



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

Linux-based Mobile Phone Middleware

Application Programming Interface

Reference Architecture

Document: CELF_MPP_RA_ FR4_20061103

Deleted: 3G Specification

Deleted: Multimedia Mobile Phone API

Deleted: FR2_20060602

WARNING : This is a working draft for review only, it is NOT a published specification of the CE Linux Forum. It is likely that further substantial changes will be made in the course of review and issue resolution. Send comments on this version to:

MppApiComments@tree.celinuxforum.org

Revision History

Revision	Comment	Reviewer	Editor	Date
1.0	Initial draft for discussion at San Francisco face-to-face		NEC, Panasonic	July 2005
1.0	Minor revisions made at the San Francisco meeting		NEC, Panasonic	July 2005
2.0	Revision for Kawasaki meeting;		Scott Preece	September 2005
2.2.1	Changed to new format		Scott Preece	2005.10.21
2.2.2	Work with figures		Scott Preece	2005.10.24
2.2.3	Revisions based on Tamagawa discussions and San Francisco comments, including NEC comments. Resolution of issues raised previously and first work on harmonization with LiPS Reference Model.		Scott Preece	2005.12.22
2.2.4	Revisions to resolve Andre's issues		Scott Preece	2006.1.11
FR1	First version for Formal Review		Scott Preece	2006.3.1
FR2	Revised to resolve reviewer comments		Scott Preece	2006.6.2
FR3	Revised based on work at Editors' meeting; title updated; copyright updated		Scott Preece	2006.7.6
FR4	Revisions following Tokyo meeting; cleanup including dropping noncommon components		Scott Preece	2006.11.3

26	0. Introduction	5
27	0.1 Scope	5
28	0.2 Vocabulary and Abbreviations	5
29	0.3 References.....	5
30	1. The Mobile Phone Domain	6
31	2. Architecture.....	8
32	2.1 Applications Domain	9
33	2.1.1 Application Layer	9
34	2.1.2 Middleware Layer	9
35	2.1.3 Kernel/Driver Layer	10
36	2.2 Communications Domain.....	10
37	3. Description of functional entities	11
38	3.1 Linux Kernel	11
39	3.2 Common API	11
40	3.3 Application Framework	12
41	3.3.1 Window Manager.....	12
42	3.3.2 Application Manager	12
43	3.3.3 UI Toolkit	13
44	3.3.4 UI Renderer	13
45	3.3.5 Event Bus	13
46	3.4 Telephony Service	13
47	3.4.1 Circuit-Switched (Voice) Communication Service	13
48	3.4.2 Packet-Switched (Data) Communication Service	14
49	3.4.3 SMS Communication Service	14
50	3.4.4 Equipment Service.....	14
51	3.4.5 Personal Information Manager Service.....	14
52	3.4.6 Data Exchange Service	14
53	3.4.7 Record and Playback Service.....	14
54	3.4.8 Light Management Service	14
55	3.4.9 Sound Service	14
56	3.4.10 User Profile Library	14
57	3.5 Multimedia Service	15
58	3.5.1 Multimedia Manager	15
59	3.5.2 Multimedia Library	15
60	3.5.3 Multimedia Driver API	15
61	3.6 Feature Services	15
62	3.7 Connectivity Service.....	15
63	3.8 Platform Management Service.....	16
64	3.9 Terminal Adaptation Function (TAF).....	16
65	3.10 Driver API	16
66	4. Data Flows	17
67	4.1 Voice communication.....	17

68	4.2 Video phone	18
69	4.3 Internet Application	19
70	4.4 Dial-up Networking with External Devices	20
71	4.5 SMS communication.....	21
72		

DRAFT

Deleted: ¶
¶
¶
¶
¶

0. Introduction	5¶
0.1 Scope	5¶
0.2 Vocabulary and Abbreviations	5¶
0.3 Reference	5¶
1. The Mobile Phone Domain	6¶
2. Architecture	8¶
2.1 Applications Domain	9¶
2.1.1 Application layer	9¶
2.1.2 Middleware layer	10¶
2.1.3 Kernel/Driver layer	11¶
2.2 Communications Domain	11¶
3. Description of functional entities	12¶
3.1 Linux Kernel	12¶
3.2 Common API	12¶
3.3 AP Framework	12¶
3.3.1 Window Manager	13¶
3.3.2 Application Controller (APC)	13¶
3.3.3 UI Toolkit	13¶
3.3.4 UI Renderer	13¶
3.3.5 Mobile Software Bus (MSB)	13¶
3.3.6 Others	14¶
3.4 Telephony processing	14¶
3.4.1 Circuit-Switched (Voice) Communication service	14¶
3.4.2 Packet-Switched (Data) Communication service	14¶
3.4.3 SMS Communication service	15¶
3.4.4 Equipment service	15¶
3.4.5 Schedule	Error! Bookmark not defined.¶
3.4.6 Data Exchange	15¶
3.4.7 Record and Playback	15¶
3.4.8 Light Management	15¶
3.4.9 Sound System	15¶
3.4.10 User Profile Library	15¶
3.4.11 Removable Media Management	15¶
3.5 Multimedia Framework	15¶
3.5.1 Multimedia Manager	16¶
3.5.2 Multimedia Library	16¶
3.5.3 Multimedia Driver API	16¶
3.6 Data Processing	16¶
3.6.1 Bar Code Library	16¶
3.6.2 OCR Library	16¶
3.7 Connectivity Service	17¶
3.8 Platform Management Service	17¶
3.9 Driver API	17¶
4. Data Flows	18¶
4.1 Voice communication	18¶
4.2 Video phone	18¶
4.3 Internet Application	19¶
4.4 Dial-up Networking with External Devices	20¶
4.5 SMS communication	21¶

73

0. Introduction

74

This document defines a Reference Architecture – a commonly-understood organization of components for implementing mobile handsets. The purpose of a reference architecture is to define a standard vocabulary for talking about products in a domain. A practitioner should recognize the components and the organization of the components of typical handsets as variations on such a reference architecture. The purpose of this Reference Architecture is explanatory rather than constraining. Typical products will have implementation architectures that modify or extend this architecture in various ways.

81

Chapter 1 is an overview of the domain, characterizing different product tiers.

82

The basic architecture is described in chapter 2

83

The functions of each component of the architecture are described in chapter 3.

84

The data and control flows between components during typical use cases are described in chapter 4.

86

0.1 Scope

87

This document defines the reference architecture of Linux-based mobile phone. This is a non-normative part of the Specification. The goal of the Reference Architecture is to provide the context for the descriptions in the normative parts of the Specification.

88

89

The Reference Architecture does not describe the internal architecture of the communication protocol stack or the Application Framework.

90

91

92

0.2 Vocabulary and Abbreviations

93

See the corresponding section in the Preface document.

94

0.3 References

95

[CS] CE Linux Forum, *Linux-based Mobile Phone Middleware Application Programming Interface – Circuit-Switched Communication Service*

96

[Preface] CE Linux Forum, *Linux-based Mobile Phone Middleware Application Programming Interface – Preface and Common Types*

97

98

[PS] CE Linux Forum, *Linux-based Mobile Phone Middleware Application Programming Interface – Packet-Switched Communication Service*

99

100

101

[TAF] Technical Specification Group Core Network; Terminal Adaptation Functions (TAF) for services using synchronous bearer capabilities (Release 4); 3GPP TS 27.003 V4.1.0 (2001-03), 3rd Generation Partnership Project

102

103

Deleted: v

Deleted: The Architecture presented in this document is based on an architecture that was originally the collaborative work NEC Corporation, Panasonic Mobile Communication Ltd., and NTT DoCoMo, Inc.¶

Deleted:

Deleted: [Preface] <provide reference to the Preface document>¶
[CS] <provide reference to the Circuit-Switched Communication Service section>¶
[PS] <provide reference to the Packet-Switched Communication Service section>

104

1. The Mobile Phone Domain

105

It is customary in the mobile handset industry to describe phones as falling into different “tiers,”

106

broad classes of phones with common characteristics – feature set, price, display size, etc.

107

Because different tiers have different performance, price, and feature requirements, they will

108

often be implemented by different software and hardware architectures.

109

Table 1 gives working definitions of a set of Reference Tiers. Note that these represent a

110

particular point in time (end-of 2004) and some of the details of specific areas will change as

111

technology changes. Many products do not fit neatly into one niche and will blend characteristics

112

of different tiers. Also, many products will add features not covered by this table.

113

The Reference Architecture corresponds broadly to the “Smart Phone” and “Multimedia Phone”

114

tiers in this table. The differences between those tiers are typically in the application of the

115

architecture (the way it is used) and in the details of the components and the physical

116

characteristics of the device. This architecture could be used for the lower tiers (“Feature Phone”

117

and “Plain-Old Mobile”), but would include capabilities and performance enablers that would

118

probably make such application economically inappropriate. However, as component costs

119

decrease and their performance increases, it might become practical to use this architecture

120

more broadly; the economies of scale resulting from reusable components might offset the

121

hardware costs.

	Tier			
Aspect	Smart Phone	Multimedia Phone	Feature Phone	Plain-Old Mobile
Focus	business focus	Personal/Entertainment Focus	Lifestyle Focus (voice plus social networking support features)	Voice
Primary Functionality	Full PDA functionality (Calendaring, address book)	Strong PIM support, personal content management features	Minimal PIM functionality (phonebook, datebook)	Phonebook and call logs
Extensibility	Extensible (downloadable features)	Limited extensibility (MIDlets or BREW)	Limited extensibility (MIDlets/BREW)	No extensibility
Multimedia	Optional	Video capture support, Media/content players, stereo	Limited multimedia support (pictures, MP3, MIDI, Simple, low-frame-rate animations)	None
DRM	Optional	Multiple DRM schemes	Hard DRM (limits on copying any media of given types)	None
Camera	Optional	2-3 megapixel camera	VGA camera or no camera	No camera

	Tier			
Aspect	Smart Phone	Multimedia Phone	Feature Phone	Plain-Old Mobile
Browser	XHTML Browser	XHTML Browser	WAP Browser (text-centric)	Embedded access to specific URLs
Display	QVGA or larger color display	QSIF or larger color display	QSIF or smaller color display	Small display (64x96), non-color
Interaction	Touchscreen UI or QWERTY keyboard plus pointing device	Specialized keypad for media/game interaction	Standard keypad plus carrier-specific keys	Standard keypad
Connectivity	3G connectivity, possibly WLAN, Bluetooth, IrDA	2.5G or 3G connectivity, possibly WLAN; High-speed USB; Bluetooth	2G connectivity; USB or serial cable	2G connectivity; proprietary accessory cable
Memory	32M RAM, 64M ROM, removable storage	64M RAM, 64M ROM, Hard Disk or large removable storage	16M RAM, 16M ROM, no removable storage	8M RAM, 8M ROM or less
Processor	120MHz	200MHz	30MHz	15MHz

122

123

2. Architecture

124

This document describes a reference architecture for mobile phones based on the Linux operating system. Any real product would be likely to have an implementation architecture that modifies or extends this reference architecture.

127

The architecture separates processing between two domains: the application domain and the communications domain. The two domains might run on separate processors, as separate processes on a single architecture, as separate virtual processors running over a micro-kernel, or other physical implementation. The Communications domain would include all activities requiring hard real-time behaviour.

132

Figure 1 is a top-level view of the architecture. Note that while the layering between the large boxes is meant to be strict (for instance, Applications interact with the Kernel only through Middleware components, not directly). The horizontal arrangement is arbitrary.

135

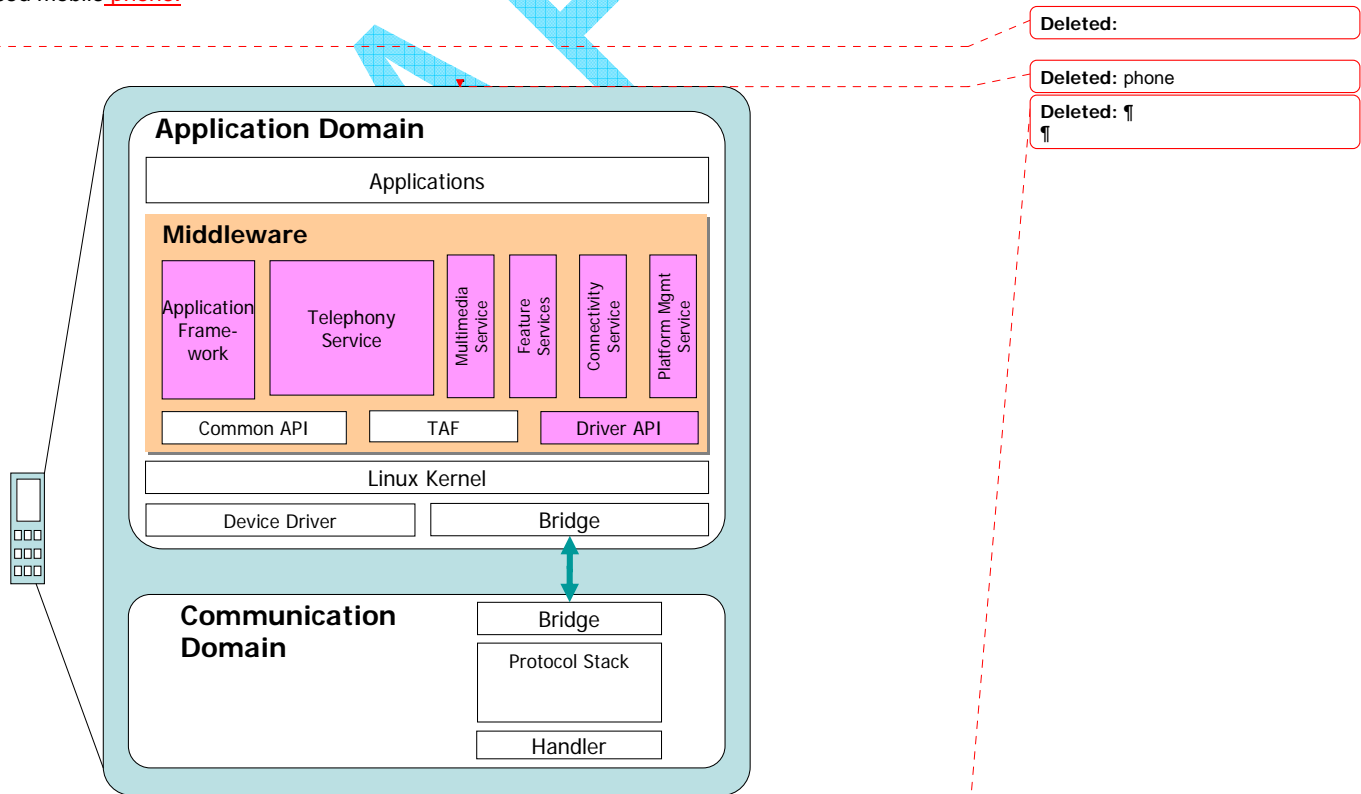
The Reference Architecture identifies this partitioning of processing domains but does not dictate the physical realization used. In Figure 1, the Bridge represents whatever mechanism is used in the particular realization to support interaction between the domains. The implementation of the Bridge is not in the scope of the Reference Architecture.

139

The blocks shown with a white background in Figure 1 are not part of the scope of the Mobile Phone API. They are shown as part of the common understanding of the structuring of a Linux-based mobile phone.

142

143



144

145

Figure 1 Overall Architecture

146 2.1 Applications Domain

147 The software running in the Application Domain contains the following 4 layers:

148 Application

149 Middleware

150 Linux ~~Kernel~~

Deleted: k

151 Driver & Bridge

152 2.1.1 Application ~~L~~ayer

Deleted: l

153 The application layer contains various applications. They are classified into the following 8
154 categories; samples are listed to illustrate the categorization, but are not meant either to be
155 requirements or to be a complete list of the applications in a given category:

156 2.1.1.1 Telephony ~~A~~pplications

Deleted: a

157 Telephony applications include Standby Screen (Idle), main menu, videophone application,
158 phone applications, phonebook and other Personal Information Management (PIM) applications,
159 and phone personalization settings.

160 2.1.1.2 System ~~A~~pplications

Deleted: a

161 System applications include Air download, Generic LCD display, Backside LCD display, PIN
162 authentication and monitor mode, other function setup, Equipment alarm, etc.

163 2.1.1.3 Multimedia ~~A~~pplications

Deleted: a

164 Multimedia applications include still image viewer, video viewer, camera app, vector graphics
165 viewer, avatar and ring tone management.

166 2.1.1.4 ~~F~~eature Applications

Deleted: Data-processing

167 Data-processing applications include OCR, barcode, SD-PIM, data transfer, memory transfer,
168 external I/F communication, user data, IR, schedule, personal productivity, voice memo, schedule
169 alarm, and data folder.

Deleted: a

170 2.1.1.5 Internet ~~A~~pplications

Deleted: -a

171 Internet applications are those that use TCP/IP to access resources on the internet, including e-
172 mail, Browser, HTML mailer, etc.

173 2.1.1.6 Internet Application Engine

174 Internet application engines are application-level services that would be used by applications that
175 include internet-enabled functions; examples include engines for HTTP, SSL, embedded
176 languages, etc.

177 2.1.1.7 Java Application Engine

178 The Java applications engine includes a Java Virtual Machine (JVM), Java Application Manager
179 (JAM), and class libraries.

Deleted: ¶
<#>Others¶
Others includes Accessory menu,
Accessories (text memo, calculator
etc.), etc..

180 2.1.2 Middleware ~~L~~ayer

181 Middleware layer contains the following components.

Deleted: l

182 2.1.2.1 Applications ~~F~~ramework

Deleted: f

183 The Applications framework provides application developers with a common framework of
184 services commonly used by mobile-phone applications.

185 | **2.1.2.2 Telephony Service**

Deleted: s

186 | The telephony service provides application developers with a framework of services for
187 | communications and handset management.

188 | **2.1.2.3 Multimedia Service**

Deleted: s

189 | The multimedia service provides video phone service (H324, for example), and multimedia data
190 | decoding, encoding, and rendering facilities.

191 | **2.1.2.4 Feature Services**

Deleted: Data processing service

192 | Most phones will have some differentiating features, such as location awareness, that will be
193 | exposed as services usable by one or more client applications.

Deleted: The data-processing service supports processing the data from various devices, e.g., bar-code reader, optical character reader

194 | **2.1.2.5 Platform Management Service**

Deleted: , etc

195 | The Platform Management Service provides the functions of system management, including
196 | installation of software and control of system processes.

Deleted: s

197 | **2.1.2.6 Connectivity Service**

198 | The Connectivity Service handles inter-device functions, such as synchronization and OBEX data
199 | exchange.

200 | **2.1.2.7 TAF (Terminal Adaptation Function)**

201 | The TAF provides the back-end services needed by the Telephony Service APIs [TAF]. It
202 | mediates between the phone software's model of network programming and the specifics needed
203 | for the particular networks supported by a specific handset. Note that the TAF could equally well
204 | be located in the Communications Domain or in the device driver layer; the placement shown in
205 | the diagram is common, but not universal.

206 | **2.1.2.8 Common API**

207 | The Common API provides application developers with standard C-language functions, including
208 | standard libraries and system calls.

209 | **2.1.2.9 Driver API**

210 | The device-driver API provides middleware and application programs access to devices and to
211 | services modeled as devices.

212 | **2.1.3 Kernel/Driver Layer**

Deleted: l

213 | **2.1.3.1 Kernel and Device Drivers**

214 | The Kernel / Driver layer contains the Linux Kernel and device drivers and the Application-
215 | Domain end of the Bridge.

216 | **2.1.3.2 Bridge**

217 | The Bridge supports communication between the Application and Communication domains,
218 | which may be implemented as separate processors or not, but will minimally have different
219 | scheduling regimes.

220 | **2.2 Communications Domain**

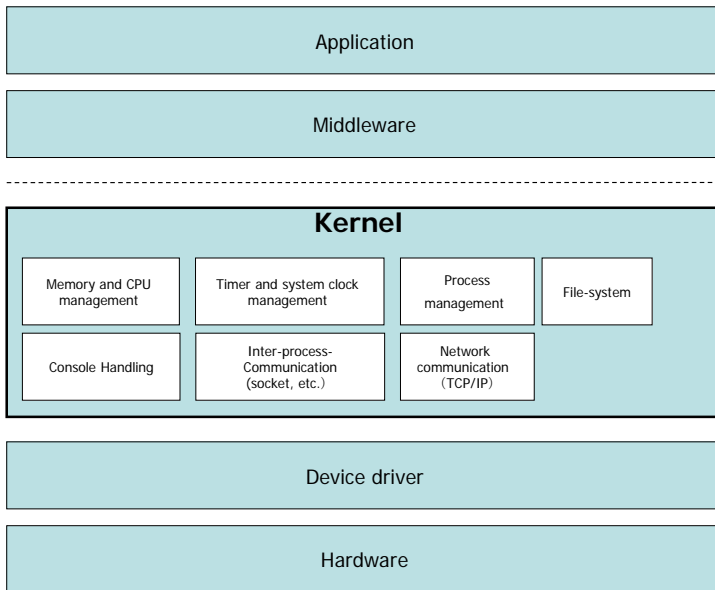
221 | The Communications Domain performs all processing that requires hard real-time behaviour,
222 | including executing the lower levels of the protocol between the device and the network. Its
223 | protocol stack contains the network stack's L1, L2, L3 layers. The Bridge supports communication
224 | with the Applications Domain.

225 **3. Description of functional entities**

226 **3.1 Linux Kernel**

227 The Linux Kernel provides:

- 228 • Memory and CPU management
- 229 • Timer and system clock management
- 230 • Process management: create, destroy and dispatch
- 231 • File-systems: files, directories, and space management
- 232 • Console handling
- 233 • Inter-process-communication: sockets, message queues, shared memory, etc.
- 234 • Network communication: TCP/IP



235
236 **Figure 2 Linux Kernel**

237 **3.2 Common API**

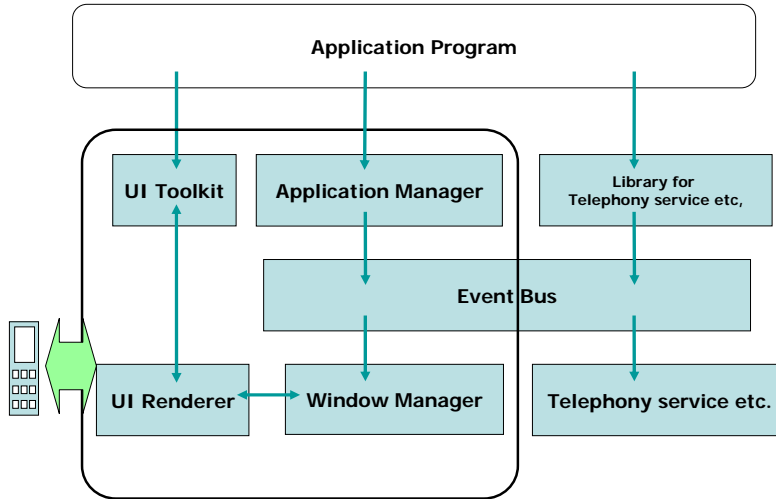
238 The Common API contains various functions for applications to use, including the standard C
239 libraries. The Common APIs conform to the POSIX standard.

240
241
242
243
244

3.3 Application Framework

Figure 3 is an overview of the application framework. This is an area where variation is common in the domain, with the specific requirements of the application set and the UI framework chosen for specific products potentially differing from this reference architecture. Usually, however, the functional separation is similar to this.

Deleted: -



245
246

Figure 3. Application Framework

Deleted: <sp>

247

3.3.1 Window Manager

The Window Manager provides unified operation and decoration for windows and controls overlap of windows (clipping and layering).

In the Application framework for mobile-phone, window manager provides:

- 251 Foreground and background control of application window
- 252 Tracking the state of applications (active, inactive, idle, not-started)

253

3.3.2 Application Manager

The Application Manager (AM) controls the start and end of applications. That is, starts applications, switches between applications, and shuts down applications at power-off. AM also manages application start status, and controls application switching operation (e.g., selection of an application to be next used as a foreground application at application switching).

Deleted: Controller
Deleted: (APC)
Deleted: APC
Deleted: APC

The Window Manager controls window stacking, focusing, and properties for each group during the period from window generation to deletion. The Window Manager receives requests from the AM through the Event Bus and resolves contention between applications, according to the priority table based on the application status. It also control windows overlapping and key focusing.

Deleted: APC
Deleted: library
Deleted: Controller

The Application Manager uses the Event Bus to communicate between applications and the Window Manager.

264 3.3.3 UI Toolkit

265 The UI Toolkit is a set of functions and facilities for the application to use to present and manage
266 interaction with the user.

267 3.3.4 UI Renderer

268 The UI renderer controls presentation of the interaction elements on the display(s).

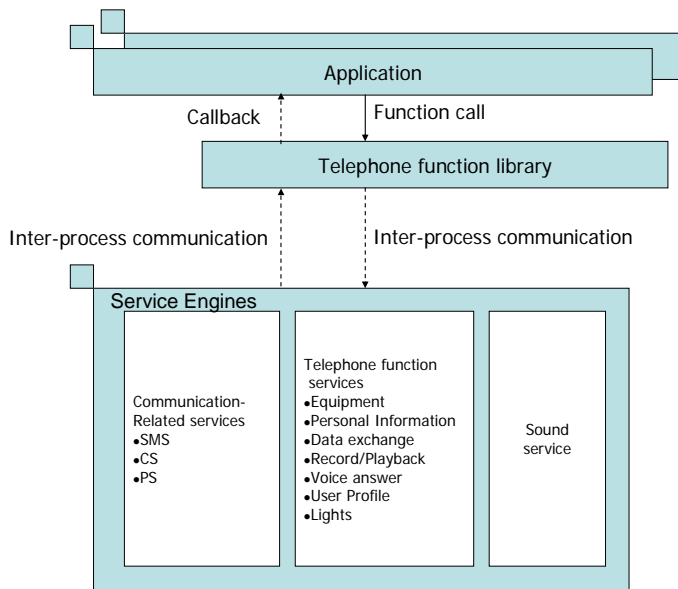
269 3.3.5 Event Bus

270 The Event Bus is a framework for passing messages between entities. It supplies communication
271 services (synchronous and asynchronous communication) between applications and services.
272 Different implementations of the reference architecture may use different kinds of inter-process
273 communication for this role. See [Preface] for more information about the programming model
274 and the portion of the Event Bus semantics that are visible in the APIs.

275 **3.4 Telephony Service**

276 **Figure 4** describes the telephony processing framework. The Telephony Service provides
277 abstract (not specific to the network technology or the handset hardware) APIs for applications to
278 use to access the functions of the handset, including both communications services and control of
279 the typical range of equipment features.

280 The architecture separates the interfaces to the various function-specific services from their
281 implementations. A specific implementation architecture might choose to make this a deployment
282 separation, as shown in **Figure 4**, or might choose to bundle them within a library implementation.



283

284 **Figure 4 Telephony Processing**

285 3.4.1 Circuit-Switched (Voice) Communication Service

286 The Circuit-Switched Communication (CS) service provides application programs with abstract
287 APIs for dialing, call disconnection, rejecting incoming calls, etc., for voice and video
288 communications. This API is available as [CS].

Deleted: <#>Others¶
<#>PICT (PICTograph) display library¶
It controls turning on/off of the upper pictograph elements such as antenna and battery. ¶
<#>Image library¶
It Converts, extracts, or compresses image data, aloes used to set or acquire image information.¶
Deleted: -
Deleted: 4

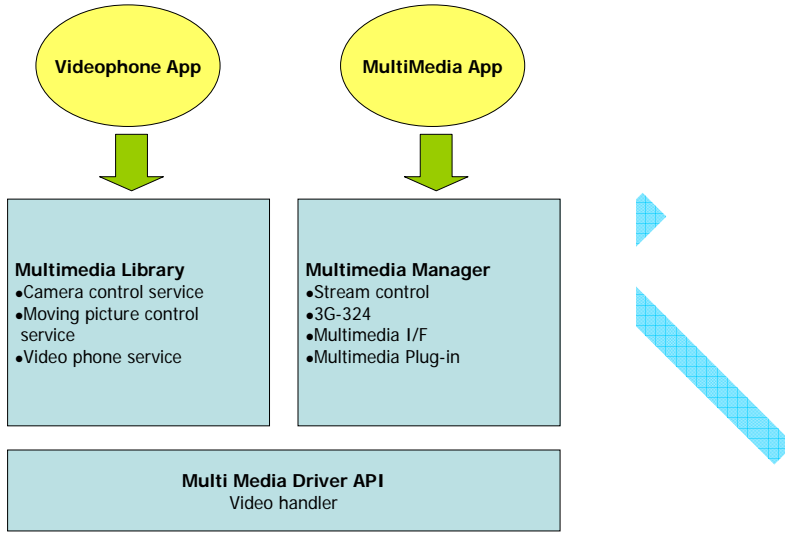
Deleted: 4

Deleted: s

289	3.4.2 Packet-Switched (Data) Communication Service	Deleted: s
290	The Packet-Switched Communication (PS) service provides abstract APIs for initiating and terminating sessions and connections, rejecting incoming calls, etc., for data communications. This API is available as [PS].	Deleted: application programs with
291		
292		
293	3.4.3 SMS Communication Service	Deleted: s
294	The SMS Communication service provides abstract APIs for sending and receiving SMS messages, receiving notification of events, and checking the status of the SMS service, etc.	Deleted: application programs with
295		Deleted: .
296	3.4.4 Equipment Service	Deleted: s
297	The Equipment service provides APIs for setting up, controlling, and reading the status of various handset hardware elements (batteries, headsets, etc.).	Deleted: programs to
298		Deleted: control
299	3.4.5 Personal Information Manager Service	Deleted: s
300	PIM provides abstract APIs for registering a schedule (calendar item), sorting the schedule, reading to-do data, etc.	Deleted: programs to
301		Deleted: data
302	3.4.6 Data Exchange Service	Deleted: s
303	The Data Exchange service provides abstract APIs for handling phone-book, memo, image, and video content, etc. on internal and removable memory stores.	Deleted: programs to
304		Deleted: e
305	3.4.7 Record and Playback Service	Deleted: s
306	Record and Playback provides application programs with record and playback of voice memo.	
307	3.4.8 Light Management Service	Deleted: s
308	The Light Management service provides APIs for programs to for control various lights.	
309	3.4.9 Sound Service	Deleted: s
310	The Sound service provides abstract APIs for functions to play and manage ring-tones and other sounds.	
311		
312	3.4.10 User Profile Library	Deleted: l
313	The User Profile library provides application programs with the ability to read and set various attributes of the user's profile, such as registered phone numbers and owner name.	Deleted: ,
314		Deleted: , and e-mail addresses.

315 3.5 Multimedia Service

316 | Figure 5 describes the multimedia service.



Deleted: -5
Deleted: .

317
318 | **Figure 5 Multimedia Framework**

Deleted: 5

319 3.5.1 Multimedia Manager

320 The Multimedia Manager provides interfaces for interconnecting multimedia functions, such as
321 video phone library, 3G-H324M protocol support, camera control library, and moving picture
322 control library.

323 3.5.2 Multimedia Library

324 The Multimedia Library provides interfaces for camera control, moving picture control, and video
325 phone services.

326 3.5.3 Multimedia Driver API

327 Multimedia Driver API provides video handler interfaces.

328 3.6 Feature Services

Deleted: Data Processing

329 This block covers various kinds of add-on functionality related to product-specific features, such
330 as special-purpose hardware, location-awareness, etc.

Deleted: dealing with external data

331 3.7 Connectivity Service

Deleted: <#>Bar Code Library¶
It provides the bar code reader functions.¶
<#>OCR Library¶
It provides the OCR access reader functions.¶
<#>Location Services¶
This provides access to the geographical location of the phone and events related to recognizing specific locations, for phones equipped with the appropriate capabilities.¶

332 The OBEX (object exchange) module supports synchronization and sharing of information
333 between devices by exchange of data objects. It provides an interface that is used by OBEX to
334 perform communication processing based on request messages from application programs and
335 to return processing results to the application programs. It also provides an interface that is used
336 by OBEX to convert objects to canonical format for interchange.

337 **3.8 Platform Management Service**

338 The Platform Management monitors the activation of each task at the time of power on and the
339 deactivation of each task at the time of power off, as well as the charging and other statuses of
340 the mobile terminal.

341 **3.9 Terminal Adaptation Function (TAF)**

342 The TAF is the interface between the phone software's networking functionality (voice and data)
343 and the network protocols that are used over the connected network(s); it is the entity that
344 actually constructs sessions and connections in response to abstract requests from clients,
345 adapting a generic view of connections to the specific interactions needed for a particular
346 network.

347 Applications do not use the TAF API directly; they use it only through the APIs provided by the
348 Telephony Service.

349 **3.10 Driver API**

350 | The Driver API provides upper layer components (middleware and application programs) present
351 | an abstract interface to device drivers, so that device-independent components do not need to be
352 | aware of the particular device drivers available on a specific handset, hardware-specific ioctls,
353 | etc. This avoids hardware dependencies, enabling development of portable middleware and
354 | application programs for mobile phones. The Driver API sits between the client applications and
355 | the actual device drivers.

DRAFT

4. Data Flows

This section describes data and control flow between components of the architecture in various domains.

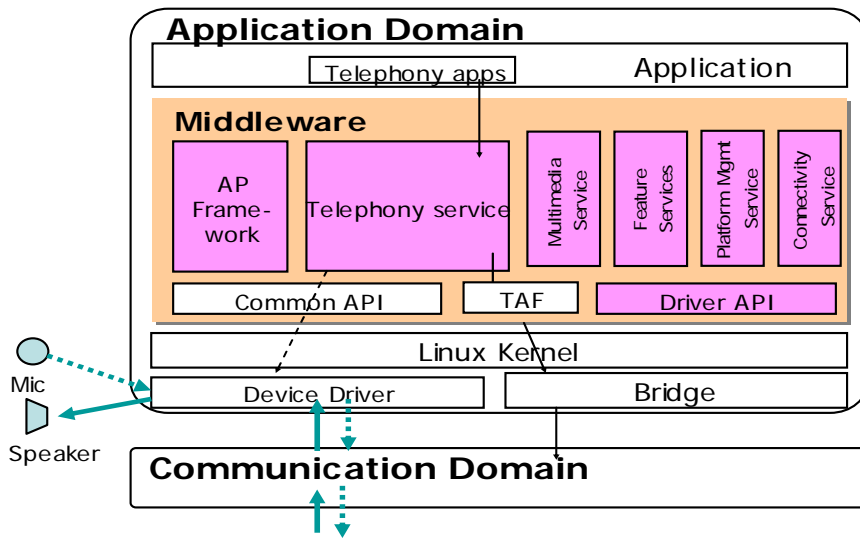
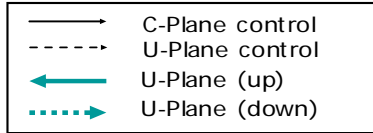
4.1 Voice communication

Telephony application controls the C-plane using the telephony service. This figure shows the flows for a mobile-originated call; for a mobile-terminated call there would also be C-plane flows upward into the telephony service to announce the incoming call. Incoming voice data is decoded by a protocol-specific codec and sent to the appropriate audio output; the user's voice is encoded and sent to the network.

Deleted: a

Deleted: Voice

Deleted: .



365

366

367

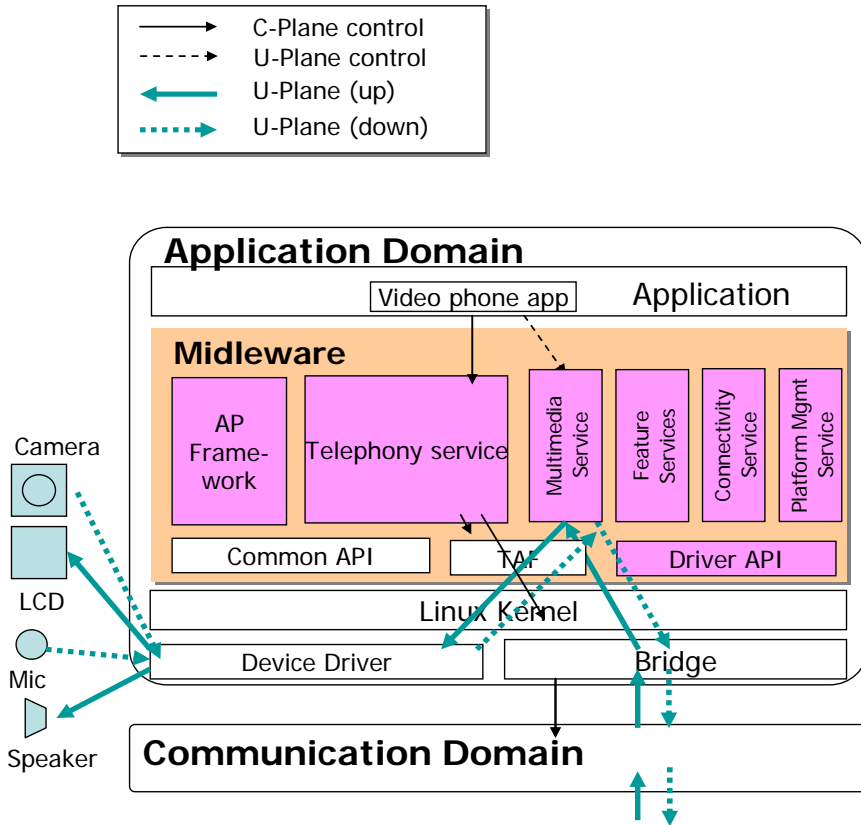
Figure 6 Voice Communication

Deleted: 6

368 **4.2 Video phone**

369 Video phone application program controls C-Plane by telephony service.

370 Multimedia service provides decoding and encoding facilities to support video telephony, such as
 371 H.324.



372
373

374



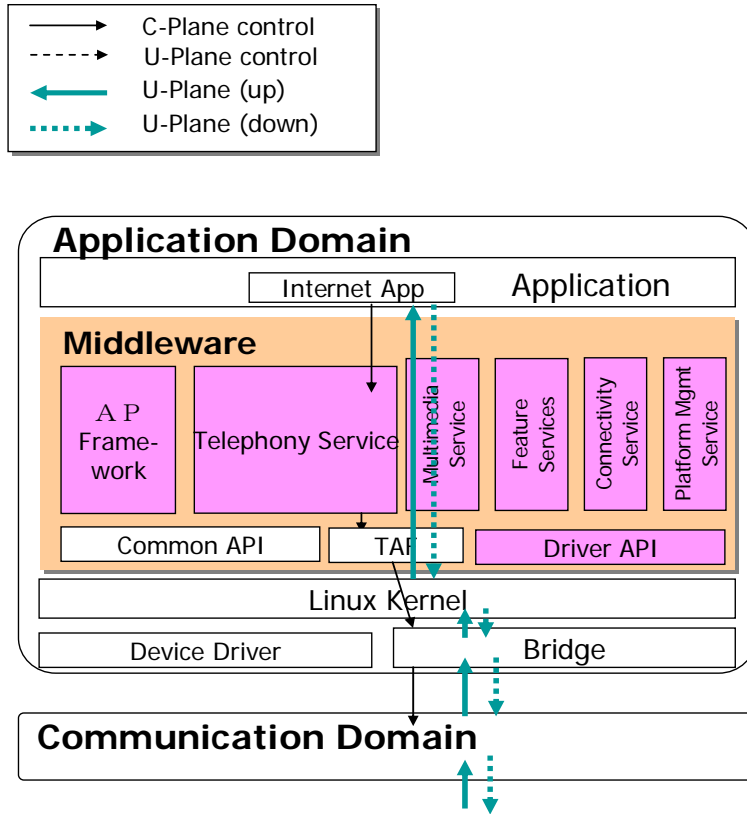
Figure 2 Video Phone

Deleted: 7

375 4.3 Internet Application

376 Internet Application program controls C-Plane by telephony service.

377 Data from the Internet is transported by a protocol stacked based on TCP/IP using the Linux
378 Kernel networking capabilities.



379
380
381



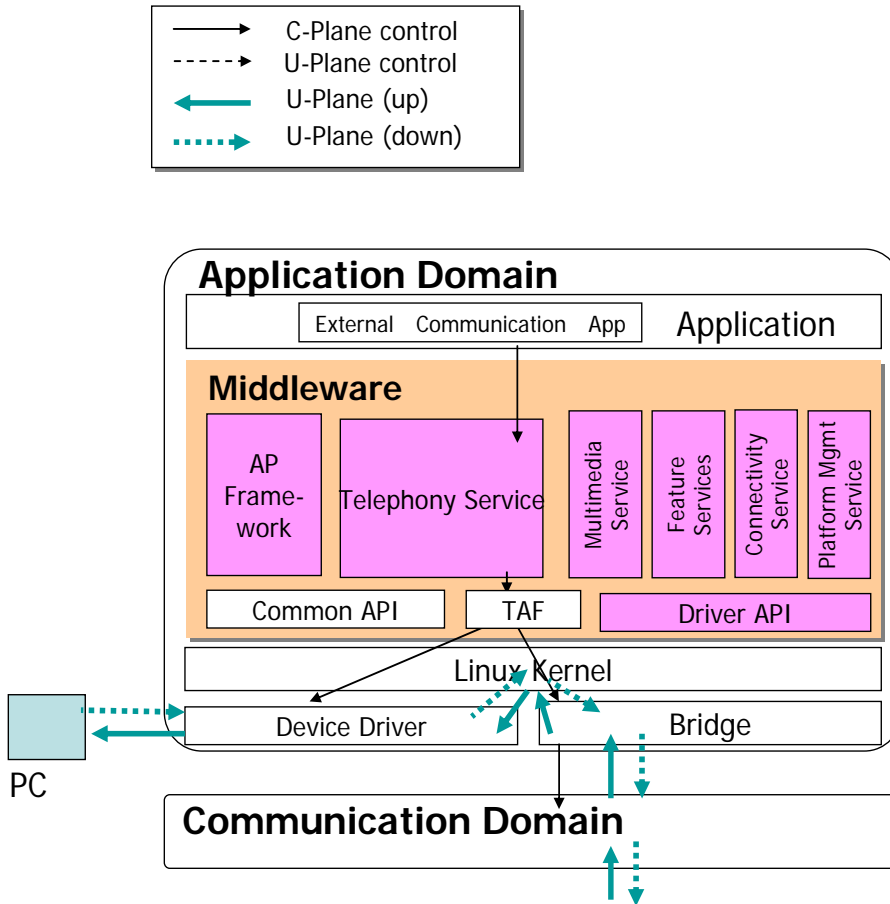
Figure 8 Internet Application

Deleted: 8

382 4.4 Dial-up Networking with External Devices

383 Applications program for dial-up networking controls C-Plane by telephony service.

384 Data are received and transmitted to an attached external device through USB or other
 385 communication bus and device driver and routed back to the internet through the communications
 386 stack.



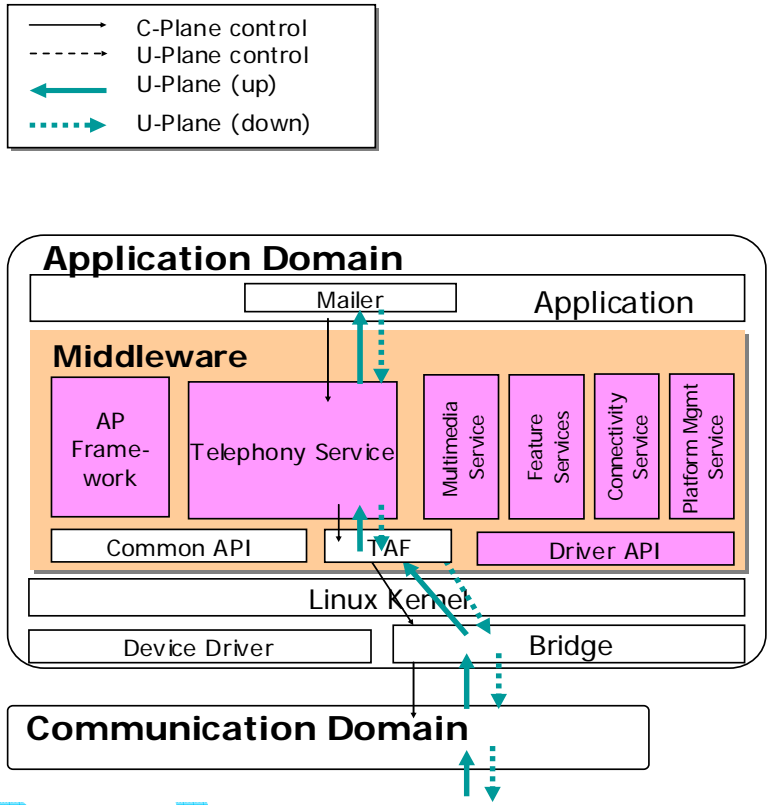
387
 388
 389

Figure 9 PPP Communication

Deleted: 9

390 4.5 SMS communication

391 Telephony Service SMS library controls C-Plane and U-Plane to send and receive short
392 messages.



393
394
395
396
397
398

Figure 10 SMS Communication

Deleted: 10