# Exploration of Game Consoles as a legitimate computing platform for in-the-field biomedical data acquisition and management

Christopher Armstrong[1], Diarmuid Kavanagh[2], Sara Lal[3], Peter Rossiter[4]
[1]*University of Technology, Dept. Medical and Molecular Biosciences Sydney, Australia*
[2]*University of Technology, Dept. Medical and Molecular Biosciences Sydney, Australia*
[3]*University of Technology Dept. Medical and Molecular Biosciences Sydney, Australia*
[4] *Forge Group, Sydney, Australia*

*Abstract*— **Biomedical research increasingly requires for testings be conducted outside the lab, in the field such as the participant's home or work environment. This type of research requires semi-autonomous computer systems that collect such data and send it back to the lab for processing and dissemination.**

**A key aspect of this type of research is the selection of the required software and hardware components. These systems need to be reliable, allow considerable customizability and be readily accessible but also able to be locked down.**

**In this paper we report a set of requirements for the hardware and software for such a system. We then utilise these requirements to evaluate the use of game consoles as a hardware platform in comparison to other hardware choices.**

*Index Terms*— **biomedical data acquisition, driver risk factors, game console platform, home gateway, remote-home technology, smart data, wireless technology**

## I. INTRODUCTION

Through a number of ongoing projects and one in particular associated with the SmartData project (funded under ARC Linkage grant, Australia) where the broad aim is to study the effects of environmental and human factors related risk factors in people whose work place requires good concentration for long periods of time, for example, in professional truck drivers. The purpose of the SmartData project for instance is to build hardware and software infrastructure that will allow profiles of drivers to be built through the continuous non-intrusive collection of physiological and psychological data in real time. The aim is to close the gap that remains in the study of physiological and psychological factors and fatigue. The effects of fatigue on drivers has been examined extensively through laboratory based research ([1]; [2]), but important road environment field research still remains to be done and has so far proven difficult to undertake due to the lack of appropriate data collection technology.

While many of the effects of fatigue can be studied in the laboratory, there are limits on the ability to collect realistic ecologically data and the acceptability of laboratory results for drawing conclusions about real on-road driver risk factors needs to be verified by more field research. This is not the only instance of the need to verify laboratory based studies by equal field based research. Much human factors research is waiting for a practical solution towards better instrumentation and technology to enable important field research. A possible solution starts with a practical technology for collecting data about participants outside the lab, preferably from their own home, places of work or even vehicles in the case of truck drivers. If we take truck drivers as an example, such technology needs to enable the capture of factors that may influence on-road driving risk that are present prior to driving long distances, factors in the home or in truck depots. Robust and reliable technology for continuous real time monitoring of physiological and psychological factors in the field would provide more ecologically valid data sets reflective of the true physiology and psychology of the sample group. In the case of truck drivers the ability to acquire and measure important data prior to long driving episodes can take current research to new areas of exploration.

It turns out that the requirement for a low cost reliable in-the-field biomedical data acquisition and management technology is a recurring theme. There may be a possible extension of our research into the investigation of home based care applications for the aged through constant round the clock non-intrusive physiological and psychological monitoring. Hence there is a need to find a cost effective hardware and software combination that could stand up to the rigor of field deployment and at the same time provide a reliable and robust platform on which to deploy and support ongoing scientific investigations.

There are many examples in the literature over the last

Manuscript received 15/10/09. This work was supported in part by the Australian Research Council Linkage grant (LP0562407).

10 years of investigations in the field of telemedicine and telecare and all require an underlying hardware and software data acquisition combination for biomedical data acquisition. Most usually involve the use of a PC or laptop as these are common and easy to obtain, however as we will illustrate they do come with limitations and still leave many questions of robust deployment and management unanswered. Indeed this paper suggests that one of the issues that has hampered the emergence from the laboratory to the home of many of the ideas and innovations investigated in the field of telemedicine and telecare is the lack of a cheap and reliable instrument for performing numerous and varied field investigations particularly in the home.

The home is a very hostile and unpredictable environment in comparison to the laboratory and it is difficult to manage successful deployment of standard PC based solutions because they are easily accessible and vulnerable to modification and re-configuration. The standard PC and laptop with a well known operating systems invites interference by both expert and novice.

We are unaware of any suitable devices available in the market that could be customised to such research needs. Therefore, this paper describes the development of a 'gateway' device, which may be used as a hub to download data from medical monitoring devices, gather information from psychological surveys and schedule physiological tests, all in the home environment. With the ability to monitor the device remotely.

This paper considers game consoles (such as the Sony PlayStation 2 and 3, Microsoft Xbox and Microsoft XBox 360) as a hardware platform for the gateway device. We use the term ‘gateway device’ to indicate the use of a dedicated computer that is placed in a person’s home, and is used to coordinate a number of other digital and electronic devices that are connected to it. As this is the start of a flexible biomedical data acquisition and management technology.

The programming language Java with an OSGi framework was chosen as our software platform because of its suitability for residential gateway devices and relative portability and security ([3]; [4]; [5]). This paper will further proceed to describe experiments undertaken in order to satisfy the base requirements with two game consoles, (i) the Sony PlayStation-2 (Sony Computer Entertainment, PLAYSTATION 2, Australia) and (ii) Sony PlayStation-3 (Sony Computer Entertainment, PLAYSTATION 3, Australia) and the relative strengths and weaknesses of each console.

Much of our previous work and our future investigations are focusing on the popular notion of the smart home and [6] [7] and there are many reasons for developing a cheap, reliable and resilient hardware and software combinations that can be deployed to the field and in particular the home environment.

## II. Requirements

The SmartData project aims to collect data with relatively simple, off-the-shelf electronic devices that monitor heart rate, physical activity, etc., and examine this in the context of self- reported results from participant's response to psychological surveys (such anxiety, emotion, mood etc.) and certain physiological tests (reaction time, ECG etc.). However, the data from such devices must be constantly downloaded onto a desktop computer, and the effort required to manually collate and analyse this data using conventional methods such as paper-based forms makes such research substantially time-consuming and expensive. Furthermore, once a research program has been started, it can be very difficult to change aspects of it during its progression, especially if the participants are geographically displaced from the main research site for long periods of time, as would be in the case of truck drivers.

It is apparent that a device is required that could be placed in a participant's home to act as a hub or gateway to collect data and feed it back to the researchers in a central location. This “gateway device” would allow the participants to provide response to electronic surveys. In order to monitor participants’ progress without physical access to the gateway device for long periods (sometimes up to a few weeks), the device would need to be controlled remotely, able to download updates, as well as subsequently upload research data after the research program commences.

It was also a requirement that the device be readily available and easily obtainable. Even better if the device was a commodity while not an immediate or interesting target for hackers or denial of service attacks. It has been noted in the literature as devices for collecting important physiological and psychological data become more prevalent and critical to quality of life then the consequences of connectivity to the Internet and the inherent dangers of hacking and denial of service increase [8].

It was also a requirement that the device should be able to work with a minimal of support equipment. That is should be self contained and able to work to a standard TV. On the basis that all homes are likely to have a TV set. With the ambition of a device that can be readily deployed worldwide to all field situations, so no legitimate scientific investigation is hampered by lack of electronic resources and does not place a burden on potential participants and scientific endeavour to source additional monitors or anything but the standard power supply of the local.

The use of a PC as a solution was ruled out because it is too hard to control the operating system versioning, software updates and numerous other malicious pieces of software which can find their way on to PC once connected to the Internet. As it was a requirement to connect to the Internet standard operating systems like Microsoft XP or Vista are too hard to maintain in a know configuration. Often Microsoft itself will install updates and changes in the background. It is also too easy for participants and others to interfere with these common operating systems. A piece of instrumentation for scientific investigation needs to be as secure as possible and in a known state of configuration to ensure reliable measurement.

It therefore logical to look for a more resilient operating system and a less well know platform that was less inviting and less well known and therefore through obscurity protected from many of the configuration management issues encountered with PC based hardware.

### *A. Hardware*

The above requirements provides a basic architecture of the proposed system, which would be similar to that in Figure 1.

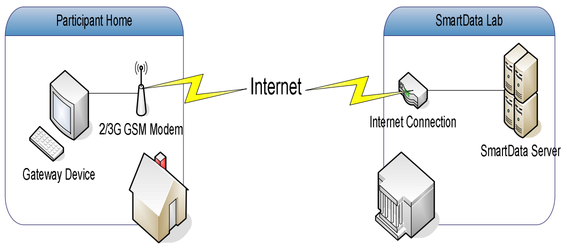

Figure 1. Architecture of SmartData system

The type of software required to collect the data from such external devices and to allow updates to program schedules, is specialized, and was developed by the SmartData project for this purpose. We wanted to reduce the time needed for software testing during development by requiring that the target device support a full Java Standard Edition runtime environment and not have the software satisfy stringent performance requirements. This means that the device cannot be a resource-constrained system and should be capable of running a 32-bit, pre-emptive, multitasking operating system with virtual memory management and memory protection. The device had to be cheap to be deployed in large numbers but customisable to permit the SmartData software to run. There was a strong desire for the device to be able to connect to participant's television sets to reduce the costs associated with purchasing separate display units. A simple way to connect the monitoring devices using a common peripheral interface such as Universal Serial Bus (USB) is required because many of these monitors need to be worn or carried by the participant whilst they are not home, such as an activity monitoring device like the Actigraph GT1M Monitor, which collects and records body movements [9]. Large data sets would be derived so the device needs secondary storage. It is also important to be able to lock down the device so that participants could not misuse the device or interfere with its normal operation.

### *B. Software Platform*

As this box will be deployed “out in the field” and has to be kept running for long periods, a software platform with high reliability that could be updated whilst the system is still running is required. Software with relative portability was also desirable in order to deploy the same or similar software on later gateway devices to be modified for deployment in trucks as well as to collect ‘real field’ driving data. We wanted flexibility to change hardware platforms and operating systems where we deemed it to be necessary. Java was chosen over languages such as C/C++ because it tends to be more portable, provides fine granularity of in-process security through the Java Permissions Application Programming Interface (API), and it's lack of direct pointer-access can help reduce common programming errors that are fatal (such as null pointer exceptions and dangling pointer dereferences) and cause security issues such as buffer overflow exploits [10].

Java served as a base for the OSGi platform, which was designed to run on residential gateway devices that have continuous uptime and need to be managed and updated remotely, without user intervention. It extends Java to provide a structure that can be used to build inter-changeable software components that are loosely coupled in the form of services, allow fine-grained security configurations, and can be updated without restarting the framework [11]. There are multiple vendors that implement the specification ([12]; [13], [14]; [15]; etc). This gives more flexibility in choosing an environment that suited our research needs and makes it easier to switch implementations later if required.

There was also the need to get Java running on the platform in order to use the PlayStation-2 as a deployment and server environment for wireless sensor motes (namely the TMote Sky and TMote Invent, Sentilla, USA). It was intended to use these motes to perform tests, such as a reaction time assessment test.

## III. Selecting Appropriate Hardware

We examined multiple computer-type devices suitable for our needs. We first looked at an emerging field of devices used such as home gateways which are Personal Video Recorders (PVRs) for watching and recording broadcasted television (TV) program and Set Top Units for watching digital TV and subscription TV services. They are mass-produced and built to interact with a TV, which makes them suitable for our application. Unfortunately, many are tied to proprietary platforms without much scope for modification, such as set top units distributed with cable-TV services, and others are not on the market for long enough so that we can make a firm decision.

PC-based systems were also considered as they are considerably cheap and have good software and peripheral support due to widespread use. Apart from being able to support a greater range of operating systems (such as Windows or Linux, etc.) there is also good support for development due to their widespread use. It has excellent backwards compatibility, and there are a wide variety of vendors that produce PC-based hardware, making it very difficult to be “locked-in” to one manufacturer. It is also possible to source hardware boards that have most peripherals integrated.

The main concern with PCs was how easily they could be used by the participants to install another operating system, whereas dedicated devices tend to require expert technical skills in order to modify them. There is also considerable variance amongst all the PC-based products in the market with most products having short lifecycles before a new variant is introduced, making it harder to predict performance if the hardware needs to be frequently changed. Using newer hardware means that we would need to thoroughly test new software drivers, which in our experience are more unstable in newer versions. This would be a more practical option in the future if we could select products with long lifecycles, and a simple method could be identified to “lock-down” the devices.

The last option being considered is game consoles. Similar to PCs, they are relatively cheap due to mass production and remain compatible with their software over the product lifecycle. Their product lifecycles are quite long, for example, the PlayStation One was available in Japan in December 1994 [16] and discontinued from March 2006 [17]. The PlayStation 2 was introduced in March 2000 [16] and continues to be available, and the Microsoft Xbox was introduced in March 2000 [18] and discontinued in 2006. They also connect easily to TV sets. Customisation is a big concern as most consoles are proprietary and like set top units are not easily modifiable. There is a risk of vendor “lock-in” because only one company or organization typically manufactures each model. The only commercially available consoles identified to allow customisation without purchasing prohibitively expensive Software Development Kits were the Sony PlayStation-2 and PlayStation-3, both of which can run Linux with fairly inexpensive modifications supported by the manufacturer. The Xbox (Microsoft Xbox) was also a consideration as it has been shown to run Linux, but it required a legally dubious and unsupported “mod-chip” in order to boot Linux and changes to its controller ports in order to access its USB hub. We ruled this option out because we wanted minimal hardware changes and less uncertain legal issues.

### A. *PlayStation 2*

The PlayStation-2 (PS2) features multi-core MIPS architecture called the “Emotion Engine” (clocked at about 290MHz) with cores designed for handling image processing, video-decompression and vector calculation [19]. There are separate processors for handling graphics, audio and PlayStation-1 games. It has about 32MB of RAM, composite TV-OUT or RF-OUT, two controller ports with memory card slots, and two USB 1.1 ports at the front.

Because of these powerful hardware features, many technically advanced purchasers of the PS2 expressed an interest in running Linux on their consoles. Due to the copyright protection implemented by Sony to prevent pirated games from being executed in the console, it was not possible to simply create a bootable Linux disc, plug in a hard drive, and install Linux. The cost of licensing an official development kit for the PS2 is also prohibitive enough so that only commercial game and application developers can afford to purchase one.

Sony published a “PlayStation-2 Linux Kit” (Sony Computer Entertainment, Sony Playstation 2 Linux Kit) that was cheap enough for purchase by consumers. The kit included a hard disk, mouse and keyboard, a disc for booting GNU/Linux, another disc containing a Red-Hat based GNU/Linux distribution, a computer monitor adaptor and an expansion card containing a network card and hard disk connector for attaching the hard disk. The kit requires older hardware versions of the PS2 that contains an expansion slot for plugging in the add-on card (specifically models before the SCPH-7000), which is something that does not appear on the newer 'slim' hardware and makes it impossible to use the PS2 Linux kit [20] with newer hardware.

This kit contains versions of Linux and GNU software that is outdated by today's standards (most sourced from the time the kit was developed), running the Linux kernel 2.2 and a GNU Compiler Collection (GCC) 2.95. This made it more difficult to cross-compile and run newer software and more difficult to use some of the newer hardware devices (such as the Actigraph) that we intended to use. As the PlayStation-2 is a specialised hardware platform with a unique variant of MIPS architecture and peripherals, the official Linux and GNU maintainers do not provide support.

Most of the software required for the research was compiled directly on the PS2 hardware or by using a cross-compiler that was obtained from the PlayStation-2 Linux Community website [20].

The PS2's CPU architecture lacks two useful instructions, 'll' and 'sc', which are used together to implement atomic swap-and-load instructions for simple synchronisation primitives. These instructions are used extensively in some open source Java virtual machines for synchronisation between threads instead of kernel based locks. For synchronisation that only needs to last for a few CPU cycles (such as incrementing an integer value), this type of locking is more efficient as it avoids the overhead of a system call and context switch into supervisor mode. It was determined not worthwhile to modify the virtual machines and adapt them for system-call based locking because these user-space locks were found to be used extensively in their source code.  Furthermore, it would seriously degrade performance by adding many more system calls for simple synchronisation needs. The kaffe open-source Java virtual machine [21] was the only one found to have been specifically made compatible for the

PS2, whilst others such as jamvm and cacaovm still relied on these specific locking instructions found in normal MIPS architectures.

Another major problem identified was with software performance. Most of the Java virtual machines trialed did not support Just-In-Time compilation for the MIPS architecture, or if they did (in the case of kaffe), they did not have the support for the PlayStation-2's unique architecture. This meant we had to turn on the slower, C-based interpreter for all the virtual machines trialed. This meant that the performance was unacceptably slow and unable to even start up the OSGi framework due to the small amount of RAM and slow execution speed.

Assessing other virtual machines such as jamvm yielded mixed results. The compiler tool-chain being used was too old (GCC 3.x) to support newer code. We did consider the option of modifying the source code of these programs so that they could be compiled with the older GCC however this was beyond the team's level of expertise.

One of our more significant reasons for not continuing with the PS2 was the lack of available hardware. The Linux Kit is only available for some regions now (having sold out in the United States and other places) and no longer includes the add-on card needed for network access and to plug in the hard disk; it only has the DVDs and VGA cable. Because it requires the older, larger PS2 and the discontinued expansion port, both these parts need to be carefully located and purchased second hand. This means that there is no guarantee that those parts will be readily available to carry out the proposed research. Even if the slim PS2 models with the cut-down Linux Kit could be used, there is no room or place to connect and mount a hard disk inside the unit.

The main support for the kit was obtained from the PlayStation-2 Linux Community Web Site [20]. Amateur software developers managed to port the PlayStation-2 specific changes to a GCC 3.3 and a Linux 2.4 kernel and published their results on this website, but due to a lack of developer resources, they have not been able to port the patch sets to more recent GCC and kernel versions (Linux 2.6 and GCC 4.x). Additional updates to the kit from Sony for newer software versions on the PS2 could not be found.

The official development kit was also not a viable option, not only because of the cost alone, but also because an agreement would need to be executed with Sony who would have to approve the software we wanted to produce and force us to print our own discs. The costs involved with the option were considered too impractical for our project and so we decided not to pursue it.

### *B. PlayStation 3*

The PlayStation 3 (PS3) has a multi-core PowerPC based Cell architecture designed for intensive multiprogramming and graphics/vector calculation. It was built for complex three dimensional computer games that need to perform intensive physics and graphics calculations. It also has 256MB of general purpose RAM and a further 256MB of graphics RAM. Unlike the PS2, the PlayStation-3 comes with a built-in hard disk drive, Bluetooth, Bluetooth technology, wireless controllers, USB 2.0 ports and 802.11b/g Wi-Fi Wireless Networking adaptor [22].

The graphics output on the PS3 is also more flexible. The PlayStation-2 required a PAL (Phase Alternating Line) or NTSC (National Television System Committee) compatible TV set or it was difficult to find computer monitors that supported required "sync-on-green" [20], but the PS3 has HDMI (High Definition Multimedia Interface) and composite outputs. This allowed the connection of a digital computer monitor via an HDMI-to-DVI adaptor as well as a High-Definition or normal analogue television (PAL and NTSC).

GNU/Linux and other operating systems are supported natively on the PlayStation-3 via the "OtherOS" facility in the System Dashboard. This is set up by loading a bootloader image file from a CD, DVD or USB flash drive with the PS3 Dashboard, which copies it into internal flash memory and uses it to boot-strap the other operating system.

Similar to the PS2 RTE, Sony has implemented protection of the graphics hardware through a "hypervisor", which sits between the PlayStation-3 hardware and the OtherOS. It still allows access to the processor cores and main RAM, but only provides a framebuffer for graphics output, not full graphics acceleration as it does for games [22].

Running Java virtual machines on the PlayStation-3 has been much more successful. IBM produces a pre-compiled version of a Java 1.5 and 1.6 virtual machine for its PowerPC achitecture that runs directly on the PS3 ([23]), and jamvm is also available. We had no problems using them to start up an OSGi environment with the Knopflerfish and Equinox OSGi implementations.

We were able to download and install most of the software we required using pre-compiled packages for the distribution that we were using (most packages came from the PowerPC variant of each distribution). The PS3's Cell chip had enough power to compile and run the Java VM and full-profile OSGi framework too. Each distribution run a newer GCC 4.x variant so there was no significant compilation issues.

The above makes the PlayStation-3 a more attractive option, and it appears that Sony supports continued work on the Linux kernel and some user-space utilities for running Linux on the PS3 [22]. At this point in time, it is possible to compile unmodified versions of the latest Linux 2.6 kernel for the PlayStation-3. However, it isn't clear how long Sony will continue to support Linux on the PS3, and they could easily withdraw their support in the future. If Sony decided to do this, we would have to reconsider the other hardware platforms for our gateway device, such as PCs.

Initally, the PS3 was not a compelling option as it is more expensive than the PS2. Even after pricing the extra hardware and software that needed to be purchased for the PlayStation-2 to run Linux, the PS3 is still more expensive compared to the PS2. The PS3 is not expected to become much cheaper for some years. However, given the relative difficulty of finding the necessary PS2 hardware, compared with immediate retail availability of a PS3 system, the extra cost may be justified.

## IV. Network Considerations And Data Transmission

Another requirement of the technology was an ability to handle large scale deployments cost effectively. To conserve both the bandwidth and network time consumed by the sink and the processing required by the server, the clients (sinks) only connect to the server on an as needed basis, usually periodically. Once a connection is established data is transferred in both directions then the client disconnects itself from the Internet. This ensures that the resources are only used as needed.

Connecting in this manner however presents its own issues. This method is not designed for anything other than non-real time information analysis. While the sinks are receiving real time data the central server will not receive this data until the sink establishes a connection, which depending on the implementation can be days. This becomes an issue if the server requires current data from all its clients or if a critical sensor image update or software update is ready for the clients.

Messages that are required to be send across the network are buffered on both the client and the server until a communication link is established. Once a link exists the messages are transferred using a transaction mechanism to ensure the data delivered through the unreliable link. If confirmation is received data is removed from the messaging queue, else the message is retransmitted when possible.

Currently there is no mechanism to deal with the situation where the server wishes to communicate with the clients. An external notification system is required to inform the clients of a server connection request which forces the clients to establish a connection.

The implemented solution was a Short Message Service (SMS) notification system whereby the server could send out SMS as required to fixed numbers attached to the sinks. If possible the clients would establish the connection to the server, or if not, send back their own SMS regarding their status. This software solution is easy to implement utilising standard Hayes commands set [24][25], commonly referred to as attention (AT) commands, to read from the mobile device. Many potential wireless devices presently have SMS capabilities build in which makes them ideal for this situation.

## V. Data Security

Due to the sensitive nature of some information, which may be acquired through the sensor devices, such as in the case of medical monitoring [26][27] or person monitoring, the data must be kept secure on both the client and server. Using the given framework the client is fairly secure against tampering, as it does not remain on the network and even further protected against attack if Dynamic Host Configuration Protocol (DHCP) Internet Protocol (IP) allocation is used. Furthermore, data does not remain on the client and is purged as necessary limiting the damage caused by a compromised system.

The server is also designed to isolate all data from each client. The data transmitted by the client is never retransmitted on the network by the server. This is to ensure that if a client is being imitated by another machine requesting information, no sensitive information is released.

A transmission issue lies in the transmission of data between the client and server, which if intercepted may reveal private information. A reliable mechanism is utilising a Secure Sockets Layer (SSL) connection mechanism to authenticate the client and server [29]. This is possible as the server and clients possess both the memory and processing power required. Utilising a signed certificate on the server makes sure that the clients only connect to the verified server. This ensures that all sensitive data transferred from the clients remain secure and only retrievable by the server.

Login data pre-shared on the client is used to verify the identity of the client upon connection to both prevent unauthorised data being transmitted and to differentiate between clients. In the case of a client being compromised and its login information is obtained by an attacker, when detected the server can invalidate the login information and reissue the affected client a new login key via SMS.

## VI. Reliability

Since the network connections of the clients are not reliable, and in some cases could experience a large number of drop outs, such as in the case of a mesh networked contained in a moving vehicle communicating via wireless Internet capabilities across the High Speed Packet Access (HSPA) or General Packet Radio Service (GPRS) protocols, data delivery in these cases can be extremely unreliable and further care must be taken to ensure data has been transmitted when compared to a more reliable connection mechanism.

To deal with this for the purposes of delivering data to the server, the client was designed to transmit its data files and wait for acknowledgements from the server before clearing the entry from its own data storage. This ensured that even if the connection was lost during transmission, or the server did not receive the data files the client would still have a copy of the data. Though this can result in redundant transmission, such as transmission errors within large files, it ensures that data is not lost.

Another reliability concern is with the update provided from the server to the clients. Though the updates will be tested before being deployed, due to the random connectivity nature of the design, the clients cannot be assumed to be in the same state. One client may have received updates to the current revision while another may not have been able to connect thus missing many of the current updates. The client needs to be able to detect what updates are available on the server and download and install the required ones automatically and also install in the correct order.

## VII. Conclusion

We examined the requirements of a proposed system to obtain physiological and survey data from a real life environment in drivers for the purposes of understanding fatigue risk factors. Given the desire to conduct research with willing participants outside the lab, we investigated the need for a computer based device that could be placed in their homes to collect physiological and survey data and the requirements for such device to interface with their television, allow the connection of external medical monitoring devices and the secondary storage of acquired data sets. We also established the need for middleware that could support remotely deployed and managed devices and our choice of Java and OSGi to satisfy these needs.

We showed that our choice of a game console as a suitable hardware device over similar devices such as personal video recorders and set top boxes was due to their high availability and stable platform details.

The PlayStation-2 was shown as a potential option because it could run Linux and allowed external hardware devices to be attached and was relatively inexpensive. However, the difficulty in attempting to compile software and run it on this platform, the difficulty of finding the needed second hand hardware, its poor performance, and lack of support, all demonstrated that the PlayStation-2 would be an inferior choice of platform. We chose the PlayStation-3 as an alternative to PlayStation-2 because it could run Linux without having to modify the console or introduce an extra “Linux kit”. It had greater hardware capabilities that increased the flexibility of the type of software that could be run and how this could be developed.

It was noted that whilst the PS3 was more expensive than the PS2 (including the extra hardware and Linux kit) and that PS3 was not expected to drop in price considerably soon, it had greater hardware availability, which may justify the extra price paid for its use in our research. At the time of writing this paper, the PS3 is our preferred computer system for the home for remote data recording and collection. We touched briefly on some network, data transmission, security and reliability issues associated with the remote management of the devices. Which is a critical requirement to the efficient and cost effective deployment of the proposed biomedical data acquisition technology.

## Acknowledgement

The SmartData project was supported under an Australian Research Council grant (ARC Linkage grant No. LP0562407), Australia. We thank Venuganan Santhakumar for technical input.

## References

[1] Lal SKL & Craig A. 2005, ‘Reproducibility of the spectral components of the electroencephalogram during driver fatigue’, International Journal of Psychophysiology, vol 55(2), pp137-43.

[2] Ting P et al. 2008, ‘Driver fatigue and highway driving: a simulator study’, Physiology and behavior, vol 94, pp 448-453.

[3] Li X. and Zhang W. 2004, ‘The Design and Implementation of Home Network System Using OSGi Compliant Middleware’, IEEE Transactions on Consumer Electronics, vol. 50, no. 2, May 2004.

[4] Zhang, H., Wang, F. and Yunfeng, A. 2005, ‘An OSGi and agent based control system architecture for smart home’, Proceedings of the 2005 IEEE Conference on Networking, Sensing and Control, pp13-18, Beijing, China, 2005.

[5] Kirchof, M. and Linz, S. 2005, ‘Component-based development of Web-enabled eHome services’, Personal and Ubiquitous Computing, vol. 9, issue 5, September 2005.

[6] Chan, M., et al. , ‘Smart homes – Current features and future perspectives’, Maturitas, 2009

[7] Chan, M.; Esteve, D.; Escriba, C. and Campo, E., ‘A review of smarthomes – Present state and future challenges’, Computer Methods and Programs in Biomedicine 91 (2008), pp55-81, 2008

[8] Lin, C.; Young, S. and Kuo, T., ‘A remote data access architecture for home-monitoring health-care applications’, ScienceDirect Medical Engineering and Physics 29 (2007) pp199-204, 2007

[9] Actigraph 2007, Actigraph GT1M Monitor / ActiTrainer and ActiLife Lifestyle Monitor Software User Manual, Actigraph LLC, March 2007, <http://www.theactigraph.com>.

[10] Van Hoff, A. 1997, ‘The case for Java as a programming language’, IEEE Internet Computing, vol. 1, issue 1, pp 51-56, Palo Alto, CA, USA.

[11] OSGi Alliance 2009, OSGi Alliance | About / The OSGi Architecture, viewed 16 March 2009, <http://www.osgi.org/About/WhatIsOSGi>.

[12] Apache 2009, Apache Felix, viewed 4 February 2009, <http://felix.apache.org/site/index.html>.

[13] Eclipse 2009, Equinox, viewed 4 February 2009, <http://www.eclipse.org/equinox/>.

[14] Knopflerfish 2008, Knopflerfish OSGi – open source OSGi service platform, view 4 February 2009, <http://www.knopflerfish.org/>.

[15] ProSyst 2009, OSGi Framework Implementations – open source Equinox and commercial – ProSyst, viewed 4 February 2009, <http://www.prosyst.com/products/osgi_framework.html>.

[16] Sony Computer Entertainment 2009, Business Development/Japan | CORPORATE INFORMATION | Sony Computer Entertainment Inc., Sony Computer Entertainment, viewed 4 February 2009, <http://www.scei.co.jp/corporate/data/bizdatajpn_e.html>.

[17] Sinclair 2006, ‘Sony stops making original PS’, Gamespot, viewed 4 February 2009, <http://au.gamespot.com/pages/news/story.php?sid=6146549>.

[18] Microsoft 2000, Xbox Brings “Future-Generation” Games to Life, Microsoft Corporation, viewed 4 February 2009, <http://www.microsoft.com/presspass/features/2000/03-10xbox.mspx>.

[19] Sony 2001, EE Overview, Sony Computer Entertainment Inc., version 5.0, published October 2001, Tokyo, Japan.

[20] Playstation 2 Linux Community 2007, Linux for PlayStation 2 Community: Linux for Playstation 2 FAQs, viewed 4 February 2009, <http://playstation2-linux.com/faq.php>.

[21] Kaffe 2009, Kaffe.org, viewed 4 February 2009, <http://www.kaffe.org/>.

[22] Sony Computer Entertainment 2 2008, Linux Kernel Overview, viewed 19 March 2009, <http://www.kernel.org/pub/linux/kernel/people/geoff/cell/ps3-linux-docs/ps3-linux-docs-08.06.09/LinuxKernelOverview.html>.

[23] IBM 2004, IBM developer kits for Java technology on Apple PowerPC hardware, viewed 19 March 2009, <http://www.ibm.com/developerworks/systems/library/es-apple.html>.

[24] HarmoniousTech Limited, Developershome. *SMS Tutorial: How to Send SMS Messages from a Computer / PC? AT Commands*, [Online] HarmoniousTech Limited, 2008 [Cited: March 25, 2009] <http://www.developershome.com/sms/howToSendSMSFromPC.asp>

[25] HarmoniousTech Limited, Developershome. *SMS Tutorial: How to Receive SMS Messages Using a Computer / PC?*, [Online] HarmoniousTech Limited, 2008 [Cited: March 25, 2009] <http://www.developershome.com/sms/howToSendSMSFromPC.asp>

[26] Jurik, A.D. and Weaver, A.C., "Remote Medical Monitoring", *Computer*, 2008, Vol 41, pp 96-99.

[27] Yuce, M.R., et al., "A Wireless Medical Monitoring Over a Heterogeneous Sensor Network", *Engineering in Medicine and Biology Society*, IEEE, 2007, pp. 5894-5898.

[28] Karlof, C.; Sastry, N. and Wagner, D., "TinySec: a link layer security architecture for wireless sensor networks" *Proceedings of the 2nd international conference on Embedded networked sensor systems*, ACM, 2004, pp. 162-175

[29] Diffie, W. and Hellman, M., "New directions in cryptography", *Information Theory, IEEE Transactions on*, 1976, Vol 22, Issue 6, pp. 644-654.

**Christopher Armstrong** is a student researcher on the UTS SmartData project. This project aims to provide a practical low cost reliable field deployable biomedical data acquisition and management technology. He is currently studying in software engineering at the University of Technology Sydney.

**Diarmuid Kavanagh** (PhD Candidate UTS, BE Electrical, BSc and Masters of Engineering Science UNSW), Is currently one of the key researchers on the SmartData project at the University of Technology, Sydney.

**Dr Sara Lal** (PhD, MAppSc, BSc, GCHE, DipLaw), is an academic at the University of Technology, Sydney. Some of Dr Lal's areas of research are neuroscience, cardiovascular, sleep disorder, cognitive function, transportation safety, and sensor and wireless technology development. Dr Lal has attracted various national and international competitive grants and has published book, book chapters, journal and conference papers.

**Peter Rossiter** founded his first computing consultancy company in Australia in 1984 (Working Computer Systems) which in 1995 morphed into Forge Research Pty Ltd becoming one of The Forge Group of companies. Since 1983 Peter has been involved with the delivery of a wide variety of software projects in many different environments and businesses. As group Forge Chairman and CEO, Peter has overseen the launch and development of several Forge spin-off companies such as QIQ (acquired by Hyperion in 2004) and most recently Integeo.