Dynamic Migration of Computation through Virtualization of the Mobile Platform

Shivani Sud, Roy Want, Trevor Pering, Kent Lyons, Barbara Rosario, Michelle X. Gong
Future Technology Research, Intel Inc
2200 Mission College Blvd
Santa Clara, CA 95024
{shivani.a.sud, roy.want, trevor.pering, kent.lyons, barbara.rosario, michelle.x.gong}@intel.com

Abstract

Virtualization and live migration techniques have long been used in the enterprise server space and have been tuned to address data center usages. These capabilities are now expanding to personal computers including desktops and laptops and more recently into smaller mobile devices such as Netbooks and Mobile Internet Devices (MID). Hardware support for virtualization in these platforms, such as that offered by Intel® Atom™ processor, enables the use of existing operating systems and virtualization software. Our experiments demonstrate that live migration can be used to dynamically offload computation to a nearby desktop computer from a Netbook, taking only 25 seconds over a 100Mbps Ethernet network and approximately 100 seconds over an 802.11n interface with a measured throughput of 70Mbps. Additionally, these experiments highlight the limitations of existing virtualization solutions for migrating computation between small form-factor mobile devices and desktop computers which have widely varying resources and processing capabilities. Finally, we discuss the challenges observed in these early experiments and raise key questions that need to be resolved to enable the design of effective systems that support this use model.

Keywords: Virtualization, MIDs, Intel® Atom™ processor, mobile, Virtual Machine (VM), live migration

1 Introduction

In this paper, we discuss the use of virtualization on small mobile computers such as smart phones, Mobile Internet Devices (MIDs) and Netbooks which are increasingly enabling low-power high-performance computing in a small portable device form factor. As an example of such systems, we focus specifically on the Intel's Atom class of mobile processors and how virtualization can be used to migrate a virtual machine between these types of mobile devices, and other IA systems such as a laptop. The overall goal of such live migration is to allow a user to move their computation from an ultra mobile device such as a MID to a more capable machine such as a laptop or desktop when it becomes available.

Mobile Internet Devices

Users are increasingly choosing smart phones as their 24/7 connection to the world. These devices are not only primarily communication devices, but personal portals to the internet and are used to access data as frequently as they are for voice. Further, mobile devices are increasingly being used as media generation devices with users capturing videos and photos, text messaging, twittering, micro blogging etc. in addition to more traditional media consumption such as listening to MP3s or watching videos, or using the device as an e-reader. Moreover, social networking applications now allow mobile users another level of instant notification when an event occurs in their social network. Statistics indicate that the number of internet users in the developing world will increase dramatically as many users who never had access to a wired Internet connection start using their smart phones to access the internet for the first time [7]. The processing power available in smart phones is already able to host operating systems such as embedded Linux and Windows Mobile 6.0. The new low power x86 based Intel's Atom processors can host full desktop operating systems such as Windows XP or Linux desktop editions. These processors are being

Processor-> / Feature	Intel Atom Z550	ARM 11 core	Qualcomm MSM7201A RISC Chipset (based on ARM 11 core)	Texas Instruments OMAP 2430 RISC Chipset (based on ARM 11 core)	Nvidia Tegra APX 2500
Frequency	2.00 GHz	620 MHz	528MHz	450 MHz	750MHz
Cache	512K	16K	-	32K(data)/ 32K(instruction)	-
Power	2.4W	0.45 mW/MHz	-	-	2.5-4W
Instruction Set	x86	ARMv6	ARMv6	ARMv6	ARMv6
CPU Core	SoC codename Lincroft	ARM11	ARM1136EJ-S	ARM1136	ARM11MP

Figure 1: Smart phone and MID processors

used in an emerging class of devices called Netbooks and Mobile Internet Devices (MIDs). Since desktop class operating systems are now available on these small mobile devices, new usage models can be enabled through the increased available compute power in ways that enrich the mobile user experience. These trends indicate that in the future smart phones are likely to become the primary computing platforms for mobile users [1]. Research is also on-going to allow these devices to take advantage of nearby computers and wirelessly share peripherals to extend their capabilities with more peripherals, such as full-sized displays and keyboards, and components not available on the mobile platform [15].

Opportunities enabled by the Atom family of processors

Atom is a Low Power Intel Architecture based SoC that support Intel x86 ISA. The Atom Z550 processor has a clock speed of 2GHz with hyper-threading and supports up to 2GB of memory and can be packaged into a handheld device consuming 220mW average power and 100mW idle power. In addition, Moblin [6] is a Linux software platform optimized for Atom that has drawn in many manufacturers, vendors and dev elopers. With processing capabilities that far outstrip many desktop computers sold only a few years ago, these processors are likely to change the landscape of smart phones, MIDs, Netbooks and other mobile computers. We compare the various smartphone and MID processors in the market today in Figure 1.

The Atom processor has hardware support for virtualization (Intel VT extensions) and allows hosting hardware accelerated VMs that provide better performance compared to software emulation based solutions. Hardware extensions to support virtualization are a capability not seen in the handheld space before [3]. This widens the playing field by allowing readily available open source and proprietary hypervisors designed for x86 platforms to be run on devices such as Netbooks and MIDs. This capability allows for the leveraging of experience and existing knowledge base in this area.

In this paper, we demonstrate the use of this virtualization as a technology for creating a container for a set of applications (like office productivity applications in the case of a mobile enterprise user) on a mobile device. Using a high bandwidth radio, we treat the VM as a unit to migrate the computation state onto a PC for better utilization of opportunistically available resources

In Section 2 we present uses of virtualization in the mobile device space and introduce a new usage model, for dynamic computation offload based on live migration and virtualization built from off-the-shelf components. In section 3, we demonstrate how we can achieve the migration of VM state in approximately 100 seconds over existing wireless interfaces. We present results that characterize the offload time using fast local area networking and discuss their viability. As this is done using existing technologies it demonstrates the concept is viable, although technology improvements can clearly benefit the approach. In Section 4, we present the research questions brought to the fore that need to be addressed to make this usage model attractive to users through seamless and transparent operation, and

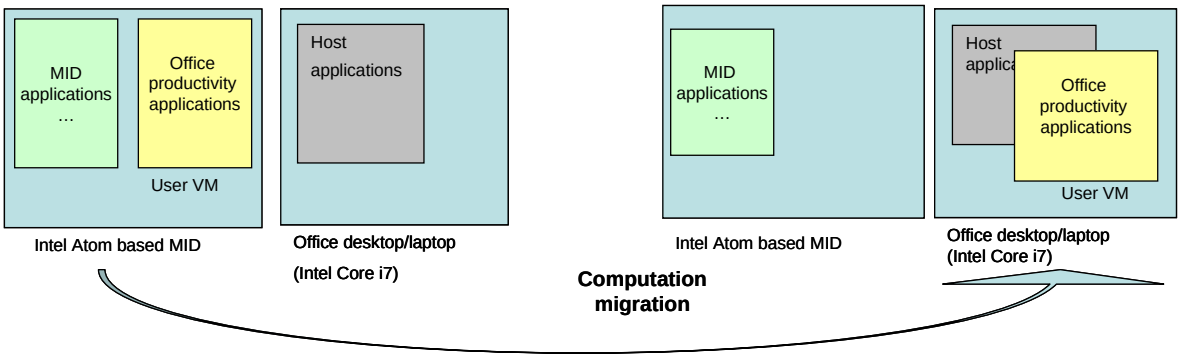

Figure 2: Virtualization and live migration provide a mechanism to migrate the dynamic computational state

compare our research with related work in Section 5. We conclude with key findings that support the processing offload vision.

2 Virtualization in the Mobile Space

Virtualization technology has long been used in the IT industry to increase reliability and availability, and optimize performance of server class machines. This technique is employed in data centers and is typically transparent to the end user, who is unaware how it is benefiting server applications. Virtualization has its roots in the mainframe era, and has since made its way to personal computers but its use is still relatively limited, and development of the broader market is still at a nascent stage.

As discussed above, the advent of MIDs that provide accelerated performance in a mobile platform, combined with the increased desire of users for full on-the-go information access, means that users can now run their favorite PC operating systems and applications on their handheld devices

Computation migration for mobile devices

We explore additional uses of virtualization in the MID space from the perspective of enhancing the user experience through better performance and interactivity with more capable processors and peripherals found on desktop machines. Our usage model is based on the dynamic computation offload from a mobile device to a PC, to take advantage of a richer computation environment. To illustrate the problems and benefits of such a solution we first describe a scenario involving a mobile enterprise user who needs constant access to office productivity tools, along with voice and data connectivity, irrespective of their location, and the optimizations that could be achieved using this approach:

> *Jane uses her home computer to check her email and reviews a presentation she needs to deliver later that morning. As the day progresses, she seamlessly migrates her work environment from her home PC to her mobile device before leaving home. While traveling she continues reviewing the presentation, adding notes as she rides the subway to work. Soon it is time for her to dial into a teleconference. On reaching her desk, her work environment seamlessly migrates from her mobile device to the office PC, and she can now use the office PC to continue reviewing the presentation, while she continues her teleconference from her mobile device.*

The processing power of a MID device is now large enough to support the full suite of Microsoft Office applications, and most of the other applications users run on their laptops or desktops. However, while their small form-factor and low power consumption make them very suitable for mobility, it also constrains long term user interaction because of the poor user interface, and the limitations of the battery. It is thus desirable for a user to leverage any additional processing resources to augment performance and overcome power constraints on the device and nearby peripherals to overcome the user interface limitations, while retaining the complete independence of functionality when environmental support is not available. Using virtualization and VM migration, we can migrate our office applications to nearby PCs and utilize their full hardware capabilities e.g. processor, display, keyboard, network as shown in Figure 2.

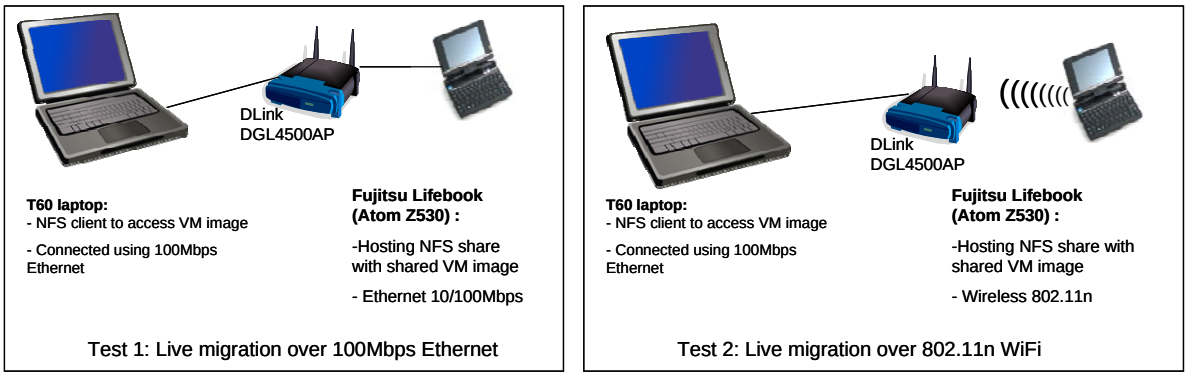

Figure 3: Two test environments used to capture live migration timings

Consider the usage scenario described in the section above, in which a seamless transition for the user can be achieved by hosting the desired office applications in another guest VM on the MID. This ensures that all applications are available to the user when (s)he is on the go. And when the user reaches a resource rich environment , such as their office, the VM hosting the office productivity applications can be migrated to a more powerful PC, while the user continues using their MID as an independent standalone device.

At a later time when the user becomes mobile again they can migrate the context back again to their MID retaining the current state of their applications and they can thus seamlessly transition computation among the most suitable computers in their environment and adapt to their situational constraints.

Live migration

Some of the technologies to enable the capability described above are currently available for experimentation. Both proprietary and open-source virtualization software are readily available for the desktop environments, such as VMWare, Xen desktop, VirtualBox and KVM. Some of these virtualization software solutions support *live VM migration* that allows a user to migrate a live VM session from one system to another.

Of the above mentioned products, KVM is an open-source kernel virtual-machine infrastructure based on hardware virtualization support available in most of today's processors (Intel's VT and AMD's SVM). KVM adds virtualization capabilities to the Linux kernel while benefiting from its memory management and scheduling algorithms, such that VMs exist as regular Linux processes. KVM leverages processor extensions for hardware virtualization and has a performance advantage over software emulation based solutions. KVM *live migration* uses an NFS mounted shared disk space to store the guest VM image which needs to be accessible from both source and target migration end-points. The KVM live migration algorithm makes use of a cyclic process of copying dirty memory pages from the source to the destination. A write access to a page that has already been copied causes it to be re-marked as dirty and it needs to be copied again in the next cycle. When the number of dirty pages falls below a certain threshold, the cycle is stopped and the rest of the dirty pages are copied to the destination; at which point the VM has been completely migrated.

3 Experiment

Wireless Docking is a term used to extend the notion of a wired docking station to wireless access to peripherals. Our usage with virtualization and live migration extends this model to the processing resources available on other computers, and we call this usage *Wireless Dock+.* So, Wireless Dock+ is the extension of the wireless docking model that allows for enhanced computation through VM migration, in addition to enabling access to functionally better peripherals on the target device.

Test Environment Setup for Wireless Dock+

As the name indicates, Wireless Dock+ should offer a cable free experience for the user. In terms of our mission for live migration this implies that when a mobile user comes into the vicinity of their office PC, VM migration can be triggered (either manually or automatically) over a short-range high-bandwidth radio channel.
To experiment with the scenario described in section 2 we employ an Atom based Netbook (Fujitsu Lifebook U820) as the mobile device. It is powered by an Intel Z530 Atom 1.6GHz processor with 512K cache and 1GB of memory. For the office computer we used a ThinkPad T60 notebook computer based on an Intel T2400 Core Duo 1.83GHz processor with 2MB cache and 2GB of memory. Both devices are connected to a DLink DGL 4500 802.11 Draft N WLAN AP. The host OS on both the T60 and Lifebook is the Ubuntu 8.10 desktop edition. The guest VM is also created using the Ubuntu 8.10 desktop edition on the Lifebook. We set up the NFS server for hosting the VM image on the Lifebook, because the mobile device should be able to operate the office applications VM independently, without requiring access to a remote server. The mobile device can then share the VM image, and support ad hoc VM migration to nearby PCs over a local wireless link. The T60 has an NFS client installed and can access the VM image from the NFS share on the Lifebook.

In our effort to use off-the-shelf and open-source components, for this experiment, we use KVM (kvm-72) as the open source hypervisor for supporting the guest VMs on both the Lifebook and T60, and makes use of Intel hardware virtualization extensions on both devices. The guest VMs are configured with a single CPU, 512 MB of memory space, and the allocated a disk image of 10GB. The network is configured with a user-space network-stack and a Cirrus graphics driver provides the graphics emulation.

We performed two tests, setups shown in Figure 3, in which we investigate the feasibility of computation handoff between a Netbook and a PC by measuring the migration time. First, we measured it with an Ethernet connection between the devices to ensure the viability of our proposal and provide a baseline. And in the second test we employ a Wifi connection. Details are presented below:

Test 1: The T60 was connected to the access point with 100Mbps Ethernet, and the Fujitsu Lifebook was also connected over a Realtek RT8139 100Mbps Ethernet interface.
Test 2: The T60 setup was the same, while the Fujitsu Lifebook connected wirelessly using the Atheros 928 chipset, providing 802.11agn Wifi with maximum theoretical physical layer data rates of 600Mpbs. There was no other traffic through the AP during the test.

For each test, we ran three scenarios:

- Basic Computation: a couple of terminals and browser windows active,
- Media Browsing: viewing a YouTube video,
- Interactive Collaboration: an openGL collaborative application.

We used the qemu[1] console window to trigger the migration process and made coarse grained measurements of the timings from when the migration was initiated to the time the VM became active at the destination computer. The measured results, summarized in Figure 4, were averaged over 3 iterations for each scenario.

App Load-> NW i/f	**Terminal and browsers windows**	**YouTube video**	**OpenGL Cobalt collaboration**
100Mpbs	20 sec	25 sec	25 sec
802.11 Draft N	110 sec	150 sec	130 sec

Figure 4: Timings of live migration from Lifebook to T60

[1] KVM has a console, which provides command line interface used to interact with the VM. This command line provides commands to set the migration speed, initiate the migration process and check the status of an active migration

Performance statistics

Our results show that for VM migration over a 100Mbps Ethernet link, typically 25 sec of migration time is needed; but with an 802.11 Draft N link, the average migration times are on the order of 2 minutes.

Using *iperf*, we determined the 802.11 Draft N throughput was about 70 Mbps. The average time for the migration was proportional to the size of the memory allocated to the VM. In the case of applications that dirty memory pages more frequently, multiple iterations are needed to transfer the dirtied pages to the destination, and add to the migration time. This explains why the YouTube test took the longest out of our three test examples. Due to the lower throughput and collision prone nature of the wireless link, the migration of same amount of data takes longer compared to an Ethernet link. This additional delay contributes to more dirty data being generated during the migration process and thus the average migration time over the 802.11 Draft N link is much longer than that over the Ethernet link. The results of our experiments show the viability and current limitations for seamless wireless *live migration* of a mobile users compute environment to infrastructural PCs (2 minutes is a long time when you are waiting).

With improvements in WLAN network technologies we can envision scenarios in which a fast WLAN or WPAN (such as UWB) can significantly reduce migration time, thus requiring less waiting on behalf of the user and therefore provide a more seamless user experience. In addition, developments in storage technologies such as Solid State Disks (SSDs) allow faster disk access than spinning media, Together these technologies provide additive performance improvements for accessing files across the local network and hence result in a better user experience.

Discussion of results

Our usage requires a high-speed short-range radio technology for live migration and dynamic computation offload; and does not depend on high-speed broadband access to the Internet or another server, so it is not affected by the latency considerations of WAN technologies. WLAN technologies are however rapidly evolving towards much higher speeds than WAN technologies and the gap continues to grow. With 802.11 Draft N interface, we saw only 70 Mbps throughput that is a fraction of the maximum theoretical physical layer data rate of 600Mbps. And with the upcoming standards for 60GHz, we can only expect the WLAN throughput rates to go higher (with a projected theoretical raw bit rate of 5 Gbps [5]). Based on our measurements and the expected new standards, we can project that a wireless migration time of around 120 seconds for 802.11 Draft N will become a couple of seconds using a 60GHz radio.

Additional optimizations for our experimental set-up can be achieved by providing faster access to the remote VM disk image on the MID using an additional radio technology such as WUSB, to augment the WLAN radio bandwidth in the migration process. Thus, we can leverage multiple high-bandwidth WLAN/WPAN technologies to achieve an optimal user experience. High speed Ethernet links (e.g. 1Gbps) are the norm on PCs nowadays however they are not typically available on MID or mobile devices, so we were not able to compare the performance over the best of class wired and wireless interfaces.

4 Mobile virtualization and migration – some research questions

Our experiments demonstrate the viability of the usage scenario proposed using existing off-the-shelf and open source components available for x86 platforms; and underlines the benefits of employing an Atom processor to support VMs and migration, on a mobile device. During the experiment described in section 3, we observed some issues that arose out of the nature of client side virtualization, and from the platform differences of mobile and desktop computers. Some of these have been identified by the research community in the recent past but there are still no off-the-shelf solutions to compare against. We summarize all our findings in the remainder of this section.

Effective platform resource utilization vs generalization

Virtualization is typically implemented by using VMs which simulate a set of generic device interfaces to the guest OS. This works well for migration when VMs are compute bound, as long as the CPU instruction set is supported and the needed optimizations to support these typical uses of VMs in data centers that provide faster access to storage and networks, are available.

In the end-user/consumer space, when using a VM to access all the resources of the host device through generic abstractions of the hardware that are presented, undermines the opportunities users might benefit from by migrating to a more powerful host. For example, consider a game that migrates between a high performance desktop computer and a mobile device. If a user has a discrete graphics processor on their device they can run Open GL based games and other applications that are tuned for enhanced graphics, but they will be deprived of the enhanced experience when running inside a VM. There are solutions available, such as VMGL [8] that counter this issue to some extent, but they are not standard and are not guaranteed to work for all applications. For a user, the use of the complete working set of applications hosted inside a VM will not be transparent if it involves the loss of performance because hardware accelerators, normally available on the base platform are no longer accessible. This holds for the myriad of special peripherals that are available on mobile devices and are an important part of the user experience, but which may not be handled by the generic VM abstractions. Some virtualization solutions improve on this by offering para-virtualized drivers that enable the host and guest VM's to cooperate and provide a better approximation of the actual hardware to the guest VM. Such drivers will need to be made available for all of the special hardware and for each host OS and guest VM that can be hosted. An important take-away is that this issue is accentuated in the mobile market because the variety of peripherals is likely to be quite different on mobile devices and PCs.

Migrating across platforms with heterogeneous resources

If we consider the usage scenario described in section 2, using live migration of the system state between a MID, and a desktop computer, the disparity in platform resources between the two becomes apparent. Live migration features have been realized for the personal computer as an evolution from the needs of servers, where VMs are migrated among servers with similar set of abstracted resources to attain highly available services in the presence of actual or anticipated failures. However in our mobile model employing out-of-the-box live migration features, the user experience will be limited to the minimum common denominator of the resources that are available among the target devices. This will limit the user experience on resource rich devices, and make it less transparent, which could deter user adoption.

In the light of our discussion in section 3 and the use of VMs for the migration of dynamic system state, interesting research questions arise. First, if and how to make it possible to provide a user with the appropriate level of abstraction of resources in the guest OS, while allowing the guest to be able to detect and dynamically adapt to resources on the target host, while maintaining the resource availability and continuity of its use after the process of migration. This would avoid the abstraction of resources down to the lowest common denominator, and instead match the capabilities of available resources on the target host. For example, it may be a scaling from a modest display resolution and screen area on a MID, to a high-definition display using a 3D accelerated graphics adapter on a desktop PC. The reverse question is equally likely, how can we migrate and adapt from a resource rich environment to a more modestly resourced environment in an effective way?

In summary, when using live migration the resulting client side experience should not deter the user by dumbing down the platform, instead they should have access to the full potential of all the hardware resources on the target. This includes both platform capabilities, and the processor performance; whether it be on a MID or a desktop computer. Live migration should have the capability to dynamically scale the user experience to take advantage of all the available resources.

One dimension of scaling of the migration process is to adapt to the available peripherals – video (display, camera), audio (speakers, Bluetooth headset), I/O (joystick, keyboard, track point, mouse),

peripheral devices connected to the platform (printer, scanner), or network connectivity (broadband, cellular, WLAN, WPAN). Some of these peripherals do not have state information that needs to be migrated (e.g. keyboard, and mouse etc.) as the human operator ensures the continuity. However, some peripherals require state changes transition across the migration process. For example, in a network interface, the connections established need to be transferred to the destination host to provide seamless operation. Also a display needs to convey the same information that the user sees on the source host as it transitions to the target host. When using para-virtualized drivers to provide a closed approximation of the hardware resources, the transmission of peripheral state during the migration process will be key to making the experience seamless for the user. There is research in the area of capability adaptation for interface virtualization [13] that discusses some of these issues.

Another dimension in scaling the migration process is adapting to new computation resources - the system memory, the number of processor cores, available, and any special purpose co-processors that are being used (e.g. TPMs). In our experiment we used an Atom processor that has 1 core and 1GB memory, where as the laptop processor T60 had 2 cores and 2GB of memory. As technology evolves the variety of options will increase for both mobile and static computers making migration more complicated, though desktop computers will typically outpace the capabilities of MIDs. When a user employs live migration to move state between a MID and desktop PC, based on today's virtualization solutions the desktop computer will be under-utilized. This observation inspires the research question: how do we dynamically scale the OS to recognize and utilize the increased processing resources (multiple cores), and memory available on the desktop PC when migrating from the mobile device. And when the migration moves state from a desktop to the MID, how does an OS recognize that it has fewer cores and less memory available and adapt accordingly. Gracefully handling these transitions at the OS level could make these transitions less visible to applications, and hence allow legacy applications to be used without modification and achieve the best possible result.

User Interface adaptation

The disparity of resources between computers is most keenly seen by changes in resources that affect the user interface. The modalities of interacting with a mobile device with significant processing power are themselves topics of active research. Even though very high-resolution displays are increasingly becoming available on the mobiles, interactions tend to be brief when compared to the use of a desktop PC. Today the interactions with mobile devices, mostly a legacy form the time when they primarily supported voice communication devices, are still evolving. Using legacy PC applications also has problems as they are honed for the keyboard-video-mouse model of interaction and it is difficult to use the same mode of interaction on a MID. There are numerous research initiatives that are addressing these issues, some employing sensor-based solutions to improve user interaction. However, this is potentially complicating VM migration as not all devices will have the same set of peripherals and sensors for interaction.

This raises the research question: how do we dynamically change the mode of interaction and user interface during migration to suit the target device, and how do we dynamically adapt the state of applications to these changing interfaces. User interface adaptation is also a hot topic in the HCI research community and also applies to scenarios such as designing web pages that display well on both mobile and desktop browsers.

Virtual Wireless I/O

One of the enabling platform features to support mobility is ubiquitous wireless access. As we described in our experiment, we used an 802.11 Draft N interface on the mobile device, and a user-space network-stack to provide network access inside the VM. The network stack limits the network performance seen by the guest, and furthermore the guest is not visible as a host by other devices on the LAN, or other guest VMs on the host, it is only visible to the hosting OS. Bridging the network interfaces makes the guest visible as a host on the LAN and lets other systems access this guest VM. This allows for constant and uninterrupted visibility (the guest maybe hosting services) as the guest VM migrates from one host to

another. Host network bridging is available only for wired network interfaces and is not available for most of today's wireless interfaces because it involves manipulation of the MAC address for the interface it bridges. This function is not currently available in the firmware API of most wireless chipsets. Another option is to use IP routing to achieve a bridging function, but this has a detrimental impact on the VM network performance. A key learning is that wireless access is a key requirement for driving mobile usage, and the lack of support for bridging between wireless interfaces is a major hurdle for effective live migration.

Security

Data security is an important requirement for most computers, and it is even more critical for mobile devices that may be exposed to a wider variety of malware while connecting to networks in un-trusted locations. Secure partitioning, or sandboxing, mechanisms, can be based on TPM to establish trust for components that provide partitioning and protection among the partitions. In that case, related research questions arise for secure migration: how do the credentials that are rooted in the trusted hardware on one platform migrate or map on to the hardware credentials of the target platform.

When a user works with a set of known host computers for wireless Dock+ there can be a shared and preconfigured set of trust credentials among the computers. But, when the user tries to use a public kiosk for the Wireless Dock+ additional security concerns will arise for protection of both the kiosk computer and the mobile device. For example, malicious software could be left on the kiosk computer by another rogue user or rogue kiosk operator. This requires establishing a trust mechanism where the source and destination computer can ensure hardened boundaries of interaction between the devices. When trust is not established, interaction can be curtailed or established in a sandbox which limits the potential damage that could result from malware.

5 Related Work

ISR [12] and SoulPad [2] are research projects that have the goal of migrating VM state to maintain the continuity of a user session between computers. ISR enables a mobile user to access their remote session stored on a network server from any other ISR instrumented computer connected to the network. This involves checking out a *parcel* from the server and then checking it back in to allow access by the next computer that needs to continue the session. High-speed connectivity to the server is required for the best operating experience. There are various optimizations available to improve this process like VM overlays [16] and Opportunistic replay [14]. SoulPad provides a passive solution for transferring VM state in a USB flash drive. State can be moved from a computer to the USB drive and then restored at another computer by physically transporting the drive and plugging it into the target machine. There are additional process migration techniques that use sandboxing for migrating groups of applications, or process domains that involve check pointing and restart of the target PC at a later time [11]. Goyal & Carter [4] also use virtualization as a means of offloading specific tasks to a server in the infrastructure for remote execution, and return the results to the source device, thus exploiting the potential performance benefit of more capable remote computers. Most other research and products around mobile virtualization solutions are proprietary in nature [9][10].

6 Conclusion

In this paper we propose the use of virtualization to enhance the utilization of processing capabilities available on MIDs, an emerging class of mobile device. Through Wireless Dock+ a computer can be configured as a communication device and at the same time serve as the mobile user's primary personal computer. Live migration techniques can be used to extend the MID using nearby high-performance and resource-rich computers, while retaining independent processing capability in their absence. Further, as our usage is based on high-throughput WLAN or WPAN radios to achieve this transition, it places no additional strain on a WAN or MAN to effect migration, and no centralized server is required for hosting the VM images. Wireless Dock+ therefore decouples WAN connectivity from the process of migration, and the migration process can be ad hoc and completely opportunistic.

As we propose using the MID as a mobile user's primary compute platform, there is no requirement to carry any additional devices or computer hardware. Our experiments show that a seamlessness user experience is possible along with modest migration times of ~100 seconds using currently available technologies: In the future this has the potential to decrease to a few seconds with the development of 60GHz radio standards (already underway). In addition, more research questions are brought highlighted by our migration experiments, arising from the heterogeneity of the devices involved, and the requirements for making the migration process transparent to users, particularly across platforms with a diverse set of system resources and user interface technologies. These questions need to be addressed to use live VM migration as a useful tool for computation transfer from a MID to a PC to right-size the user experience to the PC's enhanced processing capabilities, and then back to the MID capabilities when migrating the state back to it for enhanced mobility.

References

1. Barton, J., Zhai, S., Cousins, S.; Mobile phones will become the primary personal computing devices; WMCSA, 2006
2. Caceres, R., Carter, C., Narayanswami, C., Raghunath, M.; Reincarnating PCs with Portable SoulPads; IBM Research Center, Watson; Mobisys 2005
3. Cox L., Chen P.; Pocket Hypervisors: Opportunities and challenges; HotMobile 2007
4. Goyal, S., Carter, J.; A lightweight Secure Cyber Foraging Infrastructure for resource constrained device; WMCSA 2004
5. http://apcmag.com/60ghz_the_gigabit_wireless_we_are_waiting_for.htm: 60GHz: the gigabit wireless we are waiting for
6. http://moblin.org: Moblin Home Page
7. http://www.cellular-news.com/story/35529.php: High Growth Forecasted for the Mobile Internet 2010
8. http://www.cs.toronto.edu/~andreslc/xen-gl/: VMGL, H Andres Lagar-Cavilla
9. http://www.ok-labs.com/releases/release/ok-labs-enables-worlds-first-virtualized-smartphone-with-mobile-virtualizat: OK Labs Enables World's First Virtualized Smartphone:
10. http://www.xen.org/files/xensummit_4/Secure_Xen_ARM_xen-summit-04_07_Suh.pdf: Secure Architecture and implementation of Xen on ARM for mobile devices:
11. Osman, S., Shubhraveti, D., Su, G., Nieh, J; The design and implementation of Zap: A system for migrating Computing environments; OSDI 2002
12. Satyanarayanan, M. et al; Internet Suspend Resume; CMU (http://www.isr.cmu.edu)
13. Suh, S., Song, X., Jumar, J., Mohapatra, D., Ramachandran, U., Yoo, J., Park, I.; Chameleon: A capability Adaptation System for Interface Virtualization; Mobivirt 2008
14. Surie, A., Cavilla, H., Lara, E., Satyanarayanan, M.; Low Bandwidth VM Migration via Opportunistic Replay; HotMobile 2008
15. Want, R., Pering, T., Sud, S., Rosario, B.; Dynamic Composable Computing; ACM HotMobile 2008
16. Wolbach, A., Harkes, J., Chellappa, S., Satyanarayanan, M.; Transient Customization of Mobile Computing Infrastructure, Carnegie Mellon University; MobiVirt Workshop 2008