# Health Care Information Appliance

Poznan University of Technology
Poland

Application ID: 4DZB7H
Region ID: 8

Project team:

Wojciech Gryncewicz
Grzegorz Jachimko
Tomasz Janiszewski
Arkadiusz Waliszewski
Piotr Zieliński

Project mentor:

Jan Kniat

# 1. ABSTRACT

Health care is, and hopefully will continue to be, one of the most important issues in the modern world. Everybody agrees that prevention is better than cure. However, patients much more often see their doctor for treatment than for preventive tests and/or regular check-ups. The main purpose of our project is to help change this trend, by encouraging people to become more involved in their own health care.

At the moment, health service organisation in Poland, and probably in many other countries, does not encourage people to carry out preventive tests, because of limited funding, and long queues at doctors' surgeries. Moreover, patients do not have access to their medical history since it is placed at many specialists' files. Finally, patients often find it difficult to obtain answers to their questions and doubts, as reliable medical information is printed in specialised magazines, using medical jargon hardly comprehensible for ordinary men.

The system developed by our team – the Health Care Information Appliance – is meant to change this state, at least in some aspects. Our main aim has been to create a portable system which will gather various kinds of personal medical information, allow the patient to browse them, update them and provide the GP with easy access to information about the patient's health.

Since health related data is confidential, securing it against unauthorised access or modification has been one of our main priorities. Another priority is simplicity of user's procedures, because potential users of our system are not expected to be familiar with computers.

# 2. SYSTEM OVERVIEW

Health Care Information Appliance (HCIA) is a portable, free-standing device with easy connection to Internet. HCIA which allow users to input health related data from different sources (called below "measurement devices"), browse them and share them with an authorised medicine doctor via Internet. The medical information gathered is stored locally and backuped in a dedicated server. It is protected from unauthorised access. HCIA communicates with a dedicated Internet server, allowing automatic updates of all its software. The system is an operational environment for program modules (downloaded from the network on user/doctor request). The modules deal with various health care aspects, and control various measurement devices. They are called below "health care modules". All health modules can be easily installed and uninstalled. Some particular health care modules, which due to their subject should not be used without a doctor's supervision, can only be installed by a qualified medicine practitioner.

The HCIA user interface consists of a simplified keyboard with only a few function keys, and optionally assisted by a voice commands engine and of a LCD screen and sound feedback. The simplified keyboard and easy navigation is meant to be a help for not technically inclined users. Patients identify themselves by a hardware key and a short password. All stored information is protected by a cryptographic engine based on the hardware key, and cannot be accessed without it in any way. The same device can be used by many users, each user equipped with his/her key and password. Each user can use many different health care modules. Information regarding particular users is stored in user profiles. User profiles can be remotely accessed by authorised persons (i.e. doctors). There is a backup of the user profiles on dedicated server. All user data transfers via Internet use secured protocols.

HCIA was created with one basic assumption – to be an open architecture system. So the proposed system is just an operational environment. It is divided into a few separate modules, which perform all basic functions required.

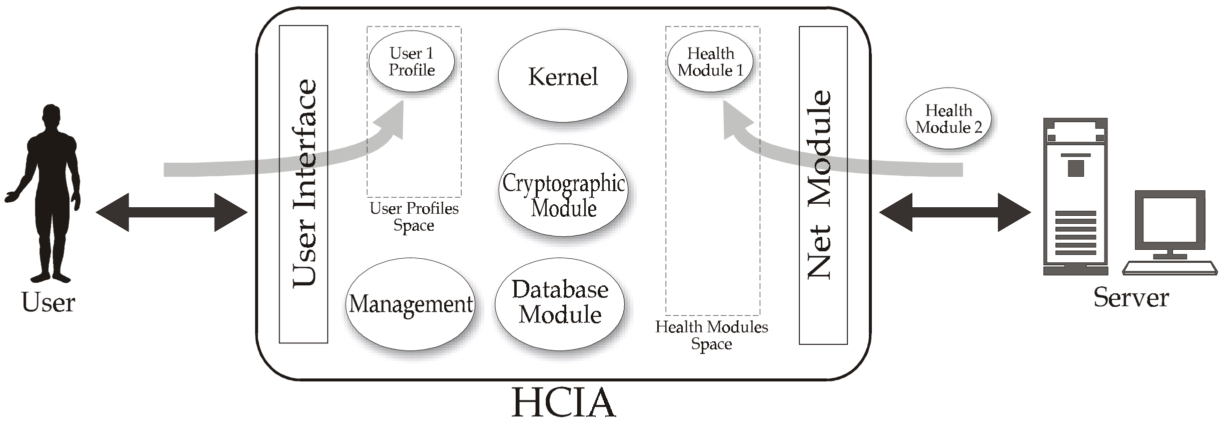

**figure 1**

The User Interface on the figure 1 consists of input (keyboard, voice command engine and hardware key), output (LCD screen and sound) and all measurement devices. Through this interface a new user can start using the system, all medical information is entered into the system, and access to the whole Health Care System is provided.

The Net Module's role is to download new modules from server, to ensure data backup and to provide any additional information required by other downloaded modules.

The Cryptographic and Database Modules are tightly connected – their task is to secure all user's information, either stored locally or transferred via net, form unauthorised access.

The Kernel provides basic, low-level system functions.

In the HCIA system there is also space for user profiles and health care modules.

In order to illustrate the system possibilities, we have implemented two example health modules. The first one is a simple module allowing the user to download and browse health related articles from the dedicated server. Our aim was to show how the system operates with Internet. All articles on the server are placed there by medicine doctors – this assures their correctness.  All articles should be written in simple terms and in easy language. Because we didn't want to create a new file format, articles are prepared in the simplified HTML standard.

The second module exploits the English Method of natural birth control – this is a very novel method, which uses three factors to determine the exact time of woman's fertility with the highest possible probability. Procedures folded in this module help gather, store and calculate information about a woman's fertility cycle. An external thermometer for this module is required, but there is a possibility of inputting measured data directly from keyboard. The thermometer is our example measurement device.

The HCIA system is based and tested on the QNX operating system. From the point of view of performance requirements the Project Kit delivered by IEEE is sufficient. Because we assume that all medical data are saved on the server side, there is no necessity of storing the same older data on the local hard disk. Hard disk drive's capacity of 260 MB is enough for HCIA needs. Because the voice command subsystem performs a lot of real-time vector calculations, a MMX™ class processor, such as the supplied K6 3D, is required.

In creating the HCIA we used the waterfall system development model with increments. After each increment we obtained a properly working version of the system. Subsequent increments extended system functionality. The HCIA system and all its components were designed in the Unified Modelling Language (UML). We have chosen this methodology because it is a world-wide accepted standard. Therefore our UML diagrams will be easy to read and understand for system analysts from many countries.

We believe that our Health Care Information Appliance is a very useful construction both for patients and doctors. It is a completely new approach to the health care protection process. The created system can be easily operated by people unfamiliar with computers. The HCIA system may also improve GP's work efficiency by faster access to a patient's medical history.

# 3. IMPLEMENTATION AND ENGINEERING CONSIDERATIONS

## 3.1 System requirements specification

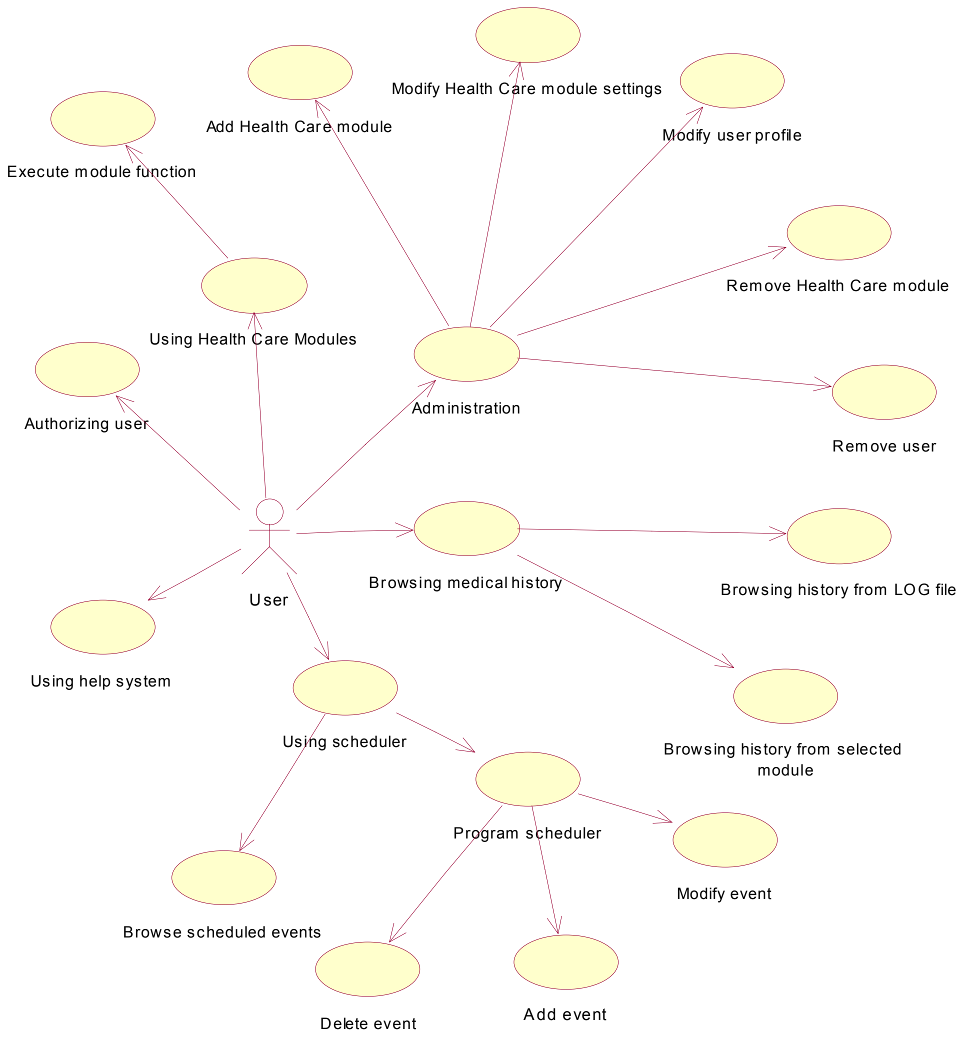

**figure 2**

We assume that there are two groups of users of the HCIA system, i.e. patients and doctors. Doctors are not presented in figure 2 – if the patient is logged into the system, the doctor can access the system at the patient's home. The doctor also has direct access to user's medical information stored on server, but this function is not part of our system, so it is not presented here. Our main concern was the comfort of the patient user. All implemented functions were analysed for maximum usability.

The functions available to the user are as follows:

*3.1.1 Authorising user* – the function is called when the patient inserts his/her hardware key into the system. This feature is necessary for system start-up. After positive authorisation, the user gains access to all stored encrypted information.

*3.1.2 Administrating* – this group of functions allows both patient and doctor to configure the system's basic modules and health care modules, add and remove health care modules, modify users settings and remove logged user from the system. As it was said, some health care modules can be installed only by a doctor – it is fulfilled by doctor's authorisation on the server where all modules are stored.

*3.1.3 Using scheduler* – this function is universal and independent of any modules, so it is implemented as part of the system. It allows to program the system so as to remind the user about certain health care events (e.g. taking a pill or carrying out a test).

*3.1.4 Using health care modules* – a list of all installed modules available for a logged patient is displayed. The patient may choose a module on request and then execute its specific functions. The module's functions are also displayed in the form of a list.

*3.1.5 Browsing medical history* – this function allows the user to browse all stored events, recorded by health care modules. There are two ways of displaying this information – displaying the history from one module selected by the user, or displaying the full history, generated from LOG file – this file saves all events from all modules. The second option should be especially useful for the doctor – it can help find correlation between different measured factors.

*3.1.6 Using help system* – all basic system functions and health modules functions have enclosed help files. This feature allows the user to become familiar with system possibilities.

## 3.2 Logical view of the system

As mentioned above, Health Care Information Appliance is a system whose task is to gather, process and store user's medical information. Now we will present the whole system from the user's point of view.

### 3.2.1 HCIA and external environment

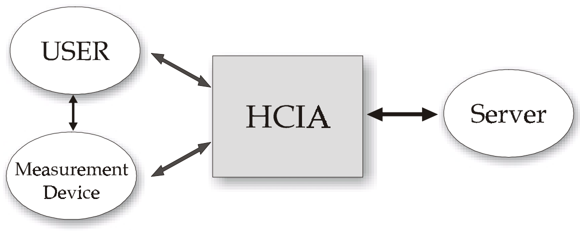

**figure 3**

Health Care Information Appliance communicates with three kinds of external entities: user, dedicated server and measurement device. Doctors communicate with the HCIA system via server – all patient's medical information is stored there, and only the authorised GP has access to this data. The GP may also access information stored locally – when he sees his/her patient at home.

Communication with the user is provided by input and output devices: input – keyboard, voice commands engine, hardware key, and output – on-screen information and sound. The measurement device is usually an input device (e.g. blood-pressure tester, temperature tester), but may need some output data as well (e.g. additional display). Communication with server includes health care modules download, software updating and backup of updated user data on server – to prevent data loss in case of device failure and to make it accessible for the doctor when the user is not on-line.

## 3.3 HCIA software

### 3.3.1 HCIA software architecture

We concentrated on creating an operational environment for HCIA rather than on creating a closed system for just one purpose. It required developing an open architecture system, which can be easily extended by installing additional specialised health care modules. We decided to create a set of basic modules, which provide all necessary functions for simple installation, configuration and management of new system extensions. Thanks to modularity, each part of the system can be very easily replaced with a newer version. Each module knows what type of functions are provided by other system parts, but does not have to know anything about how it is done.

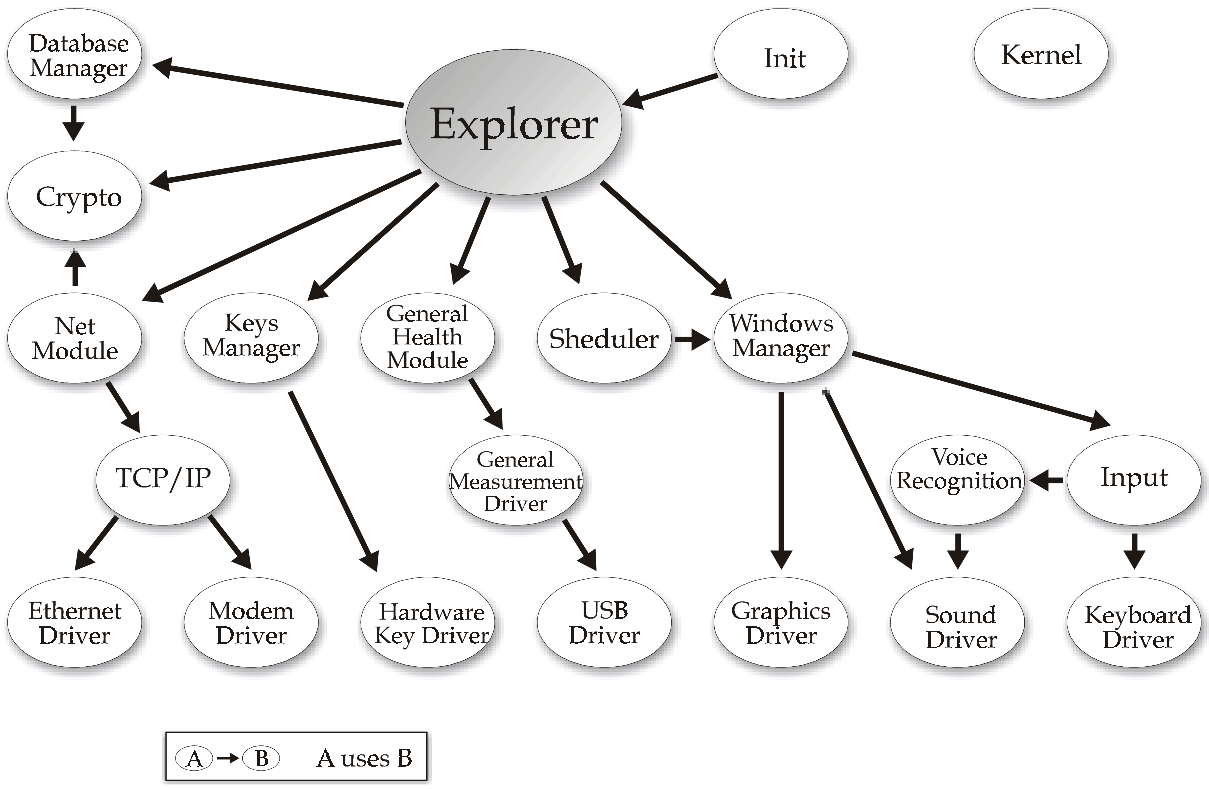

figure 4

*3.3.1.1 Kernel* – is the smallest but the most important part of the system. It provides the set basic functions, used by all other modules.

The primary role of Kernel is to provide communication between processes. However, Kernel doesn't participate in communication directly. It configures other processes, so they can communicate between each other. The only exception is the system start-up.

Synchronic communication is provided by the operating system QNX. But asynchronic communication is also required. It is realised by creating a few new processes, communicating synchronously, which are responsible for delivering all messages to the receiver.

Kernel also provides a simple interface, which is used to run and shutdown modules. It also prevents running a few copies of the same module. This function is realised as a "link count" (when the module's link count is equal to zero – the module is no longer needed, and can be freed).

Finally, Kernel makes it possible to upgrade modules – this feature is supported by QNX.

*3.3.1.2 Init* – this module is run first. Its task is to run all basic modules. It is also a module manager – this function differs from Kernel's functions (Kernel manages modules as a memory manager, and Init is a kind of small database including all information about installed modules or their versions).

*3.3.1.3 Explorer* – provides services for standard system functions. Allows Windows Manager (which provides user's interface) to operate menus filled with basic system functions (e.g. profile management, scheduler configuration).

*3.3.1.4 Database Manager* – allows storing all medical information generated by different users. At this level of the database there is only basic information (such as user identifier, module identifier, exact date and time of medical event). This is enough to generate a detailed history, because the description of medical information is stored in health care modules.

This module also provides internal data encrypting – for that purpose it uses the Cryptographic module.

*3.3.1.5 Network module* – provides high level Internet access. It allows communication with server via secured connection. It is used to download new modules or their upgrades, to perform medical data backup and to download required articles.

*3.3.1.6 Scheduler* – helps manage periodical tasks. It can be configured directly by user or by health care modules (e.g. carrying out measurements, taking a pill). It is also used by the system to check for new module updates.

*3.3.1.7 Window Manager* – it is a high level module, which provides an easy-to-use user interface, and enables fast and understandable data presentation. This module can be adjusted for user's personal needs, e.g. contrasting colours.

*3.3.1.8 Cryptographic module* – cryptographic procedures such as Triple DES, hashing function SHA-1 and Diffie-Hellman's key negotiation protocol are included. This module is used by the Database and Network modules.

*3.3.1.9 Voice recognition* – it is responsible for processing voice samples, and decides whether a sample is a correct command or not.

*3.3.1.10 Input* – uses both voice recognition and keyboard, and unifies commands from these modules. Higher level modules do not know if the user gave e.g. "left" command using voice or keyboard.

*3.3.1.11 General Health module* – this is an abstract class of modules which extends the HCIA system's range of application. It consists of functions, which gather, modify, process and present user's medical information. In our project such modules are the English Method of natural birth control and Medical Articles Downloading From Dedicated Server.

*3.3.1.12 General Measurement driver* – this is an abstract measurement device driver, using USB interface. It provides functions necessary for sending commands and gathering information from the device. In our project, an example of this driver is an external thermometer driver.

*3.3.1.13 USB driver* – this module is responsible for maintenance of the USB interface, detecting new hardware, and data transmission. This module wasn't provided by QNX – we had to write it ourselves.

*3.3.1.14 Graphics driver* – this module provides all graphics functions required (e.g. drawing lines, filling, displaying bitmaps or text). Because of performance requirements it relates directly to hardware, and uses the 32bit assembler language and MMX instructions. We use 640x480 resolution in High Color.

*3.3.1.15 Key manager and Hardware key driver* – those modules provide communication with hardware keys used in our system.

*3.3.1.16 TCP/IP, ETHERNET, Modem* – modules provided by QNX, responsible for Internet communication.

*3.3.1.17 Keyboard driver* – responsible for maintenance of all keyboard-entered commands.

*3.3.1.18 Sound driver* – maintenance of sound card. Provides functions to save and play sound samples.

### 3.3.2 Detailed specification of chosen modules

#### *3.3.2.1 Cryptographic algorithms*

##### *3.3.2.1.1 Overview*

<u>Hardware keys</u>

Some cryptographic algorithms used in HCIA are based on hardware keys – specialized cryptographic devices. Due to its their internal memory protection feature, no data or code stored in a hardware key can be accessed. Another useful hardware key's feature is its internal hardware random numbers generator. Both these features are used in the authorization process.

Encryption algorithms

In encryption processes some well known, standard algorithms were used. We decided that algorithms tested in real life would be much more reliable that any algorithm developed by our team.

The following algorithms were used in the project:

- Symmetric encryption/decryption algorithm - Triple DES (3DES)
- Hash function – SHA-1 (Standard Hash Algorithm)
- Key negotiation protocol –Diffie-Helman's protocol

#### *3.3.2.1.2 Authorization*

After the user inserts his/her hardware key, the system, reads the unique key serial number and determines if the user accesses HCIA for the first time or has a profile already set. In the first case, the new user is put on the user list and a new user profile is created. In the second case, a three-step authorization process begins. In the first step, the hardware key authorises itself against the system – HCIA checks if a valid hardware key was inserted. In the second step, HCIA authorises itself against the key – this step is necessary to prevent hardware key scanning. In the third step, the user identifies himself/herself by typing a short password – this step prevents unauthorised persons from accessing medical data even if the hardware key was stolen.

The hardware key host inside HCIA includes another security controller device, the same as the one used in the hardware key, and with same features. System versus hardware key authorization is, in fact, authorization between the hardware key and the hardware key host. Because all data stored inside security controller memory are inaccessible, this mechanism of authorization prevents medical data from being accessed even if system software is cracked and debugged.

## Authorization protocol

Authorization protocol between HCIA (its software and hardware key host component) and hardware key is shown in figure 5. The following symbols are used in the diagram:

? x=y - x and y comparison. If x is different from y, authorization procedure stops.

H(x,y) – the result of SHA-1 hash function with arguments x and y

GK – secret general key, stored in every hardware key and every hardware key host (same key in every device)

$LK_1$ – secret key 1, stored in hardware key

$LK_2$ – secret key 2, stored in hardware key

P – user password

$ID_{KEY}$ – unique hardware key serial number

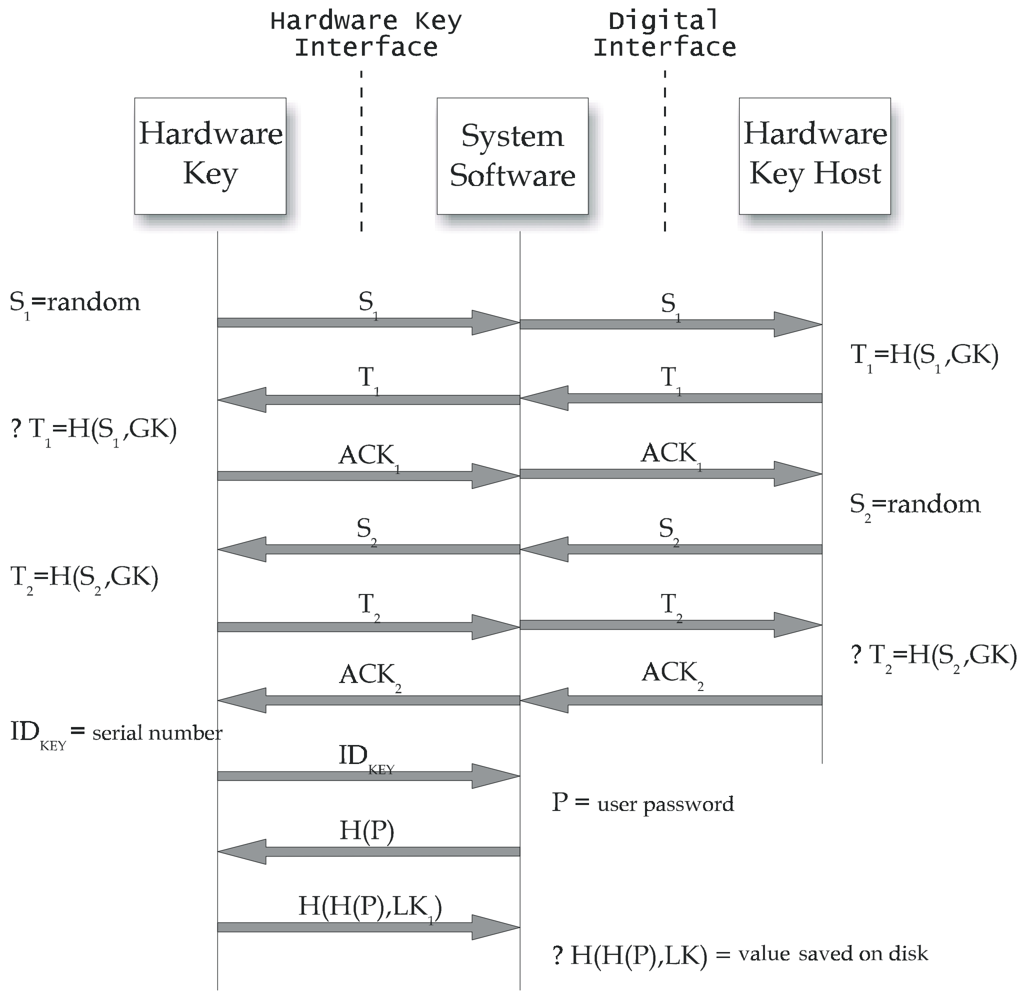

**figure 5**

### *3.3.2.1.3 Database encryption*

The authorization process is not the only prevention against unwanted access. All personal data stored in the device are encrypted, to prevent access by direct hard disk read. The encryption protocol developed by our team uses data from the authorization process, so the user data cannot be decrypted if the authorization process has been skipped.

Data stored in the database are divided into blocks. Each block is encrypted with a different encryption key, changed depending on the block write time and its identification number. In addition, every block includes some random data, so even the encryption of two identical blocks with the same key results in different encrypted data. Because encryption/decryption key generation is based on the user password and hardware key data cannot be accessed without them.

Encryption protocol:

Data stored in the database are encrypted/decrypted with the 3DES algorithm with a changing key, generated using the hardware key device. The protocol of encryption key generation is shown in figure 6. The following symbols are used in the diagram:

H(x,y) – the result of SHA-1 hash function with arguments x and y

I – data block identification number

T – data block modification time

$LK_1$ – secret key 1, stored in hardware key

$LK_2$ – secret key 2, stored in hardware key

P – user password

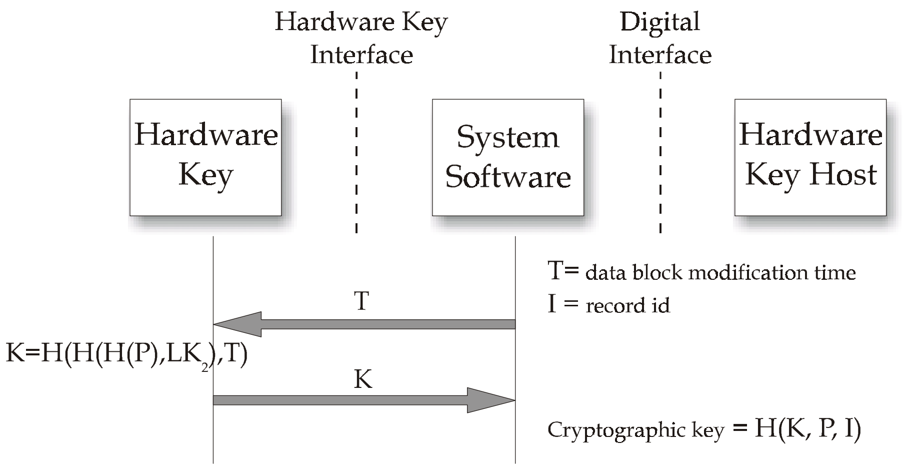

**figure 6**

The value H(P) is known to the hardware key from the authorization process.

To improve database operations time, encryption keys are cached in memory.

*3.3.2.1.4 Internet connection security*

Our system uses the Internet connection to communicate with the dedicated server. Data stored in HCIA would not be safe if Internet data transmission (such as backup) was not secured. That is why we consider connection security essential.

HCIA – server authorization is similar to hardware key authorization. In addition, all data to be sent are encrypted. We achieved it by using the Diffie-Helman's protocol.

Protocol:

- Server and client (HCIA) negotiate key K, using Diffie-Helman's protocol. From now on all data to be transmitted are encrypted using key K and 3DES algorithm.
- Server authorises itself against the client
- Client authorises itself against the server
- Data exchange begins

### *3.3.2.2 Voice recognition*

The voice commands option is intended for people who have problems with using the HCIA keyboard. There are a few basic voice commands – navigation (up, down, left, right), help, accept and back. Those functions are tightly connected with keyboard functions.

If the voice commands option is enabled, data from A/D processor via sound card driver are passed to the Voice Recognition module. The samples are probed at 11kHz with 16bit resolution. Then the module decides if during the last 1500ms any command was given. It is done by calculating the average signal level, and deciding if it is greater than the background static. This operation is repeated after every 128 samples.

If a meaningful signal is detected, it is divided into 256-element blocks, and each of them is processed by FFT (Fast Fourier Transformation). Then the processed blocks (which include real and imaginary values) are transformed into 128-element blocks, by use of L1 Norm (sum of absolute differences). The next step is reduction to 32-element blocks (logarithmic scale is used, and low frequencies are preferred).

Such pre-formed information is compared with the voice command pattern by a dynamic programming algorithm, which finds minimal distance between two vectors – the first vector is the pattern and the second is the given command. In this way, the command with the minimal difference from the tested input is found if the difference doesn't exceed a given tolerance level.

### *3.3.2.3 Health care module – English Method of Natural Birth Control*

This method was developed in the Queen Elisabeth Medical Centre. It can be used in all specific life situations – after giving birth, after miscarriage and during hormonal treatment. It is very well examined and has a very high Pearl's rate – 0,44.

In the woman cycle phase, there are three phases – before ovulation infertility, fertility and after ovulation infertility. To determine woman's fertility three factors are used –

temperature, mucus and cervix symptoms. Information about the last two factors is specified by women themselves. The first one, temperature, can be measured by our thermometer. However, the following conditions must be met:

- the measurement can take place after at least 3 hours of sleep,
- it should last not less than 1,5 minutes,
- it should be done always at the same time of the day,
- all additional conditions (e.g. illness, travel or drunk alcohol) should be noted,

The typical temperature graph is two-phased. Before ovulation the body temperature is low. During and after ovulation the temperature is usually higher (the latter phase lasts about 14 days – there might be a 2 days difference, and the temperature drops before the next cycle).

Mucus and cervix symptoms should be observed during the day, and results should be entered in the evening. All possibilities connected with these symptoms are presented in the form of radio buttons, and the woman only has to select the proper button.

The first six months cycles are used to gather information about the woman's usual fertility cycle, and cannot be used for family planning. In the next months, system using gathered data and currently entered data, will help show in which phase of the cycle the woman is, and if she is fertile or not.

#### *3.3.2.4 Health care module – Articles Downloading*

To prevent the user interested in medical knowledge from being misinformed, and to create for him/her an effective way of finding information of his/her interest, we decided to implement a health care module downloading medical articles from the dedicated server. Those articles are in the form of simplified HTML files. They can contain links to other subjects and are organized in a specific hierarchy. This service is similar to WWW service (but based on a very simple file format and simple graphic form), being a competent source of medical knowledge.

## 3.4 System hardware

### 3.4.1 External hardware interfaces

The user identifies himself/herself by inserting a small form factor called here a hardware key. The hardware key uses its own, specific 4-wire interface, described in 3.4.3.

All measurement devices are connected via USB ports. This allows their easy connection/disconnection while HCIA is running.

The HCIA can be connected to Internet using RS-232 (modem) or Ethernet interface. In future the system can be expanded via software update by USB modem support. Another possibility (for manufacturers) is including in HCIA an internal PCMCIA-based GSM modem.

While HCIA has its own internal audio system, it also has external audio ports allowing external microphone, headphones or stereo system connection.

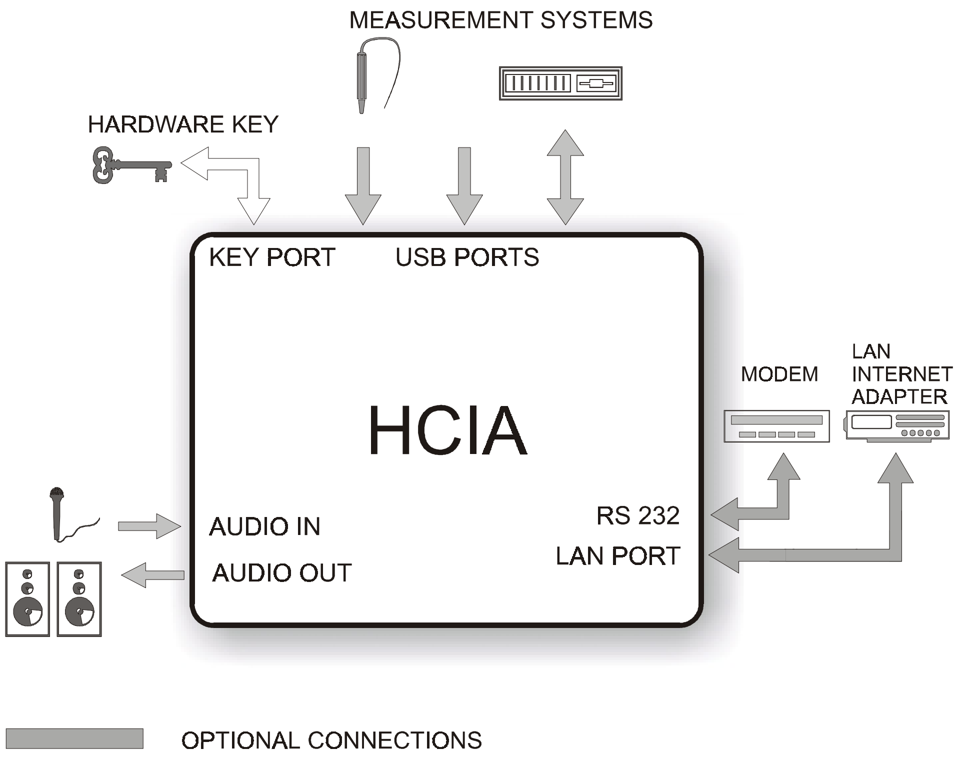

**figure 7**

### 3.4.2 HCIA internal hardware system

Hardware components included in the HCIA are:

- AMD K6-2 main processor unit
- Versa Logic VSBC-6 motherboard with PC-104 to PCMCIA interface and 32MB RAM
- M-2000 8MB flash disk (HCIA system disk)
- Calluna PCMCIA 260MB disk (HCIA storage disk)
- Color LCD
- Simplified 9 key keyboard (not implemented in the prototype)
- PC-104 sound card (developed by our team) with mono amplifier
- Hardware key host (developed by our team)
- Internal speaker and microphone

### 3.4.3 Description of the internal hardware developed by the present team

#### *3.4.3.1 Hardware key and hardware key host*

User authorisation and data security are based on the hardware key Infineon SLE66CX160S 16-bit security controller. The hardware key can be either a small plug or a chip card. It is connected to the hardware key host included in the HCIA system via its own specific 4-wire interface. The key form determines the mechanical interface; since SLE66CX is a chip card controller, the use of the chip card key form and card interface in the final product is recommended.

The hardware key host - a device inside HCIA, developed by our team, drives the hardware key. This device includes another SLE66 security controller unit – this prevents the secret data used in cryptographic processes from being accessed. In fact, even if the HCIA system software was cracked and debugged, secret data wouldn't be readable, due to SLE66 internal memory protection.

The hardware key host device is connected to the system motherboard via Opto22 digital interface.

#### *3.4.3.2 PC-104 sound card with mono amplifier*

To allow the use of the system's multimedia functions, our team has developed a PC-104 SoundBlaster Pro compatible sound card. This sound card is based on ESS 1869 sound chip, and communicates with the motherboard via PC-104's ISA compatible interface. The card includes 1W mono audio amplifier driving the HCIA internal speaker.

HCIA includes an internal microphone and speaker, so the user does not need any additional hardware to use its multimedia functions. Optionally, an external amplifier or microphone can be connected to HCIA via soundcard's "line out" stereo output and mono microphone input.

#### *3.4.3.3 USB thermometer (hardware part of example health module)*

The team has developed a simple USB thermometer device as part of the natural birth control module. This device consists of three main parts:

- National Semiconductors LM76 digital thermometer
- Microchip PIC12C509 microcontroller unit, driving the thermometer and USB bridge, and containing LM76 calibration data
- Thesys TH6503 USB Bridge chip, interfacing between microcontroller and USB line.

## 3.5 Optimization and performance

During implementation, we particularly concentrated on optimization of the most important system elements.

The first element we concentrated on is the Graphics Driver, undoubtedly the most frequently used module. Thanks to the 32-bit assembler code and MMX instruction set, we achieved a significant performance improvement in comparison to C code or other drivers.

The voice recognition module is also highly optimized – it requires plenty of computation power, so its optimization was an important part of our work. At first we optimized the FFT computation function: written in the MMX assembler code, it is based on radix-2 complex butterfly stage which computes 4 16-bit sound samples at once. It turned out to be about twice as fast as its C code version. We also improved some of the L1 Norm calculating functions from the voice recognition module, namely, the complex-to-normalized transformation function and the function calculating the sum of absolute differences between vectors. We again used the MMX assembler code. Finally, the dynamic programming algorithm calculating the minimal difference between two samples was significantly improved by reducing its computational complexity from $O(n^2)$ to $O(n \log n)$. We reduced the algorithm tolerance and, thanks to several performed tests, we created a simplified version of the algorithm. MMX instructions were also used. This part of optimization resulted in the biggest reduction of CPU usage.

In the cryptographic module the DES algorithm was optimized. It was worth optimizing this algorithm because of its frequent use. It was improved by tabling permutation data and by assembler implementation.

## 3.6 Tradeoffs

The design and implementation stages of the project required many hard decisions on our part. The first and the most important decision concerned the whole view of our system.

On the one hand, we had a vision of a specialized system designed for particular needs and purposes. Its advantages were greater usefulness at the moment of hypothetical purchase and shorter time of final commercial product release. Its disadvantages were a narrow group of potential clients and lack of expanding capabilities. Using many specialized, completely different devices for particular purposes could also lead to user confusion, due to different usage procedures etc.

On the other hand, there was the idea of a universal, expandable health care information system. Its advantages were a wide group of potential customers (depending of the number and range of available modules), expandability and comfort of use (the same user interface for all modules). Any specialized device could be implemented on the base of such a system. Its disadvantages were long preparation of the final product, and higher development costs.

We decided to choose the second option.

Short development time and narrow budget made us implement only some elements of the system. We decided to create the operational environment and two simple example health care modules. The most important aspects for us were data security and user’s comfort, so we contracted on these issues.

With regard to data security, we rejected security mechanisms which were simple but easily cracked. Instead, we decided to use safe, hardware solution, resistant against debugging, disassembling and transmission interception.

In the matter of user interface, we decided to develop an intuitive, modal window based system, with easy control by a few keys. Because of that, we couldn't use standard windows of the chosen operating system, and we had to create our own mechanisms gaining full control. We decided to add voice control, to allow partially paralyzed persons. Consequently, we had to overcome software and hardware difficulties connected with implementing such a module.

The choice of the operating system was crucial for implementation issues. We could choose between Windows CE, Linux and QNX. Each of them had advantages and disadvantages.

The first one had good programming tools, and was similar to other well-known Microsoft systems. Its biggest disadvantage, from our point of view, was lack of good documentation describing its configuration, installation and maintaining, as well as modularity problems (especially concerning device drivers).

Linux offered worse programming support but much better documentation. Additionally, it was configurable. Unfortunately, did not solve problems with low level hardware driving and modularity.

The best choice was QNX. It offered good programming tools (although outdated), conciseness, stability and modularity. Especially, its real-time features were useful for our project.

Another decision worth mentioning was the choice of external interfaces. We did not use RS232 or Centronics port and chose USB interface instead. USB provides much better parameters (including real-time transmission), it is expandable and easily reconfigurable (it provides easy connection during system work). It is also becoming a widely used standard for modern devices. Unfortunately, programming USB was much harder than programming other ports (QNX does not have any USB support, so we had to implement the driver ourselves).

## 3.7 Additional tools

During system development, several tools were created in order to simplify the system configuration and exploitation.

The first tool is the dedicated server. It was not our task to develop a fully professional server, but to create its sketch, which would allow to test net functionality of the HCIA. Our server supports encrypted transmission, backup of medical data and articles service (see example module).

Another additional tool is a simple medical article editor. The same editor is used to write system help files. It offers basic edit tools and supports articles hierarchy.

The last tool, the sound samples editor, is used to add voice sample patterns. It allows sound sampling, sound edition (mostly cutting) and has a simple noise reduction module.

## 3.8 Verification and Testing

In the verification process, testing code correctness was essential. We divided this process into two phases. The first was code inspection. After completing each module, his author checked the code for hidden logical errors. His results were then checked by another team member. The second phase consisted of module’s function results check. Specific data was entered on each function input and our task was to compare the obtained results with the expected pattern.

Testing the Natural Family Planning module was a real problem. To check its correctness, a representative group of women should test it over a period of at least 3 months. For obvious reasons, we were unable to test the module in this way. Instead, we entered data taken from reliable sources and compared the obtained results. The algorithms used in this module were tested by another group of students of Poznan University of Technology, as part of their graduation project.

Ten people of different age and sex participated in testing the Voice Command Engine. This test helped to configure the module, so that its performance was improved.

The last aspects to be tested were user comfort and usability of all system menus. The main aim was to check if the user interface was intuitive. A group of 6 people unfamiliar with the project were involved in this part of tests. Thanks to their remarks, a few elements of interface and menu layout were changed.

The whole system was also tested with reference to network operation. Because the server program is not a part of the project, for the test needs we created a “server sketch”, which provides required functions (e.g. supplying health care modules, articles and storing of the medical information backup – at the moment all information stored on the server side is not encrypted).

## 3.9 Costs

HCIA Project Kit was extended by some hardware components, entailing additional costs. Now we will list these components and their respective costs.

### *3.9.1 Sound system*

Multimedia functions are an essential part of the HCIA system. In order to create the base for such functions as voice commands, the Project Kit was extended by a sound card and microphone.

### *3.9.2 Security system*

Our team decided that medical data requires hardware protection. This entails hardware keys cost and the cost of hardware key host development.

### *3.9.3 USB thermometer*

USB thermometer was developed as an example of USB measurement device.

### *3.9.4 Keyboard*

The keyboard was not included in the Project Kit. However, it is necessary for HCIA prototype.

### *3.9.5 Costs summary*

| | |
|---|---|
| ESS1868 sound chip | - $15 |
| Microphone | - $2 |
| SLE66CX160S security controller (3 pcs) | - $36 |
| LM76 digital thermometer | - $3 |
| PIC12C509JW microcontroller | - $15 |
| TH6503 USB Bridge | - $8 |
| Keyboard | - $5 |
| Plugs, sockets and passive electronics components | - $25 |
| Plated circuit boards production costs | - $50 |

All additional circuit boards included in the project were designed by our team.

| | |
|---|---|
| **Total money spent** | **- $159** |

# 4. SUMMARY

Health Care Information Appliance was created with one aim – to be a system prepared and configured for an ordinary human being. Every person who will be using our tool will not have to waste time on configuring dozens of different options.

Thanks to this system, the user will have opportunity to get familiar with many health care aspects. All personal medical information is stored locally, and each patient has direct access to it, without necessity of searching it in dispersed archives.

The presented system performs all the functions we have intended. However, it is just a prototype, with restricted usability, portability and comfort. In a commercial version of the Health Care Information Appliance, it would be necessary to ensure system portability. It could be achieved by installing a low power consumption processor and accumulator, which would allow at least two hours of operability. Low voltage power supply would be required, in order to recharge installed accumulators. Additional possibility of Internet access should be given. We recommend PCMCIA card cellular phone modem – thanks to this solution the system could be called "fully portable".

Adding a touch screen would considerably extend the group of people who could operate our system in spite of their disability. As this is probably the most intuitive kind of interface, we originally planned to include it in HCIA. However, it would exceed the money limit, so we had to give up this plan.

The system's range of application can be easily extended by creating health care modules which can cover all important medicine aspects.

The last part of the project which should be further developed is the health care server. We presented a very simple server, with only a few basic functions. Investing in that part of the system may open new horizons.

## 5. REFERENCES

1. Kutylowski M., Strothmann W.B., Cryptographics, Read Me, Ltd, Warsaw, 1999
2. Flynn A.,Brooks M., A Manual of Natural Family Planning, Unwin Hyman Ltd, London, 1996
3. Urbaniak A., Expert system for diagnosis of women's menstrual cycle using natural family planning method, in: IFIP - System modeling and optimization, J. Doleźal J., Fidler J. (ed.) Chapman&Hall, London 1996(120-127)
4. Urbaniak A., Flejszman M., Owczarzak J., Multimedia software for natural family planning methods, European Institute of Family of Life Education IEEF European Conference, Birmingham, 1997
5. Łuczak B.,Oczkowski S.,Motławski T., Nyga G., Mikołajczak T., Womens' fertile days card editor, with elements of classification, Poznan University of Technology 1997
6. Eriksson H.E., Penker M., UML Toolkit, Wiley Computer Publishing, Stockholm 1997
7. MMX Technology Application Notes, Intel Corporation, www.intel.com
8. AMD-K6 Processor Code Optimisation Application Note, AMD, www.amd.com
9. AMD-K6 Processor Multimedia Technology, AMD, www.amd.com
10. QNX TCP/IP, QNX Software Systems Ltd
11. QNX OS - System Architecture, QNX Software Systems Ltd
12. VSBC-6 Reference Manual, VersaLogic Corporation
13. ESS1869 DataSheet, ESS Technologies
14. Microchip PIC12C509 Datasheet, Microchip Inc.
15. National Data Acquisition Linear Devices, National Semiconductors Ltd.
16. TH6503 Datasheet, Thesys, GmbH
17. SLE66CX160S Datasheet, Infineon Technologies AG
18. Universal Serial Bus Specification rev 1.1