Moby Dick, on the design of a Swiss army knife of computing

Paul J.M. Havinga, Gerard J.M. Smit
University of Twente, department of Computer Science, Enschede, the Netherlands
{havinga, smit}@cs.utwente.nl

Abstract – *Recent advances in wireless networking technology and the exponential development of semiconductor technology have engendered a new paradigm of computing, called personal mobile computing or ubiquitous computing. This offers a vision of the future with a much richer and more exciting set of architecture research challenges than extrapolations of the current desktop architectures. In particular, these devices will have limited battery resources, will handle diverse data types, and will operate in environments that are insecure, dynamic and which vary significantly in time and location. The research performed in the MOBY DICK project is about designing such a mobile multimedia system. This paper discusses the approach made in the MOBY DICK project to solve some of these problems, discusses its contributions, and accesses what was learned from the project.*

1. PERSONAL MOBILE COMPUTING

In recent years, technology drivers changed significantly. High-end systems used to direct the evolution of computer architectures and systems. Now low-end systems drive technology, due to their large volume and attainable profits. Advances in technology enable portable computers to be equipped with wireless interfaces, allowing networked communication even while on the move. Whereas today's notebook computers and personal digital assistants (PDAs) are self contained, tomorrow's networked mobile computers are part of a greater computing infrastructure.

Two trends – multimedia applications and mobile computing – will lead to a new application domain and market in the near future. *Personal mobile computing* (often also referred to as ubiquitous *computing* [17]) will play a significant role in driving technology in the next decade. In this paradigm, the basic personal computing and communication device will be an integrated, battery-operated device, small enough to carry along all the time. This device will be used as a replacement of many items the modern human-being carries around. It will incorporate various functions like a pager, cellular phone, laptop computer, diary, digital camera, video game, calculator and remote control. An important issue will be the user interface: the interaction with its owner. The device will support multimedia tasks like speech recognition, video and audio.

Wireless networking greatly enhances the usability of a personal computing device. It provides mobile users with versatile communication, and permits continuous access to services and resources of the land-based network. A wireless infrastructure capable of supporting packet data and multimedia services in addition to voice will bootstrap on the success of the Internet, and in turn drive novel networked applications and services.

However, the technological challenges to establishing this paradigm of personal mobile computing are non-trivial. In particular, these devices have limited battery resources, will handle diverse data types, and will operate in environments that are insecure, unplanned, and show different characteristics in time.

Five years ago, in 1995, we started the MOBY DICK project, whose main focus was on the design of such a novel and versatile machine [12]. The MOBY DICK project started as a joint European project (Esprit Long Term Research 20422) to develop and define the architecture of a new generation of mobile handheld computers. The design challenges have been primarily in the creation of a single architecture that allows the integration of security functions (e.g. payment), externally offered services (e.g. airline ticket reservation), personality (i.e. these devices know what their owners want), and communication (mobile internet terminal). After one year, the research themes focussed on: systems architecture of mobile computers, reconfigurable computing, energy efficient multimedia communication, and QoS over wireless access networks.

In the next section we will describe the problems to be solved when designing an architecture for such a *mobile multimedia system*. In Section 3 we describe our contribution and approach to solve some of these problems, followed by a brief introduction to current systems and research on mobile multimedia devices in Section 4. Then in Section 5 we present the main lessons that we have leaned. We conclude in Section 6 and present our vision of future research.

2. Problem Statement

The main focus of the MOBY DICK project has been on those issues pertinent to the system design level, i.e. the area of the hardware system-designer and systems programming-designer. We did not delve into the lower level details of the VLSI realisation of the mobile system itself, nor into the higher levels of the operating system and applications.

2.1. System architecture

Today, the choice of wireless devices is largely limited to simple wireless phones on the one hand, to complex and bulky laptops with wireless communication capability on the other. While these devices serve their purposes, they are neither the most integrated nor the most general: their functionality is often limited, they can operate for just a short time, and they are incapable of fully exploiting the emerging integrated wireless networks. Even while current devices have the ability to communicate and process data, they are and by large primarily either data processing devices *or* communication devices. Simply shrinking the processing devices and communication devices, and packaging them together does not alleviate the architectural bottlenecks of integrated mobile multimedia devices [9]. The real challenge is to design a device where data processing and communication share equal importance.

Multimedia functionality is a driving force for many research challenges. For example, due to the size constraints on a portable computer, the user interface must be small. This is a main reason that pens have become the standard input devices for PDAs. The shortage of area on a mobile device can cause us to trade buttons in favour of recognising the user's intention from analog input devices such as handwriting, gesture [2] and voice. Speech generation and recognition seem an ideal user interface since they require no surface area and allow hands-free and eye-free operation. However, general-purpose speech input and output places substantial storage and processing demands on a mobile device. Other research investigates the use of head-mounted virtual reality displays [16]. Main problems to be solved are the required processing power or communication bandwidth and the required weight and size (i.e. a small and light headgear).

A key challenge of mobile computing is that many attributes of the environment vary dynamically. Mobile devices face many different types of variability in their environment. Therefore, they need to be able to operate in environments that can change drastically in short term as well as long term in available resources and available services. Some short-term variations can be handled by adaptive communication protocols that vary their parameters according to the current condition. Other, more long-term variations generally require a much larger degree of adaptation. Merely algorithmic adaptations are not sufficient, but rather an entirely new set of protocols and/or algorithms may be required. For example, mobile users may encounter a complete different wireless communication infrastructure when walking from their office to the street. They might require another air interface, other network protocols, and so forth. A possible solution is to have a mobile device with a reconfigurable architecture so that it can adapt its operation to the current environment and operating condition.

Reconfigurability also has another more economic motivation: it will be important to have a fast track from sparkling ideas to the final design. If the design process takes too long, the return on investment will be less. It would further be desirable for a wireless terminal to have architectural reconfigurability whereby its capabilities may be modified by downloading new functions from network servers. Such reconfigurability would also help in field upgrading as new communication protocols or standards are deployed, and in implementing bug fixes [9]. One of the key issues in the design of portable multimedia systems is to find a good balance between flexibility and high-processing power on one side, and area and energy-efficiency of the implementation on the other side.

2.2. Wireless communication

Mobile computers require wireless network access, although sometimes they may physically attach to the network for a better or cheaper connection. Wireless communication is much more difficult to achieve than wired communication because the surrounding environment interacts with the signal, blocking signal paths and introducing noise and echoes. As a result wireless connections have a lower quality than wired connections: lower bandwidth, less connection stability, higher error rates, and, moreover, with a highly varying quality. These factors can in turn increase communication latency due to retransmissions, can give largely varying throughput, and incur a high energy consumption. Three key problems in networked wireless multimedia systems are 1) the need to maintain a minimum quality of service (throughput, delay, bit error rate, etc) over time-varying channels, 2) to operate with limited energy resources, and 3) to operate in a heterogeneous environment.

Quality of Service – Considerations of energy efficiency are fundamentally influenced by the trade-off between energy consumption and achievable Quality of Service (QoS). To deal with the dynamic variations in networking and computing resources gracefully, both the mobile computing environment and the applications that operate in such an environment need to adapt their behaviour depending on the available resources including the batteries.

Energy-efficiency – The wireless network interface of a mobile computer consumes a significant fraction of the total energy of a mobile computer. More extensive and continuous use of network services will aggravate this problem. Energy efficiency can be improved at various layers of the communication protocol stack. Adaptability of the protocols is a key issue is achieving this.

Heterogeneity – Future mobile information systems will be built upon heterogeneous wireless (possibly overlapping) networks, extending traditional wired networks to mobiles moving over a wide area. Distributed applications in mobile-computing environments must be prepared to deal with partitions – a mobile computer will, at times, not be able to communicate – or changes in the communication infrastructure with different network qualities. There may be places where they can access multiple transceivers, or even may concurrently use wired access. The interface may also need to change access protocols for different networks, for example when switching from wireless LAN coverage in an office to cellular coverage in a city. This heterogeneity makes mobile computing more complex than traditional networking.

2.3. Energy efficiency

Although the subject of low-power consumption of integrated circuits (ICs) is drawing considerable attention (“cool chips are hot”), this interest is only of recent date. There are several motivations for energy-efficient design. Perhaps the most visible driving source is the success and growth of the portable consumer electronic market. Today’s desktop computers are not intended to be carried, so their design is liberal in their use of space, weight, energy consumption, noise, cabling, and heat dissipation. In contrast, the designer of a hand-held mobile computer should strive for the properties of a wristwatch: light, small, durable and long battery life.

Batteries are the largest single source of weight in a portable computer. Minimising energy consumption can improve portability by reducing battery weight and lengthening the life of a charge. Moreover, the functionality of the mobile computer is limited by the required energy consumption for communication and computation. Unfortunately, the rate at which battery performance improves (in terms of available energy per unit size or weight) is fairly slow, despite the great interest generated by the booming wireless business. Aside from major breakthroughs it is doubtful that significant reduction of battery size and weight can be expected in the near future. It has been generally expected that the battery technology alone will not solve the low-power problem. It therefore makes sense to look for alternative strategies for energy savings and energy management. The emerge of various applications and the need to support them in a wireless setting may open new possibilities for energy-saving strategies.

The way out is *energy efficiency:* doing more work with the same amount of energy. Traditionally, energy efficiency has been focussed on low-power techniques for VLSI design. However, the key to energy efficiency in future mobile multimedia devices will be at the higher levels: energy-efficient system architectures, energy-efficient communication protocols, energy-cognisant operating system and applications, and a well designed partitioning of functions between wireless device and services on the network.

A major challenge in achieving this will be that many attributes of the system environment can vary drastically by several orders of magnitude over the short and long term. Key to these issues will be *adaptability*. Research has shown that continually adapting the system and protocols can significantly improve the energy efficiency while maintaining a satisfactory level of performance [11]. Adapting to the variability is the shared responsibility of many layers in the system design of the mobile device, including the applications.

3. CONTRIBUTIONS

The research performed in the MOBY DICK project addressed the design of an architecture for a mobile multimedia handheld computer that can cope with the requirements and difficulties mentioned above. The main focus has been on the specification of a system architecture supporting the required functions for future handheld devices.

The approach made in our research was to study *practical* solutions to the inherent problems of handheld multimedia terminals. In this field too often, system architectures, protocols, and applications are developed with a theoretical background only and with a limited scope covering one horizontal layer in a system. In contrast, this research is characterised by a strategy that traverses *vertically* through various layers of the system architecture of a multimedia hand-held system and is driven by energy-efficient design considerations.

While low-level circuit and logic techniques have been well established for improving energy efficiency, they do not hold promise for much additional gain. As the issue of energy efficiency becomes even more pervasive, the battle to use the bare minimum of energy will be fought on multiple fronts: semiconductor technology, circuit design, design automation tools, system architecture, operating system, and application design. The key to energy efficiency in future mobile systems will be the higher layers of the mobile system, its system architecture, its operating system, and indeed the entire network, with energy efficiency in mind.

In its most abstract form, a networked computer system has two sources of energy drain required for operation:

- *Communication*, due to energy spent by the wireless interface and due to the internal traffic between various parts of the system, and
- *Computation*, due to processing for applications, the operating system, and tasks required during communication.

Minimising energy consumption is a task that will require minimising the contributions of communication *and* computation, making the appropriate trade-offs between the two [9]. For example, reducing the amount of transmitted data may be beneficial. On the other hand, the computation cost (e.g. to compress the data being sent) might be high, and in the extreme it might be better to just send the raw data.

Because communication has the foremost influence on both Quality of Service and energy consumption, we concentrated in the MOBY DICK project on the *communication channels* rather than the computational elements. The communication channels contribute a significant amount of the total energy consumption of a typical mobile system. As semiconductor technology improves, computation gets cheaper, whereas communication has much less advantage of the smaller feature size. Therefore, communication relatively will get even more expensive. This property also holds for multimedia applications, even though these applications typically require a significant computational effort as well. For a significant part this is due to the limitations of most current hardware and operating systems that are unable to differentiate between various traffic streams.

Specific contributions of our research are the design of an energy-efficient architecture for mobile multimedia systems and a reconfigurable connection switch, as well as the design of crucial wireless network functions (i.e. MAC protocol, adaptable network interface, and a model for adaptable error-correction) that are energy efficient and can support multimedia traffic [3][4][5][6].

3.1. System architecture

The traditional architecture of a mobile is centered around a general-purpose processor with local memory and a shared-bus that connects peripherals to the CPU. However, in such an architecture several problem areas can be identified. Main problem areas are energy consumption, performance, and Quality of Service guarantees.

A large fraction of system time and power budget in a shared bus architecture is devoted to bus transactions. Busses are significant sources of power dissipation due to high switching activities and large capacitance. This architecture requires frequent traversal of multimedia streams over the bus and through the layers of the operating system software, and possibly also through to a network protocol stack which is composed of transport, network, link and medium access (MAC), and physical layer protocols [14].

Current systems based on a shared bus architecture are able to deliver the required performance for various multimedia applications not only by using the rapid advance in technology, but also by careful design and use of the interface modules. The process to achieve this requires a huge amount of effort of both the hardware designer of the I/O interfaces and the system designer. There are many subtle device issues that can influence the overall I/O performance of a system. Minor changes in the hardware or software configuration can have severe consequences for the performance of the system. The reason for these problems is often the interconnect and the interconnection protocols. Since a shared bus cannot give QoS guarantees, a single device or application can reduce the throughput that is available for all devices.

By designing a *connection-centric* architecture that moves processing power closer to the data stream, it is possible to solve these problems. The whole system is based on connections between modules. Each connection is associated with a certain QoS. This approach is especially well suited for continuous media data (e.g. audio, video, etc.), where the processing is actually of a very specialised nature (e.g. signal processing, compression, encryption, etc.) and needs to be carried out in real-time. In contrast to memory-centric (or CPU-centric) systems, a connection-centric system is decomposed out of application-specific modules. In such a system the data traffic is reduced, mainly because unnecessary data copies are removed. For example, in a system where a stream of video data is to be displayed on a screen, the data can be copied directly to the screen memory, without going through the main processor. The CPU is thus moved out of the data flow datapath, although it still participates in the control flow. The role of the CPU is reduced to a controller that initialises the system and handles complex protocol processing that are most easily implemented in software.

The approach used in our research [3] to achieve a system as described above is to have autonomous, reconfigurable modules such as network, video and audio devices, interconnected by a switch rather than by a bus, and to offload as much as work as possible from the CPU to modules placed in the data streams. Thus, communication between components is not broadcast over a bus but delivered exactly where it is needed, work is carried out where the data passes through, bypassing the memory. To limit the communication overhead and the required buffering, the granularity of the tasks on the devices is rather coarse, and the application is partitioned in large blocks. The programmability of each module is more fine-grained and is controlled by the individual autonomous module.

The interconnect of the architecture is based on a reconfigurable communication network switch, called *Octopus*, which interconnects a general-purpose processor, (multimedia) devices, and a wireless network interface. Conceptually, the architecture is analogous to a self-routing packet switch. The connection-oriented approach using fixed sized cells and the asynchronous multiplexing are key factors. This not only eliminates the need to transfer a large number of address bits per access, it also gives the system the possibility to control the QoS of a task down to the communication infrastructure. All connections are identified with a connection identifier, which is used to identify the type of data, and the module destination address. This identifier provides the mechanism to support lightweight protocols that provide data-specific transport services that are associated with a certain QoS. This is an important requirement since in a QoS architecture all system components, hardware as well as software, have to be covered end-to-end along the way from the source to the destination.

The Octopus testbed

A key goal motivating the design has been simplicity and flexibility. Our goal was to build a testbed from off-the-shelf VLSI components that was easy to design and test.

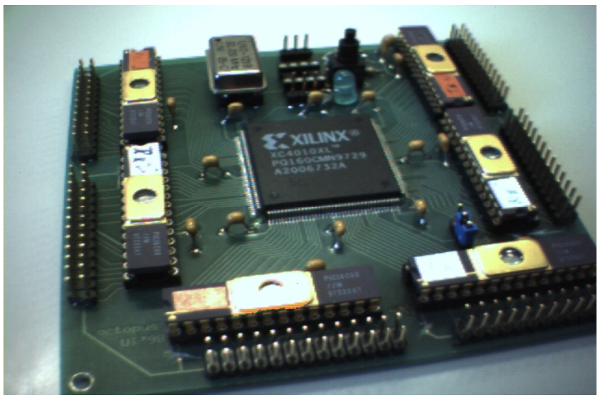

Figure 1: Testbed implementation *Octopus*.

With this testbed we are able to quickly explore the design space of the system. Therefore, the prototype interconnection module is build using a Field Programmable Gate Array surrounded by several low-end and low-power micro-controllers.

A prototype of the Octopus switch has been implemented on a single small printed circuit board with a standard Field Programmable Gate Array of Xilinx (i.e. XC4010XL) and six low-end micro-controllers (i.e. Microchip PIC 16C66). The main reason for using FPGAs in our design is *flexibility*. This approach creates a test-bed for interconnection structures, arbitration mechanisms, and clocking mechanisms. With an FPGA it is relatively simple and efficient to implement the datapath of a system. However, the FPGA is not suitable for high control complexity. For example, it would require relatively large area if the FPGA has to implement the connection setup protocol, handle timeouts and perform retransmissions. Traditional microprocessors are much better equipped to deal with larger and more complex control structures, and although they are capable of handling the data-flow as well, they typically can only achieve this with a much lower performance than with a hardware implementation. By combining an FPGA with a traditional processor, it is possible to build a system that uses a mixture of hardware and software in order to exploit the best features of both domains effectively.

The realised prototype shows that it is relatively simple to implement an architecture that is suitable for becoming the interconnection structure of a mobile computer. The combination of a FPGA surrounded by several small micro-controllers proved to be a flexible prototyping testbed. The complexity of the architecture is low, which make it feasible to built the switch in a custom IC.

3.2. Wireless communication

Another important aspect in mobile multimedia systems is wireless communication protocols that provide multimedia services to mobile users.

We have shown that energy-awareness must be applied in almost all layers of the network protocol stack. To achieve maximal performance and energy efficiency, *adaptability* is important, as wireless networks are dynamic in nature. Furthermore, if the application layer is provided with feedback on the communication, advantage can be taken from the differences in data streams over the wireless link. To allow this, feedback is needed from many layers: the physical layer provides information on link quality, the medium access layer on effectiveness of its error correction, and the data link layer on buffer usage and error control.

Multimedia applications are characterised by their various media streams. Each stream can have different quality of service requirements. Depending on the service class and QoS of a connection a different policy can be applied by the communication protocol in order to minimise energy consumption. For example, by avoiding error-control overhead for connections that do not need it and by never transmitting stale data, efficiency is improved. This combination of limited bandwidth, high error rates, and delay-sensitive data requires tight integration of all subsystems in the device, including aggressive optimisation of the protocols that suits the intended application. The protocols must be robust in the presence of errors and they must be able to differentiate between classes of data, giving each class the exact service it requires.

The access to the wireless channel is controlled by data link protocols. Many protocols for wireless networks are basically adaptations of protocols used in wired networks, and ignore energy issues. A first step in improving the energy efficiency of the wireless network protocols is to *eliminate useless activity* of the wireless interface. There are various reasons for this useless activity. It has been shown that for typical applications like a web-browser or e-mail, the energy consumed while the interface is 'on' and idle is more than the cost of actually receiving packets. That is because these applications have little demanding traffic needs, and hence the transceiver is *idling* most of the time. Furthermore, in a typical wireless broadcast environment, the receiver has to be powered 'on' at all times to be able to receive messages from the base station, resulting in significant energy consumption. The receiver subsystem typically receives all packets and forwards only the packets destined for this mobile. Another cause is due to the *inactivity threshold*, which is the time before a transceiver will go in the 'off' or 'standby' state after a period of inactivity, which causes the receiver to be in an energy consuming mode needlessly for a significant time. Significant time and energy is further spent by the mobile in switching from transmit to receive modes, and vice-versa.

The next step is to *reduce the amount of data*, which must be pushed through the channel. This goal can be reached in a number of ways. One is to reduce the *overhead of a protocol* which influences the energy requirements due to the amount of 'useless' control data and the required computation for protocol handling. Another step is to avoid *collisions* that typically may occur in broadcast networks. This causes the data to become useless and the energy needed to transport that data to be lost. The high *error rate* that is typical for wireless links is another source of energy consumption for several reasons. First, when the data is not correctly received the energy that was needed to transport and process that data is spoiled. Second, energy is used for error control mechanisms. This includes energy spent in the physical radio transmission process, as well as energy spent in computation, such as signal processing and error control at the transmitter and the receiver. Finally, because in wireless communication the error rate varies dynamically over time and space, a fixed-point error control mechanism that is designed to be able to correct errors that hardly occur, spoils energy and bandwidth. If the application is error-resilient, trying to withstand all possible errors spoils even more energy for needless error control. Reducing the amount of data is also an application-layer issue. For example, the application might change the compression rate or possibly reduce the data resolution. Instead of sending an entire large full-colour image, one can send black-and-white half-size images with lossy compression.

The goals of low energy consumption and the required support for multiple traffic types lead to our communication system that is based on reservation and scheduling strategies [6]. For each connection a different set of parameters concerning scheduling, flow control and error control is applied [5]. The wireless network is composed of several base-stations that each handle a single radio cell possibly covering several mobile stations. The base-station controls access over the wireless channel based on communication requests for connections of the mobiles by dividing bandwidth into transmission slots. The key to providing QoS for these connections and the energy efficiency of the mobiles will be the scheduling algorithm that assigns the bandwidth. The premise is that the base-station has virtually no processing and energy limitations, and will perform actions in courtesy of the mobile. The main principles are: avoid unsuccessful actions by avoiding collisions and by providing provisions for adaptive error control, minimise the number of transitions by scheduling traffic in larger packets, synchronise the mobile and the base-station which allows the mobile to power-on precisely when needed, and migrate as much as possible work to the base-station.

The testbed

We have implemented a test-bed of the network interface that we can use to experiment with the various techniques and mechanisms for e.g. error control and MAC protocol. It is build with off-the-shelf components (a Xilinx FPGA, a microcontroller, and memory) to allow a short design cycle. Figure 2 shows a photograph of the network interface implementation.

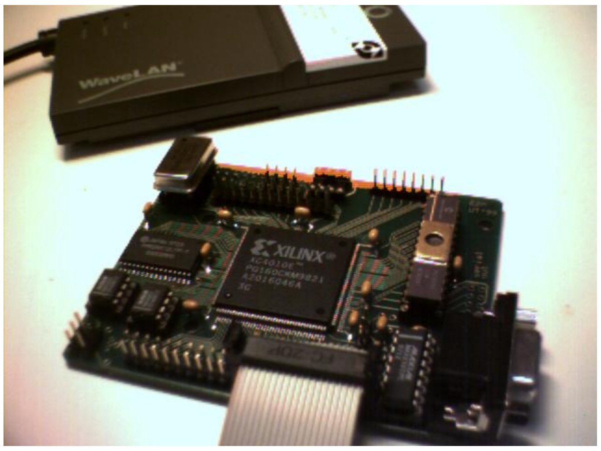

Figure 2: Network interface implementation.

We use a WaveLAN modem as the physical layer. Migration of some functionality from the mobile, for example to the base-station, allows reduction of the complexity of mobiles. Added complexity in the base-station or other parts of the fixed network is justified because they can be better equipped and are not battery powered.

4. Related Work

The growing popularity of mobile systems has spawned much interest and research by industry and universities in both computer science and electrical engineering. Most of the current research, however, often tackles just one horizontal layer of the design. Although this research is valuable, and must be applied whenever suitable, we will provide here merely a brief overview of those systems and current research that look into the problem of designing a mobile multimedia device in an integrated fashion.

Currently, there is a broad consensus that the existing mobile devices are by far not capable of supporting the required multimedia functionality. Some reasons are: processing power, energy consumption, communication bandwidth requirements, etc. About the solution to solve this problem there is much less consensus, however. Within the notion of mobile computing, there is considerable latitude regarding the role of the portable device. Is it a terminal or an independent, stand-alone computer? How many purposes shall the device serve? Many different architectural choices are possible, each with a different partitioning of functions between the wireless device and remote servers. These design choices greatly affect the issues mentioned in this paper.

Several architectures have been proposed that address mobile multimedia computing. Only few systems address energy reduction. Systems like the *InfoPad* [15] and *ParcTab* [7] are designed to take advantage of high-speed wireless networking to reduce the amount of computation required on the portable. These systems are portable terminals and take advantage of the processing power of remote compute servers. No local computation, except for appropriate coding/decoding of the I/O data, is done at the pad. Such devices are known as *thin clients*, since the client itself does little work. This approach simplifies the design and reduces power consumption for the processing components, but significantly increases the network usage and thus potentially increases energy consumption because the network interface is energy expensive. These systems also rely on the availability of a high bandwidth network connectivity and cannot be used when not connected.

In the Ubiquitous Communications project (*Ubicom*) [16] at the Delft University of Technology the clients also depend heavily on the wireless communication network. This project aims at developing a campus-wide system for wireless communication that is capable of supporting multimedia applications. The target is a visual geographic information system that uses augmented reality techniques to display information on a mobile user's headset; information is super-imposed on the user's view using a retinal scanning display. To minimise energy consumption in the mobile unit, the main processing power is located in the backbone network.

The *Merlin* project of the University of California at Los Angeles (UCLA) [11] is developing mobile computing and wireless communication technologies with the focus on creating a wireless I/O-network subsystem that can be used to create many different types of wireless connected multimedia nodes: handheld computes, wireless cameras, wireless IP phones, etc. The subsystem is composed of a wireless network processor, codecs, and radio to provide all the necessary wireless networking and multimedia processing capabilities. In the architecture of *WAND*, a low-power embeddable module built at UCLA for creating multimedia wireless terminals, the general-purpose processor is moved out of the packet flow data path, and the data streams flow directly between the radio and the speech and image codecs. A full-fledged PC or PDA may be adjunct to WAND, but its presence is optional and, in many wireless terminals unnecessary.

Other research mainly concentrated on specific topics, and did not cover the system architecture of a mobile computer as a whole. There is much research on multimedia processors, hardware accelerators, and heterogeneous multiprocessor architectures mainly targeted for DSP algorithms (e.g. [1][8][10]). In recent years much research has been done in providing QoS over a wireless link. Access protocols for these systems typically only address network performance metrics such as throughput, bit efficiency, and packet delay. However, thus far, little attention is given to energy conserving protocols, and researchers mainly focuses their effort on energy reduction by circuit design. Very recently there is a growing interest in energy-efficient design, although mainly concentrating on medium access and link-layer energy reduction techniques.

5. Lessons Learned

In the Moby Dick project the following lessons have been learned:

- *Energy-efficiency is crucial in the architecture of a mobile multimedia system*. The increasing demands for performance and integration of computer systems will be accompanied by increasing levels of energy consumption. The requirement of portability of hand-held multimedia computers and portable devices places severe restrictions on size and energy consumption. Without a significant energy reduction techniques and energy saving architectures, battery life constraints will limit the capabilities of these machines. More extensive and continuous use of network services will only aggravate this problem since communication consumes relatively much energy.

- *Apply a system-wide layer integration/co-operation*. The key to energy efficiency will be achieved in the integrated design of all layers of the system, its system architecture, its operating system, and the entire network. Co-operation or integration of the various layers significantly improves energy efficiency of the system because it reduces wasteful operations and data streams retain a high locality of reference.

  The art of low-power design used to be a narrow speciality in analog-circuit design. As the issue of energy efficiency becomes even more pervasive, the battle to use the bare minimum of energy will be fought on multiple fronts: semiconductor technology, circuit design, design automation tools, system architecture, operating system, and application design. We have shown that there is a vital relationship between hardware architecture, operating system architecture and applications architecture, where each benefits from the others. In our architecture we have applied several supplementary energy-reduction techniques on all levels of the system. In particular we have looked at the integration of the wireless network in the system. The combination of limited bandwidth, high error rates, and delay-sensitive data requires tight integration of all subsystems in the device, including aggressive optimisation of the protocols to suit the intended application.
- *Use a Quality of Service framework*. We have demonstrated in our research and in particular in the design of the system architecture, the switching network, and the wireless network design, that Quality of Service is not only important to provide an adequate level of service for a user, but can also be used as a tool to achieve an energy-efficient system. Users and applications request a certain QoS level. The system then operates in such a way that it will try to satisfy these requirements, but never gives more quality than required and necessary. As the mobiles must remain usable in a wide variety of environments, they must be flexible enough to accommodate a variety of multimedia services and communication capabilities and adapt to various operating conditions in an (energy) efficient way.
- *Communication is more important and has more influence than computation*. In the MOBY DICK project we concentrated on the communication channels (both internal in the system and the wireless network) rather than the computational elements, since the contribution of communication channels to the costs of the total system is increasing rapidly, and communication has a significant influence on both the energy-efficiency and the Quality of Service. A general theme in our approach was to reduce the amount of communication and avoid 'useless' and inefficient computation, which consequently reduces energy dissipation and increases performance of the system.
- *Focus on the core issues.* Early in the project we had a wide view of problems that we thought were important to solve. Therefore, we started with many topics (like data consistency and security). After one year, we focussed on those items that seemed to be the most compelling and in which we had the most expertise. Therefore, we concentrated on the key issues of energy-efficiency and Quality of Service and took a vertical, system wide, approach with the main emphasis on those issues pertinent to the system design level, i.e. the area of the hardware designer and systems-programming designer. We were not dealing with the lower level details of VLSI realisation, nor the higher levels of the operating system and applications.

  We were not concerned in building a low-power system, but merely an *energy-efficient* system. Therefore, the test-bed we have made is designed to evaluate the *energy efficiency* of designs, and is *not* designed to be low power! The actual implementation of the test-bed is therefore primarily designed to be *flexible*, and suitable for doing experiments with various design alternatives. Because of this, the implementation test-bed used various flexible, but certainly not low-power components (i.e. we have used Xilinx FPGAs).
- *Build experimental computer systems*. Already early in the project we started building parts of the system, not just as a proof of concept, but also because it focussed us to the real problems of a systems design. We have found that this approach was very valuable since this pointed us to some practical problems that we would not have seen when we stopped just after simulation. Also, at the same time, we noticed that some potential problems were not as difficult to solve as expected. While it is certainly easier to quit after the simulation stage, we found that the results at that stage lack reality and are often wrong. This is in particular true for a system wide approach, in which the interaction between different layers plays a significant role.

Although our testbed currently consists of various small printed circuit boards containing Field Programmable Gate Arrays (FPGAs), micro-controllers, and memory, the complexity of these designs is low. This low complexity will make it possible to transfer the architecture to a (large) custom IC.

6. CONCLUSION

Our vision of a personal computing device is even more compelling today than it was in 1995. The trend in wireless terminals has been to shrink a general-purpose desktop PC into a package that can be conveniently carried. Even PDAs have not ventured far from the general-purpose model, neither architectural nor in terms of usage model. Both the notebook and the personal computer generally use the same standard PC operating system such as Windows or Unix, same applications, use the same communication protocols and use the same hardware architecture. The only difference is that portable computers are smaller, have a battery, a wireless interface, and often use low power components.

A key challenge of mobile computing is that many attributes of the environment vary dynamically. Mobile devices face many different types of variability in their environment. Therefore, they need to be able to operate in environments that can change drastically in short term as well as long term in available resources and available services. Merely algorithmic adaptations are not sufficient, but rather an entirely new set of protocols and/or algorithms may be required. For example, mobile users may encounter a complete different wireless communication infrastructure when walking from their office to the street. A possible solution is to have a mobile device with a reconfigurable architecture so that it can adapt its operation to the current environment and operating condition. Adaptability and programmability should be major requirements in the design of the architecture of a mobile computer.

We are entering an era in which each microchip will have billions of transistors. One way to use this opportunity would be to continue advancing our chip architectures and technologies as just more of the same: building microprocessors that are simply complicated versions of the kind built today. However, simply shrinking the data processing terminal and radio modem, attaching them via a bus, and packaging them together does not alleviate the architectural bottlenecks. The real design challenge is to engineer an integrated mobile system where data processing and communication share equal importance and are designed with each other in mind. Integrating current PC or PDA architecture with a communication subsystem, is not the solution. One of the main drawbacks of merely packaging the two is that the energy-inefficient general-purpose CPU, with its heavyweight operating system and shared bus, becomes not only the center of control, but also the center of data flow in the system and a main cause of high energy consumption.

Clearly, there is a need to revise the system architecture of a portable computer if we want to have a machine that can be used conveniently in a wireless environment. A system level integration of the mobile's architecture, operating system, and applications is required. The system should provide a solution with a proper balance between flexibility and efficiency by the use of a hybrid mix of general-purpose and the application-specific approaches.

We, in the academic world, have the luxury of building a machine that demonstrates a vision of a computer system without concern for compatibility. The MOBY DICK project demonstrates the importance of building experimental computer systems. Simulation results do not reduce the perceived risk of new technologies sufficiently for industry to adopt them.

Future work

Although we have come up with solutions to a number of problems in the field of mobile multimedia computing, many others have remained unsolved or received only minor attention. We attempt to give a few suggestions for future research.

Having an energy efficient architecture that is capable to handle adaptability and flexibility in a mobile multimedia environment requires more than just a suitable hardware platform. First of all we need to have an operating system architecture that can deal with the hardware platform and the adaptability and flexibility of its devices. Optimisations across diverse layers and functions, not only at the operating systems level, is crucial. Managing and exploiting this diversity is the key system design problem. A model that encompasses different levels of granularity of the system is essential in the design of an energy management system and in assisting the system designer in making the right decisions in the many trade-offs that can be made in the system design. Finally, to fully exploit the possibilities offered by the reconfigurable hardware, we need to have proper operating system support for reconfigurable computing, so that these components can be reprogrammed adequate when the system or the application can benefit from it.

The lessons learned from the design of the MOBY DICK architecture serve as a first step towards a system-level design of an energy-efficient mobile multimedia computer. Research in these items will be continued in *Chameleon* project [13] that will in particular perform research in reconfigurable computing for these systems.

Acknowledgements

We thank all of the many people who contributed to the success of the Moby Dick project. Special thanks go to Gunnar Hartvigsen, Terje Fallmyr, and Tage Stabell-Kulø from the University of Tromsø, Norway; Alberto Bartoli, Gianluca Dini, Luigi Rizzo, and Marco Avvenuti from the University of Pisa, Italy; Sape Mullender, Arne Helme and Ties Bos from the University of Twente.

REFERENCES

[1] Abnous A., Seno K., Ichikawa Y., Wan M., Rabaey J.: "Evaluation of a low-power reconfigurable DSP architecture", *proceedings 5th Reconfigurable Architectures workshop (RAW'98)*, March 30, 1998, Orlando, USA. (URL: http://xputers.informatik.uni-kl.de/RAW/RAW98/adv_prg_RAW98.html)

[2] "Rock 'n' Scroll – Button-free Tilt and Gesture Input for Itsy", http://www.research.digital.com/wrl/projects/RocknScroll/RocknScrollOverview.htm.

[3] P.J.M. Havinga, "Mobile Multimedia Systems", *Ph.D. thesis University of Twente*, February 2000, ISBN 90-365-1406-1, www.cs.utwente.nl/~havinga/thesis.

[4] Havinga P.J.M., Smit G.J.M.: "Octopus: embracing the energy efficiency of handheld multimedia computers" , *proceedings fifth annual ACM/IEEE international conference on mobile computing and networking (Mobicom'99)*, pp.77-87, August 1999.

[5] Havinga P.J.M.: "Energy efficiency of error correction on wireless systems", *proceedings IEEE Wireless Communications and Networking Conference (WCNC'99*), September 1999.

[6] Havinga P.J.M., Smit G.J.M., Bos M.: "Energy efficient wireless ATM design", *ACM/Baltzer Journal on Mobile Networks and Applications (MONET), Special issue on Wireless Mobile ATM technologies*, Vol. 5, No 2., 2000.

[7] Kantarjiev C. et al.: "Experiences with X in a wireless environment", *Mobile and location-independent computing symposium*, Cambridge MA, August 1993.

[8] Leijten J.A.J.: "Real-time constrained reconfigurable communication between embedded processors", *Ph.D. thesis, Eindhoven University of Technology*, November 1998.

[9] Lettieri P., Srivastava M.B.: "Advances in wireless terminals", *IEEE Personal Communications*, pp. 6-19, February 1999.

[10] Mangione-Smith W.H., et al.: "Seeking solutions in configurable computing", *IEEE Computer*, pp. 38-43, December 1997.

[11] Sheng S., Chandrakasan A., Brodersen R.W.: "A Portable Multimedia Terminal", *IEEE Communications Magazine*, pp. 64-75, vol. 30, no. 12, Dec., 1992.

[12] Smit G.J.M., Havinga P.J.M., et al.: "An overview of the Moby Dick project", *1st Euromicro summer school on mobile computing*, pp. 159-168, Oulu, August 1998; Moby Dick homepage: www.cs.utwente.nl/~havinga/mobydick.html.

[13] Smit G.J.M., Martinus Bos, Paul J.M. Havinga, Sape J. Mullender, Jaap Smit: "Chameleon - reconfigurability in hand-held multimedia computers", *proceedings First International Symposium on Handheld and Ubiquitous Computing*, HUC'99, September 1999.

[14] Srivastava M.: "Design and optimization of networked wireless information systems", *IEEE VLSI workshop*, April 1998.

[15] Truman T.E., Pering T., Doering R., Brodersen R.W.: The InfoPad multimedia terminal: a portable device for wireless information access", *IEEE transactions on computers*, Vol. 47, No. 10, pp. 1073-1087, October 1998.

[16] Toetenel H.: "The ubiquitous communication program", *Euromicro summer school on mobile computing '98*, Oulu, pp. 181-189, August 1998, http://ubicom.twi.tudelft.nl.

[17] Weiser M.: "Some computer science issues in ubiquitous computing", *Communications of the ACM*, 36(7):75-84, July 1993.