# System Development Requirements of Intelligent Object Framework

Sasa Savic[#1], Hao Shi[#2]
[#]College of Engineering and Science, Victoria University
Melbourne, Australia

***Abstract*— The primary aim of the research project is to explore and evaluate several wireless communication standards, protocol and application standards for device control and management of smart homes or industries over the wireless medium, in order to better understand the requirements needed for the proposed Intelligent Object Framework (IOF). As the technology is evolving, certain aspects of communication components have been neglected and not standardized across platforms. Companies have been developing proprietary communication protocols and applications to control devices. This trend introduces an issue of complexity and confusion amongst integrators, developers and users. The IOF's primary objective is to introduce a communication framework and application to remove the platform and device dependency from the scope. Implementation of the IOF involves several software packages to be pre-installed to a dedicated system to enable seamless communication and message exchange between different system components. Apart from the software package pre-installation, the system requirements outline two platforms suitable for the implementation of Intelligent Objects. Intelligent Object hardware discrepancies must be obvious to satisfy the end result for the proof of concept. This is achieved by introducing an embedded system platform and a smart-device platform in implementation of the IOF.

***Keywords*—** Intelligent Object Framework, IOF, Mobile Network, Wireless Network, System Development Requirements.

## I. Introduction

Wireless communications have become a standard in our lives. This is due to the technological advancements made on the frontier of telecommunications. From the early stages to telemetry systems to the current communication standards wireless communication has stretched far and wide. Wireless technology is best known for its capabilities and usage in personal devices such as laptops and mobile phones. Essentially each one of the devices mentioned has an embedded wireless module. Depending on the configuration of the module, main processing is usually done on its main baseband sub-system of processors but the device is still managed by the main processor of the device that is attached to. It may be noticeable that wireless modules are also of embedded nature, adding to the complexity of embedded devices that inherit their functionality. This enables wireless embedded modules to be integrated into wide variety of applications not only laptops but also smartphones.

### *A. Wireless Standards*

There are a number of applicable standards currently implemented by wireless infrastructure. The primarily focus is Wi-Fi technology [1, 2] and the other telecommunication mediums are equally important for data exchange. Infrared (IR) and Radio Frequency Identification (RFID) are commonly used on credit cards and access cards while Bluetooth are used for mobile phones and laptops. All wireless technologies primarily give the same desired outcome but the major difference is performance and range. Performances vary with different types of wireless standards and technologies. The range must also comply with FCC standards and can only operate in a specified frequency range. Wireless devices implement signalling methods defined by the specification. For Bluetooth, it is commonly referred to as Frequency Hopping Spread Spectrum (FHSS) whereas Wi-Fi (802.11) standards [3, 4] use Direct Sequence Spread Spectrum (DSSS) and Orthogonal Frequency Division Multiplexing (OFDM) for secure communication and also to accommodate multi-clients. Commonly used wireless technologies are part of IEEE 802.11 standards [3] as listed in Table 1.

TABLE I
WI-FI PROTOCOL, SPEED AND COVERAGE [3]

| **802.11 Protocol** | **Release** | **Indoor Range (meters)** | **Speed (Mbit/s)** |
|---|---|---|---|
| - | 1997 | 20 | 2 |
| a | 1999 | 35 | 54 |
| b | 1999 | 38 | 11 |
| g | 2003 | 38 | 54 |
| n | 2009 | 70 | 150 + |

### *B. 3G & High Speed packet Access (HSPA) Standards*

In 2008, Telstra was the first telecommunications provider in the world to offer 3GPP service to its customers [5]. The provided service was the very powerful EDGE (Enhanced Data rates for GSM Evolution) technology. It improved data transmission rates, which was an extension on top of the standard GSM (Global System for Mobile communications). HSPA was implemented on top of the EDGE infrastructure. The current HSPA provides data rates up to 56 Mbit/s on the downlink and 22 Mbit/s on the uplink with the multiple input, multiple output (MIMO) technologies. The HSPA also specifies an IP based architecture meaning that 3G/EDGE enabled device gets an IP assigned to it.

This revolutionizes the way devices communicate with one another drastically. It gives a user more choice of the type of data he/she is to receive. Voice and data streaming simultaneously down the same communication pipe means the user can have a lot more integration options. One particular example is enabling remote devices to fit into the networking infrastructure a lot easier, and in turn, allowing engineers and developers to create wide variety of different applications and systems. It turns a smartphone into a super-wireless device able to receive and send data from anywhere in the world.

Fig. 1 demonstrates the 3G infrastructure and how it spans across a scaled wide area network. A smartphone device such as a mobile phone connects to the GPRS network via a range of different sub-systems within the GPRS (General Packet Radio Service) framework. The first entry point is the Radio Network Controller (RNC) that manages the encryption and decryption of data sent to and from the smartphone device. It is also responsible for multiplexing of different users by implementation of a transport layer protocol ALCAP (Access Link Control Application Protocol). From there on, the signals are forwarded to Serving GPRS Support Node (SGSN). The responsibility of SGSN lies in the delivery of data packets from and to the mobile-base stations that are within its geographical service area. This includes packet routing and transfer. The next interface, i.e. connected to SGSN, is the Gateway GPRS Support Node (GGSN), which is the main component of the GPRS Network. The GGSN’s responsibility lies in ability to internetwork the GPRS network and external packet switching network like the Internet or the X.25. The X.25 protocol is amongst the oldest packet-switched services available today. The protocol standard consists of nodes that handle leased lines and old telephone service connections as well as the ISDN (Integrated Service Digital Network).

All the communication that occurs on the GPRS Core Network is seamless to users, and from user’s point of view - provided that they are in the range of the towers, the internet connection never drops as the packet switching and handover processes are handled by the core network components.

## II. Operating System Requirements

Windows Mobile Device Center support was crucial in the development of the HTC Windows Mobile Application. Problems were encountered when applications were deployed on Windows Vista. Occasionally the system needed to be restarted in order for the USB driver to recognize the device after device disconnection. As this was a slight annoyance, Windows 7 fixed the synchronization problems that were exhibited in Windows Vista. To the slight annoyance, in order for Windows Mobile application to deploy to an emulator, Windows Virtual PC needed to be installed to enable virtual network adapter support. This was essential for testing of application features without the need of physical device to be connected. Each time physical device is connected and disconnected, development time was lost due to synchronization and application deployment. As Windows 7 became the system of choice, minimal system requirements need to be met as shown in Table 2. As software requirements increased in the initial planning phase, certain system characteristics needed to reflect the project scope changes in order to divert the trouble of having to make system upgrades.

TABLE II
Windows OS Requirements

| Component | Requirement |
|---|---|
| Processor | Microsoft states that minimal processor speed for Windows 7 support is 1GHz or faster on a 32-bit or 64-bit processor. These requirements don’t reflect on the required application installation, thus development system was Intel Core 2 Duo 2.2GHz processor. |
| Memory | Minimum volatile memory requirements to operate Windows 7 comfortably on a 32-bit processor are 1GB, 512MB more than required by the older brother Windows Vista, although for a 64-bit processor, the recommended RAM size is 2GB. |
| Graphics | A 128 MB GPU, which supports DirectX 9 graphics with WDDM 1.0 driver support or higher. |
| Storage | 16GB minimal available hard disk space for a 32-bit processor and 20GB for a 64-bit processor is required to run Windows 7, although based on the application number required to develop Intelligent Object Framework, a 120GB hard drive was available at disposal. |

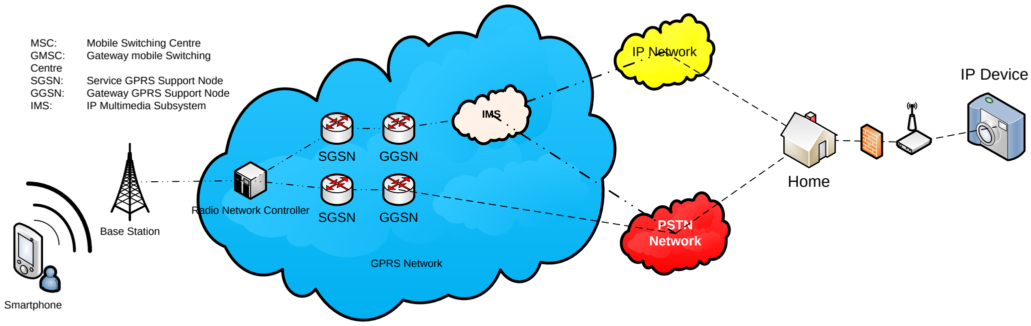

MSC: Mobile Switching Centre
GMSC: Gateway mobile Switching Centre
SGSN: Service GPRS Support Node
GGSN: Gateway GPRS Support Node
IMS: IP Multimedia Subsystem
Smartphone
Base Station
Radio Network Controller
SGSN
GGSN
SGSN
GGSN
IMS
GPRS Network
IP Network
PSTN Network
Home
IP Device

Fig. 1 3G/GPRS/EDGE network diagram [5]

## III. Software Requirements

To aid the development of the IOF, a number of software packages needed to be installed to provide the required platforms needed for development. The chosen development platforms were Microsoft products with the additional embedded system module, Tibbo [6]. The mixture of two different platforms and technologies paves the way for the IOF to prove its hypothesis on control of multiple platform independent devices by application of the framework itself.

### A. *Microsoft Visual Studio*

Being the integrated development environment of choice for the development of Intelligent Object Framework was not considered by chance. Certain support aspects needed to be taken into consideration. Microsoft Visual Studio 2010 was the first choice of development environment, but was abolished due to lack of current support for Windows Mobile development. Visual Studio 2008 had this covered. As the IOF consists of several underlying components, each using a range of different libraries from the .NET Framework's BCL (Base Class Library) provided by Visual Studio. Visual Studio provides a range of products such as Microsoft Visual Studio C++, as Microsoft's implementation of C and C++ compiler and Microsoft Visual C# targeting .NET Framework that is used as a primary development tool for the development of Intelligent Object Framework's Management Studio and the internal components; the development of an Intelligent Object Mobile Device that use Windows Mobile SDK explained in the next section. Table 3 illustrates the minimum system requirements needed for successful installation and operation of Microsoft Visual Studio.

TABLE III
Microsoft Visual Studio System Requirements

| Component | Requirement |
|---|---|
| Operating System | Microsoft Windows 7, Microsoft Windows Vista, Microsoft Windows Server 2003 R2, Microsoft Windows XP SP2 |
| Processor | Minimum CPU requirements is 1.6GHz, although it is favorable a higher processor to cope with the IDE. We can see the discrepancy between system requirements for Windows 7 alone, but as we progress in installation of packages, the system requirements increase. Recommended 2.2 GHz or higher. |
| Memory | Minimum requirements are 384 MB Ram, although recommended is 1GB of RAM. |
| Graphics | A 128 MB GPU, which supports DirectX 9 graphics with WDDM 1.0 driver support or higher, with 1024x768 display. |
| Storage | Minimum 2.2 GB of hard-drive space and this may increase as new packages are being installed. |

### B. *Windows Mobile SDK*

Windows Mobile is a development platform for mobile devices based on the Windows CE (Embedded Compact) that is used in a currently wide variety of Windows phones. Choice of development framework ranges from native C++ code, to managed C# and VB.NET incorporating XML and Web Services. As HTC mobile, running Windows Mobile 6.0 is chosen as one of primary Intelligent Objects, Windows Mobile SDK is a dependency in the project, thus it needed to be installed on the development machine. The Compact Framework provides synonymous benefits as the .NET Framework, although it is designed specifically for resource-constrained devices, such as Personal Devices and Mobile Phones. High level figure of .NET Compact Framework which is incorporated as an extension to Microsoft's .NET Framework and included in Windows Mobile SDK as shown in Fig. 2, illustrating the architecture.

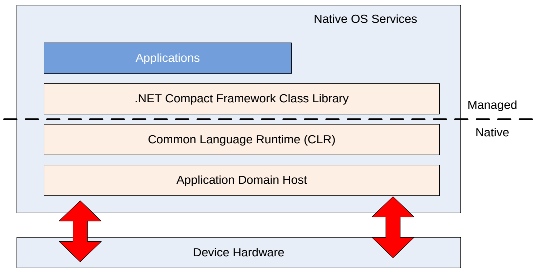


Fig. 2 High-level architecture of Compact Framework [7]

Being a subset of .NET Framework, it consists of base class libraries as well as additional libraries that are specific to mobile device development. Coupled with .NET Framework, Compact Framework provides common features access such as networking support, particularly access to TCP/UDP networking stack that is be used as main communication medium for Intelligent Objects.

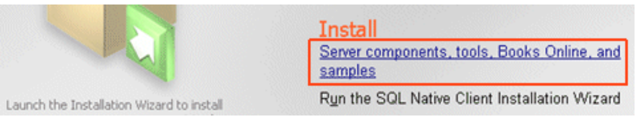
## C. Microsoft SQL Server

Adoption of a relational database model is imminent in successful design and deployment of Intelligent Object Framework. As host of Microsoft product follows, Microsoft's SQL Server 2005 is chosen as a database server. The reason that IOF database model fits in the design of the IOF is thoroughly explained [8, 9]. Table 4 outlines the minimum and recommended system requirements for installation of MSSQL 2005.

Installation of SQL Server can somewhat be difficult to a general user so when installing, certain requirements need to be addressed. For most installations, if system of choice is Windows 7 on a 32-bit processor, user needs to install x86-based option. When in the first installation screen as shown in Fig. 3 once a CD has been inserted or an exe installation file executed. User must make sure that he/she is the administrator of the machine before attempting to install any enterprise software package as certain registry entries must be written. If user does not have the right privileges to the system, the installation will fail.

TABLE IV
MICROSOFT SQL SERVER SYSTEM REQUIREMENTS

| Component | Requirement |
|---|---|
| Operating System | Microsoft Windows 2000 Server SP4 or later, Windows Server 2003 SE, EE, DE with SP1 or later, Windows Server 2008, Windows XP, Windows Vista, Windows 7. |
| Processor | Minimum 600 MHz, although 1GHz or higher is recommended. |
| Memory | Minimum 512 MB, although 1 GB or more is recommended. |
| Storage | Approximately 350 MB of available HDD space, and additional 450 MB HDD space for other tools and sample databases. |


Launch the Installation Wizard to install

Install

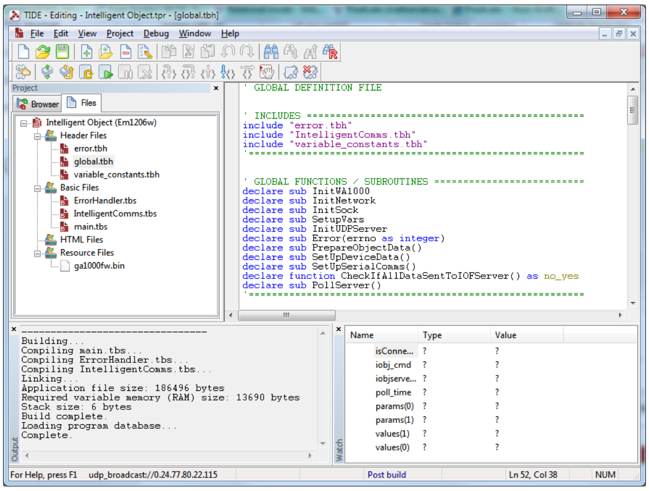
Server components, tools, Books Online, and samples

Run the SQL Native Client Installation Wizard

Fig. 3 Installing Server Components

## D. Tibbo Integrated Development Environment (TIDE)

The IOF requirement specifies that a successful integration between two devices must lie in the platform difference. Intelligent Object Management component of the system must exhibit the ability to control at least two devices. Second Intelligent Object Device is an embedded module EM1206. TIDE is a development environment of Tibbo Technologies that allows EM1206 applications to be written, compiled and uploaded to the target platform; the EM1206. TIDE has the ability to connect to the target platform, monitor and control it. This process is called cross-debugging. Fig. 4 shows TIDE environment, which is necessary in a successful development of a secondary Intelligent Object (device).



```
' GLOBAL DEFINITION FILE


' INCLUDES =============================================================
include "error.tbh"
include "IntelligentComms.tbh"
include "variable_constants.tbh"
'======================================================================


' GLOBAL FUNCTIONS / SUBROUTINES ==========================================
declare sub InitWA1000
declare sub InitNetwork
declare sub InitSock
declare sub SetupVars
declare sub InitUDPServer
declare sub Error(errno as integer)
declare sub PrepareObjectData()
declare sub SetUpDeviceData()
declare sub SetUpSerialComms()
declare function CheckIfAllDataSentToIOFServer() as no_yes
declare sub PollServer()
'======================================================================
```

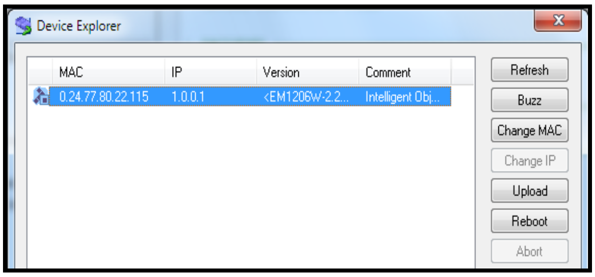
```
------------------------------------
Building...
Compiling main.tbs...
Compiling ErrorHandler.tbs...
Compiling IntelligentComms.tbs...
Linking...
Application file size: 186496 bytes
Required variable memory (RAM) size: 13690 bytes
Stack size: 6 bytes
Build complete.
Loading program database...
Complete.
```

| Name | Type | Value |
|---|---|---|
| isConne... | ? | ? |
| iobj_cmd | ? | ? |
| iobjserve... | ? | ? |
| poll_time | ? | ? |
| params(0) | ? | ? |
| params(1) | ? | ? |
| values(1) | ? | ? |
| values(0) | ? | ? |

For Help, press F1 udp_broadcast://0.24.77.80.22.115 Post build Ln 52, Col 38 NUM

Fig. 4 Tibbo Integrated Development Environment showing different views

TIDE environment must initially be set up to support EM1206 embedded device. In first step, the latest firmware for EM1206 needs to be downloaded from Tibbo website. Once TIDE has been installed, select Project and click on Select/Manage Target. Provided the device has been connected by the Ethernet cable, it shows up on the Device Explorer. Select Upload and find the newly downloaded firmware file as shown in Fig. 5.

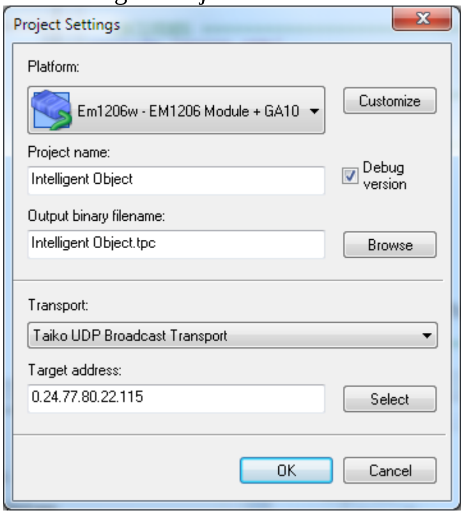


Device Explorer

| MAC | IP | Version | Comment |
|---|---|---|---|
| 0.24.77.80.22.115 | 1.0.0.1 | <EM1206W-2.2... | Intelligent Obj... |

Fig.5. Device Management

Once the firmware has been uploaded, go to Project and select Project Settings. Correct platform needs to be selected to properly configure the development environment. Below Fig. 6 illustrates the correct settings for Intelligent Object device.



Project Settings

Platform: Em1206w - EM1206 Module + GA10

Project name: Intelligent Object

☑ Debug version

Output binary filename: Intelligent Object.tpc

Transport: Taiko UDP Broadcast Transport

Target address: 0.24.77.80.22.115

Fig. 6. Project Settings

Platform EM1206 with an additional GA1000 setting is selected, a project name is specified and transport set to UDP Broadcast Transport. Target address, being the device connected to the Ethernet module is selected. This concludes the final settings and the user is ready to begin developing in TIDE environment for EM1206 embedded device.

## IV. Intelligent Object Platform Requirements

This section thoroughly describes the platform requirements for each Intelligent Object (device). Chosen platforms are HTC Mobile running Microsoft Windows Mobile Operating System and Tibbo Embedded Device – EM1206 with the additional GA1000 Wireless Embedded Module.

### *A. Microsoft Windows Mobile on HTC*

To successfully complete the Intelligent Object Framework proposition, one of the crucial hardware components is Microsoft Windows enabled smartphone. Another platform could have been chosen such as the Android operating system on a HTC device or equivalent, but since Microsoft products were at disposal, it is the obvious choice. Microsoft Windows Mobile 6 is powered by Windows CE operating system which is primarily developed for small handheld embedded devices. It is a distinctly different operating system and the kernel as opposed to the desktop Windows. Windows CE is classified a Real-Time Operating System (RTOS) for real-time applications. The Windows CE ROTS are either event-driven or time-shared. In most cases event driven RTOS switches between tasks only when an event of a higher priority needs execution. This is classified as preemptive priority, as opposed to time-share which switches tasks and processes on regular interrupts. Fig. 7 shows Windows CE Architecture.

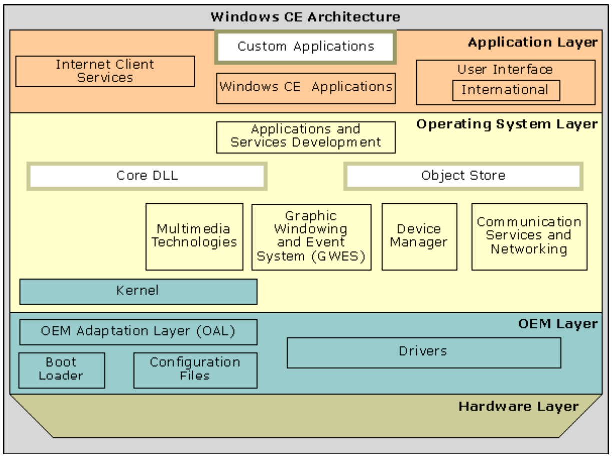


Fig. 7. Windows CE Architecture [7]

The diagram shows that over the hardware layer, board support package (BSP) lies that includes OEM adaptation layer, boot loader with configuration files and device drivers. This software is specific to a hardware board. Moving up the layer, the kernel sits directly on top of the OEM layer and it acts as a connector of application software and the hardware (through the device drivers). The kernel is the core component of the Operating System and is surrounded by a number of libraries and core dynamic link library (DLL) components that offer accessibility and interaction from the user. The Core DLL libraries are built on kernel calls and most of their functionality is based around calls to other modules of the operating system such as the communication and networking services, graphics and events and device management. Further up the layer are the Windows CE applications which are native to the environment. This is also where the custom applications are built. Part of the IOF is the Object (device) itself, thus the focus of the project lies in custom application layer. The Intelligent Object device application is built using Microsoft SDK tools mentioned in the earlier section.

### *B. Tibbo Operating System (TiOS) on EM1206*

Tibbo Operating System is a lightweight streamlined operating system that runs fast enough on a mid-range microprocessor in tiny memory space [6]. It features a dual-process system. Master Process, which is in charge of communication and event generation, and second process is the Virtual Machine

(VM). Fig. 8 shows the block diagram of Tibbo Operating System structure.

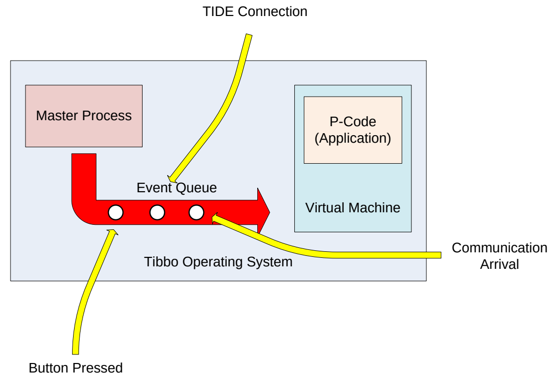


Fig. 8. Tibbo Operating System block diagram [6]

The Virtual Machine (VM) is in charge of executing the application. The instructions that are fed to the VM are defined as P-Code (pseudo-code) and are what the Tibbo compiler produces. It is not native code so the primary objective of VM is to understand and execute P-Code.

### C. *GA1000 Wireless Embedded Module*

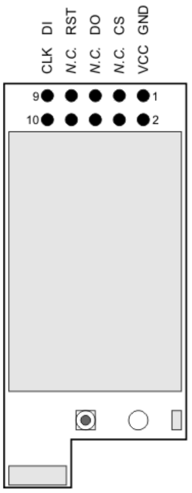
The GA1000 embedded module is an add-on to the EM1206 embedded device. It implements 802.11b/g Wi-Fi standard and it further expands the scope of potential applications on the EM1206 platform. The GA1000 utilizes the serial-port-interface (SPI) occupying five I/O control lines. Table 5 describes the I/O pin layout which requires to be mapped inside the application. The initialization code illustrates the mapping to the device I/O datelines. By following the implementation, described below, the developer can gain substantial access to the device's functionality with **wln** object, which one of the core-components of Tibbo Operating System. The **wln** object allows the user to initialize W-LAN sessions and connect to networks and in-turn communicates with other devices.

Furthermore, Table 6 describes the application configuration necessary for correct mapping to IO lines. The initialization procedure is amongst the first functions that run on boot time of the application, right after the TiOS operating system environment is set up. The **io.state** reset line must first be pulled to low, followed by a signal **HIGH.** This is because the Wi-Fi module requires a hardware reset for correct operation, hence why it is dedicated a hard reset pin. The main lines that map back to the main processor serial port interface (**SPI**) are **CLK** (clock signal), **DO** (data output), **CS** (chip select), **DI** (data input) and **RS** (reset). The **WLN_***serial port interface*, are specifically mapped addresses to General Purpose IO Lines (GPIO). The WLN (Wireless LAN Interface) structure consists of constant address variables that map back to the GPIO. **WLN_RST** (reset) maps to line 11 (Pin P1.3), **WLN_CS** (chip select) maps to line 15 (Pin P1.7), **WLN_DI** (data in) maps to line 12 (Pin P1.4), **WLN_DO** (data out) maps to line 15 (Pin P1.5), **WLN_CLK** (clock signal) maps to line 14 (Pin P1.6).

The wireless embedded module serves its purpose as the primary communication medium. Once added and programmatically configured, the Intelligent Object's primary purpose is to send and process received data from the IOMS [8, 9]. The reciprocal process is further controlled by the TiOS.

TABLE V

I/O PIN ASSIGNMENT [8]



GA1000 Module

| Pin # | Function | Description |
|---|---|---|
| 1 | GND | System ground |
| 2 | VCC | Positive power input, 3.3V |
| 3 | CS | Chip select |
| 4 | N.C | No connection |
| 5 | DO | SPI port (Data Out) |
| 6 | N.C. | No Connection |
| 7 | RST | Reset |
| 8 | N.C | No Connection |
| 9 | DI | SPI port, data in |
| 10 | CLK | SPI port, clock |

## V. CONCLUSIONS

In this paper, system development requirements of the IOF are described. First wireless technology and wireless standards are reviewed. The wireless standards cover IEEE 802.11 Network standards and Telstra 3G & High Speed packet access (HSPA) standard. Then the required software packages for Implementation of the IOF are detailed. The packages include Windows Mobile SDK, Microsoft SQL Server, Tibbo Integrated Development Environment and Microsoft Visual Studio which enable flawless communication and message exchange between different system components of the IOF. The integration of an embedded system platform and a smart-device platform makes the IOF achievable.

TABLE VI
VISUAL STUDIO SYSTEM REQUIREMENTS [9]

```
sub InitWA1000()
          ' Allocate memory ==================
          wln.buffrq(6)
          sys.buffalloc
          '====================================
          ' Reset Line ========================
          io.num=WLN_RST
          io.enabled=YES
          io.state=LOW
          io.state=HIGH
          '====================================
          ' Map Interface Lines ==============
          wln.csmap=WLN_CS
          io.num=WLN_CS
          io.enabled=YES
          wln.clkmap=WLN_CLK
          io.num=WLN_CLK
          io.enabled=YES
          wln.dimap=WLN_DI
          wln.domap=WLN_DO
          io.num=WLN_DO
          io.enabled=YES
          ' ==================================
          ' Load GA1000 Firmware
          romfile.open("ga1000fw.bin")
          if wln.boot(romfile.offset)=NG then
                    Error(1)
          end if
          isProcOk=YES
end sub
```

## ACKNOWLEDGMENT

Sasa Savic would like to thank Victoria University for the Research Training Scheme (RTS) for his PhD study.

## REFERENCES

[1] Alliance, Z. (2006). "Zigbee Vision for the Home: Zigbee Wireless Home Automation", Zigbee Whitepaper.

[2] Alliance, Z. (2007). "Zigbee Enables Smart Buildings of the Future Today", Zigbee Whitepaper.

[3] IEEE. (2009), "IEEE 802.11", http://www.ieee802.org/11/, Viewed on 9 November 2013.

[4] IEEE (2010), " Official IEEE 802.11 Working Group Project Timelines", Viewed on 22 December 2013. http://www.ieee802.org/11/Reports/802.11_Timelines.htm

[5] Ericsson. (2008) "World first in delivering innovative 3G technology to Telstra.", Retrieved on 12 December 2013. http://www.ericsson.com/thecompany/press/releases/2008/06/1226569,

[6] Tibbo. (2009). "Tibbo Operating System", http://docs.tibbo.com/taiko/index.html?syscomponents.htm , Retrieved on 15 February 2013.

[7] Microsoft (2010), "Benefits of .NET Compact Framework." http://msdn.microsoft.com/en-us/library/aa446549(v=MSDN.10).aspx, Viewed on 9 November 2013.

[8] Savic S. and Shi, H. (2011), "An Intelligent Object Framework for Smart Living", *Procedia Computer Science* 5 (2011) 386–393.

[9] Savic, S. (2010) " Intelligent Object Framework", thesis for Master of Science in Computer Science, School of Engineering and Science, Victoria University, June 2010, 138 pages.