A Device-independent Design Scheme on Interactive Animation and Game

LI Xin-yu, SONG Ying, HUANG Xia, XIANG Li-sheng, SHEN Qing

***Abstract*—Today, there are almost 136 million Internet users and 500 million mobile users in China. Wireless networks are striding toward wide band service. Animations and games are urgent to seize hold of both Internet and wireless market simultaneously, i.e., to attract more users. However, this is a difficult task indeed, because animation and game designers must take environments of both Internet and wireless network and consistency across devices in consideration. A scheme to solve above design problems with minimum labor cost is proposed. Also, many practical solutions to problems posed by limit physical resources of mobile are discussed.**

***Index Terms*—Interactive Animation, Mobile Game, Heterogeneous networks, Device independent design.**

I. Introduction

IT and communication are developing rapidly. The growth rate of IT industry in China keeps above 20% every year. Today, among 136 million Internet users in China, above 2/3 is using high-speed internet. In America, more and more high-speed internet users are watching movies, playing games to kill time via stream video/audio platform. The community predicts, by 2010, the wide band users in world will rise to 500 million. In the world, the increase of mobile user exceeded 638 million in 2006. In China, there are more than 500 million mobile users using UMPC, PDA and mobile phone to meet their needs.

The combination of wireless business with wideband networks is inevitable and can't be hindered. Producers of popular recreational applications, such as animations, games, wish more and more users would adopt their software, whether through Internet service or wireless access. They wish that they could hold both Internet and wireless markets and possess all potential consumers, including users with mobiles and TVs, in addition to PC users. However, it is also a difficult task indeed. Since animation and game designers must take different factors into account, such as runtime settings, Internet or wireless network. It is troublesome for an application to adapt to diverse physical facilities with various functions and capabilities. That is, if mobile phone and personal computer solve same calculation problem, the difference of physical resources and I/O operations between them will lead to heavy burden in program design and implementation.

Facing such challenges, we present a scheme on the premise that labor cost is minimum and attain much better results.

We need to introduce some definitions in order to state the main idea.

Definitions 1: If an application program can allocate and use a certain device's physical resources to implement desired capacities and functions, we call it a special program oriented to a specific device (or a device-dependent program). And if an application program can allocate and use the physical resources through a general operating system, we call it a device-independent program.

Definitions 2: Say "⟷"as "…is equivalent to …", "→" as "is a", while "+" and "|" are connectors who connect a required component, an optional component, respectively. More specifically,

<Device-independent interactive animation/game > ⟷ <physical device> + <leading OS> +<application> + <interaction>

<Physical device > → <PC> | <laptop> | <UMPC> | <PDA> | <mobile phone>

<Leading OS> → <Windows> | <UNIX> | <Linux>| <Symbian>

<Application>→<animation>| <game>

<Interaction> → <The interactions controlled by user > | < The interactions controlled by system>

That is, a device-independent interactive animation (game) is an (a) animation (game) to be played with different physical devices, under diverse operating systems, and be interacted with various interactions.

Physical device can be a PC, a laptop, a UMPC, a PDA, or a mobile phone.

This work was supported by National High Technology Project (2005AA114090).

Li Xinyu is the general manager of Hunan Talkweb Information System Corp. He is a senior architectural designer and interested in system architecture, mobile communication, etc.

Song Ying is the vice general manager of Hunan Talkweb Information System Corp. He is interested in mobile communication, multimedia, etc.

Huang Xia is the assistant of general manager of Hunan Talkweb Information System Corp. She is also an International Financial Manager. She is interested in demand analysis, product design, etc.

Xiang Lisheng is a vice technical supervisor in Hunan Talkweb Information System corp. He is interested in MM, animation, etc. E-Mail: 13974838381@hnmcc.com

Shen Qing was a professor in the Institute of Computer Science at National University of Defense Technology, China. Now he is the first chief technical consultant of Hunan Talkweb Information System corp. He is interested in pattern recognition, AI, MM, animation, etc. E-Mail:sq1950224@yahoo.com.cn

A Leading OS can be one of these operating systems, Windows, UNIX, Linux, or Symbian.

Application can be an animation or a game.

Interaction may be controlled by a user or by a system.

II. Extend SMIL to realize interactive functions

SMIL is a multimedia manipulation language recommended by W3C. It can synchronize and combine multimedia components. SMIL 2.1, W3C recommendation 13 December 2005, includes many new features, such as tiled background picture in visual presentations, fades in audio presentations, etc. But so far it cannot process backtracking in media, which is the basis of switch in tree structure and backtracking in graph structure, and only supports media to play in stream format. Backtracking is a required function in interaction between human and machine/human.

We extend SMIL compliant with XML specification and add some elements/attributes, such as Select, Parents, Option, Getinfo, to realize interactive SMIL links. This is so-called TW-SMIL, which can implement interaction between human and machine/human. In client, such as a mobile phone, decoder, developed in J2ME, fulfils media playing.

TW-SMIL 1.0 indicates a SMIL document as selectable by adding a Meta statement “selectable” in “head” tags. For example,

<head>
<meta name="selectable" content="TW-SMIL"/>
</head>

Note that a general SMIL document contains no such Meta element.

Select element

Name attribute of select element can be used as common identifier of all options in player menu. The related attributes include parents, option, and Getinfo:

1) Parents (support backtracking)

Parents are the immediate predecessor option through which one can arrive at this Select and which can be used in navigation backtracking. Parents may have 0 or many options. If the value of parents is 0, it represents a root; and if value of parents is greater than 1, it may implement the reuse of SMIL and present a navigation graph to a user. “src” of parents indicates the location of SMIL navigation file.

2) Option (support branching)

Select element may have 2 or above option attributes. Option id is used in interaction of systems, option name is a description of player menu, and option src indicates location of corresponding SMIL file.

3) Getinfo

Getinfo provides an easy access to information needed for implementing interaction between human and machine/human from system or user. Getinfo itself possesses id, name, value, from, src, etc.

For example, when getinfo is used to represent “ratio of current popularity”, it will have the following descriptions,

<getinfo id=1 name=”excellent” value=”XX” from=”system_dbms” src="..\dbms\statistics\select percent from XXXX where …… =xx”</getinfo>

<getinfo id=2 name=”good” value=”YY” from=”system_dbms” src="..\dbms\statistics\select percent from XXXX where ……=yy”</getinfo>

<getinfo id=3 name=”bad” value=”ZZ” from=”system_dbms”

III. A scheme to realize device-independent programming

Definition 1 and Definition 2 imply that a device-independent application can implement the same predefined function with different physical devices under various operating systems. Specifically, Definition 2 requires that applications be able to run at 2 entirely different kinds of computing machines, PC and mobile phone, which may be equipped with Windows or Symbian. We shall present the state-of-art technology in mobile phone.

It is known to all, Java Virtual Machine technique is ubiquitous and pervasive in high-end office and amusement equipments. Contrarily, C, C++ and C# still have not widely used in these products. From the evolution process of numerous cell phones, such as Nokia, Motorola, we can estimate computation power, development trend and compatibility to Java of current mobile phones.

While CPU used in these new-style phones is mainly ARM4T at 104/123/165/220 MHz, sometimes ARM5 at 220 MHz is also used. Main memory of a phone is 2.3~8 MB. The computation power of a phone configured as above may only be equal or sometimes inferior to, a Pentium III PC pre-installed with Windows 95. But these phones still can do many calculating works in addition to human communication. Not only can they support data service of GPRS Class 10, but also support all built-in JAVA Virtual Machine service, CLDC1.1 and MIDP (Micro Information Device Profile) of J2ME. The Nokia intelligence phones pre-install a Symbian 6.0/7.0/8.0 OS, while Motorola intelligence phones pre-install Linux. More and more mobile phones support MIDP, and are ready for "intelligence phone".

Until now, the technical scheme to accomplish device-independent application program is clear. That is, using JAVA calculation fulfills all “device-independent” application programming. Concretely, fulfill server platform in J2EE, implement web-based client’s program (Applet) running in PC (and UMPC) in J2SE, and accomplish client’s program (MIDP) running in a mobile phone in J2ME.

Since the J2EE and J2SE have already been admitted and employed extensively by the community, they are not discussed more, and it is J2ME that will be focused on.

IV. Key points in J2ME developing

According to our practice, when Java architecture is deployed, a smart global scheme must be designed first. We must solve the problem in two directions: while we keep kernel routines device-independent, we should also provide effective ways to make it fit a device-specific platform. Thus, software components division and standardization, module or subroutine design is done first. Then, write kernel routines and place them into module with meticulous care, so as to realize the reuse of kernel routines as much as possible at Applet level and MIDP level. Thirdly, write the I/O routines respectively, which cannot be reused.

Under such a technical scheme, we estimate that at least 50~70% labor cost could be saved. Table 1 shows the labor savings in developing an application, Bridge. The rate of reduced cost depends closely on the manifestation of a work, the complexity of picture and the switching density (frequency) between keyboard and screen display. In other word, when routines contain more and more I/O operations, the saved labor cost is less and less.

Table 1 One example of labor savings in program development

Module	Main Functions	JAVA Swing Lines of Code	Reusable in J2ME Lines	Reusable Ratio
Function	Sequence, whoNeeds, etc,	159	159	100%
Engine	mousePressed, actionPerformed, etc,	473	330	69%
Paint	drawCa, showCard, showPushedCa, showContract, etc,	785	0	0%
Dispatcher	AIPushCa, dispatcherAttack, distributer, etc,	1004	804	80%
prepare	disArrangeCa, prepareCa isSupportFileSystem, etc,	213	213	100%
rule	calcuValue, findLowest, roleAAttack, roleBDefence, roleCAttack, roleDDefence, etc,	1125	1125	100%
calcuAndDraw Points	calcuPoints, updateTable,etc,	254	52	20%
MakeContract	makeAContract, setContract, etc,	398	398	100%
Total		4411	3081	69%

Some problems related with J2ME developing closely will be discussed below:

1. Similarities and differences between J2ME and J2SE

Essential contents in J2SE are still used in J2ME, mainly including: 1) Basic idea of OO: concept of class and object, inheritance, polymorphism, etc; 2) Syntax foundation: data types, key words, and operators, etc; 3) Exception handling; 4) Multi-thread and multi-session processing.

The contents not used in J2ME mainly include: 1) “javac” and “java” command in JDK; 2) some classes are not included in J2ME, and even if they are included, the methods of them are reduced. For example, the Applet, AWT, Swing, can't be used at all. Table 1 shows only 69% of source code in JAVA Swing may be reused in J2ME.

The currently unfulfilled functions of J2ME include:

1) To migrate source code from J2SE to J2ME directly (without rewritten), especially some special methods (written in J2SE to show some scenes) used frequently in cartoon, game, etc. In such case, almost 30~50% labor cost should be increased to achieve device-independent programming.

2) To modify content in buttons dynamically.

3) To accept Chinese input on canvas.

4) To operate local resources, such as address/phone dirctories, received messages, etc.

2. To deal with the limited calculating resources of mobile phones

J2ME application program uses two kinds of memory management, global and peak memory management. Global memory management is used to decrease the total memory requirements, while peak memory management is used to decrease the memory requirement when the memory requirement of a certain program is increasing.

In terms of safety, the CLCD’s garbage collector is not so good as J2SE’s. When overloading or assigning object too fast, the collector won't collect garbage in time and the efficiency of application will undermine. Considering such a limit in J2ME, we should decrease the memory requirement as much as possible. There are several efficient ways.

A. The simplifier the program, the better the application

According to J2EE design principle, an application should be divided into a lot of objects (classes), and all subroutines are defined as methods of corresponding classes (In fact there are no standalone functions or procedures in Java.) But J2ME design requests a program as simple as possible to improve efficiency. Under such a consideration, we declare each module or subroutine with Midlet if possible, and pack several Midlets into a Midlet package. In this way, the application manager can manage Midlet and its corresponding resource efficiently.

B. The smaller the program, the better the application

The size of the J2ME program is very important to valid deployment. We should delete those unnecessary modules or subroutines inside the class which may not be used currently, to reduce the whole size of an application. When an application is deployed in a wireless network, the smaller application needs less time to be downloaded and deployed. And the compatibility of a smaller program with other applications is much better than a larger one.

C. To reduce the total memory needs for the program as far as possible, mainly:

1) Adopt a simple data type (scalar type) if possible, instead of a complex type such as an object, array, etc.

2) Declare less objects if possible. While declaring an object, the system should allocate corresponding space on running heap memory. So it is a good idea to declare an object only when it is used instead of declaring many objects long before they are used. Also, don't relate a class with excessive number of instances. And if an object is not used in program any more, its identifiers should be assigned as null immediately.

3) Declare a data type according its true accuracy needs. Declare shorter data type, such as Boolean, byte, short, etc. instead of int, if possible. Of course, these trivial tricks have less impact on the programs than on J2ME.

4) Reuse any thing as far as possible. Let many identifiers from the same object be reused in different periods of a program's lifecycle, for example, the reuse of some large arrays, allocated memory, "lazy initialization" of instances, etc. Although this contradicts with the principle of software engineering design, but it benefits smaller equipments, such as mobile phones, severely.

5) Avoid creating an object inside iteration.

6) Check the current usage of memory frequently. The related methods include FreeMemory and TotalMemory. For example, when the memory overflows, handle the OutMemoryError exception by program and don't wait the operate system to do it.

7) Release the resources just in time. You'd better release the resources such as file, server connection, etc, immediately while it's no longer used. Program should implement a necessary garbage clear itself, rather than depend on garbage collector or host environment.

8) Use local variables if possible. In J2EE developing, a programmer is accustomed to setting data members as global data type in a class but using fewer local variables. These global data members should be supported by data management and stack operation from system. This will lead to unnecessary waste of CPU. Assigning value to local variants can bypass several steps to get values from a class' data member, thus leading to the decrease in CPU consuming. Although this will abandon the advantage of data encapsulation in an object, when the program runs in a small machine such as a mobile and deals with a great deal of data, the processing speed should be preferred.

3. The necessity of test on real mobile phone and its implementation

Various J2ME emulators on PC bring great convenient for the J2ME application development. We can not only set a series of break points and watch/set the values of every global/local variant, implement debugging with emulator, but also inspect the program's current running result on the emulator screen (including appearance, color, etc.) But the physical capacity of a true mobile phone is far inferior to PC. Especially, when animations or fighting games run on a mobile with limited CPU speed and memory size, we always feel the speed is too slow, and the frame refresh delay is too long, etc. Further, if the mobile phone can't provide a dynamic adjustment to memory while the program requests a larger memory, it will cause a machine down. So testing the application on a real mobile is a necessary.

The main test items include: 1) function, 2) operability, 3) Whether the package size of Jad plus Jar can be supported by most cellular phones? (For example, the acceptable size of a Nokia S40 is only 64KB.) 4) Whether the speed of a program running can be accepted or not?

Delivering a Jad+Jar package to the machine may take one of the ways as below: 1) OTA, 2) A data line, 3) Infrared transducer, 4) Blue tooth.

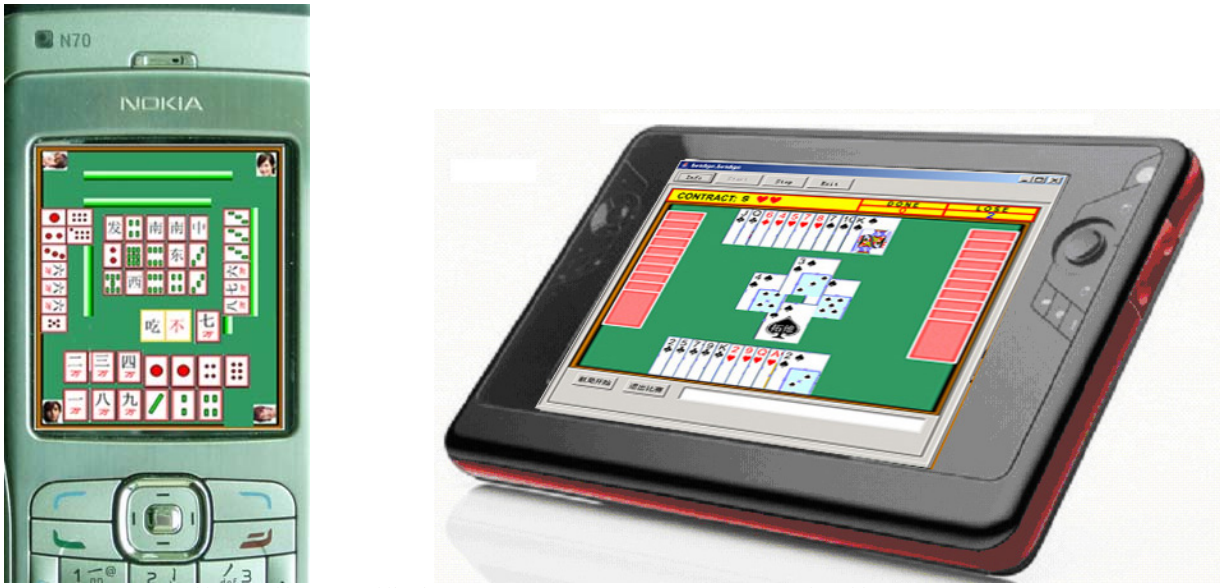
4. A reasonable appearance design

Compared with the resources such as memory and CPU etc., the screen of a mobile phone is the most precious physical resource. Whether how large a screen of a mobile phone reaches, it can't be compared with UMPC. The first important design principle for pictures shown on a mobile phone is "brief and appropriate". Nice or elegant only is a second consideration. In other words, first you must let the player get a complete, fluency playing, and then pursue elegant pictures. Efficient methods include: turning graphic elements to vectors, appropriate description, and frames switching just on time…

It is noted that the MIDP only supports the PNG format in 256 color resolutions currently. More color layers are not available yet.

Figures 1 a) and 1 b) show the picture designed for an interactive Chinese mahjong in mobile phone developed in MIDP and an interactive game of bridge in UMPC developed in Java Swing respectively. As any card in bridge can't be inversed (even in a mobile phone or Web), so we design all pictures in one direction only and set another suit beside the digital (Fig. 1b). In this way, you can see the card clearly in two directions (in row or in column). Thus, we get a brief and appropriate expression.

As the more detail technology scheme for Interactive playing in GPRS, one can see reference [1].



a) Chinese mahjong on mobile phone b) Bridge on UMPC

Fig 1 Examples of device-independent application

V. Conclusion

This paper provides two definitions to restrict problems discussed in a limited scope and points out the general requirement for creating, testing, and deploying an interactive popular application adapted to different networks and different devices simultaneously. Focusing on this requirement, we provide design scheme to solve the real-world problems caused by mobile phone with very limited physical resources.

References

[1] Li Xinyu, Song Ying, Xiang Lisheng, Shen Qing, A New Fashion of Digital Animation--Interactive Mobile Animation, the 2007 International Conference on Wireless Networks (ICWN'07)
[2] Loren Terveen, Elena Papavero, Mark Tuomenoksa, DynaDesigner: A Tool for Rapid Design and Deployment of Device-Independent Interactive Services. http://www1.acm.org/sigchi/chi95/proceedings/workshop/lgt2bdy.htm