P- 1720

K C T LIBRARY COIMBATORE-641 006

SPY ON MOBILE

By

GUNSEKARAN R.
Reg. No. 71203621016
Of

Kumaraguru College of Technology
Coimbatore

A PROJECT REPORT
Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements
for the award of the degree
of

MASTER OF COMPUTER APPLICATIONS
JUNE 2006

BONAFIDE CERTIFICATE

Certified that this project titled SPY ON THE MOBILE is the bonafide work of Mr. Gunasekaran R. who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

16/6/06
PROJECT GUIDE

HEAD OF THE DEPARTMENT

This candidate with University Register No. 71203621016 was examined by us in the Project Viva code Examination held on 30/6

30/6
INTERNAL GUIDE

30/6/06
EXTERNAL GUIDE

ABSTRACT

Spy On Mobile – a product named **Otran** is a software product engineered at Kumaraguru College of Technology targeting the mobile devices theft safety. In the present scenario of developed community, the amount of mobile device users has grown enormously. Along with it, the circumstance where mobile users are getting into situation of loosing their mobile either by their mistake or by a planned robbery from unethical people living around. With the present technological models, particularly with the facilities among Cellular Service Provider in India it is impractical and almost impossible to get the lost mobile once it leaves owner's hand.

The product comes up with the idea of spying the context of the mobile users and reporting the unauthorized usage of the mobile device to a personally agreed mobile phone number with SMS as the communication medium for intimation.

The mobile devices the users are almost totally dependant on it for personal information storage. When such a device gets onto an unauthorized hand it gives security specific issues which are normally not handled by the mechanism built onto the mobile device. This product addresses this issue by blocking illegal people's access to personal information and helps emergency retrieval of data from a lost device using another device remotely. An included support for logging the communication medium, when the owner feels to monitor the conversation taking place by a third person using his mobile device. Product promises customer usability orientation as,

"Install, configure and forget about it until you feel the need"

The project is developed as a proof of concept model. The idea is to exploit the assumption that people who steal are not much technically skilled in this problem domain and are defeated by its **hidden mode of operation**. The product is a step forward in problem domain although it doesn't promise 100% safety mechanism.

It has been developed on using Symbian C++ Series60 SDK v2.0 with Microsoft Visual C++6.0 IDE.

ACKNOWLEDGEMENT

I express my deep sense of gratitude to **Dr. Joseph V. Thanikal,** Principal, Kumaraguru College of Technology, Coimbatore for his kind patronage and for the facilities offered to complete the project.

I thank **Dr. M. Gururajan,** Head of Department, Department of Computer Applications for his encouragement and for those of his words that keeps firing the curiosity among the young people to get onto innovative ideas.

I wish to express my profound thanks and gratitude to **Mrs. V. Geetha,** Assistant Professor, Department of Computer Applications and Project Guide, for her moral support in completing the project and guiding to make a clear shape till the finishing touchups to the project.

I wish to place on record my sincere thanks and appreciation to Symbian Developers Forum members, for their assistance whenever I had technical difficulties.

In memory of **Professor. Richard Feynman** *and his indefinite curiosity.*

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS AND NOMENCLATURE

IMEI	International Mobile Equipment Identity is a unique 15-digit code used to identify an individual GSM mobile device to a GSM network.
IMSI	International Mobile Subscriber Identity, a unique 15-digit code that is attached to every SIM Card.
S60	Mobile devices following Series 60 configuration.
SIM	Subscriber Identification Module
SMS	Short Message Service
Remote Terminal	A mobile device having SMS facility with user knowing the original device's passes code.
CSP	Cellular Service Provider.
OTRAN	Tamil word meaning Spy.

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

The project titled Spy on Mobile – Product named '**Otran**' is a software utility developed aiming at the mobile devices that run on Symbian platform, to provide a safety system getting embedded into the devices memory and be active all time.

The software product focuses on providing the mobile device owner a tracking mechanism, for the device itself on circumstance where they tend to loose the device. The mobile owner can fix the service code (IMSI) he/she is using along with the personally agreed safety condition mobile number. The software works in background as a spy, detects the change to service code and if it finds to be non-confirming to the owners specification, the remote safety number is intimated about this via SMS communication medium.

The personal contacts information stored by the device's owner in the device's Phonebook is automatically archived to a safety location by the spy and so it is not available for the unauthorized device user. In later session, when the owner gets back the mobile device onto his hand, contacts can be restored.

The software spying in background, also monitors the user activities along with the parsing the communication medium of command sequences from a remote terminal along with the authentication passwords.

The software also helps retrieval of the personal contacts information from a remote terminal device using the command specifications of the software through SMS medium.

The software also provides locating the device by generating a high pitch sound even if the device is in Silent Profile mode.

"The software provides all these without unauthorized users' knowledge"

OBJECTIVE OF THE PROJECT

The main goal of Spy on Mobile is to help the mobile device owners feel having their mobile safe to most possible extent. The project is engineered to come out with a product that is more user oriented and device safety oriented.

The goals are achieved by specific objectives, such as:

- To provide a simple user interface.
- Spy developed to work without the knowledge of the unauthorized users.
- Invisible mode of user interface.
- Highly simple to remember and use Remote Terminal commands for interaction with the Spy.
- Quick responses from the Spy upon a Remote Terminal Command.
- Occupy very low amount of Space/Time Complexity measures.
- Online Context sensitive help in user interface.
- Auto launch feature that helps the user not to worry about needing to execute to software.
- Adjust to the network situations on the command – response context.
- Easy to do, restore point in the software functionality.
- Minimum usage of third party library to keep the cost of usage low.

1.2 BACKGROUND STUDY

1.2.1 Study of the Existing System

The existing system in the mobile devices for tracking them is made out in four ways namely,

a) Location Tracking via Cellular Service Number

- The service provider has the capability of tracking the **Cell ID** of particular mobile numbers. This is available when the mobile device owner decides to file a law suit that his mobile device is missing to the concerned Police Department, where by the Police officials talk to the Cellular Service Providers in order to track the Cell ID of the mobiles existence.
- The Cellular Service Providers places an order to its technical team to track the device having particular IMEI.

The drawbacks in this model can be summarized as:

- A long chain of activities required to track the device.
- This is not a practically possible way, since Cellular Service Providers will not be ready to congest their network for a matter of tracking a single mobile device because the cost of doing so will be more than even the cost of the original itself.

b) Locating via Sound

The context where the user himself has kept the mobile device somewhere around the room and forget the location where he kept, the mobile device can be located by making a call from another phone so he can hear the ring tone.

The drawbacks in this model can be summarized as:

If the user has kept mobile device in Silent Profile, this won't help.

In both the above methods the possibility of getting the lost mobile back to hand is almost impossible.

c) Personal Contacts Information Recovery

The user can synchronize the contacts in mobile device's Phonebook to computers Address Book, once in a while.

The drawbacks in this model can be summarized as:

- Not every mobile device user owns a computer.
- Still the unauthorized user has access to the personal contacts.

d) Communication Channel Logging

The mobile devices contain a built-in logging system which the users normally use to monitor the communication channel of the device.

The drawbacks of the built-in system are,

- Open access to everyone handling the device.
- Not a unified logging model.
- No invisible mode of operation.
- Possibility of unauthorized cleanup so information is lost to the device owner.

CHAPTER 2

SYSTEM ANALYSIS

System analysis is conducted by considering the objectives like identifying major user needs that commonly exist among the mobile device users in the device's safety measures. Other developmental analysis like evaluating the feasibility of the system, allocating the resource required for the development of the system, establishing the cost and schedule constraints and to create system definition.

2.1 DEFINING THE PROBLEM

Spy on Mobile is a software system targeting the mobile device environments helping the mobile device owners in theft alert in circumstances where they loose their mobile devices and their personal contact information. It also helps logging of communication channel.

Owing to the number of drawbacks in the existing system, the new solution is proposed in such a way to provide additional support to the existing system making the device and data more secure. The proposed system gives an easy to use user interface in a hidden mode of operation, for specifying the settings and controlling the mode of operations of the spy. The owner doesn't need to care much about the execution of the system since everything in done automatically at background in invisible mode.

2.2 STUDY OF PROPOSED SYSTEM

The proposed system aims to increase the safety measures in the mobile device along with added security feature for the device's personal contact information. The proposed system comes with a usability oriented simple to use user interface for configuring the working mode of the whole system.

The Spy on Mobile has two kinds of users: **Mobile device owner and unauthorized user**. Mobile device owner is the one having access to the user interface via authenticated login. Unauthorized users have no way to login to the

user interface and are even unaware of its existence. User interface helps configuring the mode of activities of the Spy.

During the act of change in the Cellular Service [IMSI, SIM Card] the spy gets automatically signaled and blocks the thief from access to user interface part. It then starts to archive the personal phonebook contacts to its private storage where no one has direct access. Only the spy interface can access that private archive which is secured using secret pass code mechanism.

Further the mobile owner is given a remote commanding provision in order to talk to the Spy without the thief's knowledge about the conversation. The summary of the commands provided goes as:

- Making sound alarm
- Remote Contact Retrieval
- Mode Reset

Communication channel can be optionally set to be monitored to have a log of all communications taking place. The user interface part help setting this mode and checking out the log.

The software un-installation control is embedded into the user interface so only the mobile device owner can uninstall the software.

Benefits of the Proposed System

The benefits of the proposed system are summarized as below,

- Added support for existing safety system.
- Remote access of personal contacts from remote device in emergency situation.
- Device locating even when placed in Silent Profile.
- Hidden mode Communication logging.
- Restorable personal contacts information.

2.3 DEVELOPING SOLUTION STRATEGIES

2.3.1 FEASIBILITY ANALYSIS

Feasibility study is an evaluation of the system proposed regarding its workability, impact on the target environment, ability to meet user-needs and effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study, before it is approved for development.

In **Spy on Mobile** project, the feasibility is analyzed and the system is checked for workability and the impact on the target environment and also the ability to meet the mobile user's needs and the effective use of resources.

2.3.1.1 Technical Feasibility Study

Here the system requirements are assessed in terms of inputs, outputs, files, programs, procedures and staff required for the development and implementation of the Spy on Mobile. Care was taken in deciding will the project will be possible to be completed as one man team. An investigation was carried out to identify the type of equipments required, method of developing the system and methods of running the system once it had been developed. The emulator used to develop was analyzed for ability to provide functionalities to develop the system. It was identified that certain portions of the project requires porting to the real mobile device for execution.

The mobile device manufacturers being spread vastly coming out with variety of devices, the devices of S60 configuration was chosen as the target platform. The Symbian Operating System which lies as the common ground provides the API library for making this system to solid reality. There has been certain third party public and non-public libraries selected to use. Some of those non-public libraries were been considered reverse engineered solely for development purpose and will not be documented or released to any public use.

Therefore, the proposed system is technically feasible.

2.3.1.2 Economic Feasibility Study

Economic feasibility deals with analysis of cost against benefits. Here the user purchasing the products will yield benefits like a simple insurance to their device, because the software product will increase the probability of the safety constraints of their device and personal contacts information to an extent.

Being developed as a product to be sold across a large market the cost to get a licensed copy of the product is marginally low that every mobile device owner can afford to. As the product does not need any other paid libraries to be used inside, the user is not needed to pay any extra amount of money in using this software. Since there is no hardware cost involved because of the fact that this product is targeting only S60 mobile devices, the users who have these devices can add this utility easily.

The results of the evaluation show that this project was economically feasible and justified its development for effort being invested. Being engineered as a product, the low cost feature will cover large number of customers for this product, so making better trend for marketing.

2.3.2 SYSTEM SPECIFICATION

The following are the hardware and software requirements to develop and deploy the product,

2.3.2.1 HARDWARE REQUIREMENTS

DEVELOPMENT ENVIRONMENT

Processor	:	Intel Pentium IV, HT
Operating Frequency	:	3.0 GHz
RAM	:	1024MB DDR
Hard Drive	:	160 GB Seagate SATA
I/O Device	:	104 Keyboard, Mouse
Communication Device	:	USB Bluetooth Dongle
Mobile Device	:	Nokia 6600, N70 Series

TARGET MOBILE DEVICES

All Symbian OS running mobile devices of S60 specification.

2.3.2.2 SOFTWARE REQUIREMENTS

DEVELOPMENT ENVIRONMENT

Operating System	:	Microsoft Windows XP
SDK	:	Symbian Series60 SDK v2.0
IDE	:	Microsoft Visual C++ 6.0
Tool Backgrounds	:	Active Pearl 5.8.7 Build 813 Java Runtime 1.3.1
Communication Driver	:	IVT BlueSoleil HCI mRouter Runtime from Nokia

TARGET MOBILE DEVICES

Operating System	:	Symbian OS v7.0
Libraries	:	AVKON Library MobInfo API Library

2.3.2.3 SOFTWARE ENVIRONMENT

Symbian OS v7.0

Symbian OS is the operating system licensed by the world's leading mobile phone manufacturers. Symbian OS is designed for the specific requirements of open, data-enabled 2G, 2.5G and 3G mobile phones. Symbian OS is already available in the Sony-Ericsson P800, the Nokia 9200 Communicator series, etc. With the introduction of Symbian OS v7.0s, the range of mobile phones with Symbian OS will expand even further, beginning with the Nokia 6600.

Symbian OS is a 32-bit multitasking operating system, wherein events often happen asynchronously and applications are designed to interact with one another.

Short Message Service (SMS), much related are to our project, consists of an SMS stack with a messaging API to send and receive SMS and provides the following features:

- The SMS stack is implemented as a plug-in protocol. The GSM (03.40) SMS protocol is provided.
- The GSM SMS stack can be used as a bearer for the WAP protocol module.
- Transmission and reception of GPRS SMS.
- SMS: send and receive streamed SMS messages. Enumerate, read, write and delete access to the SMS storage areas of the phone and SIM. Receive messages that match a specified text.
- 7-bit SMS alphabet, 8-bit SMS alphabet and UCS2 data coding schemes are supported.
- Supports sending and receiving concatenated SMS messages.
- Scheduled sending: on a specific date/time, "now" or upon request. Specify and review scheduled actions.

Series 60 Developer Platform 2.0

The Series 60 Developer Platform enables application and service developers to produce smart phone products for the Series 60 Platform. By providing a standardized platform while maintaining a high degree of flexibility, the Series 60 Developer Platform is well positioned to maintain its market leader status in the smart phone sector.

Series 60 Developer Platform 2.0 contains the following technologies:

- J2ME Java APIs
  - MIDP 2.0
  - CLDC 1.0
  - Wireless Messaging API (JSR 120)
  - Mobile Media API (JSR 135)
  - Bluetooth API (JSR 82)
- XHTML browsing over TCP/IP
- MMS messaging with Synchronized Multimedia Integration Language (SMIL)
- OMA Digital Rights Management (DRM) (forward-lock)
- OMA Client Provisioning
- Symbian OS v7.0s native APIs

The target platform requirements,

Display	176 x 208 pixel, 256 color display.
Input	Two soft-keys, five-way navigation, an application launching and swapping key, as well as Send and End keys. To improve and facilitate text input, it includes a Clear key and an alpha toggle key. It uses a standard 12-key number keypad with alpha printings.
Processor	It is recommended that the target device use a 32-bit ARM processor.
Code Size (ROM)	16 MB
RAM Usage	8 MB

TABLE 2.3.2.3.1: **Series 60 device Requirements**

Series60 C++ SDK

C++ is the native programming language for the Series 60 Developer Platform and the underlying Symbian OS. However, the use of C++ in Symbian OS differs from C++ use in other environments in that it is much more tightly controlled. Symbian OS was written specifically for devices where physical resources would be constrained and/or limited. Typically, a Symbian device is rarely rebooted, and thus it has to be capable of running error-free for months or even years, so its main focus is on controlling power consumption, error handling, and clean-up. Therefore, there are strict rules relating to memory management, and C++ in Symbian OS differs from standard C++ in, for example, string handling.

Series60 adds a User Interface Layer (AVKON) onto the underlying UIKON from Symbian OS v6.1. AVKON provides a set of UI components and an application framework designed specifically for Series60 devices. Applications are normally split into two parts, the engine and the UI, to aid maintainability and flexibility. The application engine, which is sometimes known as the application

model, deals with the algorithms and data structures needed to represent the application's data. The application UI, sometimes called the app, deals with the on-screen presentation of the application data and the overall behavior of the application. Model-View-Controller Paradigm is a common design pattern used in UI application of Series60.

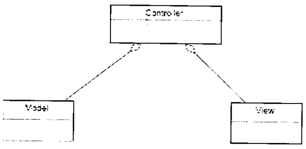


FIGURE 2.3.2.3.1: **Model-View-Controller Paradigm**

The MVC pattern forces a separation in the application design, allowing the model code to be reused.

Microsoft Visual Studio IDE

Microsoft Visual Studio 6.0 can be used as an environment for Symbian C++ development although future versions of Symbian SDKs may not be supported. For the time being, Microsoft Visual Studio remains a viable option if this environment is already available to the developer. However, purchasing this product is not recommended for continued Series 60 Platform development. This IDE provides debugging facilities, but many of the developer tools are accessible only through the DOS environment. A wizard for developing Symbian projects (including Series 60 AVKON frameworks) on this platform is also available.

MobInfo API v1.01 for Symbian OS v7.0, v7.0s

MobInfo offers minimal but complete and easy services to apps and components that need some telephony services but are not telephony-oriented apps themselves. Thus the Mobinfo API does not intend to provide a complete API to 3rd parties for telephony; it rather aims to provide a manageable and easy path to Etel features that many developers seek and usually have to employ works around to get to, through undocumented and non-public areas of the system. Mobinfo is a shared dynamically linked and loaded library with a static interface, whose exported classes are not intended for derivation. Its purpose is to easily enable the following application usage scenarios.

ETel Core Library

The API defines a core set of functions that are supported by almost all telephony devices and services. Applications can use it directly to access general telephony devices. The interfaces are also derived from by other APIs, for example the Etel GSM APIs, as the basis for more advanced functionality.

The API uses the Symbian OS client/server framework. The API provides R classes that send requests to the Etel server. The server in turn passes requests to an appropriate plug-in module that handles the physical device.

2.4 USECASE DIAGRAMS

Use Case Analysis is a valuable tool during the analysis phase to make up the **Requirement Workflow**.

Use Case diagrams make system, subsystems and classes approachable and understandable by presenting the outside view of how these elements may be used in context. Use case diagrams are also important for testing executable systems through forward engineering and for comprehending executable systems thought reverse engineering.

The use case diagrams for project are given in Figure 2.4.1 – 2.4.3.

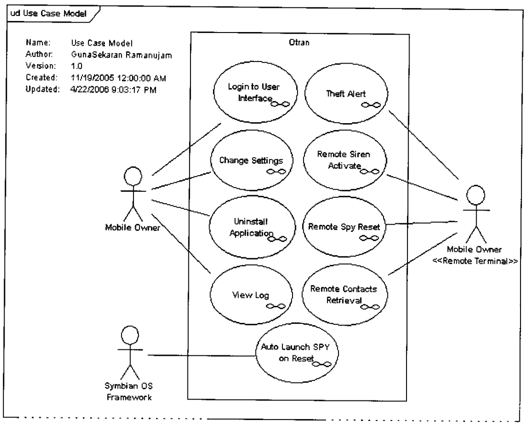


FIGURE 2.4.1: **Use Case Diagram**

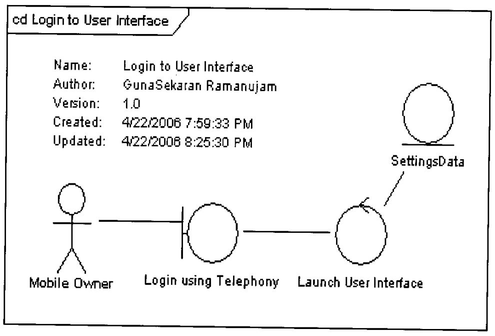

cd Login to User Interface
Name: Login to User Interface
Author: GunaSekaran Ramanujam
Version: 1.0
Created: 4/22/2006 7:59:33 PM
Updated: 4/22/2006 8:25:30 PM
SettingsData
Mobile Owner
Login using Telephony
Launch User Interface

FIGURE 2.4.2: **User Interface Use Case Diagram**

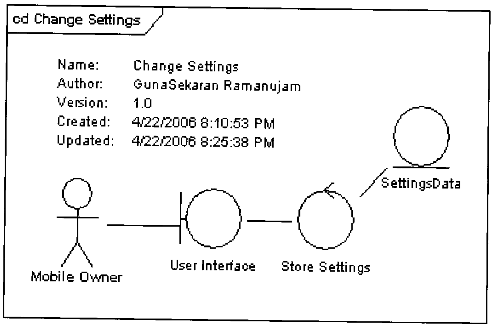


FIGURE 2.4.3: **Settings Change Use Case Diagram**

CHAPTER 3

SYSTEM DESIGN AND DEVELOPMENT

Design is the pivotal point in the system development lifecycle. It translates the system requirements into an operational system. System design is the process of planning a new system that will complement the workings of the existing system. The design is clearly based on the common requirement gathered from various categories of user belonging to the context of the project.

Software design is an iterative process through which requirements are translated into a "blueprint" for constructing the software. But in this project the development is made using the Prototyping Approach, where the design model is revised every time along with changes in requirements and the technological influence. Every time the prototype is revised the old prototype is updated and not thrown off.

3.1 Architectural Design

Architectural design involves identification of software components, decoupling and decomposing them into processing modules, conceptual data structures and specifying relationship among the components.

For Spy on Mobile it involves three separate levels namely:

a) User Interface

b) Spy

c) Auto launch on Reset

The simple structural split of the product goes as in Figure 3.1.1.

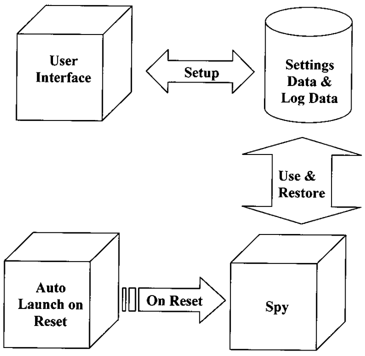


FIGURE 3.1.1: **Global model of Spy on Mobile**

a) User Interface

The GUI interface for the mobile device owner, for altering the settings details which in turn is used by Spy. These settings data are stored to the commonly used file "Otran.ini". The user interface is not available directly in the Applications Menu of the Series60 device in order to hide the existence of the software from unauthorized users. The user interface can only be launched by the Spy. The telephony user interface is the interface for the users to request the Spy to launch the user interface. The Spy when it detects a dialing of the commonly agreed pass code prefixed with a '*', it launches the user interface and clears the dialed entries of the pass code from the log, so it wont be visible for others.

The way this is achieved by, passing the pass code as command line argument to the application from the Spy. Since the pass code to not known to any other application, there is no way to launch the user interface application.

The user interface's modules are designed for,

- Configure the Settings file data
- Enable/Disable Communication Logging
- View the Log
- Clear the log
- Enable/Disable Application Uninstaller
- Activate/Deactivate Spy Mode
- Restoring the contacts from archive after Spy mode is reset.

The User Interface has an additional module **ActiveMobileInformationEngine** which is designed to assists the users in gathering device based information's IMEI and IMSI, so relieving the user from need to know the special codes to get those information for their device.

The Application Manager of the device, automatic launches the user interface during installation due to the parameters given to it. The user interface is designed to be intelligent about this, so where by it will automatically restart the device when exited, so that **OtranRecog** module will launch the Spy during boot sequence. During normal launch the user interface doesn't restart the mobile when exited.

User interface is designed so that it will provide good information messages upon user actions over it, which makes the software more user- friendly.

The user interface's class model goes as in Figure 3.1.2.

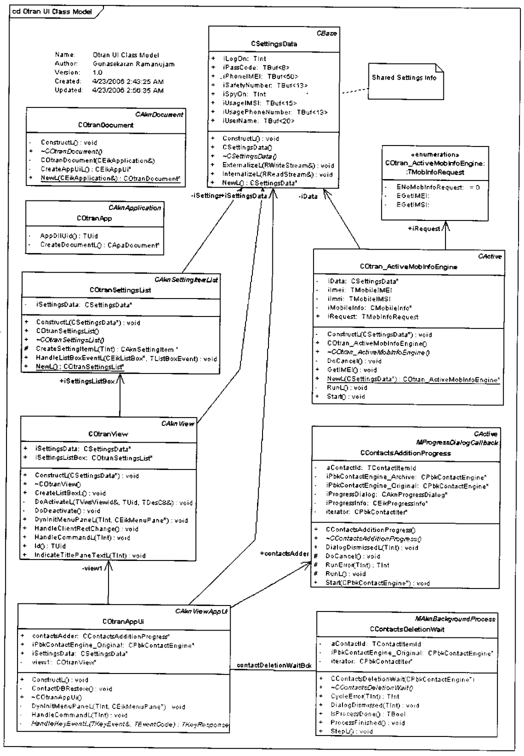

cd Otran UI Class Model
Name: Otran UI Class Model
Author: Gunasekaran Ramanujam
Version: 1.0
Created: 4/23/2006 2:43:25 AM
Updated: 4/23/2006 2:56:35 AM
CBase
CSettingsData
+ iLogOn: TInt
+ iPassCode: TBuf<8>
+ iPhoneIMEI: TBuf<50>
+ iSafetyNumber: TBuf<13>
+ iSpyOn: TInt
+ iUsageIMSI: TBuf<15>
+ iUsagePhoneNumber: TBuf<13>
+ iUserName: TBuf<20>
+ ConstructL() : void
+ CSettingsData()
+ ~CSettingsData()
+ ExternalizeL(RWriteStream&) : void
+ InternalizeL(RReadStream&) : void
+ NewL() : CSettingsData*
Shared Settings Info
CAknDocument
COtranDocument
- ConstructL() : void
+ ~COtranDocument()
- COtranDocument(CEikApplication&)
- CreateAppUiL() : CEikAppUi*
+ NewL(CEikApplication&) : COtranDocument*
CAknApplication
COtranApp
- AppDllUid() : TUid
- CreateDocumentL() : CApaDocument*
-iSettings
+iSettingsData
-iData
«enumeration»
COtran_ActiveMobInfoEngine::TMobInfoRequest
- ENoMobInfoRequest: = 0
- EGetIMEI:
- EGetIMSI:
+iRequest
CAknSettingItemList
COtranSettingsList
- iSettingsData: CSettingsData*
+ ConstructL(CSettingsData*) : void
+ COtranSettingsList()
+ ~COtranSettingsList()
# CreateSettingItemL(TInt) : CAknSettingItem *
+ HandleListBoxEventL(CEikListBox*, TListBoxEvent) : void
+ NewL() : COtranSettingsList*
+iSettingsListBox
CActive
COtran_ActiveMobInfoEngine
- iData: CSettingsData*
- iImei: TMobileIMEI
- iImsi: TMobileIMSI
- iMobileInfo: CMobileInfo*
+ iRequest: TMobInfoRequest
- ConstructL(CSettingsData*) : void
+ COtran_ActiveMobInfoEngine()
+ ~COtran_ActiveMobInfoEngine()
- DoCancel() : void
+ GetIMEI() : void
+ NewL(CSettingsData*) : COtran_ActiveMobInfoEngine*
- RunL() : void
+ Start() : void
CAknView
COtranView
+ iSettingsData: CSettingsData*
+ iSettingsListBox: COtranSettingsList*
+ ConstructL(CSettingsData*) : void
+ ~COtranView()
+ CreateListBoxL() : void
- DoActivateL(TVwsViewId&, TUid, TDesC8&) : void
- DoDeactivate() : void
+ DynInitMenuPaneL(TInt, CEikMenuPane*) : void
+ HandleClientRectChange() : void
+ HandleCommandL(TInt) : void
+ Id() : TUid
+ IndicateTitlePaneTextL(TInt) : void
CActive
MProgressDialogCallback
CContactsAdditionProgress
- aContactId: TContactItemId
- iPbkContactEngine_Archive: CPbkContactEngine*
- iPbkContactEngine_Original: CPbkContactEngine*
- iProgressDialog: CAknProgressDialog*
- iProgressInfo: CEikProgressInfo*
- iterator: CPbkContactIter*
+ CContactsAdditionProgress()
+ ~CContactsAdditionProgress()
+ DialogDismissedL(TInt) : void
# DoCancel() : void
# RunError(TInt) : TInt
# RunL() : void
+ Start(CPbkContactEngine*) : void
-view1
+contactsAdder
CAknViewAppUi
COtranAppUi
+ contactsAdder: CContactsAdditionProgress*
+ iPbkContactEngine_Original: CPbkContactEngine*
+ iSettingsData: CSettingsData*
- view1: COtranView*
+ ConstructL() : void
- ContactDBRestore() : void
+ ~COtranAppUi()
- DynInitMenuPaneL(TInt, CEikMenuPane*) : void
- HandleCommandL(TInt) : void
- HandleKeyEventL(TKeyEvent&, TEventCode) : TKeyResponse
contactDeletionWaitBck
MAknBackgroundProcess
CContactsDeletionWait
- aContactId: TContactItemId
- iPbkContactEngine_Original: CPbkContactEngine*
- iterator: CPbkContactIter*
+ CContactsDeletionWait(CPbkContactEngine*)
+ ~CContactsDeletionWait()
+ CycleError(TInt) : TInt
+ DialogDismissed(TInt) : void
+ IsProcessDone() : TBool
+ ProcessFinished() : void
+ StepL() : void

FIGURE 3.1.2: **User Interface Class diagram**

b) Spy

Spy level is the core part of the Spy on Mobile project. Spy is designed to operate at the background hidden from any kind of users, with class model as in Figure 3.1.5. The Spy keeps itself active all the time and monitors the context for the users operations over the mobile device.

The booting sequence of the Spy is auto launched during mobile device's boot sequence. The **Service Valuator** checks for the IMSI verification with its settings data as in Figure 3.1.3. When it detects a conflict in it, which is due to a possible change of SIM Card without device owner's knowledge the Spy shifts from Normal Mode of operation to Spy Mode.

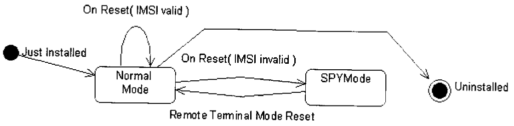


FIGURE 3.1.3: **Spy Statechart Diagram**

The control flow of the Spy goes as in FIGURE 3.1.4.

In Normal mode, the Spy may optionally set to Deactivated mode in user interface so the device owner can change his SIM Card to a new one. The Spy immediately after the reboot automatically activates the Spy Mode.

In Normal mode, the Spy's **Call Monitor** module keeps watching the Telephony applications user interface for a match against the pass code prefixed with '*' being dialed. When a match is found it launches the user interface and removes the dialer entry of pass code from log.

In both Normal/Spy mode, both **Call Monitor** and **MTMEngine** module logs the communication details which can be later viewed through the user interface later. By default this feature is turned off for efficiency considerations, but can be turned on using the user interface.

During the boot sequence of the Spy the **Application Uninstall Manager** module takes care that the Spy's entries in Application Manager of the device are removed so that no unauthorized users can remove the application from the device. This also adds additional score to applications hidden mode operations. The uninstaller entries can be restored through the user interface which is only available in Normal Mode, but that should be done before device reboot occurs; else the Spy hides the entries again.

In Spy/Normal mode, the **Sound Player** module integrated with **MTMEngine** module helps locating the device using loud sound played, which can be activated using a SMS command from a Remote Terminal. This helps when device owner forgets where he kept the device inside the room or if the thief is inside the closed proximity. Refer Table3.2 fro appropriate commands.

In Spy mode, when the device owner has got the mobile back on hand, the device can be reset to Normal mode from Spy mode by sending SMS commands from a Remote Terminal to device, which is parsed by **MTMEngine** module's **Command Parser.**

In Spy mode, **Contacts Engine** module archives the contacts from the mobile devices default phonebook to a private archive so hiding the personal contact details from the unauthorized users. These contacts can be restored using user interface once the Spy mode is reset. Being integrated with **MTMEngine** module this provides an added feature of accessing the contacts via SMS commands from a Remote Terminal. On receiving the command the archive made previously is searched, but in Normal Mode the default phonebook is searched. Refer Table3.2 fro appropriate commands.

In Spy/Normal mode, the **SettingsFileMonitor** module monitors the changes to the settings file "Otran.ini" which might be a possible cause of device owner's changes to settings via user interface, updates the local settings data, which drives the further activities of the Spy. The file monitor restores the data when Spy detects the possible action of cracking.

c) Auto Launch on Reset

The Spy is designed in a model to initiate its boot sequence along with devices boot sequence. But the Spy is not designed so intelligent so that it can itself manage this, due to the technical infeasibilities. The Spy out sources this technical requirement to **OtranRecog** module. This module uses the Recognizers logic in Symbian devices, which is similar to file type handlers in our Windows family operating systems. The recognizer is auto launched by the Symbian OS framework during the mobile device's boot sequence, which in turn launches the Spy.

OtranRecog module is built with intelligence to prepare the mobile device's context to such that Spy is comfortable with. This is achieved by waiting for the Telephony application to launch before Spy is launched.

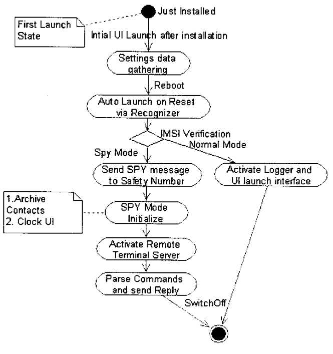

Just Installed
First Launch State
Intial UI Launch after installation
Settings data gathering
Reboot
Auto Launch on Reset via Recognizer
IMSI Verification
Normal Mode
Spy Mode
Send SPY message to Safety Number
Activate Logger and UI launch interface
1.Archive Contacts
2. Clock UI
SPY Mode Initialize
Activate Remote Terminal Server
Parse Commands and send Reply
SwitchOff

FIGURE 3.1.4: **Spy Logical Activity Diagram**

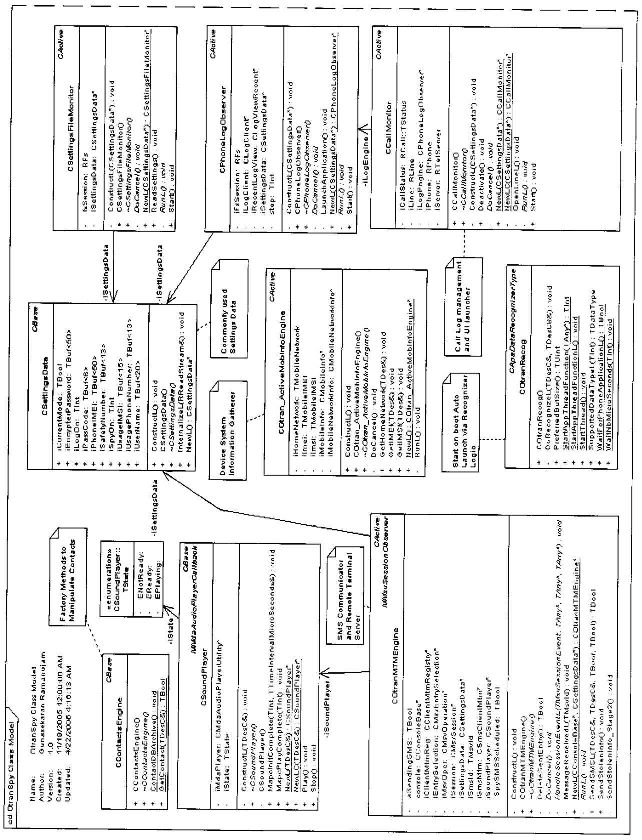


FIGURE 3.1.5: **Spy Class Diagram**

3.2 FILE DESIGN

Spy on Mobile uses a single file "Otran.ini", to store settings information to setup the working configuration of the Spy. The editing of the settings is done via user interface. Since this is a simple file with minimum number of fields it didn't require much file organization considerations and decided to be a simple **Sequential Access File** via object streaming concept of Symbian OS. The objects persistence mechanism is used for file manipulation. Since **CSettingsData** is derived from **CBase**, all this functionalities are provided by framework itself.

Field Name	Field Type
User Name	TBuf<20>
Usage Phone Number	TBuf<13>
Usage IMSI	TBuf<15>
Phone IMEI	TBuf<50>
Safety Phone Number	TBuf<13>
Pass Code	TBuf<8>
Spy On	Tint
Log On	Tint

TABLE 3.2.1: **Settings File Design**

As the file "Otran.ini" contains highly sophisticated private information, it is stored in an encrypted form so that no external application has visibility into it. The inbuilt Symbian OS Secure storage mechanism called **Encrypted Streams and Stores** is used in storing and retrieval of the settings information. The key used is IMEI number of the devices.

K C T
1720
LIBRARY
COIMBATORE

3.3 Remote Terminal SMS Commands

The Spy provides a provision for commanding via SMS medium from a Remote Terminal. In course of this products development the main thing that was considered to be addressed while designing is,

"I have always wished for my computer to be as easy to use as my telephone; my wish has come true because I can no longer figure out how to use my telephone" - **Bjarne Stroustup**

So the design of the commands were made very simple to use and easy to remember and the main advantage of design is limited set of commands.

Command Type	Command Format	Mode
Audio Alarm Locating	Otran Siren <passcode>	Normal/Spy
Spy Mode Reset	Otran Unblock <passcode>	Spy
Remote Contact Retrieval	Otran Contacts <passcode> #<firstname> [<lastname>]	Normal/Spy

TABLE 3.3.1: **Remote SMS Commands Design**

3.4 User Interface Design

The launch sequence of the user interface is done with a special procedure using Telephony interface to request Spy to launch the user interface as in Figure 3.4.1.

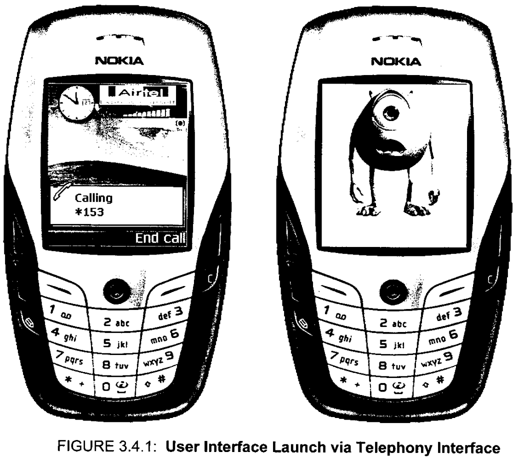
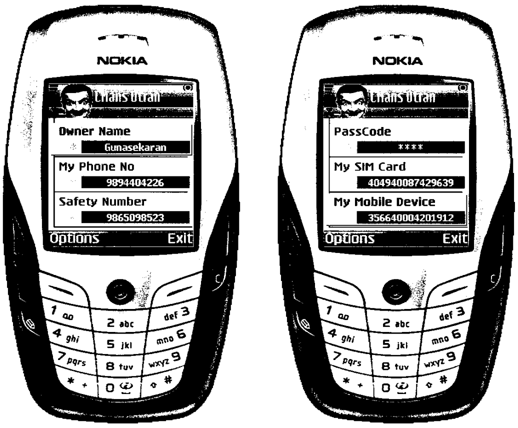


FIGURE 3.4.1: **User Interface Launch via Telephony Interface**

The settings information is used to customize the software for customer specific usage which is achieved via filling the form in user interface as in Figure 3.4.2.

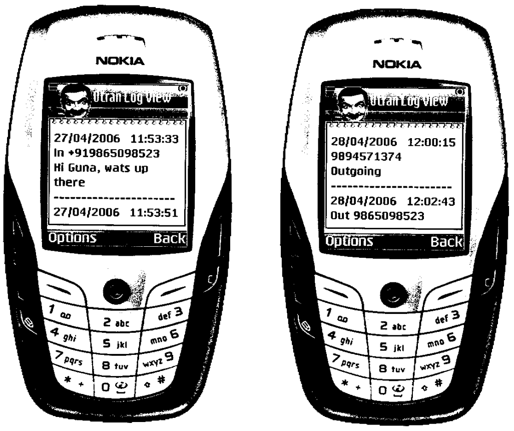


Owner Name
Gunasekaran
My Phone No
9894404226
Safety Number
9865098523

PassCode
****
My SIM Card
404940087429639
My Mobile Device
356640004201912

FIGURE 3.4.2: **Configuring Settings using UI**

The communication channel can be secretly monitored using OTRAN's Logging feature, which can be turned on in user interface. The user interface provides the place to view the log as in Figure 3.4.3.

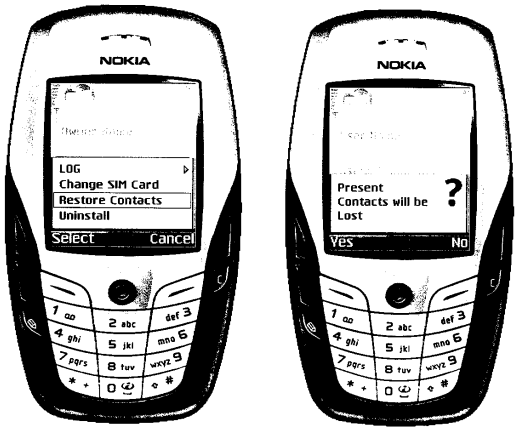


```
27/04/2006  11:53:33
In +919865098523
Hi Guna, wats up
there
----------------------
27/04/2006  11:53:51
```

```
28/04/2006  12:00:15
9894571374
Outgoing
----------------------
28/04/2006  12:02:43
Out 9865098523
```

FIGURE 3.4.3: **Communication Channel Logging**

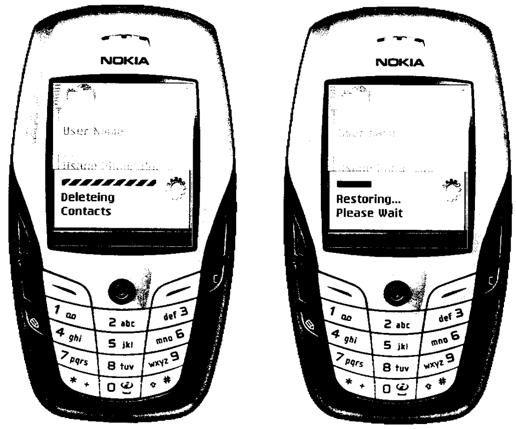
The contacts are archived to private location by the Spy when it detects unauthorized device usage. The user interface provides a facility to restore the contacts by the device owner as in Figure 3.4.4.



Present
Contacts will be
Lost

FIGURE 3.4.4: **Contact Restore**

User interface gives a visual representation of the contacts restoring process as shown in Figure 3.4.5.

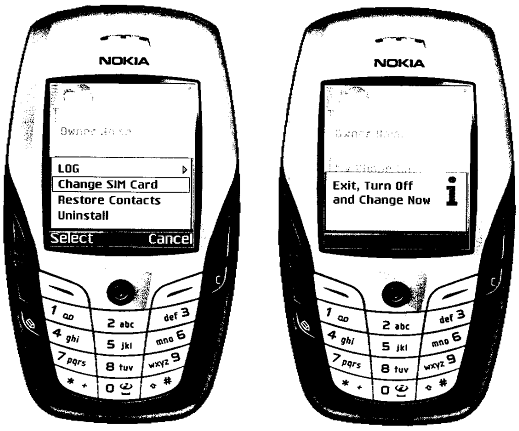


Deleteing
Contacts

Restoring...
Please Wait

FIGURE 3.4.5: **Contacts Restore Progress**

When the device owner wishes to change SIM Card himself and doesn't want the theft alert Spy sequence to be made by Spy, this can be done using user interface as shown in Figure 3.4.6.



Exit, Turn Off and Change Now

FIGURE 3.4.6: **SIM Card Change Procedure**

The specialty about product OTRAN in the problem domain it addresses is the hidden mode of its operation so no others except device owner are aware of its existence. Figure 3.4.7 shows how it is hidden in various possible locations.

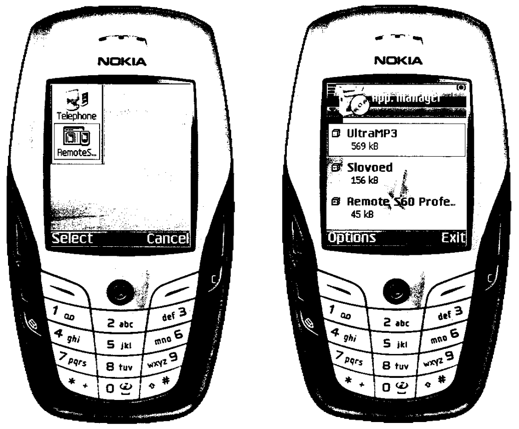


FIGURE 3.4.7: **Spy Invisible Mode of Operations**

The final product was made into a self extracting installation file Otran.sis which can be installed either in device itself after downloading or directly from PC. The Figure 3.4.8 shows the installation process in PC, using the Bluetooth physical interface.

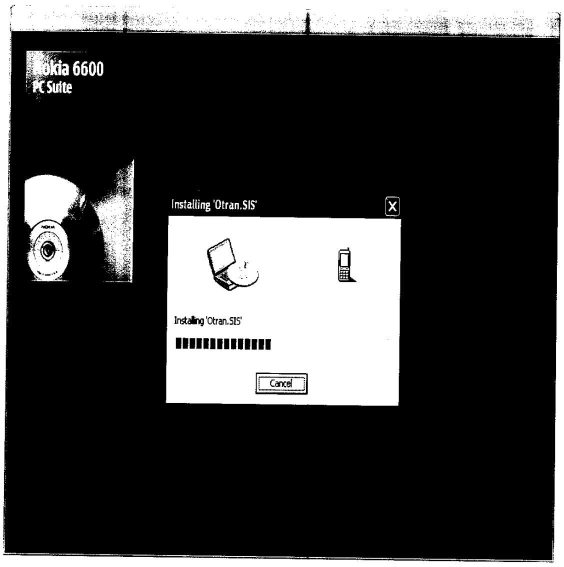


FIGURE 3.4.8: **Otran Installation Sequence using PC**

3.5 DEVELOPMENT APPROACH

The commonly used approaches for the development and testing of software programs are:

1.) Tradition Bottom-Up Approach

2.) Contemporary Top-Down Approach

The Bottom Approach of development were followed where the modules are being developed separately in an integrated way with their own files to support operational activities; then the subsystems were integrated as the planning and control activities are developed. This approach helped the project better since the requirements were not finalized and the initial moment and **Prototyping** approach being followed.

The complete document sharing the report of the document phase was discussed with guide and development continued after approval. The changes said by the guide have been made at places, based on results of technical and economical feasibility studies.

CHAPTER 4

SYSTEM TESTING AND IMPLEMENTATION

4.1 SYSTEM TESTING

System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before product is delivered for marketing. Test data is specially designed to show that the system will operate successfully in all its aspects and produce expected results under expected conditions.

The product being targeting the S60 Mobile devices, in order to provide the customers with a genuine product by having Symbian Signed Certificate the recommendations and guidelines by Nokia on design so that it satisfies the Symbian Signed Criteria. The product will be considered for revision in order to go through Symbian Signed program, when it reaches production stage.

But the product for now doesn't satisfies Intellectual Property Rights (IPR) defined by Symbian, due to the technical factors handled inside the product design, for now it may not be able to pass through Symbian Signed program. But these things are being considered to be resolved before the taken to production.

The other two issues specified in the recommendations and guidelines by Nokia are satisfied,

- The content of the application is applicable
- The developer has the rights to use all the brands visible in the application.

White box Testing

As this product's working model is mainly based on the random control flow the following tests were made on the system,

- All independent paths within a module have been exercised at least once.
- All logical decisions were checked for the truth and falsity of the values.
- All loops were executed to check their boundary values.
- Internal data-structure was tested for their validity.

Black box Testing

The functional requirements were tested with the black box testing approach to find the following categories of errors,

- Interface errors
- Errors in data structures or external database access
- Performance errors
- Initialization and termination errors
- Incorrect or missing functionalities

Beta release of the product was made to online users' community and the bugs found till final release were fixed.

Testing Strategy

Unit Testing

All the individual modules are tested to uncover all possible errors within its boundary, along with its specified functional requirements.

Integration Testing

All unit tested modules were combined and a program structure was built as dictated by design. The trace for new errors getting into was checked. The final program structure was tested for its operations.

System Testing

The main part that was done during the testing session of this project based on the guidelines from Nokia as,

- The application speed does not compromise the use and purpose of the application.
- The application does not consume the device's running power and memory excessively.
- The application does not affect the use of the system features or other applications.
- The application does not cause any harm to the user, other applications, or data.
- There are no bottlenecks in the code.
- Only necessary data is saved into the flash memory.
- Communication to and from the device/application is kept within reasonable limits.
- The application must be able to handle exceptional, illegal, or erroneous actions.

Code Testing

The logic of the code has been tested well by well executing the code with verification against possible outputs. The coding conventions were verified against the **Symbian OS C++ Coding Conventions v2.0 specification**.

4.2 SYSTEM IMPLEMENTATION

As the developed system is a product and not any users based solution for some organization, the implementation stage is the production environment of the product.

CHAPTER 5

CONCLUSION

The complete design and development of the product OTRAN is presented in dissertation. A simple to use user interface along with the hidden mode operation of the Spy brings a big advantage to the product, which the users can exploit to get the maximum benefit.

Being **Symbian OS C++ Coding Conventions v2.0 specification** followed in the coding session a scope for further expansion and implementation of any changes, will be easier in future. Maximum care and concentration has been focused to troubleshoot this project.

This project is very simple to use by the target audience and is a successful one. The developed system is flexible to changes and these changes can be made with little efforts.

Scope for future Enhancements

Though the system has been developed to satisfy the target audience needs in the context of this product, enhancements are always possible. The modularity based designed in helps changes introduction an easy option, along with the code reuse among other product verticals a viable feature.

The products next version is also considered with web based interface for common safety number service, enhanced personal information archiving, mobile cell location tracking feature are announced. Research on these areas has been planned before getting into development in order to find the technical ideas to develop them.

The system is not a 100% safety system for now since it has to sit on the volatile flash memory of the device and cannot withstand Hard Reset of device. The project was developed as a proof of concept model. To further enhance the safety model, it should be added on to ROM by device manufacturers.

REFERENCES

1. Jo Stichbury 'Symbian OS Explained: Effective C++ programming for Smartphone', John Wiley & Sons Ltd.

2. Digia 'Programming for Series 60 Platform and Symbian OS', Symbian Press

3. Steve Babin 'Developing Software for Symbian OS', Symbian Press

Websites:

www.Symbian.com – Developer Discussion Forum.