Chapter1 – Introduction

1.1 WiFi Client:

This application works as a Windows Service. This application keep track of all the available WiFi access points. When the Signal strength of the WiFi source to which the host is connected goes below a given threshold value, the application connect the host to some other better available network. Only those WiFi AP's will be considered whose profile is already available (Wireless Profile is an XML which contains the information about the WiFi network like SSID, password, applied encryption and other information like protection etc. This XML is required as an input for the application to connect to WiFi source). We assume that the profile XML's are already shared at a common shared location which is accessible to the host device and application. This will be discussed in details in further chapters.

Chapter2 – Background

In this section we will cover some related topics and the underlying technologies which are used for developing the applications.

2.1 WiFi and WLAN:

WiFi refers to the technology which uses radio waves to transfer data between electronic devices. Wireless local area network (WLAN) connects two or more devices with each other using wireless methods like Spread spectrum or OFDM radio. The use of wireless media provide the ability to mobile devices that these can be moved within a local area (in coverage range) without being disconnected. Most of the WLAN products like modem are based on IEEE 802.11 standards.

According to the WiFi Alliance any WLAN product based on IEEE 802.11 standard is referred as WiFi.

2.2 WLAN Infrastructure and related terms:

In this section we are going to discuss some important about wireless networking using WiFi.

2.2.1 Wireless Network Interface Card:

The Wireless Network Interface card (WNIC) is an integral component of the devices which support WiFi. This is the essential piece of hardware required for WiFi. WNIC works on Physical and Data link layer of OSI Network Model. WNIC controls the communication between wireless devices like Ethernet card does with wired networks. WNIC has an antenna for radio transmission.

2.2.2 WLAN Architecture:

All components that can connect into a wireless medium in a network are referred to as stations. All stations are equipped with wireless network interface controllers (WNICs).

Wireless stations fall into one of two categories: access points, and clients. Access points (APs), normally routers, are base stations for the wireless network. They transmit and receive radio frequencies for wireless enabled devices to communicate with. Wireless clients can be mobile devices such as laptops which are equipped with a WNIC.

The basic service set (BSS) is a set of all stations that can communicate with each other. Every BSS has an identification (ID) called the BSSID, which is the MAC address of the access point servicing the BSS.

There are two types of BSS: Independent BSS (also referred to as IBSS), and infrastructure BSS. An independent BSS (IBSS) is an ad hoc network that contains no access points, which means they cannot connect to any other basic service set.

An extended service set (ESS) is a set of connected BSSs. Access points in an ESS are connected by a distribution system. Each ESS has an ID called the SSID which is a 32-byte (maximum) character string.

A distribution system (DS) connects access points in an extended service set. The concept of a DS can be used to increase network coverage through roaming between cells.

DS can be wired or wireless. Current wireless distribution systems are mostly based on WDS or MESH protocols, though other systems are in use.

2.2.3 Wireless Access Point (WAP):

The wireless access point allows the wireless devices to connect with wired devices. The modern day wireless routers contains WAP which connect the wired network and a router further can spread it over WiFi.

The wireless access points are managed by a WLAN Controller which handles automatic adjustments to RF power, channels, authentication, and security. Further, controllers can be combined to form a wireless mobility group to allow inter-controller roaming. The controllers can be part of a mobility domain to allow clients access throughout large or regional office locations. This saves the clients time and administrators overhead because it can automatically re-associate or re-authenticate. The Hotspots are common example of WAP.

2.2.4 Types of WLAN:

The Wireless local area network has following modes of operation:

Infrastructure Mode: In this mode all the stations communicate through an access point i.e all data is transferred via access point. All the stations connected to an access point should have same SSID as the access point. In case WEP is enabled then all the stations should have the same WEP key and also the same other authentication parameters. This is a centralized type of wireless network and access point works as the controller device. These devices normally combine three primary functions; physical support for wireless and wired networking, bridging and routing between devices on the network, and service provisioning to add and remove devices from the network.

Ad Hoc Mode: In this mode the devices can connect to each other directly without using any access point. This mode uses Independent Basic Service Set (IBSS). It is decentralized type of wireless network. In ad Hoc mode all devices have equal status on a network and are free to associate with any other ad hoc network device in link range.

Peer To Peer (WiFi Direct): In this mode the stations connect directly to each other. Wi-Fi Direct is not the same as ad-hoc networking: The most significant difference between traditional ad-hoc wireless networking (traditional peer-to-peer networking) and Wi-Fi Direct is security.

In Windows ad-hoc networks, the highest level of security supported is WEP in mixed client environments (Windows 7 will support WPA2 provided all adapters support it, as well). Wi-Fi Direct, as mentioned, supports WPA2.

Another difference, Wi-Fi Direct devices can also simultaneously connect to existing wireless networks. More granular control and better discovery of devices also differentiate Wi-Fi Direct from ad-hoc networking. Many of the latest mobile phones and tablets like Samsung Galaxy S3, Nexus etc support WiFi Direct. Unlike WiFi direct the Ad hoc network cannot connect to hotspot and infrastructure mode simultaneously.

Chapter3 - Technologies behind this project

In This chapter we will cover the basic technologies behind this project.
We have developed the two independent services; one enables the Wireless hosted network and create the virtual router so that other devices can connect to internet using this router.
This work is based primarily on the following technologies:

1. VirtualWiFi
2. Wireless Hosted network
3. Internet connection Sharing.

We will explain each in detail, the following information is directly taken from their actually sources.

3.1 VirtualWiFi:

(*Reference: http://research.microsoft.com/enus/um/redmond/projects/virtualwifi*)
As explained by the above reference, VirtualWiFi (previously known as MultiNet) is a virtualization architecture for wireless LAN (WLAN) cards. It abstracts a single WLAN card to appear as multiple virtual WLAN cards to the user. The user can then configure each virtual card to connect to a different wireless network. Therefore, VirtualWiFi allows a user to simultaneously connect his machine to multiple wireless networks using just one WLAN card. This new functionality introduced by VirtualWiFi enables many new applications, which were not possible earlier using a single WLAN card. For example,
With VirtualWiFi, you can connect to a guest's machine or play games over an ad hoc network, while surfing the web via an infrastructure network.
You can use VirtualWiFi to connect your ad hoc network, which may contain many nodes, to the Internet using only one node.
VirtualWiFi can help make your home infrastructure network elastic by extending its access to nodes that are out of range of your home WiFi Access Point.
There are several other applications of VirtualWiFi. One such application is called Client Conduit, which is a very useful tool for fault diagnosis and recovery in Wireless LANs. Client Conduit is a tool that provides a thin pipe of communication for disconnected clients to exchange diagnosis information with the back end servers. The thin pipe is achieved by running

VirtualWiFi on the connected clients. These clients dynamically connect to disconnected clients over an ad hoc network, and send messages from them to the back end servers. VirtualWiFi enables this thin pipe without requiring the connected client to explicitly disconnect from the infrastructure network. A more detailed description of Client Conduit can be found in the paper titled: "Architecture and Techniques for Diagnosing Faults in IEEE 802.11 Infrastructure Networks".

Another application of VirtualWiFi that increases the capacity of wireless ad hoc networks using orthogonal channels is called Slotted Seeded Channel Hopping (SSCH). SSCH uses VirtualWiFi to virtualize a wireless card with as many instances as the number of orthogonal channels. It then connects each virtual wireless card on a different orthogonal channel. Furthermore, SSCH proposes a novel scheme of partial synchronization that can be used with VirtualWiFi. The details of the SSCH protocol are described in another paper, titled: "SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks".

The third application of VirtualWiFi is called WiFiProfiler, which enables clients to cooperatively diagnose the root cause of various wireless problems.

Clients, including the ones that are disconnected from the WLAN, use VirtualWiFi to form an information plane, which is different from the data plane (the WLAN). All clients then exchange configuration information over this information plane and use this information to diagnose the root cause of various wireless failures. The details of this idea are described in a technical paper, titled: "WiFiProfiler: Cooperative Diagnosis in Wireless LANs".

3.2 Wireless Hosted Network:

(Reference: http://msdn.microsoft.com/enus/library/windows/desktop/dd815243(v=vs.85).asp)

The wireless Hosted Network is a new WLAN feature supported on Windows 7 and on Windows Server 2008 R2 with the Wireless LAN Service installed. This feature implements two major functions:

The virtualization of a physical wireless adapter into more than one virtual wireless adapter sometimes referred to as Virtual Wi-Fi.

A software-based wireless access point (AP) sometimes referred to as a SoftAP that uses a designated virtual wireless adapter.

These two functions coexist in a Windows system together. Enabling or disabling the wireless Hosted Network enables or disables both virtual Wi-Fi and SoftAP. It is not possible to enable or disable these two functions separately in Windows.

With this feature, a Windows computer can use a single physical wireless adapter to connect as a client to a hardware access point (AP), while at the same time acting as a software AP allowing other wireless-capable devices to connect to it. This feature requires that a Hosted Network capable wireless adapter is installed in the local computer.

The driver for the wireless adapter must implement the wireless LAN device driver model defined by Microsoft for use on Windows 7. To receive the Windows 7 logo, a wireless driver must implement the wireless Hosted Network feature.

There is at most one wireless Hosted Network enabled at any time on the local computer and only one wireless adapter will be used by the wireless Hosted Network. If there is more than one Hosted Network capable wireless adapter, Windows will choose one adapter for use with the wireless Hosted Network. When the Hosted Network APIs are used, the Hosted Network capable wireless adapter is virtualized to at most 3 logical adapters:

A station adapter (STA) for use by client or ad hoc wireless applications. The STA adapter inherits all the settings of the original physical wireless adapter and exhibits the same behaviors as the physical adapter. Conceptually, one can view the STA adapter as identical to the physical adapter after virtualization. The STA adapter is always in the system as long as the corresponding wireless physical adapter is present.

An AP adapter for use by the wireless Hosted Network to host SoftAP. The AP adapter is present in the Windows system only after the wireless Hosted Network is invoked for the first time (when the WlanHostedNetworkStartUsing, WlanHostedNetworkForceStart, or WlanHostedNetworkInitSettings function is first called). Once created, the AP adapter will remain in the system until the wireless Hosted Network is disabled. If the wireless Hosted Network is enabled at some later time, the AP adapter will show up in the system again.

A virtual station adapter (VSTA) for use by hardware vendors to extend the wireless Hosted Network capability in Windows.

The VSTA adapter is optional and can only be created in the system by the corresponding IHV service. Unlike the AP adapter, the VSTA adapter exists in the Windows system only from the time when the IHV service initializes the adapter until the time the IHV service releases the adapter.

Virtual Wi-Fi maps the logical adapters to NDIS ports. The binding of the STA, AP, and VSTA adapters to specific NDIS ports is decided by Windows. The STA adapter is always bound to Port 0. The AP adapter is bound to the next available NDIS port when virtualization starts, and the binding remains the same until virtualization ends when wireless Hosted Network is disabled. The VSTA adapter is bound to the next available NDIS port when it is initialized by the corresponding IHV service and the binding remains the same until it is released by the IHV service.

It is possible for the VSTA adapter to be created for use by IHVs without creating the SoftAP adapter.

The following combinations are valid for a physical adapter with virtualization:

. STA adapter.

. STA and AP adapters.

. STA and VSTA adapters.

. STA, AP, and VSTA adapters.

Except for the STA adapter case, all other combinations are only valid when the wireless Hosted Network is enabled. As for the single STA adapter case, it is the physical adapter if the wireless Hosted Network is disabled. If the wireless Hosted Network is enabled, it is the STA adapter when the wireless Hosted Network has never been invoked in the system.

Layer 2 bridging is prohibited between the AP adapter and any other adapters in the system.

The same restriction applies to the VSTA adapter when it is present in the system.

The wireless Hosted Network feature in Windows implements a SoftAP. However, this SoftAP is not designed to replace hardware-based wireless AP devices. In particular, if the wireless Hosted Network is running when the computer goes to sleep (standby), hibernate, or before the computer restarts, the wireless Hosted Network will be stopped. The wireless Hosted Network will not automatically restart after the computer resumes from sleep, hibernate, or restarts. In addition, SoftAP does not provide the DNS resolution. In the case where an external DNS server is not made available using Internet Connection Sharing (see the discussion of ICS below), fully qualified domain name (FQDN) resolution between any two computers or devices connected with the SoftAP, including the computer hosting the SoftAP, would only work if both entities

mark the network type of the SoftAP network as PRIVATE (HOME or WORK in the network category pop-up). Since the machine hosting the SoftAP always marks the SoftAP network type as PRIVATE, only the computers or devices connected to SoftAP need to mark the SoftAP network type as PRIVATE in order for FQDN resolution to work.
SoftAP and ad hoc networking are mutually exclusive on the same physical adapter. If SoftAP is running on the AP adapter and a user or application starts ad hoc networking on the STA adapter, SoftAP will be shut down. Iif ad hoc networking is running on the STA adapter, an attempt to start SoftAP on the AP adapter will fail.

3.2.1 Supported Scenarios for Wireless Hosted Network

The wireless Hosted network enables two major scenarios for Windows computers:

- The ability to provide a wireless Personal Area Network (wireless PAN) for use with various other wireless devices.
- Network connection sharing for use by other computers and devices.

The wireless PAN is the primary scenario enabled by the wireless Hosted Network on its own. Once the wireless Hosted Network is started on a computer, any wireless-capable device supporting WPA2-PSK/AES will be able to connect to the softAP just as if it is connecting to a regular hardware AP. Devices connected to the wireless Hosted Network form a wireless PAN, where they are able to exchange information with the Windows computer hosting the SoftAP as well as among themselves.
Network connection sharing for use by other computers and devices requires the use of Internet Connection Sharing (ICS). In this scenario, the public interface of ICS is the shared connection while the private interface is the virtual adapter hosting the SoftAP. The shared connection can be an Ethernet, wireless LAN, or wireless WAN connection. In the case of a wireless LAN connection, the public interface of ICS can be either from another wireless LAN adapter or the station virtual adapter on the same physical wireless adapter that hosts the SoftAP. The most common use for network sharing is sharing an Internet connection, where the network on the public interface of ICS has access to the Internet.
The wireless Hosted Network interacts with Wi-Fi Protected Setup (WPS) , another important new feature in Windows 7 and Windows Server 2008 R2 with the Wireless LAN Service installed.

The wireless Hosted Network and WPS support a scenario that provisions a WPS-capable device for a non-WPS capable hardware AP. In this case, the SoftAP hosted on Windows is invoked in the background to push the hardware AP profile onto the WPS-capable device.

3.3Internet Connection sharing:

(*Reference:http://msdn.microsoft.com/enus/library/windows/desktop/dd815252(v=vs.85).aspx*)

Internet Connection Sharing (ICS) is a feature in Windows provided through the SharedAccess Service. Strictly speaking, SharedAccess enables network sharing through a computer where the shared network access does not necessarily provide access to the Internet. We use the term ICS and SharedAccess interchangeably in this section, since Internet Connection Sharing is a major scenario for the wireless Hosted Network and the ICS term is better known to the user community.

Wireless Hosted Network is closely tied to ICS to enable both the wireless personal area network (PAN) and the Internet sharing scenarios. This section provides general recommendations to application developers on how to integrate wireless Hosted Network and ICS using the public wireless Hosted Network and ICS APIs.

3.3.1 Modes of ICS:

The ICS Service operates in one of the two possible modes:

Standalone mode: Only the DHCPv4 server function is operating when the ICS service is invoked. This is a special operation mode for ICS and is only made available through the wireless Hosted Network. A user or application is not able to directly start and stop standalone ICS through public ICS APIs or netsh commands. Starting the wireless Hosted Network typically involves starting ICS in standalone mode to use the DHCPv4 server function to provide private IPv4 addresses for connected devices.

Network communication for the connected devices is limited to sending and receiving network packets between a connected device and the local computer hosting the wireless Hosted Network and among the connected devices themselves. This effectively enables the wireless personal area network scenario for the wireless Hosted Network.

Full mode: All the features of ICS are operating when the service is invoked, such as network address translation and DHCP server functions for both IPv4 and IPv6. This is the normal mode of operation for ICS. A user or application may start and stop full ICS mode through public APIs or netshell commands. For example, this service can be stopped using net stop sharedaccess from an elevated command prompt. Combining wireless Hosted Network with full ICS, Network communication for the connected devices is not limited to the wireless PAN. Any connected device has access to network (such as the Internet) through the shared network connection from the computer running the wireless Hosted Network. This effectively enables the Network sharing scenario for the wireless Hosted Network.

In this section, we use the term full ICS to mean the case where all the ICS functions are invoked in ICS Service to provide access to all the full ICS features with the wireless Hosted Network.

The two ICS operation modes are mutually exclusive with full ICS taking higher precedence. The ICS Service may transition from standalone mode to full mode, but not from full mode to standalone mode. The ICS standalone mode was introduced in Windows 7 and on Windows Server 2008 R2 with the Wireless LAN Service installed in conjunction with the wireless Hosted Network feature. It is not available in previous versions of Windows.

Any full ICS operation involves two different network adapters in the system:

The public interface. This is the network interface with access to the Internet. It is this interface that the local computer running ICS uses to share the Internet with clients and devices connecting to it via SoftAP.

The private interface. This is the network interface that other devices use to connect to the local computer that is running ICS. A DHCPv4 server is running on this private interface to provide private local IP addresses to the other remote computers.

When the public interface does not have Internet access, the DHCP server on the private interface continues to provide local IP addresses to the connected devices. Standalone ICS only involves the private interface on which SoftAP is running; it does not involve any public interface.

At any time, there is at most one instance of full ICS running on the local computer. If full ICS is already running on the local computer, starting another full ICS exhibits the following functional behaviors:

If the public and private interfaces of the new full ICS are the same as the existing full ICS, starting the second full ICS is equivalent to a no-op.

If the new public interface is different from the old public interface, but the new private interface is the same as the old private interface, starting a second full ICS has little impact on the connected devices on the same private interface. The ability to access the Internet may change with the new public interface.

If the new private interface is different than the old private interface, ICS functions will stop working on the old private interface and start applying to the new private interface. Any remote device connecting to the local computer using the old private interface will lose IP connectivity to the local computer.

When full ICS is already running, invoking a second full ICS is disruptive to remotely connected devices using the old private interface as long as the second ICS integration uses a different new private interface.

To manage and use the ICS service to support ICS integration with wireless Hosted Network, a software application must first obtain an INetSharingManager interface. The INetSharingManager interface provides access directly or indirectly to all the other COM interfaces in the ICS API. The get_SharingInstalled method on the INetSharingManager interface reports whether the local computer supports connection sharing. The get_EnumEveryConnection method on the INetSharingManager interface retrieves an enumeration interface for all connections in the connections folder. The get_INetSharingConfigurationForINetConnection method retrieves an INetSharingConfiguration interface for the specified connection. Methods on the INetSharingConfiguration interface can be used to query and change ICS settings.

3.4 Hosted Network and ICS Integration:

When full ICS is not running, starting a wireless Hosted Network also internally starts the ICS Service in standalone mode with only the DHCPv4 server function to allocate IP addresses for the connected devices on the wireless Hosted Network interface. The subnet address range for the standalone DHCPv4 server is 192.168.173.0/24. This is different from the subnet range of 192.168.137.0/24 that is used with full ICS.

Starting a wireless Hosted Network with full ICS employs the following logic:

If full ICS is not already running, starting a wireless Hosted Network also starts the ICS Service with standalone DHCPv4 server.

If full ICS is already running and the private interface is the wireless Hosted Network interface, just start the wireless Hosted Network.

If full ICS is already running but the private interface is not the wireless Hosted Network interface, the wireless Hosted Network will be started without the DHCPv4 server function on the wireless Hosted Network interface.

The impact of the logic above highlights the following facts:

ICS does not transition from full mode to standalone mode.

Standalone mode can only be invoked by the wireless Hosted Network when ICS is not running in full mode.

If ICS is running in standalone mode, it will be preempted into full mode if a user or application starts ICS in full mode.

Transitioning from standalone mode to full mode in ICS will be disruptive to connected devices in the wireless PAN if the private interface of full ICS is not the same as the one for SoftAP.

It takes time to start or stop the ICS Service on the local computer in either full or standalone mode. An application should check the state of ICS Service using the NotifyServiceStatusChange function to make sure that the ICS Service is not in the start/stop pending state before starting or stopping the wireless Hosted Network for use with ICS integration.

3.5 Native WiFi API:

This project uses the Native WiFi API provided by the MSDN for development of both Hotspot and client application to switch to the best network.

3.6 Related Work:

There are some open source projects, Managed WiFi API, This project is a .NET class library allowing you to control Wifi (802.11) network adapters installed in your Windows machine programmatically. I have used this API to develop the WiFi Client service.

Source: http://managedwifi.codeplex.com/

Chapter4 - WiFi Client

In this chapter we will explain the second application named as WiFi Client.

This application is also developed as a windows service which keep the device connected to best network.

The use of this application is restricted to trusted environment for example in a corporate building which has more than one WiFi routers. The application will keep track of the connectivity of the host device, and if the connection is dropped or goes below a fixed threshold then it will search for the available networks and will connect to the best available network. The best network means the network with high signal strength and whose profile is accessible.

The design of this application requires that the wireless profiles of all the WiFi networks should be placed in a shared location which is accessible to the host machine and application.

4.1 Wireless Profile:

Wireless profile is a required parameter for connecting to a WiFi network in Native WiFi API. This is a file in XML format which keeps all the information regarding the WiFi network like SSID, Authentication, Encryption, shared key and other information. The profile XML format varies from network to network based on its type.

The more information about WiFi Profile and its various formats can be found at: http://msdn.microsoft.com/enus/library/windows/desktop/aa369853(v=vs.85).aspx and http://msdn.microsoft.com/enus/library/windows/desktop/ms707341(v=vs.85).aspx

4.2 How to get the Wireless Profile XML for a WiFi network:

Though the WiFi Profile XML can be formed by the filling the appropraiate properties value in a suitable WiFi Profile XML format provided by MSDN, but the easiest way to get the profile xml is explained below:

When we connect to any WiFi network and make it a default connection (by agreeing to the option “Connect automatically”) the profile is saved.

We can export this profile xml by following command:

1. To show the saved profiles:
   netsh wlan show profiles
2. To Export the profile to an XML file

```
netsh wlan export profile name=”name_of_the _profile” folder=”path of folder”
```

4.3 Example of the exported WiFi profile XML:

Here is given an example of the WiFi profile XML

```
/****************************************************/
<?xml version="1.0"?>
<WLANProfile xmlns="http://www.microsoft.com/networking/WLAN/profile/v1">
        <name>DLink</name>
        <SSIDConfig>
                <SSID>
                        <hex>444C696E6B</hex>
                        <name>DLink</name>
                </SSID>
        </SSIDConfig>
        <connectionType>ESS</connectionType>
        <connectionMode>auto</connectionMode>
        <MSM>
                <security>
                        <authEncryption>
                                <authentication>WPA2PSK</authentication>
                                <encryption>AES</encryption>
                                <useOneX>false</useOneX>
                        </authEncryption>
                        <sharedKey>
                                <keyType>passPhrase</keyType>
                                <protected>true</protected>

        <keyMaterial>01000000D08C9DDF0115D1118C7A00C04FC297EB0100000033C633F
CA3CDED4EA06B20CD24B49D8700000000002000000000010660000000010000200000000DD2
CDFB4843018864E9068A3DE3A9F82C2D71FAB86BC143CF475564146F3B3CA000000000
E80000000020000200000000A72E2A5136816FBDC849DD480B0027776D9C246029005807C
E2C12DE5B6383631000000001774590659F8B20052ACE58F4F7F6DA040000000917870A377
```

```
97327ECE7ABA5727AFC068D720324F8A5B2FF5305B714D611467EBF966A9538384F9B61
BB22257F94E52003081698D18CD0B4140E9B2A5B69BA0FA</keyMaterial>
                        </sharedKey>
                </security>
        </MSM>
</WLANProfile>
/*********************************************************/
```

4.4 Modification in the exported profile xml:

Here one point is to be noted that the value of <keyMaterial> is encrypted.

If we use this profile xml to connect to the same WiFi network on some other machine, it will prompt for the key again.

Therefor the keyMaterial needs to be changed to the actual key (not encrypted form).

To use the actual key instead of the encrypted keyMaterial, we need to change one more parameter in the above xml file which is <protected>.

Change the value of <protected> to “false”. And then change the <KeyMaterial> to actual key.

Following is the example of modified xml:

```
<?xml version="1.0"?>
<WLANProfile xmlns="http://www.microsoft.com/networking/WLAN/profile/v1">
        <name>DLink</name>
        <SSIDConfig>
                <SSID>
                        <hex>444C696E6B</hex>
                        <name>DLink</name>
                </SSID>
        </SSIDConfig>
        <connectionType>ESS</connectionType>
        <connectionMode>auto</connectionMode>
        <MSM>
                <security>
                        <authEncryption>
                                <authentication>WPA2PSK</authentication>
                                <encryption>AES</encryption>
                                <useOneX>false</useOneX>
                        </authEncryption>
                        <sharedKey>
                                <keyType>passPhrase</keyType>
                                <protected>false</protected>
                                <keyMaterial>LeLeh1234</keyMaterial>
                        </sharedKey>
                </security>
        </MSM>
</WLANProfile>
```

4.5 How does the WiFi Client application works:

This application uses the NativeWiFi API and is developed in C#.NET.

I have taken the help from an open source library Managed WiFi which provide the C# version of actual NativeWiFi API (written in C+) by using PInvoke.

The application does the following:

1. Keep pinging for checking the connectivity of the host if the connectivity is dropped i.e the current connection is dropped then:
2. Check for the available networks
3. Get the signal strength of every network.
4. Sort the networks according to their signal strength.
5. Take the available wireless profiles from the shared location for each network in the sorted list.
6. Connect to the best network i.e network with highest signal strength whose profile xml is available.

4.6 WiFi Client application’s Algorithm:

The following is the algorithm of the WiFi Client application:

```
On Service start:
    Run Always:
        IF not connected to internet:
            Get All Available Networks.
            Sort networks in descending order by sig. strength.
            Loop For all networks in sorted list:
                IF profile Xml is available:
                    Connect to the Network.
                    Break Out of Loop.
                ELSE:
                    Continue.
            END Loop.
#End of program.
```

4.7 Implementation and High Level Design:

This section gives the high level design of the WiFi Client application. Following are the various design diagrams

4.7.1 Use Case Diagram:

The below is the high level use case diagram. The user starts and stops the service and rest of the actions are controlled by service itself. Please note that the availability of the internet connection also works as passive actor as if the connection is dropped or current signal strength goes below a given threshold the service works accordingly. The available WiFi networks are also as passive actors as host can connect to anyone of these and profiles contains the information like password and authentication type of the trusted networks. The host connect to a given network only if the profile for the network is available at the given location.

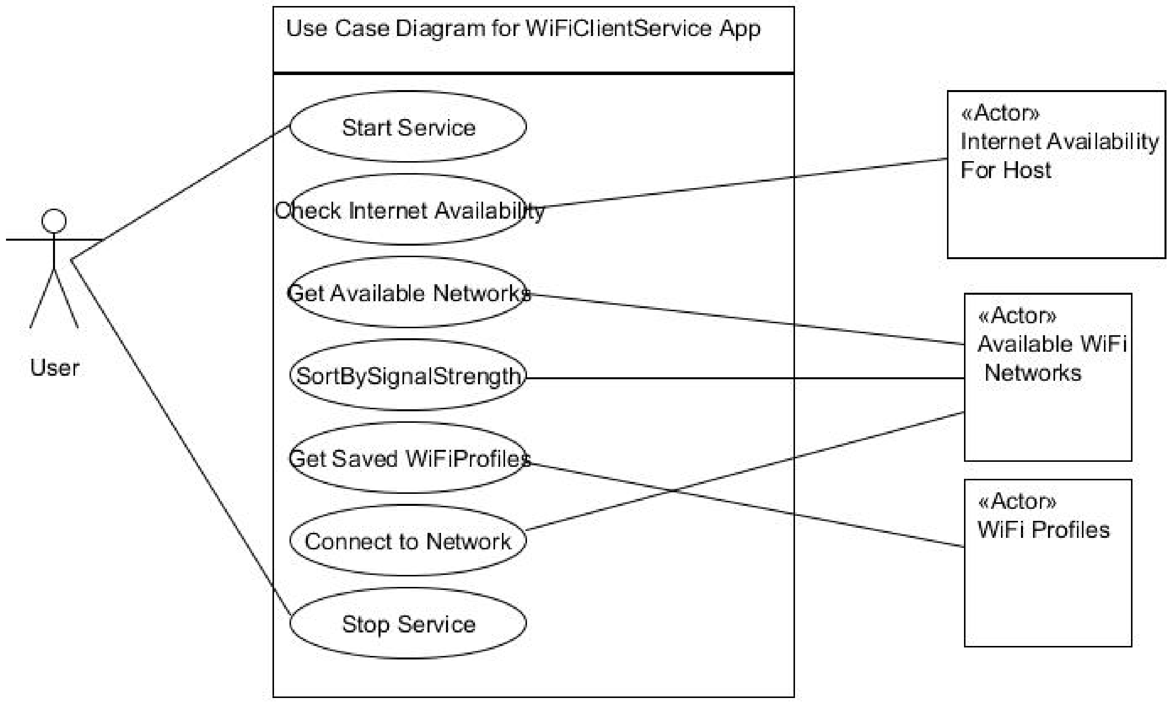
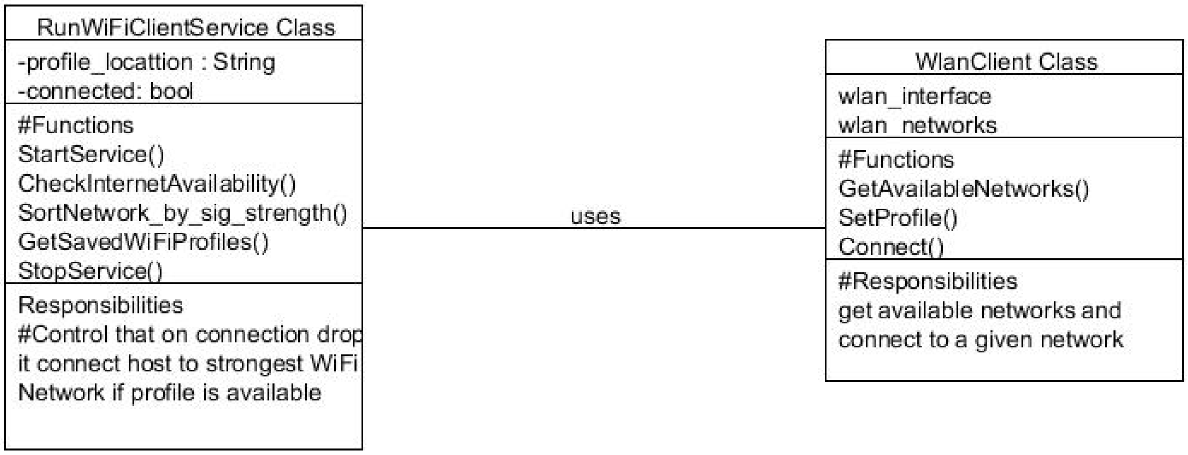

Use Case Diagram for WiFiClientService App
Start Service
Check Internet Availability
Get Available Networks
SortBySignalStrength
Get Saved WiFiProfiles
Connect to Network
Stop Service
User
«Actor» Internet Availability For Host
«Actor» Available WiFi Networks
«Actor» WiFi Profiles

Figure 4.1 Use Case diagram for WiFi Client App.

4.7.2 Class Diagram:

The following is the class diagram of the WiFi Client Application. This application has two major classes, the RunWiFiClientService which uses the Wlan Client class to get available networks and connect to the best possible network.


RunWiFiClientService Class
-profile_locattion : String
-connected: bool
#Functions
StartService()
CheckInternetAvailability()
SortNetwork_by_sig_strength()
GetSavedWiFiProfiles()
StopService()
Responsibilities
#Control that on connection drop it connect host to strongest WiFi Network if profile is available
uses
WlanClient Class
wlan_interface
wlan_networks
#Functions
GetAvailableNetworks()
SetProfile()
Connect()
#Responsibilities
get available networks and connect to a given network

.Figure 4.2 Class diagram for WiFi Client App

4.7.3 Sequence Diagram:

The following is the sequence diagram for WiFi Client application. It shows the interaction between various classes and their sequence. The RunWiFiClientService class act as the controlling class it checks if system is itself connected to internet and then interact with the WlanClient class to get the all available WiFi networks. If the connection is dropped or current signal strength goes below, the service gets the available WiFi networks using WlanClient class, sort them according to the signal quality. Now RunWiFiClientService class get the profile for the best network and connect to it. If the profile is not available, then the next best network is tried. This works in a loop and terminate when system is successfully or all the network has returned failure.

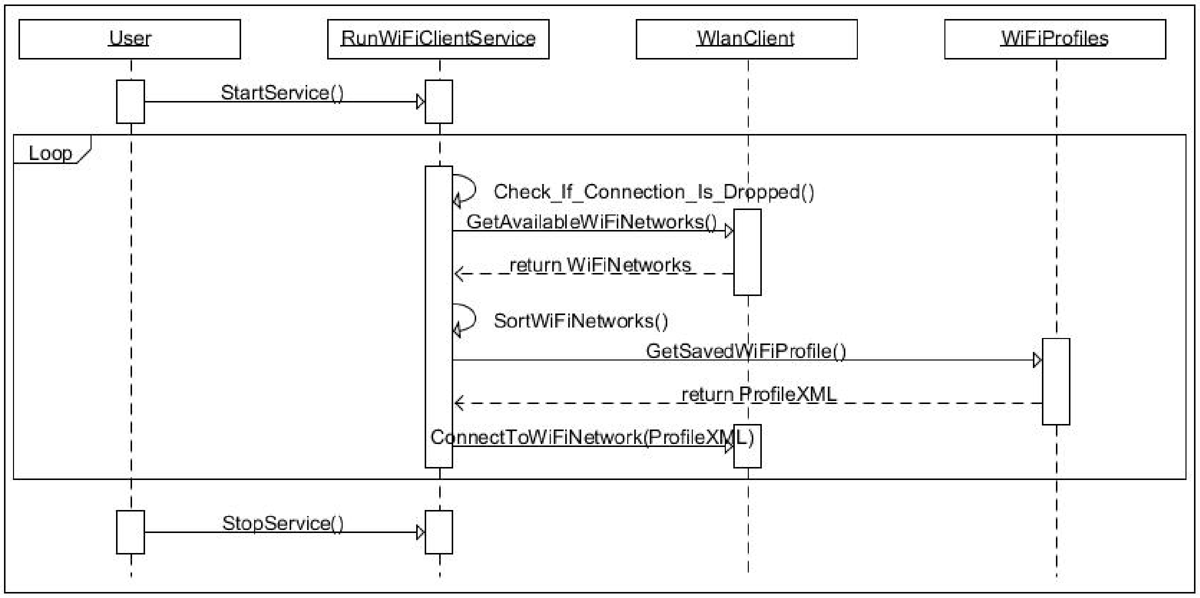


Figure 4.3 Sequence diagram for WiFi Client App.

Chapter5 – Conclusion

WiFi Client Service:

This is the smart application which keeps the host device connected to best available network always. The service find out if the device is connected to internet or not, in case the device is not connected to internet it will search for the available WiFi networks and connect to the best available network. Here the best network implies to the network with highest signal strength.

This application is supposed to be run in a trusted environment and the application will only connect to those networks whose details are already shared in a location which is accessible to the host device.

The shared details are basically saved profile xml's of the WiFi networks which are required as the input parameter for connection setup.

Directions for future Work.

In WiFi client application the profile xml's can be created automatically at run time by accessing the properties of a WiFi network and then creating the profile from a suitable format. This will obviate the need of storing entire profile xml file corresponding to a network and only SSID and password will be required to store.

In WiFi client app we can store the profile xml in an encrypted form which is decrypted by the application at the run time. For this we can use symmetric encryption scheme like AES.

References

1. Virtual WiFi :

http://research.microsoft.com/en-us/um/redmond/projects/virtualwifi/

2. Wireless hosted network:

http://msdn.microsoft.com/en-us/library/windows/desktop/dd815243(v=vs.85).aspx

3. Internet Connection sharing:

http://msdn.microsoft.com/en-us/library/windows/desktop/dd815252(v=vs.85).aspx

4. Virtual Router Plus (Codeplex)

http://virtualwifihotspot.codeplex.com/releases/view/101496

5. Managed Wifi API:

http://managedwifi.codeplex.com/

6. Native Wi-Fi API:

   6.1 Native Wi-Fi Enumerations:

   http://msdn.microsoft.com/en-us/library/windows/desktop/ms706273(v=vs.85).aspx

   6.2 Native WiFi Structures:

   http://msdn.microsoft.com/en-us/library/windows/desktop/ms706276(v=vs.85).aspx

   6.3 Native WiFi functions:

   http://msdn.microsoft.com/en-us/library/windows/desktop/ms706274(v=vs.85).aspx

7. Wireless profile samples:

http://msdn.microsoft.com/en-us/library/windows/desktop/aa369853(v=vs.85).aspx

Checklist of items for the Final Dissertation Report

This checklist is to be attached as the last page of the report.

This checklist is to be duly completed, verified and signed by the student.

1.	**Is the final report neatly formatted with all the elements required for a technical Report?**	Yes
2.	Is the Cover page in proper format as given in Annexure A?	Yes
3.	Is the Title page (Inner cover page) in proper format?	Yes
4.	(a) Is the Certificate from the Supervisor in proper format? (b) Has it been signed by the Supervisor?	Yes Yes
5.	Is the Abstract included in the report properly written within one page? Have the technical keywords been specified properly?	Yes Yes
6.	Is the title of your report appropriate? **The title should be adequately descriptive, precise and must reflect scope of the actual work done.** Uncommon abbreviations / Acronyms should not be used in the title	Yes
7.	Have you included the List of abbreviations / Acronyms?	Yes
8.	Does the Report contain a summary of the literature survey?	Yes
9.	Does the Table of Contents include page numbers? (i). Are the Pages numbered properly? (Ch. 1 should start on Page # 1) (ii). Are the Figures numbered properly? (Figure Numbers and Figure Titles should be at the bottom of the figures) (iii). Are the Tables numbered properly? (Table Numbers and Table Titles should be at the top of the tables) (iv). Are the Captions for the Figures and Tables proper? (v). Are the Appendices numbered properly? Are their titles appropriate	Yes Yes Yes Yes Yes Yes
10.	Is the conclusion of the Report based on discussion of the work?	Yes

11.	Are References or Bibliography given at the end of the Report? Have the References been cited properly inside the text of the Report? Are all the references cited in the body of the report	Yes Yes Yes
12.	Is the report format and content according to the guidelines? The report should not be a mere printout of a Power Point Presentation, or a user manual. Source code of software need not be included in the report.	Yes

Declaration by Student:

I certify that I have properly verified all the items in this checklist and ensure that the report is in proper format as specified in the course handout.

Place: **Signature of the Student**

Date: **Name: Hitender Prakash.**

ID No: 2011HZ13065