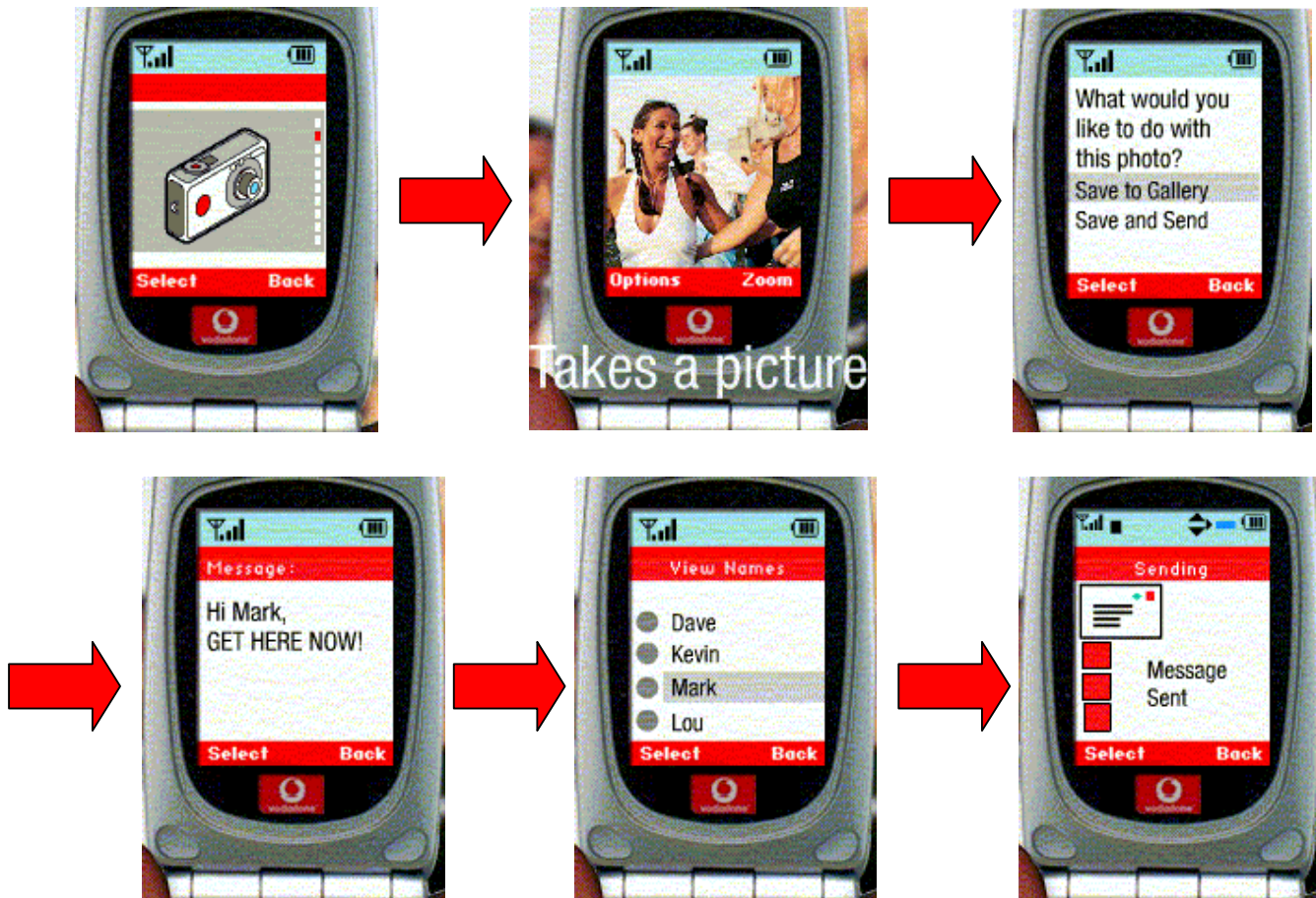


Figure 14: The access to the picture messaging (MMS) functionalities

#### 6.2.5 Technical implementation

The technical implementation of Vodafone live! is completely based on WAP and Java MIDP 1.0. Furthermore specific changes have been made in the software of the handset in order to give it a Vodafone look and feel and in order to allow easy access to the Vodafone live! menu.

The Java specification used in the Vodafone live! compatible handsets is based on J2ME MIDP 1.0, but has some Vodafone specific API extensions. The specification is therefore called the *Vodafone Service Class Library* (VSCL). Currently, two versions of VSCL exist: VSCL 1.0.1 for devices launched in 2002 (like the Sharp GX10) and VSCL 1.1.0 for devices launched in 2003. Besides MIDP 1.0 as a base for VSCL, Vodafone used the work from its Japanese subsidiary, J-Phone, by using the J-Phone Specific Class Libraries (JSCL 1.0). The VSCL specification can be depicted as follows.

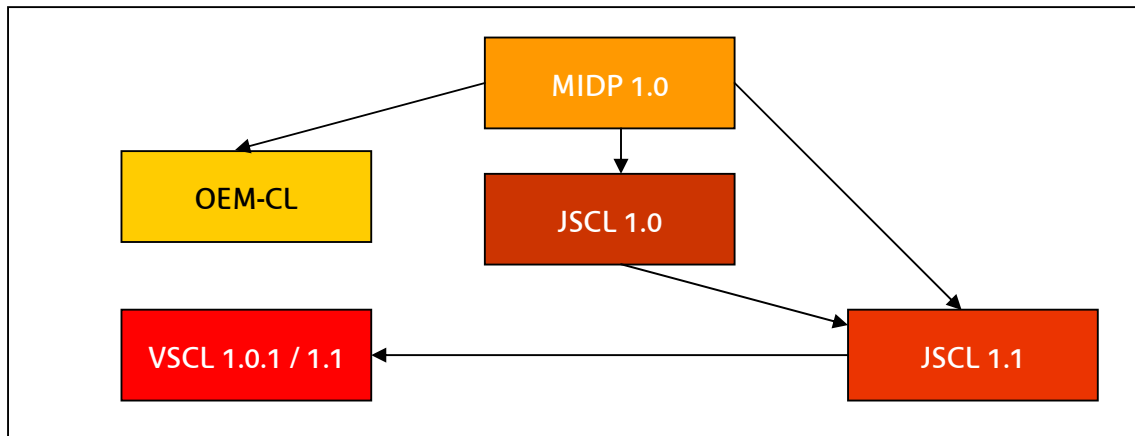


Figure 15: VSCL practical package diagram

In this figure, the OEM-CL block depicts OEM class library packages which can be created by handset manufacturers on top of the MIDP 1.0 package. VSCL provides a mechanism for handset manufacturers to integrate their pre-existing OEM class libraries with the VSCL implementation in order to maintain backward compatibility (integration in VSCL-NG for VSCL next generation).

Specific differences between MIDP 1.0 and VSCL are:

- ② JAD file contains optional fields *MicroEdition-Profile* (name and version of the J2ME profile – MIDP-1.0) and *MicroEdition-Configuration* (name and version of the J2ME configuration – CLDC-1.0) that are only used in the manifest for MIDP 1.0.
- ② The JAD file contains new fields which are not used in MIDP 1.0: *MIDxlet-Resident* (optional; indicating the capability of standby application), *MIDxlet-Network* (optional; indicating the request of network connection under execution), *MIDxlet-Copy-guard* (optional; security attribute allowing to prohibit copy, transfer or move of application) and *MIDxlet-API* (mandatory, value VSCL-1.0.1; indicating the required API packages other than the J2ME profile MIDP)
- ② For security reasons, applications are permitted to make HTTP connections only to the domain from which they were originally downloaded (or a subdomain of it).
- ② New packages in VSCL:
  - ③ System packages: system API that allows a MIDlet to control device specific functions such as vibration, backlight, input keys, media control, etc.
    - ④ *com.vodafone.v10.system.device* (get handset status [key push (including the 8-direction key), remaining battery, field intensity, incoming events using events listeners: calls, mail (i.e. MMS, SMS, cell broadcast, push), alarms, ringer] and control handset functions [vibration, LCD backlight (blink or on/off), key auto repeat])
    - ④ *com.vodafone.v10.system.media* (media [still image, animation movie, sound, ...] control functions [play, stop, ...])
  - ③ MIDlet package: defines the lifecycle of a MIDlet
    - ④ *com.vodafone.v10.midlet* (especially to implement resident MIDlets which run as a wallpaper application)

- ④ Sound package: provides sound control functions (play, stop, ...)
  - ⊗ `com.vodafone.v10.sound` (support for both sequence data and wave data and ability to control multiple data independently and asynchronously [e.g. shooting sound and a background music in a game])
- ④ Graphics package: user interface API that provides rich extensions such as display Sprite data and 3D polygon data
  - ⊗ `com.vodafone.v10.graphics.sprite` (rich graphical expressions using minimal data)
  - ⊗ `com.vodafone.v10.graphics.j3d` (for the creation of 3D graphics with the Micro3D program library using a 3D tool for example available for download on one of Vodafone's partner sites)
- ④ Utility package: provides some calculation functions
  - ⊗ `com.vodafone.v10.util` (FixedPoint holding fixed point data values and Vector2D, a 2 dimensional vector calculation class)

## 6.2.6 Analysis

### 6.2.6.1 Innovative mobile services

Picture messaging and Java are at the core of Vodafone live!. This places Vodafone slightly ahead of some of its European rivals (in the UK, both Orange and T-Mobile have already launched a picture messaging service, while the fourth operator, O2, launched a suite of Java games in September. In Vodafone's other core European markets, Germany, Italy and Spain, T-Mobile, TIM and Telefónica Móviles, respectively, have all launched both MMS and Java services already.).

Although, these services are innovative, it's not very original. The Vodafone live! offer is more original in the following areas.

### 6.2.6.2 New user interface

As we have already seen, Vodafone live! is centred on integration. In its purest form, seen on the Vodafone-exclusive Sharp GX10, Vodafone live! involves completely replacing the handset's interface with one designed by the operator, which is really different to what we have seen so far. The GX10 is the result of 12 months' co-operation between Vodafone and Sharp. It supports Vodafone's content standards, is fitted with a dedicated "Vodafone live!" button, and comes emblazoned with a Vodafone logo.

### 6.2.6.3 New way to communicate around mobile services

For years, operators and vendors have been pushing technology acronyms and not service benefits to their customers who, quite understandably, remained widely indifferent. Vodafone live! is the start of a new marketing approach for mobile operators: it focuses on applications and pushes the technology and network aspects in the background.

Vodafone has also broken with tradition by marketing MMS, unified messaging, instant messaging and mobile Java services under new names: Picture Messaging, Vodafone Mail, Vodafone Messenger and Games Arcade.

### 6.2.6.4 A powerful position vis-à-vis handset manufacturers

The Nokia 7650 supports fewer Vodafone-defined features than the Sharp and Panasonic terminals. However, the offer of a Nokia terminal is the testimony of Vodafone's leadership and purchasing power (which derives from its presence in 28



countries). Previously, Nokia focused on offering services through its mobile portal, Club Nokia. Therefore, Nokia's willingness to work with mobile operators on service propositions such as Vodafone live! was being questioned. This is not the case any more. Some research institutes think that we are seeing the start of a shift in the branding approach of leading terminal manufacturers; in future, many of them will allow their brand to coexist with those of operators.

#### **6.2.6.5 Combination of J2ME and WAP**

In the Vodafone live! offer, Vodafone uses both WAP (WAP 1.2 and parts of WAP 2.0) and J2ME technology (MIDP 1.0) as main building blocks. These two technologies are clearly used as complementary technologies and Vodafone has shown that they do not exclude each other. Although Vodafone uses the previous versions (MIDP 1.0 and WAP 1.2), the same conclusion goes for the new versions (MIDP 2.0 and WAP 2.0) and Vodafone will probably continue its Vodafone live! offer using these new technologies.

### **6.3 Japan**

Even when operators introduce Java platforms, Java itself does not make services mature nor creates a new market immediately. In Japan, the country where non-vocal services were first introduced on mobile terminals, the growth was supported by the expansion of existing features by operators and handset manufacturers, and the expansion of services by content providers relying on those features. The same situation applies for Java. Java itself does not produce any merit. The important point is to see how Java produces synergies as a part of mobile handset services.

In a mobile phone service market, there are four major players: operators, handset manufacturers, users and content providers. They have different points of view on mobile phone services.

- ⊗ Operators play a central role. They are willing to provide services to attract users based on software (developed by content providers) and hardware (built by manufacturers). Services have to be cost effective and have to match with what the users are ready to pay for.
- ⊗ Handset manufacturers can easily add new features, which expand the existing ones. On the other hand, new features demand high investments. Therefore, handset manufacturers' risks are high in a matter of cost effectiveness depending on whether or not users adopt those features.
- ⊗ Users do not care how services are provided (browsers, SMS, mail, ...) as long as they can get the same functions with the same usability at the same cost. Even if new features are added to mobile terminals, users will not adopt them if they do not find any merit compared to existing services. New added features may never create any profits.
- ⊗ Content providers care about how many users will use the new services that they offer thanks to the new features. Implementation costs are their major concern.

#### **6.3.1 Operators and handset manufacturers**

One important aspect of the Japanese market is the influence of operators on handset manufacturers: **more than in Europe, operators in Japan can decide the technical specifications of handsets.** Because there is no SIM card for 2G in Japan, a

handset can only be used with one network: handsets are operator-specific. This is why there are some very close links between handset manufacturers and operators.

### 6.3.2 Content providers

Companies providing contents for service such as i-mode are as diverse as companies having a web site. Subcontractors can design/develop mobile Internet sites, while several companies propose content distribution platforms and hosting services. On the games market, all the traditional video game editors are present, as well as broader entertainment companies (such as Bandai), and companies devoted to mobile contents.

### 6.3.3 Available services










There are a lot of services available for mobile users in Japan. On top of the traditional service, voice transmission, users can choose to have access to Mobile Internet services, as an option (about € 3). It corresponds to an Internet Service Provider (ISP) service. In return, the user gets an email address and access to a full range of multimedia services.

Mobile Internet services are not always linked with the network technology (2G, 3G) but require handset evolution: the video content download service EzMovie is offered on 2G by KDDI, and its competitor iMotion on 3G by NTT DoCoMo.

### The importance of branding

Let us look first at the Internet: it can be defined by users as a technology and a panel of services which cover email, Internet browsing, et cetera. In the same way, marketing teams in Japan have branded their mobile Internet service using a brand: the world famous i-mode for NTT DoCoMo, J-Sky for J-Phone and finally EzWeb for KDDI.

In the same marketing effort, each individual service has got a specific branding that can be found on all documentations and phones. The following table lists some of the services offered by the operator and how they have been branded:

List of services	NTT DoCoMo	J-Phone	KDDI
			
<b>Mobile Internet services</b> Global service offering access to mobile data services (similar to an ISP)	<b>i-mode</b> 	<b>J-Sky</b> 	<b>EzWeb</b> 
<b>Messaging services</b> Messaging services (email with possible attachments like music, images)	<b>i-mode mail</b> 	<b>J-Sky mail</b> <b>Sha-mail</b> (sha=image) 	<b>E-mail</b> 









<b>Web browsing services</b> Offers navigation through content providers sites.	<b>i-mode</b> 	<b>J-Sky</b> 	<b>EZWeb</b> 
<b>Interactive services (Java)</b> Allows the user to download small applications and run them on their phone.	<b>iAppli</b> 	<b>Java</b> 	<b>EzPlus</b> 
<b>Video content download services</b> Enables download of small videos to be played locally on the device.	<b>iMotion (3G)</b> 	No service yet apart from sha-mail.	<b>EzMovie</b> 

Table 5: Operator services offer and branding in Japan

The power of branding is so strong that a high percentage of customers think that i-mode is different to the Internet. NTT DoCoMo customers even requested the Sha-mail service offered by J-Phone.

In Japan, operators only brand services like i-mode, i-appli. Their role is not to produce content but to reference content providers and offer them the necessary infrastructure (specifically billing) needed to exploit their content.

#### 6.3.4 Java introduction

The plan to deploy the Java programming environment on i-mode compatible cell phones was first announced in a March 1999 joint press release from DoCoMo and Sun Microsystems, owner of the Java language. Java, after all, was originally conceived as a simple, robust computing environment that could be embedded in ROM chips and used as the basic operating system for small, wireless-enabled computing devices, from cell phones or PDA's to set-top boxes and even the proverbial wired fridge. With Java as the OS onboard such devices, together with additional hardware like RAM memory chips, small-size, low-power processors, and enhanced input/output devices (like MPEG-4 video chips), they become much more like "real" computers.

A wireless Java device can download and store files, execute applications, display dynamic Web content and continuously update itself with newer versions of onboard software (although it will still take a few more product cycles before cell phones will have all of these features).

##### 6.3.4.1 DoCoMo

DoCoMo actually introduced its Java service, branded *iAppli* on January 2001. A new line of handsets, the 503i series (Java-enabled), was launched at the same time. For the launch, users of the new 503i phones could access Java games, dynamic stock quotations, and other content on sites operated by Cybird, Sanwa Bank, Sega, Tsutaya,



Kabu.com, Nomura, et cetera. 32 different types of content were available, 80% of which were games. The first two terminals were produced by Fujitsu and Matsushita/Panasonic, other handset manufacturers followed quickly (NEC, Mitsubishi, Fujitsu and Sony). DoCoMo heavily promoted the iAppli service right from the beginning, and the customers responded very positively: 70,000 terminals sold the first day.

In its communication, DoCoMo strongly emphasized that there were actually 2 types of Java applications:

- ② Stand-alone applications that can be downloaded to the handset and used repeatedly without any access to the network – examples of these include karaoke, puzzles, some games, and personal info management software
- ② Online applications that work with a connection to the service's content server and automatically obtain fresh data – examples include applications involving weather forecasting and real-time tracking of stock market prices.

So far, free games (made by independent developers) are mostly present on DoCoMo's platform since its technical documentation is public and a simple web server is enough to distribute applications. On the other hand, applications developed by professionals are fee-based. DoCoMo has created the same model as the one used for i-mode, encouraging fans to develop and put online their own service.

The following chart shows the popularity of the categories of the official Java-enabled sites. This classification is derived from the DoCoMo's i-mode menu. Even it is not fully reliable, it gives a good idea of the tendency: **games come first** (one third of the services), followed by **melodies**, **screen savers** and **horoscope/test applications**.

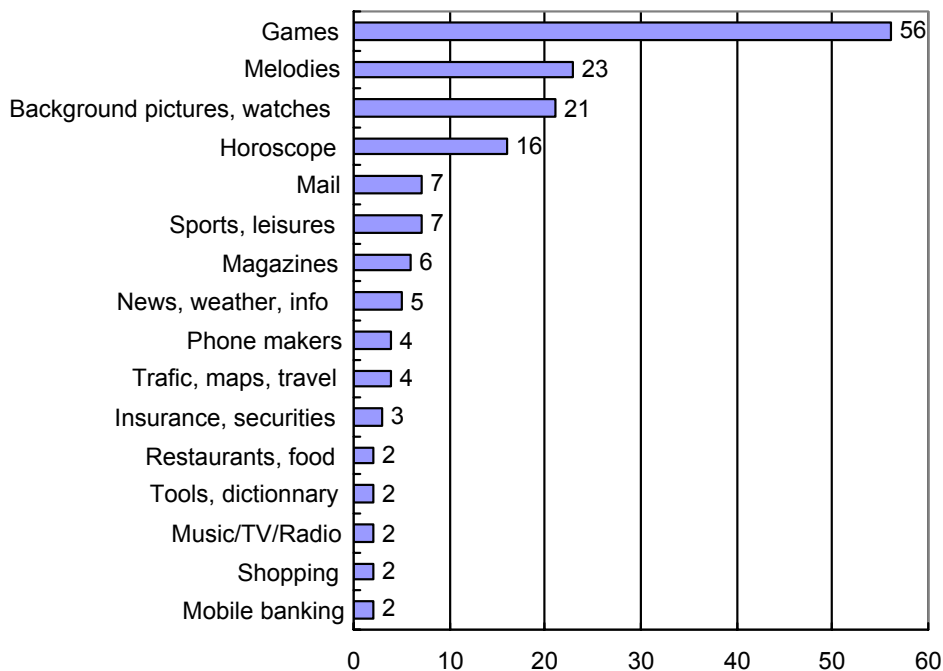


Figure 16: Repartition of official Java-enabled DoCoMo sites

Early November 2001, there were 4,800 Java applications referenced on the i-appli portal, including both official and non-official applications. There are about 150 official i-appli sites, representing less than 500 applications. This means that the independent developer community has produced more than 4,000 Java applications so far for DoCoMo's service.

#### **6.3.4.2 J-Phone**

J-Phone followed DoCoMo by launching its Java service in June 2001. It brought several innovations, especially larger applications (from 10kB to 30kB) and a dedicated 2D-3D animation engine. The first terminal was produced by Sharp, followed by Toshiba and Mitsubishi. The name of the service was simply *Java Appli*.

#### **6.3.4.3 KDDI/au**

KDDI/au was the last to go, launching its *EZ Plus* service at the beginning of July. It had the largest maximum application size (50kB) and a sophisticated security model. Interestingly, when J-Phone and KDDI launched their Java service, DoCoMo's iAppli had already 2.5 million users (after less than 6 months).

#### **6.3.5 Java standards**

As we will see below, in the Java terminology, a basic set of functionality is called a profile. DoCoMo developed its own Java profile, called DoJa (=DoCoMo Java), while the 2 others used the MIDP standard profile. However, since every operator added some specific functions to palliate some lacks of the profile (in an extension library), there is **no compatibility** between any of the Java services currently used in Japan.

#### **6.3.6 Access to Java services & contents**

##### **6.3.6.1 Discovery of new applications**

Before a user can run Java applications, he first needs to download them and thus to discover the application he is interested in. Applications can be found depending on the business model of operators on official or unofficial sites. The easiest way to discover an application is through the operator portal; from the operator first page, users can select a link to the Java applications operator portal.

Japanese operators have put a lot of effort making Java content discovery as easy as possible: simple user interface, communication through directory brochures, monthly magazine, leaflets... More information on this can be found on the communication strategy part of this document.

The operators in Japan have created a market based on subscription to official content provider sites. The role of their portal is to categorize content and facilitate user access to these sites. The final goal for the operator is to have the customer to subscribe to the content provider site.

Regarding Java content, Japanese operators act as a services directory provider, like the *GUIDE* in the French Minitel, or *Palmsoft* for Palm software.

Access to official sites is promoted through all the communication channels of the operator. The easiest way to access the service is through the operator main menu. The steps are the following:

1. Press the button for direct access to operator's portal or navigate through user interface
2. Select the Java service menu that is on the welcome page of all operators
3. Select a category of content (Wall-paper, Games, Fortune-telling...)
4. Choose a content that you want to buy
5. Read the content site page for more information
6. Subscribe to the service through pin code
7. Download the application that you want to get

In the NTT DoCoMo model, there are a lot of independent developers that have been developing Java applications and thus there are a lot of other sites referencing Java content. Several sites have taken the role of content portal to reference the work of independent developers. On these sites, you can find references to access these applications.

#### ***6.3.6.2 Access to Java applications***

Once the Java application has been downloaded, the user needs to launch this application. If the user uses the application on a regular basis, he can configure an automatic start-up time from the application menu: the application will automatically start at a given time of the day.

From the user point of view, Java contents are not different from other downloadable contents such as ringing melodies and images. For those two types of contents, users access a download site through the browser and simply click on the link corresponding to the required resource.

Once the application is downloaded, it is stored and can be accessed through a list. Basically, the Java virtual machine must be launched each time an application is started, which takes 10 to 30 seconds.

## 7 Conclusion

As this document shows, the two technologies WAP 2.0 and MIDP 2.0 do not exclude each other: although these two technologies both offer the possibility to access to several types of content using a mobile phone, the access to a specific type of content is not limited to either the use of MIDP 2.0 or WAP 2.0. A certain content type (for example the train planning) can in some situations better be accessed using an MIDP application, although in other situations the same type of content can better be accessed via WAP. The two technologies can therefore be said to be complementary instead of exclusively.

Furthermore, the download of any Java application is done using WAP for the application discovery phase. Therefore, users not only use their telephone to use downloaded applications, but continue to use WAP services for application discovery, discovering information provided by their operator (the WAP pages of the operator will not only contain a list of available applications divided in categories, but will also show the most popular applications [most downloaded], promoted applications [for example application or game of the week / month], et cetera). Finally, the user will continue to use WAP content services for his occasional needs for mobile content services.

As we saw in the use cases described in this document, both Orange with its Orange World offer, the Japanese operators and Vodafone with its Vodafone live! offer successfully introduced a combination of WAP-like and MIDP-like access services (Vodafone used its subsidiary J-Phone to learn from the Japanese market). These use cases are probably the best evidence of the non-competitiveness of the two technologies WAP and MIDP.

## References

- Mobile Information Device Profile for Java™ 2 Micro Edition version 2.0 Specification, JSR-118 Expert Group
- K-Lab consultation on Orange Java APIs, MOOS project, November 2002
- WAP 2.0 White Paper, English release, Cédric Nicolas, Ubicco, March 2002
- WAP 2.0: Impact on Services, on mobile devices, architecture, J. Vittu, O. Laurent, Y. Léchervy, October 2002, FTR&D/DMR/SMO/644v1.0
- Java launch in Japan – Java services & content market, Java strategies and advances in the technology, P. Calvet, F. Feurtey, July 2002, BD/FTR&D/DMR/SMO/02.203/PC
- Vodafone live! – survey of data services suite, version 0.1, 02/12/2002
- Vodafone live! websites: <http://www.vodafone.co.uk/live> and <http://www.vodafone.nl/live>
- Via Vodafone website: <http://via.vodafone.com>
- JAD/JAR Creation Guidelines, VSCL 1.0.1, September 2002, Vodafone AG, GMT-VG/GJP-NW/SPC303J, Revision 0.92
- Architecture Design Specification of Vodafone VSCL Java API, Version 1.0.1, August 2002, Vodafone AG, GMT-VG/GJP-HS/SPC001E
- Architecture Design Specification of Vodafone VSCL Java API, Version 1.1.0, August 2002, Vodafone AG, GMT-VG/GJP-HS/SPC001E