MOCHA: Modular and Configurable Handset Software Architecture

Kibum Kim
Center for Human-Computer Interaction
Department of Computer Science, Virginia Tech
Blacksburg, VA 24061, USA
kikim@vt.edu

Jason Dongwon Kwak, Seung Hwan Jo
UMTS R&D, Mobile Communication Division
Telecommunication Network
Samsung Electronics Co., Ltd., Korea, 442-600
{jkwak, samjos}@samsung.com

***Abstract*—This paper describes Mocha, an open mobile S/W platform and application developed by Samsung Electronics. Mocha's key features are its efficiency (it fits in regular phones), its portability (it covers different stack/OS/chipsets for CDMA, EV-DO, GSM/GPRS, and UMTS), its modularity and configurability (it meets customers' varying needs), its extensibility (it can be adapted to include new features and interfaces), its interoperability based on *de-facto* standards such as OMA, 3GPP, 3GPP2, CDG; and its security (it can withstand the attack of malicious applications). Among these key features, in this paper, we outline the multimedia messaging service (MMS) that inspired this system and then describe its design, architecture, and prototype implementation.**

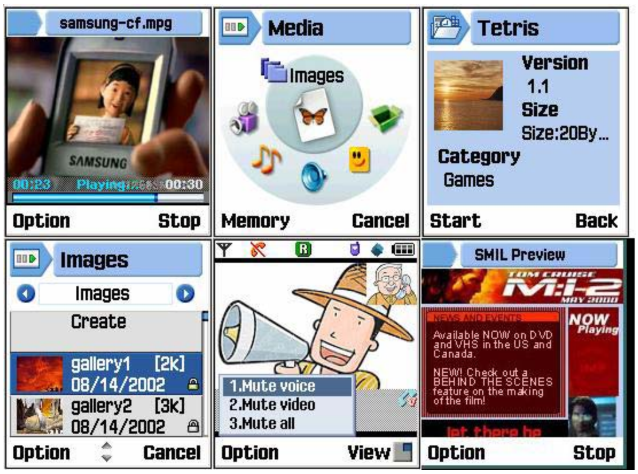
Keywords-mobile communications; handheld devices; service adaptability; social communication

I. INTRODUCTION

The progress made in cellular technology and the development of hybrid, multi-purpose handheld devices seem destined to move multimedia communication off the desktop and promise handy transmission of data, as well as voice service, over the next generation of 3G packet-switched cellular networks. Currently, Short Message Service (SMS) is the main medium for short, pithy messages (e.g. "C U L8R," "WER R U"); however, the amount of data that can be sent at any one time under this method is limited to 160 or fewer characters. As a result, while SMS might provide a quick and fun method for social communication, it is limited. On the other hand, mobile MMS enables mobile phone users entry into a brave new world of communication, wherein they can send and receive text, images, cartoons, audio and video clips, and other rich media content. Compared to SMS technology, MMS can provide a much more personal, versatile, expressive, and efficient method of mobile communication. To support these new multimedia- and information-centric services, numerous wireless technologies and solutions have been introduced. However, most current systems are specialized, tailored for specific protocols stacks, chipsets and operating systems. As a result, multiple applications must be redeveloped and modified for various wireless stacks/OS/chipsets and for different level of platforms.

II. MOCHA

Mocha (Modular and Configurable Handset Software Architecture) is a device-independent platform developed for mobile devices by Samsung Electronics., Co, Ltd. Software Center [1].



Available NOW on DVD and VHS in the US and Canada.

NEW! Check out a BEHIND THE SCENES feature on the making of the film!

Figure 1. Features of multimedia applications on Mocha

It addresses the "next generation" market, covering from 2.5G to 3G. Since its basic configuration requires just 500KB ROM, and MIPS of ARM7TDMI@40MHz, Mocha is efficient enough to fit in regular or "normal" phones. Also, it is portable enough to cover different bearers (CDMA IS95A,B, cdma2000 1x, EV-DO, EV-DV, GSM/GPRS, UMTS, TDMA), OS (REX, RTK, Nucleus, pSOS+, RTXC, AMX), and Chipsets/Stacks (QUALCOMM, SCOM, PCI, Philips).

Mocha is modular and configurable, so it meets a variety of customer needs, including dual-CPU video phones and smart phones (see figure 1). New features and interfaces can be added to Mocha, making it easily extensible, and based on de-facto standards it is also interoperable. With encrypted virtual tables and access control of APIs, Mocha will be impervious to the attack of malicious applications. Mocha possesses numerous modules, which are composed of six subsystems according to the specific services: Core Subsystem,

0-7803-8784-8/04/$20.00 © 2004 IEEE

GWES (Graphics, Window and Event Subsystem), Communication Subsystem, Security Subsystem, AVMS (Audio-Visual and Multimedia Subsystem), Application Framework and Application Service. Figure 2 shows the overall architecture of Mocha.

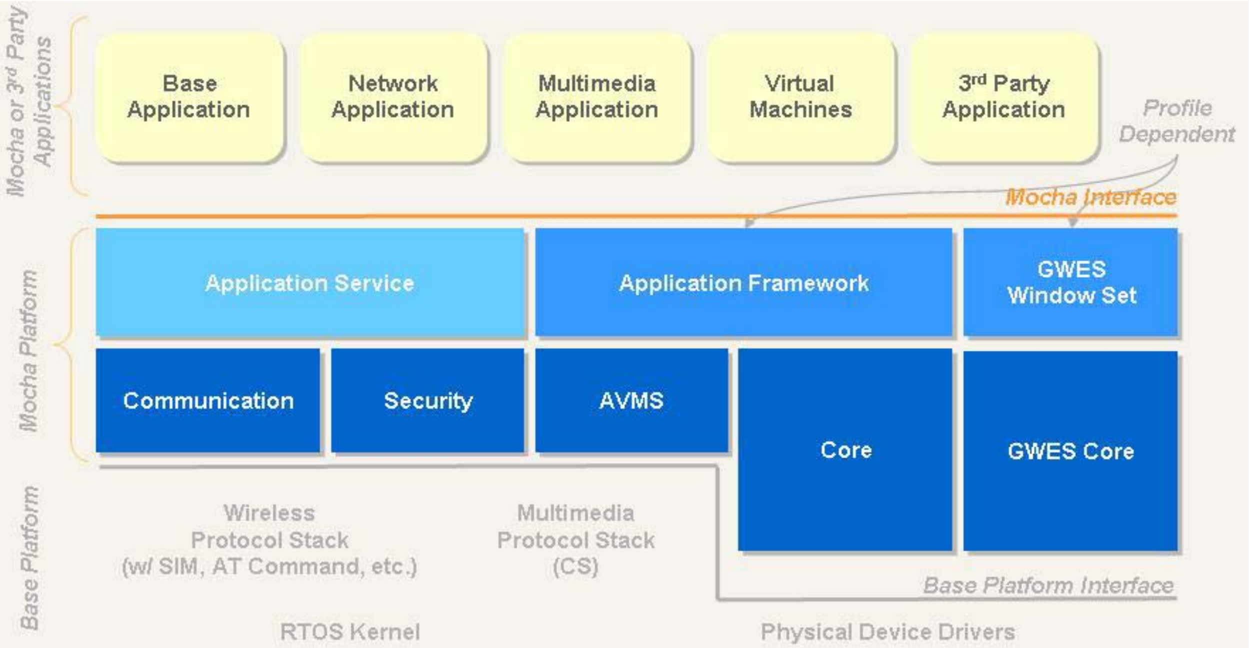


Figure 2. Top-level Mocha architecture

Mocha also provides toolkit. This toolkit includes platform simulator, device builder, diagnostic monitor, font tool, image converter, resource compiler, SIM (Subscriber Identity Module) editor, file explorer, and 3D data converter.

Platform simulator is a front-end GUI for testing Mocha applications by dynamically loading device configuration information. It provides GWES (Graphics, Window, and Event Subsystem) for frame buffer drawing, LCD speed control and IME window. Key recording and replay is another feature of platform simulator.

Using device builder, developers can test event mapping for regions, memory size, and LCD position and size. Another useful tool, diagnostic monitor, supports resource monitoring, event filtering, and event injection.

Font tool converts true type fonts to Mocha font types, and image converter changes Windows BMPs to Mocha bitmap resources. Figure 3 shows some of tools in Mocha toolkit.

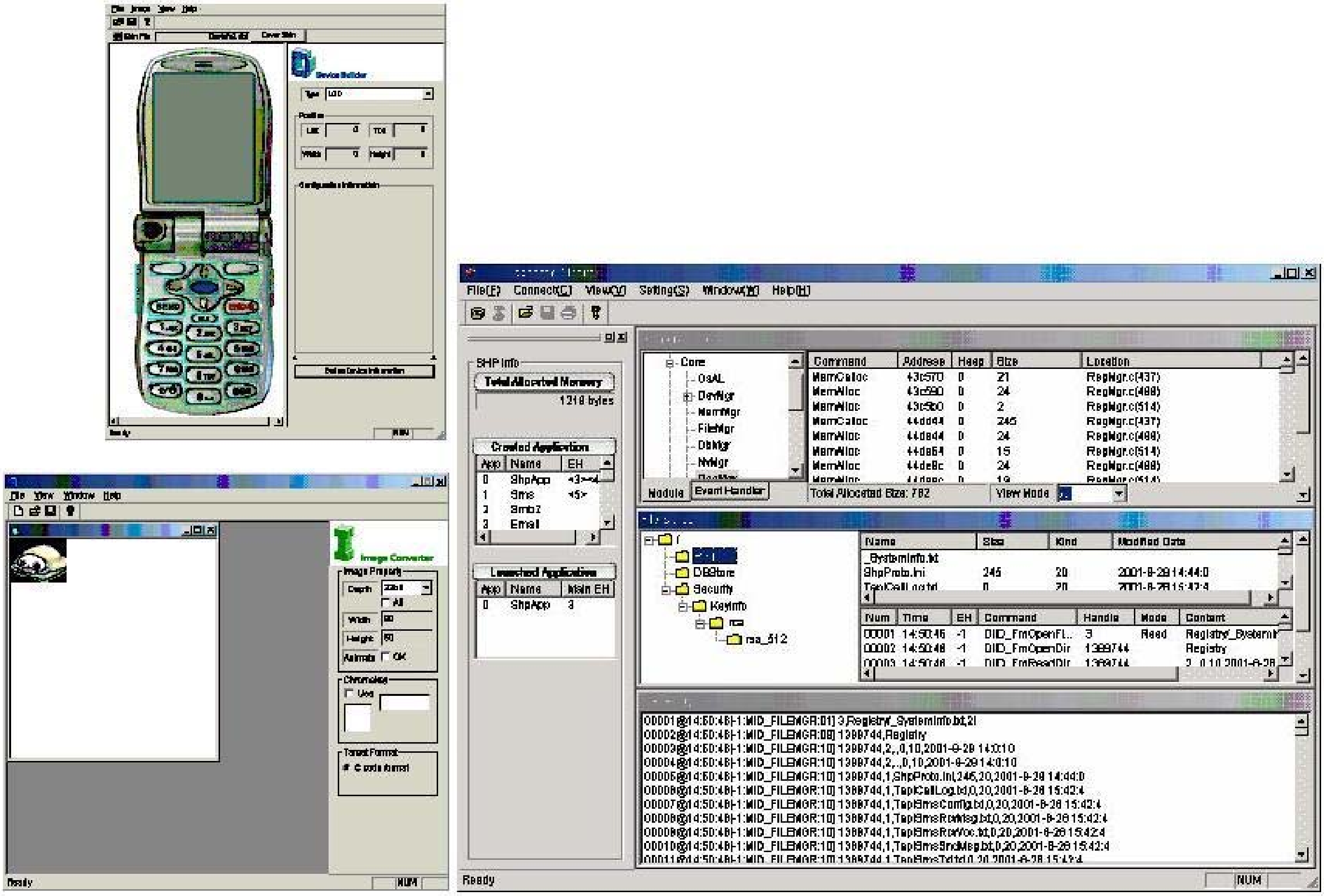
These Mocha's efficiency, portability, modularity, configurability, extensibility and interoperability make it quick and suitable to develop new applications without modifying for various wireless stacks/OS/chipsets and for different level of platforms

Acknowledgment

We are grateful to the UMTS R&D Department, Mobile Communication Division of Samsung Electronics. This research was performed while Kibum Kim was an intern in the UMTS R&D group at Samsung Electronics.

References

[1] Software Center, Samsung Electronics Co., Ltd, "Mocha: Part1. Programming Mocha Applications, Programmer's Guide," 2003.



App	Name	EH
0	ShpApp	<3><4
1	Sms	<5>
2	Smb2	
3	Email	

App	Name	Main EH
0	ShpApp	3

Command	Address	Heap	Size	Location
MemCalloc	43c570	0	21	RegMgr.c(437)
MemAlloc	43c590	0	24	RegMgr.c(488)
MemAlloc	43c5b0	0	2	RegMgr.c(514)
MemCalloc	44dd44	0	245	RegMgr.c(437)
MemAlloc	44de44	0	24	RegMgr.c(488)
MemAlloc	44de64	0	15	RegMgr.c(514)
MemAlloc	44de8c	0	24	RegMgr.c(488)

Name	Size	Kind	Modified Date
_SystemInfo.txt			
ShpProto.ini	245	20	2001-8-29 14:44:0

Num	Time	EH	Command	Handle	Mode	Content
00001	14:50:46	-1	DID_FmOpenFl...	3	Read	Registry_System
00002	14:50:48	-1	DID_FmOpenDir	1389744		Registry

```
00001@14:50:48|-1:MID_FILEMGR:01] 3,Registry\_SystemInfo.txt,2|
00002@14:50:48|-1:MID_FILEMGR:08] 1389744,Registry
00003@14:50:48|-1:MID_FILEMGR:10] 1389744,2,.,0,10,2001-8-29 14:0:10
00004@14:50:48|-1:MID_FILEMGR:10] 1389744,2,..,0,10,2001-8-29 14:0:10
00005@14:50:48|-1:MID_FILEMGR:10] 1389744,1,ShpProto.ini,245,20,2001-8-29 14:44:0
00006@14:50:48|-1:MID_FILEMGR:10] 1389744,1,TapiCallLog.txt,0,20,2001-8-28 15:42:4
00007@14:50:48|-1:MID_FILEMGR:10] 1389744,1,TapiSmsConfig.txt,0,20,2001-8-28 15:42:4
00008@14:50:48|-1:MID_FILEMGR:10] 1389744,1,TapiSmsRcvMsg.txt,0,20,2001-8-28 15:42:4
00009@14:50:48|-1:MID_FILEMGR:10] 1389744,1,TapiSmsRcvVoc.txt,0,20,2001-8-28 15:42:4
00010@14:50:48|-1:MID_FILEMGR:10] 1389744,1,TapiSmsSndMsg.txt,0,20,2001-8-28 15:42:4
```

Figure 3. Device builder, diagnostic monitor, and image converter in Mocha toolkit