RADICAL SIMPLICITY FOR UNIVERSAL ACCESS

INTRODUCTION

Even the poorest of the poor will pay for a service, if that service improves in someway their quality of life. Several corporates are now addressing rural markets and they have the need for information and communication infrastructure in remote rural locations.

For achieving this the Simputer project was conceived during the organization of the Global Village, an International Seminar on Information Technology for Developing Countries, conducted during Bangalore IT.com event in October 1998.[1]

If the right service is made accessible in the right way information technology can impact the lives of the people all over the world .The Simputer is a low cost portable alternative to PCs, by which the benefits of IT can reach the common man. It has a special role in the third world because it ensures that knowledge of English is no longer a barrier to handling a computer. [1]

Simputer's About:

- Bridging the Great Digital Divide
- Affordable Computing: A device which can bring benefits of IT to the people[7]

The key to bridging the digital divide is to have shared devices that permit truly simple and natural user interfaces based on sight, touch and audio. It is designed to be modular and extensible and based entirely on free software from the open source initiative.

Developers

Simputer was developed indigenously by scientists from the Indian Institute of Science and technologists of Encore Software. They have started a commercial venture, PicoPeta Simputers Pvt. Ltd., to ensure that it does not, like many other promising projects, remain on the drawing board. The Simputer Trust is a non-profit trust created basically to develop technology that will help take information technology to rural areas. The Managing Trustee of the simputer trust is

Mr.Vinay Deshpande. The other trustees are Vijay Chandru (IISc), Shashank Garg (Encore), Vivek.K.S (IISc), Swami Manohar (IISc), Mark Mathias (Encore), and V Vinay (IISc). Rahul Matthan (Trilegal) is the legal counsel for the Simputer Trust and has played a key role in defining the Simputer General Purpose License.

Affordable Computing

The projected cost of the Simputer is about Rs 9000 at large volumes. It uses all off-the-shelf components so they are mature technology components that do not have a very high price.[4]

Simputer Vs Palm

Simputer is not a PDA, it is a lot more powerful than a typical PDA; it has a 206 MHz RISC processor. The screen size is 320x240, with memory of 32 MB RAM. It is a complete Linux machine, it runs x-windows; we can run many x-applications by simply recompiling for the ARM processor. We have xterm and xclock running on the prototypes of the Simputer. [7]

Similar devices

The designs similar to the simputer are: the Japanese Morphy One, the Pengachu and more recently the Brazilian 'VolksComputer'.[1]

SIMPUTER ARCHITECTURE

Hardware

CPU	Intel's StrongARM SA-1110 CPU running at 206 MHz
Memory	16-64 MB of SDRAM 08-32 MB Flash for non-volatile storage
Display Options	240x320 LCD Colour or Monochrome Display Panel with backlight
Input Device	Touch-panel Overlay on LCD Display with a plastic stylus (Pen) Direction and Selection Keys
Audio Interface	Audio Codec Support for external head-set
SmartCard Interface	SmartCard Reader/Writer
USB Interface	USB Port
Connectors in Basic Unit	SmartCard Connector RJ-11 Telephone Jack USB Type-A Connector AC Adapter Input
Power Supply	2xAA-sized NiMH batteries Internal charge management Operates with external AC Adapter

System Software

Operating System	Linux Kernel 2.4.18
Soft-Modem Algorithms	V.34/V.17 Data/Fax Modem Technology
Network Protocols	TCP/IP, FTP, Telnet, PPP, HTTP etc.
Perl/Tk scripting environment	

Application Software

IMLI	IML browser
Tapatap:	Input method
Dhvani	Text-to-Speech Software
Internet Access	Browser, Email, File-Transfer
Music	MP3 Player
PIM Applications	Notepad, Address Book, Calculator

Accessories

Expansion Docking Cradle

Compact Flash [CF-II] Slot, USB Slave and Serial Port

Development Tools

Software Development Kit [SDK] consisting of

- ARM Cross-compilation Tool-chain
- Utilities[8]

HARDWARE

Smartcard interface

Smartcard driver for the simputer is a driver for the Philips TDA8008 smartcard controller. It uses the ALPAR protocol to communicate with the controller. The controller supports ISO7816-4 complaint cards following the T=0 and T=1 protocol. Drivers are available for Linux/x86, for use with the smartcard development board and for Linux/arm, for use with the simputer.

The Smartcard driver for the Linux/x86, uses the serial port to communicate with the development board. In the case of the arm platform, the driver uses a kernel module to communicate with the controller. This is because a polling serial driver is used rather than the default interrupt-mode serial driver in Linux. Currently, the smartcards that have been tested include BULL-CP8, Schlumberger Payflex and Philips DS Personalization cards. [2]

SYSTEM SOFTWARE

The boot loader : Blob

The boot loader used on the simputer is called blob. Blob for the simputer is present at offset 0 on the flash of the simputer, and is responsible for loading the kernel and ramdisk images from flash to DRAM. It also displays a logo on the LCD of the simputer when it is loading the kernel and ramdisk. This logo is also stored on the flash and can be replaced by our favourite logo too.[1]

The Simputer Linux kernel

The Linux kernel for the Simputer is based on the arm patches to the Linux kernel maintained by Russel King and Nicolas Pitre. This has been modified to work on the Simputer. The current kernel version is 2.4. The source code includes the drivers for flash, audio , keypad on the simputer, and simputer touch screen and frame buffer.[1]

The Ramdisk image

The Ramdisk image is transferred to DRAM by blob, during boot time, and is used by the kernel as a root file system on ramdisk. It contains the GNU C library and associated libraries, a shell, busybox, various startup scripts and a flash driver to mount the flash file system. The Ramdisk image is a compressed ext2 file system created using the loopback file system.[1]

The flash file system

The flash file system on the simputer contains libraries required for XFree86 clients (libX11, libXaw, libXt...), Perl 5.5, Perl/Tk, perl modules required for IMLI (XML::Parser, MIME::Base64), rxvt, xclock, chimera, Tapatap, wmx, kernel drivers for the simputer, all compiled for the arm. The flash file system is an ext2 file system. It also includes various X11 fonts that are required by the applications.[1]

Modifications to X-Windows and Window Manager

Modifications to XFree86 4.0.1

X window gives some X windows related developments including the modified wmx window manager used on the Simputer and the patch to X needed for portrait mode 4-bit display. XFree86 4.0.1 runs in portrait mode (required by most handhelds), for displays whose depths are 8, 16, 24 and 32 bits.[1]

Modifications to the window manager (wmx)

wmx-5 requires clicks from both the mouse-buttons and some keyboard clicks for effective usage. Since the right mouse click is not easy on a touch screen, and since the simputer does not have a keyboard, modifications have been done for easy usage on the simputer. A launch menu has also been added. The modified wmx follows the same licensing model as wmx.[1]

The Simputer Softmodem

The modem for the Simputer has been written in software and currently works at 2400 bps without error correction. The code currently is capable of uploading a file from the Simputer to the remote host and has been implemented in user level.[1]

The softmodem distribution includes the following files:

1. A user level V22 softmodem (2400 bps)
2. A driver for the telecom codec (ucb1300)
3. Userland utilities for controlling the telecom codec like going off/on hook, changing sampling rates, buffer sizes etc.

SIMPUTER SOFTWARE PACKAGES

IMLI: The IML browser

IMLI is an abbreviation for IML interface. The purpose of IMLI is to provide a simple and consistent interface for displaying information and developing applications that are simple, user friendly. IMLI supports display of Indian languages, and is also integrated with a speech-synthesis system, that is capable of synthesizing voice in Indian languages. The speech synthesis system is distributed separately. It uses a protocol called ITP, IML transport protocol.[1]

The novelty of the Information Markup language (IML) browser (user-interface of the simputer) is:

- uniformity across diverse applications
- ease of use
- support for multilingual text and speech output
- support for smart card usage.

Tapatap: cool character composition

Tapatap is a method for generating keystrokes to be sent to other applications, for devices, where a keyboard is absent. Tapatap uses a 3x3 grid for recognizing characters. For example, each character of the Kannada alphabet can be generated by "tapping" on the cells of the 3x3 grid in a particular sequence. The figure generated by connecting the "tapped" points, roughly resembles the way the character is written. Tapatap starts of in "letter" mode; it can be changed to go into

"number" mode by clicking on the button at the bottom. This brings up the numeric telephone style keypad, for number entry. Clicking again on the button at the bottom, brings it back to the "letter" mode. [1]

DHVANI: The Simputer Text-to-Speech Software

DHVANI gives resources needed to set up text-to-speech synthesis in Indian languages. Using images in conjunction with voice output in local languages makes the Simputer accessible to a larger fraction of the Indian population. Currently, Dhvani has a Phonetics-to-Speech engine which is capable of generating intelligible speech from a suitable phonetic description in any Indian Language. In addition, it is capable of converting UTF-8 text in Hindi or Kannada to this phonetic description, and then speaking it out using the Phonetics-to-Speech engine.[2]

Technical Description

Text to Phonetics routine

The Text-to-Phonetics routine for Kannada/Hindi reads a text in UTF-8 format and converts this text into phonetic description. Indian Languages are, by and large, phonetic in nature, so this task does not present major challenges. Hindi, however, turns out to be an exception. Eg:why is karna, i.e. to do, pronounced karna and not karana? The Hindi routine uses a new algorithm to determine where this implicit vowel occurs; this algorithm works correctly for all basic words but goes wrong on compound words, like DevNagar ,which it would pronounce as Devangar. Kannada poses no such problems and is relatively straightforward. Sample UTF-8 files for Hindi, Kannada, and

phonetic demo files for Hindi, Kannada, Tamil and Malayalam are included with the distribution. [1]

Phonetics-to-Speech Engine

The Phonetics-to-Speech Engine works by diphone-concatenation. It uses a database of about 800 basic sounds, which are pitch-marked. All the engine does is to read the phonetic description, identify the appropriate diphones, concatenate them at pitch-marks, and play out the resulting signal. To reduce the database size, we use an open-source implementation of the GSM 06.10 RPELTP compression standard. This reduces the database size to about1MB (even though it appears to be 2MB due to fragmentation).All basic sounds are recorded at 16000Hz as 16 bit samples.[1]

INFORMATION MARKUP LANGUAGE (IML) 0.8

Abstract

IML is an XML (Extensible Markup Language) application so it follows the Internet standards.It is used for describing the content and applications handled by a Simputer. The goal of IML is to enable handheld computers, that have limitations in the display size and input capabilities, to access and render content from the Internet. Use of simple icons to be selected by pointing with a stylus and text-to-speech output in the local language are the two primary means of communication. The user is not expected to be familiar with the currently widespread user interface paradigm -windows, slidebars and pull-down menus. [1].

IML Encoding

All application data is transferred between IML applications in a simple self-extracting format called IML encoding. The underlying assumption is that all data is in the form of a simple three column table. It can contain an arbitrary string, including characters that may have special meaning to XML, or to the IML encoding scheme itself. The following example illustrates this coding scheme:

Name Magic Value

---

name s Manohar

address s CSA Department, ISRO, Bangalore 560012

DOB s June 21 1960

Status e567rty Married

phone ru87iii89 3092368

choice t345tyu pizza

The table has six rows and 3 columns.

The IML encoding of the above table will be a single long string as shown below.

S[6,3:4:Name1:s10:S. Manohar7:address1:s38:CSA Department, ISRO, Bangalore 5600126:choice7:t345tyu5:pizza3:dob1:s12:June 21 19605:phone9:ru87iii897:30923686:status7:e567rty7:Married

The general format of the string is as follows: start with an 'S[' followed by the number of rows and the number of columns in the table, separated by a comma. The ':' (colon) is used as a separator between the contents of each cell in the table. For example, in the encoded string above, after the initial "S[6,3:", (indicating a table with six rows and three columns) 4:Name indicates that the next 4 characters are to be read as a key, followed by a single character ('s' for secure variable), followed by a 10 character string which is the value for the key just encountered. [1]

IML Overview

The basic unit of an IML document is a *page* . An IML document can be thought of as a deck of pages. A page contains three types of elements:

Immediate Content: These elements are a small subset of the content markup present in HTML. Since the display of a Simputer is much smaller than desktops, the range of features displayed is limited. For instance, frames are not very meaningful. Some elements in this category are *text, table, itemized text, image*, *audio* etc.

Generated Content: These are elements useful in generating content either by applications running locally or by remote applications. Some elements in this category are *appl, input, output*, *select* etc.

Control elements : These elements control the rendering of the other elements. Some elements in this category are *iml, head, b, i, u, large, small, lang* etc.

IML Syntax

IML syntax is governed by the rules of XML syntax and its grammar is specified by a Document Type Definition (DTD): the details of using tags, attributes, entity references and so on are defined in the XML language specification and the details about IML element, attribute names and their nesting etc. are specified in the IML DTD.

<title> *-- Title element*

Description:

Element for holding the title of the document. Example of title can be seen in various IML examples presented in this document.

<iml> *-- The root element*

Description:

This is the root element of an IML document and holds all its content.

Attributes of <iml>: None.

<head> *-- Header Information element*

Description:

Currenlty the head element simply contains a title element. Subsequent versions of IML will include elements for author information and other meta information about the document.

Attributes of <head>: No attributes.

<text> *-- Text element*

Description:

For presenting text content.

The text element is distinct from text directly occuring inside a page element in that its attributes enable pieces of text to be hyperlinked. Otherwise, the presentation of text in this element can be modifed by the use of b, i, u, small and large elements.

<itemize> *-- Element for listing*

This element provides a simple listing environment. The attribute *style* controls whether it is a numbered list or a bulletized list. A list element can contain most of the other elements.

Attributes of <itemize>: style = enum|bullet

<image> *-- Image display element*

Description:

The image element unlike its HTML counterpart, is quite complex and allows for interesting image presentation and manipulation. IML images have the notion of a canvas on which the pixels are rendered. Several images can be placed on this canvas and moved around on the canvas. Since this is an empty element, the various values of the attribute control the effect of the image element.

Attributes of <image>:

The following is the list of attributes:

id: attribute identifies each image entity and is useful in image manipulation, both by subsequent image elements as well as by the update element (discussed later).

mode: attribute can have one of three possible. values create|src|draw.

When mode=create, the width and height attributes specify the size in pixels, of the canvas on which subsequent images can be placed.

When mode=src, the image file (only gif format is supported) specified by the src attribute is placed on the canvas created by an earlier image element, identified by the tagid attribute.

The xoff and yoff elements can be used to specifiy the offset from the origin of the canvas at which the image is to be placed.

When mode=draw, it is possible to specify line and arc drawing using the *draw, coord, extent, fill* attributes. The *config* attribute can be used to scale or move the images placed on the canvas by preceding image elements.

audioformat, audio, audiomode and speak: These attributes are valid only when mode=src, and they allow audio content to be associated with an image. Look at the semantics of the <audio> tag for more information.

<***setattr***>
Description:Changing various attributes of elements

<***input***> -- *Input element*
Description: The input element is the basic element for obtaining user input, primarily in forms.
Attributes:
There are several attributes to the input element. these are
type: text|password|check|radio|
width: width of the text area
height: height of the input text area
var: name to be asociated with this input field
magic: one of r|e|s|t (respectively, read, exclusive,secure and transient)
value: value of the variable

The var attribute has an important role in enabling IML to deal with smartcards. The var attribute can take any alphanumeric string as its

value with a special case of a name starting with the underscore('_') having a special significance. See IML and smart card for details.

For a variable name starting with an underscore, the value of the variable is to be obtained from the smartcard, using the appropriate passwords input by the user. The level of security and access is defined by the magic attribute as follows:

magic=r : The variable is readable by anyone without a password. For changing that

variable, a password is required.

magic=e: The variable is readable and writable with passwords.

magic=s: The variable is secure. It is readbale with a password, but cannot be

changed from the IML environment.

magic=t: The variable is transient and can be accessed using a password, but

has no permanence beyond the current session. (i.e. the value is not stored on the smartcard)

Display and markup in IML

Indian language audio and display

Markup:

```
<iml>
<head><title>Indian Languages</title></head>
<page>
This is a demo of Indian languages being displayed and
synthesized<br/><br/>
<!--
Hindi : speech through phonemes, Hindi text using ISO8859-1 data
-->
<audio audioformat="tts" speak="s3m pHy6 tt1r G3000 2p k2 G3000
0s v2 g1t G3000 k1r t2 G3000 h9 G10000 \n"/>
<lang script="hindi">rnÃ =e bu Ã´Jtd;T</lang><br/>
<!--
Hindi : speech through itrans, Hindi text using ISO8859-1 data
```

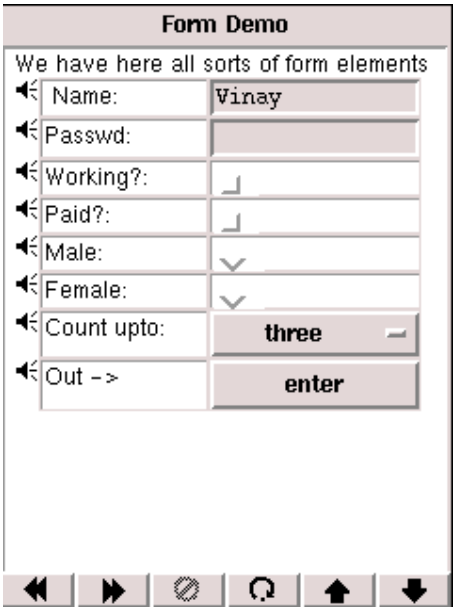
```
-->

<audio audioformat="hindi" audiomode="itrans" speak="hi.ndI me
s{}vAgata"/>
<lang script="hindi">rnÃ =e bu Ã´Jtd;T</lang><br/>
<!--
Kannada : speech through phonemes, Kannada text using ISO8859-1
-->
<audio audioformat="tts" speak="s3m pHy6 tt1r G3000 n3m m7l l1
r1n n5  G3000 0s v2 g1 t3 s5t t1 d8 G10000 \n"/> <lang
script="kannada">OÃšÂ«ÃšÂsÃšÂ¥ÃšÃ†Ã‡
Ã‘Ã›ÃŠVÃš}Ãš</lang><br/>
<!--
Kannada : speech through itrans, Kannada text using ISO8859-1 data
-->
<audio audioformat="kannada" audiomode="itrans"
speak="kan{}naDadal{}li s{}vAgata"/> <lang
script="kannada">OÃšÂ«ÃšÂsÃšÂ¥ÃšÃ†Ã‡
Ã‘Ã›ÃŠVÃš}Ãš</lang><br/>
<!--
Kannada : speech through itrans, Kannada text using UTF8
-->
<audio audioformat="kannada" audiomode="itrans"
speak="kan{}naDadal{}li s{}vAgata"/> <lang script="kannada"
mode="utf">à²¸à³• à²¸à³• à²µà²¾à²—à²¤</lang><br/>
<!--
Tamil : speech through phonemes, Tamil text using itrans data
-->
```

```
<audio audioformat="tts" speak="s3m pHy6 tt1r G3000 11n g1 ll15k
k15 G3000 n1l v1 r1 v15 G3000 k6 r3 g3 r1 d15 G10000 \n"/> <lang
script="tamil" mode="itrans">tamiJil{} vaNak{}kam{}</lang><br/>

<!--
Telugu : speech through itrans, Telugu text using itrans data
-->
<audio audioformat="telugu" audiomode="itrans"
speak="sim{}p{}yUtar{} mIku s{}vAgatam{} palukutO.ndi"/> <lang
script="telugu" mode="itrans">telugulO    s{}vAgatamu</lang><br/>
<!--
English : speech through phonemes, Text using ASCII(UTF8)
-->
<audio audioformat="tts" speak="s3m pHy6 tt1r G3000 v7l k1m 0s
G3000 y6 G3000 tt6 G3000 d3s G3000 3 n13 gHy5 r8 sh1n  G10000
\n"/> <b>Welcome in English</b><br/><br/><br/>
</page>
</iml>    [1]
```

User input form



Mark Up:

<iml>

<head>

<title>

Form Demo

</title>

</head>

<card>

We have here all sorts of form elements

<table>

<tr><td> Name: </td><td><input type="text" width="15" height="1"

var="var0" value="_name" magic="s"/></td></tr>

<tr><td> Occupation: </td><td><input type="text" width="15"

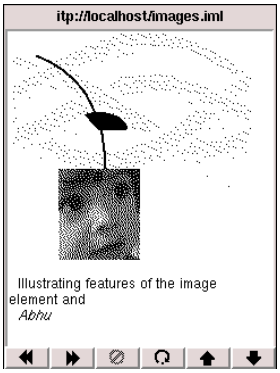
height="1"

var="var0" value="_occupation" magic="s"/></td></tr>

```
<tr><td>Passwd: </td><td><input type="password" width="10"
var="var1" value="Hello!"/></td></tr>
<tr><td>Working?: </td><td><input type="check" var="var2"
value="working"/></td></tr>
<tr><td>Paid?: </td><td><input type="check" var="var3" value="get
paid"/></td></tr>
<tr><td>Male: </td><td><input type="radio" var="var4"
value="male"/></td></tr>
<tr><td>Female: </td><td><input type="radio" var="var4"
value="female"/></td></tr>
<tr><td>Count upto: </td><td>
<select var="var5">
<option value="1" label="one"/>
<option value="2" label="two"/>
<option value="3" label="three"/>
<option value="4" label="four"/>
<option value="5" label="five"/>
</select>
</td></tr>
<tr><td>Out -> </td><td>
<output label="enter" method="exec" anchor="persist.pl">
<collect varid="var0"/>
<collect varid="var1"/>
<collect varid="var2"/>
<collect varid="var3"/>
<collect varid="var4"/>
<collect varid="var5"/>
</output>
</td></tr>
</table>
```

```
<update class="input" varid="var5" label="three" do="set"/>
</card>
</iml> [1]
```

Image Display




```
 <iml>
<page>
<page>
<image id="img1" mode="create" width="210" height="200"/>
<image id="img0" tagid="img1" mode ="src"
src="/simputer/imgs/spec/big.gif" border="0"/>
<image id="img4" tagid="img1" mode ="draw" draw="line"
coord="20,20,100,50, 75,180" width="2" smooth="1" dash="-" />
<image id="img5" tagid="img1" mode ="draw" draw="line"
coord="20,20,100,50, 75,180" width="2" smooth="1" />
<image id="img6" tagid="img1" mode ="draw" draw="arc"
coord="50,70,100,100" start="0" extent="120" fill="black" />
<image id="img7" tagid="img1" mode ="src"
src="/simputer/imgs/spec/abhu.gif"
border="0" xoff="40" yoff="120"/>
<br/>
```

Illustrating features of the image element and

 Abhu

</page>

</iml> [1]

USAGE

Text entering

There are two options on the simputer for entering text:

1. one is a soft keyboard, that can be brought up on the touch screen and you poke at it to enter one character at a time.
2. The second option is to use a novel character entry software called tap-a-tap which is similar in spirit to graffitti, but quite distinct. But to enter tons of text using the Simputer, you can attach a USB keyboard. Simputer is not recommended as a mass data-entry device.

Smart Card

The built-in smart card reader/writer of the Simputer is a critical feature that makes the Simputer an ideal device for almost any kind of transaction. In addition, the smartcard is the mechanism that allows a Simputer to be shared among a group of users.

Rural communities could own several simputers and hire these out for usage to individuals based on the ownership of a SmartCard. It has a built-in chip. Each user's Smart Card would contain the minimum "personalization" information required to log into a Community Server(Simputer) which would maintain personalized data about the user, which he can carry around with him. It can hold several hundred letters like bank account information, personal information - driving licence, person's identity, picture and signature. Once inserted into the smart card interface the simputer will read the profile from the smart card and also update changes if any during the current transaction cycle. User profiles can be stored in flash memory as accessible files and also in the smart card. Sharing would bring down the cost of the Simputer to that of owning only a simple smart card, and paying for the usage of a shared Simputer. It is better viewed as a "personalization" and security device.[1]

Storage

Internal storage, of limited capacity, is already available through Flash memory. USB is the medium for access to external peripherals. Products like the M-Systems Disk-on-Key Flash Disk are now available on USB. They can provide reliable storage ranging from 16MB to 1GB in capacity. [1] The Smart Card should not be seen as a storage medium of any significant capacity, though capacities could increase as technology advances.

APPLICATIONS

The architecture of the Simputer integrates various devices such as Smart Card reader, a Modem, a Touch Screen, a Multi-lingual Text-to-Speech system. This makes Simputer an ideal device for:

e-governance

- Smart Card enabled citizen services(Voter IDs, driving license, ration card, etc.)
- Data collection and processing
- Land and revenue records
- Education, health care and information access
- e-mail device [2]

Microbanking

- A Smart Card pass book
- Synchronizing transactional details through modem connectivity
- Interactive multi-lingual transaction log book
- Human error eliminated, increasing the integrity of the calculations [2]

Education

- Interactive text book
- Massive data storage at low costs compared to books
- Universal interface for education in any language at any level
- Automatic adjustment of content based on progress.
- Entertaining and engrossing medium
- Regular download of new educational data without reliance on infrastructure or additional expense[2]

Communication

- Cheap communications device
- High performance communication technologies for the masses
- Data and text transmission, as well as voice
- Potential centralization of the communications network
- Simplifying usage through storage of preferences of each user on a Smart Card
- Simplifying communication by removing the barriers of language and literacy
- Universality of data transmission achieved through use of icons and text-to-speech [2]

Market pricing and agriculture

- A friendly companion to know the current prices of his produce
- A trader looking for right market to sell or buy his goods
- An interactive assistant for a farmer to implement the best farming practices
- Both market and weather forecasting data instantaneously distributed

- Digitization of the barter system via organization of secure transactions using smart cards [2]

Health

- Interactive data collection device for a health worker
- Simple education medium for healthy practices
- Preliminary diagnosis of common ailments via an expert system
- Health schedules, data storage, advice on livestock
- Communication barrier broken between health service workers and rural patients
- Telemedicine : remote health care advice [2]

Technology in everyday life

- Usage in restaurants to automatically report orders to the kitchen
- Digital Assistant and diary options for personal home use
- Portable entertainment on a versatile platform
- Distribution network organization; Simputers carried by delivery agents
- Inventory management made easy
- Integration with Global Positioning Systems for directions and way-finding
- Voice transmission over standard telephone lines in emergency situations
- Global satellite digital broadcasts for educational and entertainment purposes [2]

READING ALOUD

WorldSpace data broadcasting is considered a cost-effective way of making digital content available to a large percentage of the world's population. WorldSpace makes news, music, education and entertainment programmes available to more than two billion people in Africa, the Middle East and Asia.

The tie-up with WorldSpace is a major initiative for the simputer trust, one of the first start-ups. Presently, the data is being picked up by receivers mounted on standard personal computers. Once the content is downloaded via satellite, the Simputer can be disconnected from the receiver and taken around for use by the target users. [5]

PILOT PROJECT

The long-distance education pilot project will get under way soon in the tribal district of Bastar in India's northern Chattisgarh state. The state government is implementing the project for schoolchildren. As many as 2,000 students will benefit, in the first phase from this programme. These students will receive digital content beamed via satellite from WorldSpace radio broadcasts.[5]

CONCLUSION

What makes the Simputer unique ?

- Portable and a mobile device
- Sharable and affordable
- Integrated Smart Card and Modem
- Multi-lingual text-to-speech system
- Imli makes knowledge of English no longer a barrier to the use of IT
- Images allow universal comprehension of IML content
- Relies on non-proprietary software

The simputer platform technology, being a cost effective platform can be used to develop several other products such as thin clients, cost effective e-commerce device and in embedded systems.[2]

REFERENCES

- www.simputer.org
- www.picopeta.com
- www.express-computer.com/20020401/indcomputes.shtml
- www.wired.com/news/technology/0,1282,44642,00.html
- www.news.bbc.co.uk/1/hi/sci/tech/1560771.stm
- www.geek.com/news/geeknews/2002Jul/gee20020708015274.htm
- www.anchlia.8k.com/AboutSimp.html
- www.ncoretech.com/simputer/
- www.learningchannel.org/views/editorials/EditorialJuly2001.shtml
- www.windows.idg.net/english/crd_internet_192403.html

ABSTRACT

A rapid growth of knowledge can only happen in an environment which admits exchange of thought and information. Indeed, nothing else can explain the astounding progress of science in the last three hundred years. Technology has unfortunately not seen this freedom too often. The solutions to bridging the much hyped and talked about digital divide can come from within the developing world itself Problems of access to telecommunications in the developing world have often paled into insignificance beside those of gaining access to a working computer capable of connecting to the internet. For a vast mass of the rural poor for whom a computer is probably as remote an option as a trip to the moon, the Simputer can well become the power button to prosperity.

Simputer (Simple Computer) is a low-cost, portable alternative to personal computers. It is pegged as the first of its kind in the world as it promises to ensure that knowledge of English is no longer a barrier to handling a computer. It permits simple and natural, user-friendly interfaces based on sight, touch and audio.

ACKNOWLEDGEMENT

I thank God Almighty for the successful completion of my seminar.

I express my sincere gratitude to Dr. M N Agnisharman Namboothiri, Head of the Department, Information Technology.

I am deeply indebted to Staff-in-charge, Miss. Sangeetha Jose and Mr. Biju, for their valuable advice and guidance. I am also grateful to all other members of the faculty of Information Technology department for their co-operation.

Finally, I wish to thank all my dear friends, for their whole-hearted co-operation, support and encouragement.

CONTENTS