A simple implementation to roaming between GPRS and WiFi Networks[1]

M. Adda, J. Sanders and A. Peart

University of Portsmouth, Portsmouth, UK
mo.adda@port.ac.uk
jim@jlsnet.co.uk

Abstract

Roaming between networks of different protocols and structures is one of the common situations, especially with the advances in the Internet technology, and the available services that users need to constantly access for their daily life. In this paper, we present the development of a software solution that is deployed to permit a seamless roaming between the two wireless network protocols 802.11b (WiFi), a wireless local area network, which is increasingly being used in homes and offices, and GPRS, a wireless wide area network which works as a data extension to the GSM mobile phone network. The solution is implemented and tested in hardware. The proposed solution tested for several applications demonstrates a reduction on network error swapping and shows higher performance.

1- Introduction

The modern generation of subscribers do not want to be tied to a desk, study or office. Technologies such as mobile phones, PDA's and laptop computers have become an everyday tool in people lives enabling them to become more mobile. Mobile technology has advanced in recent years allowing people to communicate and access information via a number of channels, while roving between different networks. However, the types, protocols, purposes and incompatibilities of wireless networks in existence today WiFi, GPRS, UMTS, WiMAX, Bluetooth, and HiperLAN/2 cause some concern to network engineers. In this paper we will only consider the roaming between WiFi and GPRS networks as shown in figure 1.

The main problem, when roaming between different wireless network types, is that a mobile node must first detect the loss of a wireless connection and therefore its route to a destination. The mobile node then needs to find another available wireless connection, reset its routing tables and change protocol types, before it can then communicate across the new link. During this process the correspondent node does not know when or where the mobile node has moved and therefore does not know where to send its replies. This process takes time and may result in huge packet loss, during which any real-time network applications, such as streaming video and audio, file transfers and games may be likely to timeout.

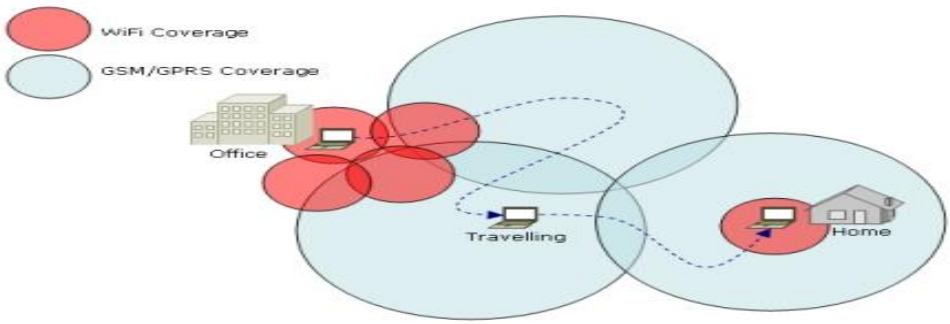

Figure 1 - Mobile Roaming Between WiFi and GPRS Networks

Several solutions have been proposed to deal with these problems. Mobile IP is a good solution to wireless roaming, but it has its downfalls. Firstly, there is need for more specialised hardware. Secondly,

[1] Support from INTEL Corporation (UK) and Orange (PCS) Ltd.

when a mobile node travels across a number of different wireless networks, it will build up a 'daisy chain' number of foreign agents. Traffic may have to be routed across this chain of different networks before it gets to the mobile node. This will add a large degree of latency and limit the bandwidth availability to the slowest link in the daisy chain [1]. In Occasionally Connected Computing, OCC, such as Intel OCC [2] and IBM projects, MQ Anywhere [3], nodes operate regardless of how or when they are connected to a network or the Internet. However, this works well for applications that operate on batch style processing, such as e-mail, file transfer or newsgroups, but would be less suitable for activities such as streaming media which require a constant real-time connection. Intel, Nokia and a number of other organisations are developing SoC (System on Chip), a single piece of silicon, which includes multiple DSP's (Digital Signal Processors) along with high-speed reconfigurable logic processors, in order to execute multiple wireless protocols [4]. Currently, SoC allows the switching between WiFi and HiperLAN/2. Efforts are being directed towards GPRS and UMTS. Software Defined Radio (SDR) modifies or replaces software programs or firmware, to change the radio's functionality, communications protocols and frequency bands. Such flexibility allows for easy upgrade to new modes and protocols without the need to totally replace hardware. The concept of SDR is rapidly gaining commercial popularity, not only for the GPRS and mobile phone industry, but also for wireless computer networks such as WiFi [5, 6].

Both SDR and SoC are still under development. Both would require new hardware to replace current wireless network devices. Mobile users who do not wish to replace their existing expensive hardware would need a software solution to work on top of their current set ups.

2-Proposed solution

In this section, we describe the mobility solution to seamless swapping between wireless networks. The NetSwap Driver is a piece of software, which will represent a 'virtual network interface' in a Mobile Node. The driver sits on top of all other network device drivers. Its purpose is to fool any Internet application into thinking that the mobile node has a constantly connected network device. The NetSwap driver will then act as a 'proxy', utilising the other different network devices (e.g. GPRS and WiFi) depending on their availability. There is a standard for communicating with most new network drivers called NDIS (Network Driver Interface Standard). NDIS is fast becoming an integral part of all Windows operating systems and all Windows compliant Network adapters are shipped with NDIS compatible drivers [7].

The NetSwap Driver is responsible for monitoring all connections. It routes the traffic from the application to one physical card at a time, depending on availability and priority order. Priority may be based on cost, routing metric, bandwidth, or it can be user defined. For example, WiFi may be of higher priority than GPRS due to it being a quicker connection. The main job of the NetSwap Driver is to mislead any applications into using the 'virtual network interface' before it uses any other network interface. One method of doing this would be to force the metric values for all other interfaces in the system up, thereby giving the virtual interface the lowest metric. For example, the driver could give itself a metric of 10 and move Ethernet to 11, Wireless to 31 and GPRS to 51. Therefore, Internet application sending network traffic will go through the NetSwap Driver with the new lower metric. The NetSwap driver must then force all packets to be sent through a special router (see figure 3 – NetSwap Gateway) by encapsulating them in a second IP header. It should then communicate with the special router by letting it know which interfaces are currently available on the mobile node. This communication would be done at regular intervals while switching between interfaces.

The NetSwap Gateway, figure 2, is a router sitting on the Internet, which acts as a central point to relay all traffic through. As discussed earlier, the mobility problem occurs when a mobile node swaps between two networks and the correspondent server is unaware of where the mobile node has moved to, rendering it unable to reply. The NetSwap Gateway must have a static IP address as it is used as a 'known point' and should not change. Its job is to forward all traffic to and from the mobile node's current address. There is an 'always on' connection as far as any applications running on both the mobile node and the correspondent are concerned, and the fact that the traffic goes through the NetSwap gateway should be transparent to both ends.

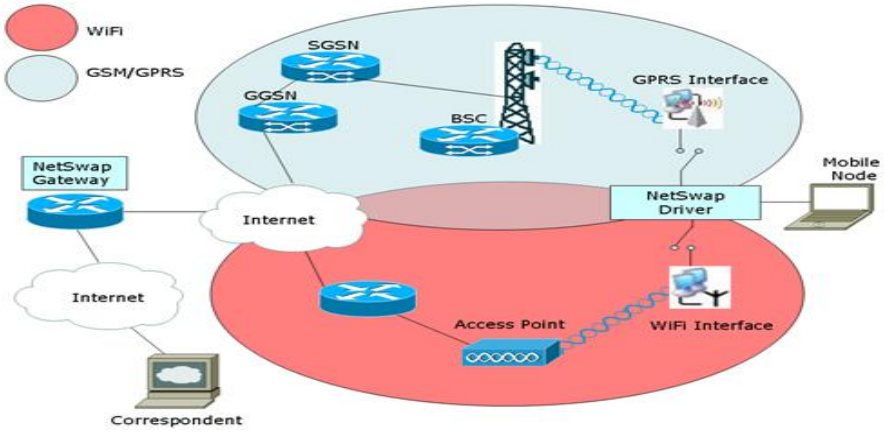

Figure 2- Switching Between Network Interfaces

3 – Implementation

Although we have simulated the solution with NS-2 simulation tool, in this paper we only present the hardware support to the implementation of the roaming problem. Intel Corporation have supplied both a GPRS Adapter (Intel PRO/Wireless GPRS 3110 PC Card) and an 802.11b WiFi Network Adapter (PRO/Wireless LAN 2100 3B MiniPCI Adapter). Orange (PCS), a UK mobile phone operator, have kindly donated a SIM card with a year's subscription and unlimited connection to their GPRS network to this research project, see figure 3.

Using this hardware, some simple tests have been done by running a few different network applications and observing what happens when network adapters lose connection during the swapping process (roaming). The simulation was used to support these results.

Figure 3 - Intel PRO/Wireless GPRS 3110 PC Card with Orange SIM card and Intel PRO/Wireless LAN 2100 3B MiniPCI Adapter.

3.1 **Equipment Set-up - Architecture**

The GPRS and WiFi cards have been installed in a desktop PC of sufficient specification and configured with optimal settings. In order to ensure fair testing, both cards are from the same manufacturer (Intel) and have had external antennas connected to improve signal strength.

An access point (Intel 2110B series) has been installed in an optimum position to provide sufficient wireless signal to the test equipment, and GPRS coverage from Orange is good. Before tests began, both WiFi and GPRS report full signal strength, see figure 4. The Access to the Internet is obtained through a Cisco 2611 router and a NTL cable modem, see figure 5.

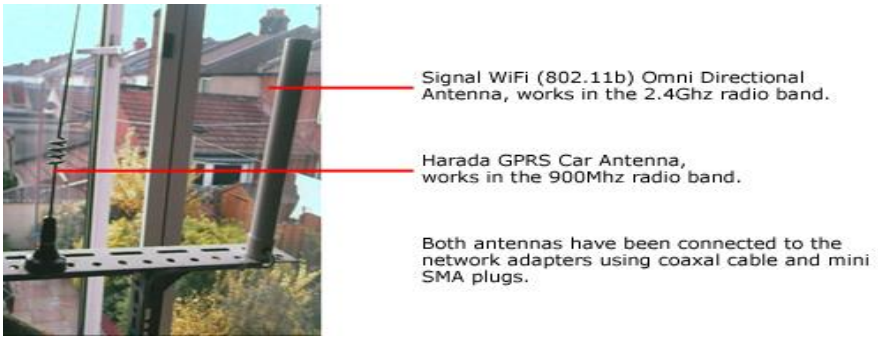

Figure 4 – Physical Equipment Antennas

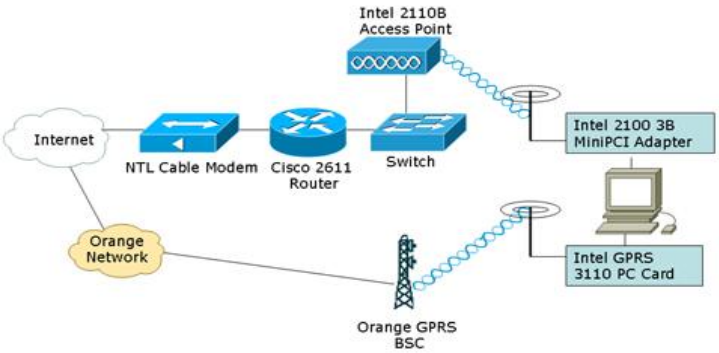

Figure 5 – Test Equipment Set-up Architecture

3.2 Application Configuration

We have used following software to conduct the experiment and support the NetSwap driver. **CuteFTP** was set-up to transfer files from a correspondent server on the Internet. It was later found that most FTP connections are "stateless". This means that connections between client and host are created and if connection is lost halfway through a file transfer the FTP server will have no knowledge of any clients trying to re-establish a transfer, therefore in normal circumstances after changing interfaces a new file transfer will need to be restarted again. The NetSwap solution should solve this problem by fooling each end into thinking there is an 'always on' connection even if the user swaps between the interfaces are swapped between. **Microsoft Windows Media Player**, which was used to stream audio from the online radio channel http://www.southernfm.com, was configured with the default buffer (5 seconds) and detect connection speed option. **Icomera** was configured with an Icomera Gateway and setup to use the Intel GPRS and WiFi interface cards. **Intel PROSet** was installed and the control panel was used to bring interfaces up and down to simulate the action of interfaces failing. IP addresses were manually assigned to the WiFi card to eliminate DHCP assignment problems. **Ping** was used as a test tool to see if correspondent hosts on the Internet were working or not. A couple of tests were done by running ping with the "no timeout" option, which would mean ping would keeps pinging until stopped by the user. This was done by using the –t parameter (e.g. **"ping www.jlsnet.co.uk –t"**)

3.3 Recording Findings and Recreating problem in Physical Hardware

As it is hard to measure lost packets in a real life scenario without protracted use of packet sniffing software, it was decided to simply measure the performance of the applications in time. This was measured using a stopwatch while running the various applications listed above in real life use-scenarios. Three case scenarios were tested:

- Starting with both WiFi and GPRS online and recording what happens when WiFi is taken down midway through a test
- Switching WiFi back on while GPRS is online. Does it swap back to the faster, lower metric interface in the tests?
- Starting with WiFi and GPRS on and switching GPRS off midway through a test

3.4 Simulating the effect of "Roaming"

Since the GPRS and WiFi adapters have been installed in a desktop computer, the equipment set-up is not very mobile. It is therefore not easy to roam between different wireless network types to get low signal etc. Rather than move the wireless adapters out of radio range of the transmitters (access point and GPRS BSC), a preferable way to reduce signal is to cover the antennas up. A tube covered in layers of aluminium foil, when placed over the antenna, was found to reduce signal to each interface enough to lose connection. This is an effective way of simulating losing range of a wireless network, see figure 6. This approach can be used with both the WiFi and GPRS antennas and makes roaming tests fairer. Due to the fact that the equipment is staying in one place, signal strengths can be controlled more reliably and signal loss can be reproduced on demand.

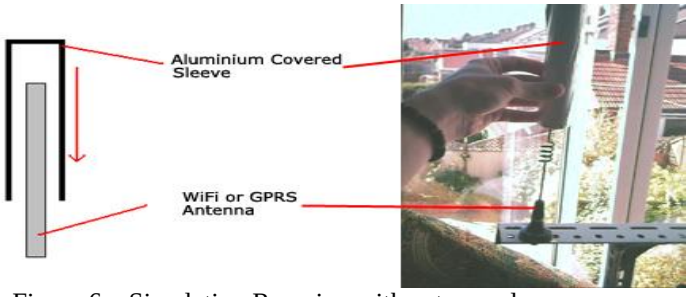

Figure 6 – Simulating Roaming with antenna sleeve

Figure 7 from window Task Manager shows network interface utilisation as can be seen in the higher resolution graphs, at the point where a wireless interface goes down there is a short period of time before normal communication resumes on the second interface. During this, any of the tests performed timeout or take a great deal of time to recover, usually with adverse effects.

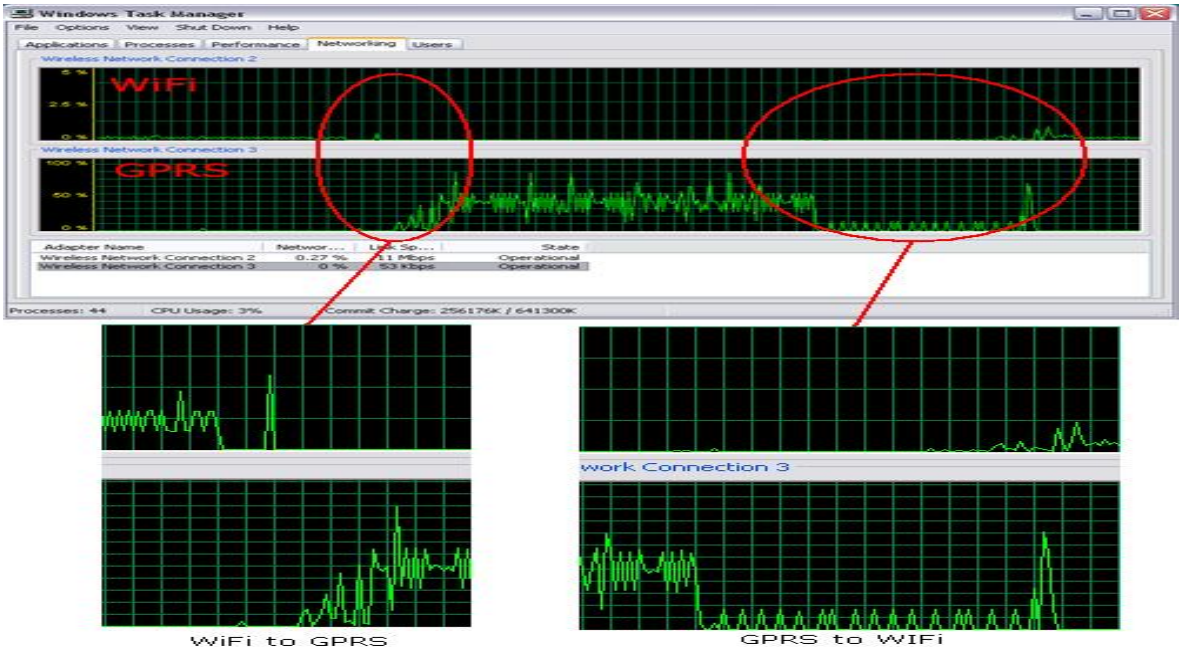

Figure 7 – Output from Tasks Manager showing network utilisation

The first test performed was to observe what happens when interfaces are swapped between during an FTP session. As expected, due to the stateless nature of FTP, the FTP session is unable to resume on a second interface after a wireless interface loses connection. Figure 8 shows the utilisation of both the GRPS and WiFi wireless interfaces in the following scenario; Initially both WiFi and GPRS are connected and then WiFi looses radio range and, as a result, its network connection. At the beginning of this test, the FTP session was working across the WiFi interface (red line). As soon as WiFi looses connection, the FTP session is terminated and will not restart without user intervention. Therefore, GPRS (blue line) does not ever get utilised for this test. As FTP is not a very good "seamless" protocol, it was decided to move onto an application, which may perform a little better when network connection is lost. Windows Media Player was chosen to stream audio from an Internet radio station.

The following tests produced some rather varied and unexpected results. Due to the fact that Media Player uses a buffer to read its audio stream ahead of what is actually being played, it was thought that this might offer a bit of leeway as interfaces were swapped between. The first test was to observe the effects, when both WiFi and GPRS interfaces are online and the WiFi interface loses connection while streaming media, see figure 9. These results were more as expected. Initially, Media Player was streaming media across WiFi due to its lower metric value. As the WiFi connection goes down, the buffer starts to run out. After the point where the buffer is empty, there is a delay while the routing tables are reset and a new route is established using GPRS. The buffer is then reloaded and streaming media resumed. This whole process takes approximately 26 seconds. Had the Media Player buffer been longer, there is a possibility that, as the interfaces changed; it may have appeared to swap a little more seamlessly. This is due to the fact that, during the time where the buffer is being utilised after an interface goes down, the routing tables could be reset. If this process is done in time, before the extended buffer runs out, streaming can resume without a break.

Figure 10 shows the effect that a WiFi reconnection has on Media Player stream, which is currently streaming over GPRS. The results are quite curious as they show that, at the point where WiFi is re-established, the buffer on GPRS is instantly cut and reloaded on WiFi. At the point where the WiFi buffer is fully loaded, it is again instantly cut and, strangely, reloads back on GPRS. The end result is that both WiFi and GPRS interfaces are online, but Media Player is still streaming its media across GPRS. Due to this, a long period of time (157 seconds) passes before streaming media is resumed, albeit on the more expensive, slower metric, GPRS. It is not known why such an effect is produced.

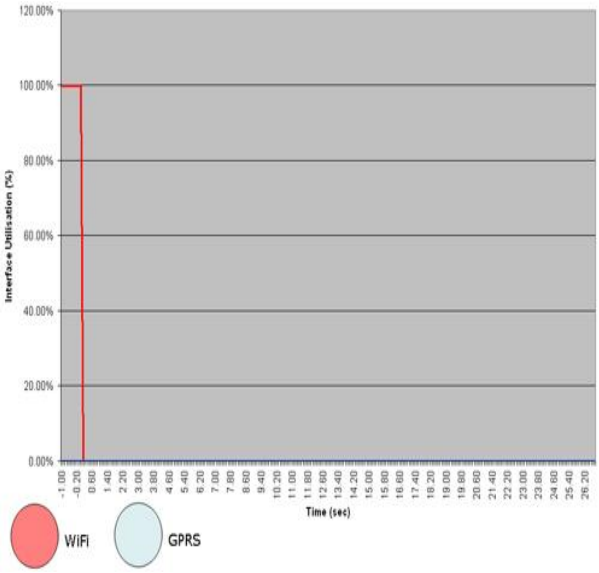

Figure 8 – FTP session using real hardware

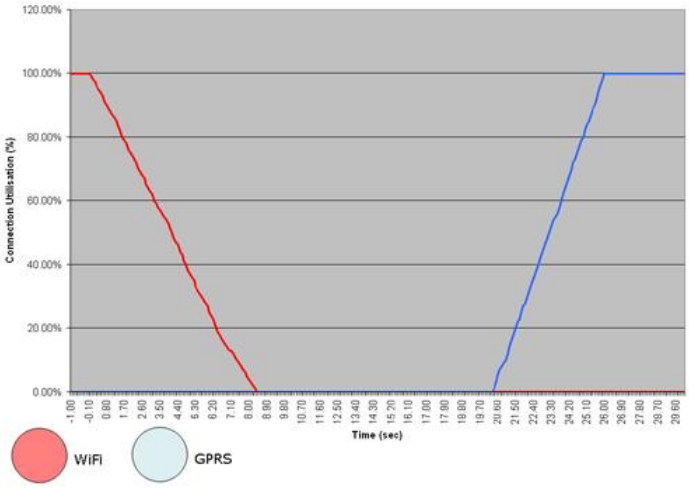

Figure 9 – Streaming Media, WiFi > GPRS

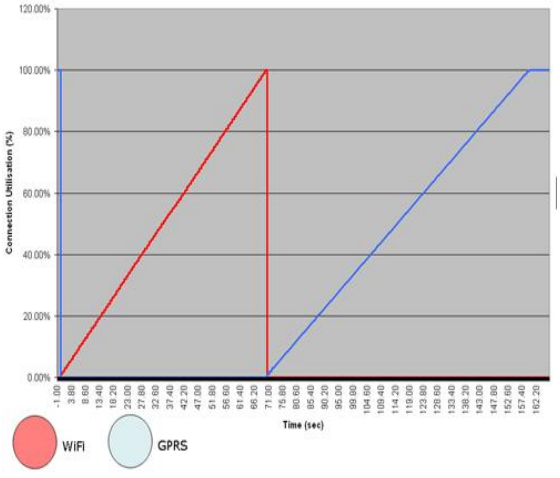

Figure 10 – Streaming Media, GPRS > WiFi > GPRS

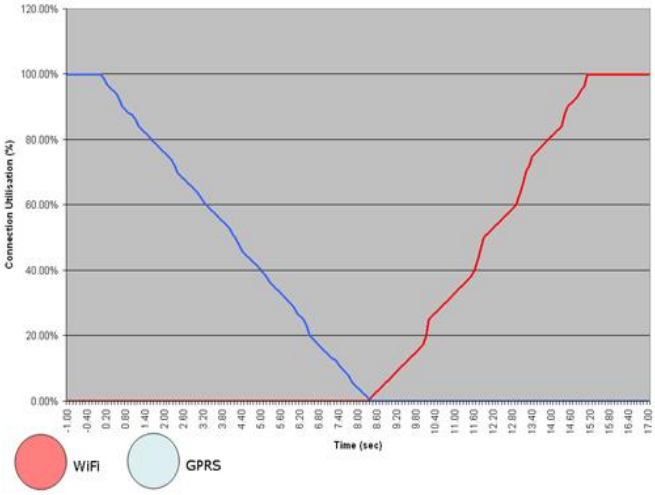

Figure 11 – Streaming Media, GPRS > WiFi

Following on from the previous test with both WiFi and GPRS interfaces online and media streaming over GPRS, a GPRS connection loss is now simulated. Figure 11 shows the best Media Player results so far, the GPRS buffer runs out and the WiFi buffer reloads almost instantly. There is a total streaming media loss of approximately 15 seconds.

A quick test was performed using the Ping tool, `C:\>ping www.jlsnet.co.uk -t`, in an attempt to discover how long it takes for it to start using another wireless connection after one goes down. After the 4th ping poll, the WiFi antenna was covered simulating the loss of signal. There are four "Request timed outs" before the pings resume on GPRS. The default timeout for ping on the Windows XP Operating System is 750ms. Therefore, the total loss is 3000ms (3 seconds).

4- Conclusion

After implementation and testing of the NetSwap solution, some encouraging results were produced. These results proved that the NetSwap solution is able to cut down the packet loss and crossover time as a mobile node roams between different types of wireless network. As the NetSwap solution is not protocol dependent it is not restricted to GPRS and WiFi protocols. The solution can be used with any type of IP based network interface, including wired connections such as Ethernet. The NetSwap design is a software based solution, which means the driver could be implemented on any type of mobile node (including Laptops, PDA's or Mobile Phones) and the gateway can be implemented on any type of router (e.g. Cisco router or Linux box).

References

[1] Gigaport 2002, How Mobile IP Works,
http://www.gigaport.nl/netwerk/access/ta/mip/en_mobileip.html , 2003

[2] Occasionally Connected Computing
http://cedar.intel.com/cgi-bin/ids.dll/content/
content.jsp?cntKey=GenericEditorial::mobile_occasionalconnect
&cntType=IDS_EDITORIAL&catCode=CVS, 2003

[3] MQ Anywhere,
http://www-3.ibm.com/software/integration/wmqe/ , 2003

[4] SOC - System-On-Chip,
Retrieved from http://soc.ece.ubc.ca/, April 2004

[5] A Software-Defined Radio for the Masses, http://www.arrl.org/tis/info/pdf/020708qex013.pdf, Part 1, Page 13, Accessed 2003

[6] Radio Free Intel: Research and Development at Intel
Retrieved from http://www.intel.com/labs/radio/, 2004

[7] What is NDIS?, SisTech - NDISOne,
Retrieved from http://www.ndisone.com/whatisndis.html, 2003