Restriction, Modification and Extension of Consumer Devices for Prototyping Ubiquitous Computing Environments

Matthias Kranz, Albrecht Schmidt
Research Group Embedded Interaction
University of Munich
Amalienstrasse 17, 80333 Munich, Germany
{matthias, albrecht}@hcilab.org

Abstract

In this paper, we report on several work-in-progress projects of our research group and other systems that exploit commercially available off-the-shelf components for deriving research prototypes and systems in the field of ubiquitous computing. We discuss the benefits of restricting, modifying and extending the built-in functionality of these systems to ease the development process. Using the approach presented reduces the amount of low level knowledge, of e.g. electrical engineering, to achieve the actual goal of building new appliances and ubiquitous computing systems. The approach presented empowers researchers from many domains to build prototypes that can be experienced by users. We present an example of an extended commercial product, the Nokia 770 tablet PC.

1 Introduction

Building infrastructures and smart appliances for ubiquitous computing scenarios is still a challenging task. In most cases there is not standard hardware platform that can be used and hence development spans over various domains and requires expertise in different fields. When looking at research prototypes published over the last years, it can be seen that many of such novel computational enriched environments or appliances are built from scratch. To do this knowledge on various platforms, ranging from simple microcontrollers, embedded hardware and sophisticated location and context-aware systems may be required. In various projects it can be seen that a significant part of the research effort (in terms of money and man-power) is spend on creating a platform which is essentially only the vehicle for the research to be carried out. Additionally, many research prototypes are only working in very limited laboratory settings and are not easily deployable into the world outside. Creating, demonstrating and evaluating a convincing ubiquitous computing scenario, especially when it involves users directly, is in general expensive to do as for meaningful results deployment in real environments is often inevitable.

We argue that using existing commercial, off the shelf hardware can reduce time and costs for creating ubiquitous computing systems. This approach, appearing at a first glance as tinkering and hacking, can lead to more real-world oriented systems that are more targeted at usability and utility. Thereby this approach offers in many respects greater benefits than many build-from-scratch systems and can help to quicker assess the research questions. The time reduction in the development phase can resources to focus on actual research on novel smart appliances and new forms of human computer interaction. Even so creating a specific hardware platform educates the researcher in this area, it does in many cases not help to advance the state of the art.

In the following we first introduce main challenges for creating prototypes. Then we highlight one approach that can be taken for creating prototypes from custom devices.

2 Challenges in Prototyping

When designing and implementing a prototype from scratch many challenges arise – in particular related to building the hardware. In the project TEA [15], Smart-Its [9] and Equator [8] as well as in work we conducted more recently we can identify, among others, the following issues.

2.1 Networking and wireless communication

In most ubiquitous computing prototypes connecting the prototype to a network is essential. Options include short

range wireless connection, wide area networking (access over modem or DSL), and Ethernet connections.

For networking many components (e.g. serial to Bluetooth connectors, one-chip Ethernet components) are readily available. This suggests at a first glance that it is easy and straightforward to add networking to a prototype. However, it often shows that there is still a significant effort in implementing a reliable system.

2.2 Minimization, Robustness, Power consumption

When prototypes should be deployed beyond the lab size, power consumption and robustness are important issues. Our experience shows that the effort for creating a system that can be handed out to others over an extended period of time (e.g. a learning appliance for use in a sports centre over a duration of 4 weeks) involves much more effort than creating a demonstrator for use in the lab or in a restricted study. In many cases, however, the unrestricted use is central for research in user interfaces and applications.

In comparison, off-the-shelf devices are often already very robust, the electronic is minimized and optimized for power consumption. E.g. when creating a wireless interactive display, it is close to impossible to these parameters a mobile phone.

2.3 Hardware update due to unavailability of components

For custom developments, the update-cycle in available components forces to adapt the prototypes over the runtime of a project, even so the functionality of the prototype is not altered. A typical approach is to first create a small number of prototypes (1-10) that are used for testing and in small studies. In a later phase, a larger number of prototypes is required (e.g. for larger study or field test). We have experience in several projects that even so the prototypes tested and explored in the first phase had to be redesigned due to the unavailability of components (e.g. RF-Modules or small displays).

Similarly, over the course of a project the capabilities of standard hardware systems change (e.g. increased network bandwidth, higher resolution screen, more power efficient). For custom developments, the same redevelopments are required. When basing prototypes on existing devices these improvements come without additional effort.

2.4 Operating system and low-level software development

Creating custom hardware from scratch offers a lot of freedom with regard to the software implementation. However, this also implies that software has to be developed for a specific hardware. In several of our projects this required a complete development of system and driver software before any application specific software could be created. In other cases it *only* requires the porting of a available operating system.

In many cases, however, the *new* developments are mainly recreating the functionality available in conventional system. The effort for the standard base system appears much bigger than the work put into the really different features that are pushing the research.

There are many further challenges in the development of interactive prototypes such as the integration of high quality interactive screens, the inclusion of large memory or compatibility with other devices.

Most of the time creating the prototypes is not extremely difficult, as modules (e.g. networking, memory, screens) are available but it still requires resources. If the research contribution is meant to be in the domain of human computer interaction and applications much of the work in these areas, even so very interesting, does provide little new findings and diverts attention from the central issues. This is also reflected in recent publications in ubiquitous and pervasive computing where the amount of purely device-oriented papers continually decreases.

3 Approaches to Prototyping

Building prototypes based on existing systems can in many cases make the overall development easier and quicker. Attaching sensors to an existing networked device may be much less effort than creating such a device from scratch. Often the platforms, that one would like to extend are not meant for extension, but can still be used as enough information is available to use them as a basis for functional prototypes.

By appropriating or *hacking*, in a positive sense, extensible platforms can be derived from standard commercially available off-the-shelf components. An interesting example is provided by the rope interfaces [3]. The objective was to create an engaging interface for children to navigate in one dimension. The prototype (hack) used consisted of an off-the-shelf mouse with scroll wheel and a rope that was used to move the scroll wheel. Technically this is trivial, but the approach is very well suited to implement the idea and to create the user experience.

In the following we discuss three different approaches for appropriating off-the-shelf devices for prototyping.

3.1 Restriction and Repackaging

One straightforward approach is restricting the use of the existing device. The obvious way for doing this is to repackage the object as it can be seen from the mouse example above. A similar approach with a keyboard was used in a public exhibit at the Cybernarium. The selection of the viewpoint in the 'Aula Regia' was done with a small number of buttons. These buttons where on a layout of the floor plan on top of a box. Inside the box was a standard keyboard – not visible to the user. For the user, it appeared as a normal floor plan with several buttons [16]. Both these examples show that repacking standard hardware allows also the development in general advances, e.g. both prototypes could now be done (without additional effort) wireless as the base devices (mouse and keyboard) are available as wireless devices.

Physically repacking existing devices and hence restricting the set of interactive features (e.g. the buttons or screen space) is another option. For example, a Compaq IPAQ was provided with a different housing to make it look like a completely different device. The screen was restricted and only visible through a round hole in the new housing[14]. The development effort for this new interactive device was much less than for creating a complete new device that offers all the features included in the IPAQ. For creating new shells 3D printing is a very interesting option.

3.2 Modifcation and Extention

Modifying and extending existing devices can be done on hardware and software level. A simple example of a hardware extension is to use a mouse as a sensor by replacing the button with a different switch, e.g. [4]. It is obvious that replacing a mouse button physically by a binary sensor in a wireless mouse is easier than creating wireless sensor from scratch.

Various example have shown that adding new hardware, especially sensors and actuators, to existing devices is an easy way of extending their functionality and of obtaining novel devices [1].

The decision of which option to take depends on the task. A prototype with the affordances and form factor of the envisioned device surely is favourable has to be preferred to alternatives.

3.3 Example Developments

In this part we would like to present two examples that used commercial off the shelf devices to derive a richer device. The presented Nokia 770 tablet PC and the RCore are examples for extention of commercial products.

3.3.1 Nokia 770

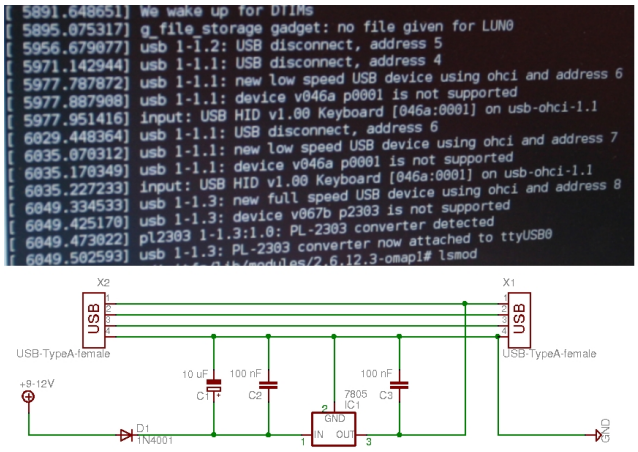

Figure 1. Hacking the Nokia 770. The upper part of the image shows the output of *dmesg* after connecting and removing several USB devices, e.g. a USB Keyboard and a USB-to-serial connector (here a PL-2303 chip was used in the converter). The lower part of the image shows the necessary circuit for providing power to the USB chip in the Nokia 770. More information and the schematics in Eagle format can be downloaded at [10].

An example of an extension we recently did is the Nokia 770 touch-screen Tablet PC, a small hand-held device with Bluetooth and Wavelan connectivity. This already runs a small Debian-based Linux system, called Maemo [12], on an ARM processor. The system can be used as any PDA for managing contacts and web access as well as gaming. Nokia used a open system approach, enabling the interested user to add any software and functionality to this device by providing all information necessary to access even the internals of this device.

This comprises for example a hidden root-mode and the possibility to activate usb-root functionality for the mini-usb port on the device, originally designed only as usb-client port for data synchronization between the Nokia 770 and a host PC. The kernel already contains support for USB devices like keyboards, mice, storage devices and even USB-to-serial converters. We are currently using the Nokia 770 in our research [10] for realizing several projects where similar hardware cannot be built or bought for a reasonable and comparable price.

As the Nokia 770 directly does not recognize connected USB components, we further investigated what is necessary to provide a fully working USB interface. It proved that the USB power was missing. We designed a small interface that

provides the necessary 5 Volts to the USB chip. When the device now is configured as USB host, arbitrary devices can be added.

Now that the Nokia 770 is *open* for additional components, we e.g. added a sensor platform, the Particle Computer [6, 9, 2]. It uses serial line communications which are now provided by a USB-to-Serial converter using the PL 2303 chip. The Nokia 770 can now be used to provide more processing power e.g. for computing the data of various sensory inputs for which the Particle Computer alone is insufficient. The Nokia 770 also augments the Partlce Computer with its rich interaction capabilities, like Bluetooh, WiFi and a sufficiently large touch screen.

The Nokia 770 is currently available for about 350 EURs in the stores. We tried to build a similar interaction device of commercially existing components. We chose a eDIP240-7 touch screen [7] and a Lantronix WiPort [11]. Both components are available for aroung 150 EURs each. The interfaces of these components are easy-to-use serial interfaces. The eDIP240-7 already provides comfortable built-in functions to generate graphic primitives and cusomizable touch screen events. The resolution, however, is 240x128 pixels and therefore very limited. Also, the basic prices already sum up to nearly the price for the complete Nokia 770. Still, power supply components and a suitable housing are missing. For rapid prototyping, a 3D model could be constructed and be printed out on a 3D printer. Also, due to the lack of a library for the eDIOP240-7, a complete development would have had to be done. For the Nokia 770 existing GTK code can be used and ported.

This example vividly demonstrates that for many cases using an existing commercial product and applying hacking, extention and modification to it results in a better overall system at a reduced effort in time and money.

3.3.2 RCore

The last example we want to present shows the extension and hacking of a commercial product on the lowest level. The RCore [13], a wireless router core for the uPart Particle Computers, is by default nothing but a receiver module for uPart Computer modules. It comprises a ChipCon 1010 [5] RF system-on-chip module and a Particle Computer interface connector. By this connector, standard add-on boards for the Particle Computer platform can be used with the RCore. This is possible due to the careful design done by the engineers at Particle Computer.

While it is originally designed only as router core, it itself is an interesting ubicomp platform. It uses a standard 8051 compatible microcontroller. For this platform, free compilers exist, for example the SDCC [17]. Also, the price is 35 EURs compard to 125 EURs for a fully-featured

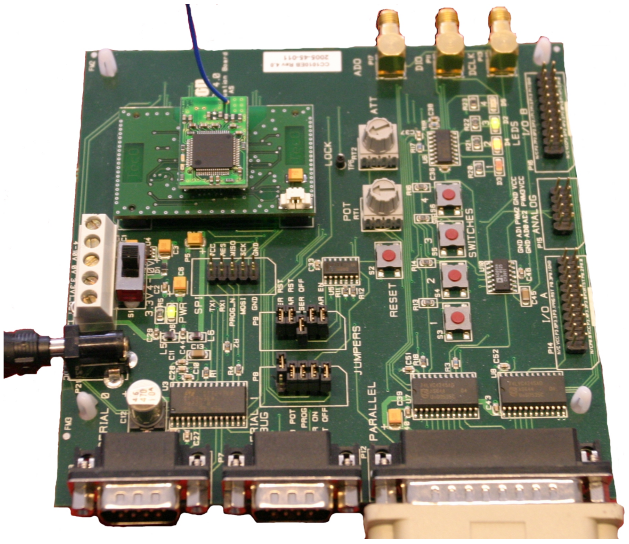

Figure 2. Development board for ChipCon 1010 based devices. The RCore is attached via an adapter board in the top left part of the image.

Particle Computer which is not always needed, e.g. for simple sensor data acquisition. So, if just a small platform with RF capabilities with standardized interfaces is needed, this is an execellent choice. The RCore so far proved to be sufficient for small sensor system while having more flexibility than the uParts also sold by Particle Computer.

4 Potentials

As we have shown, systems, either initially open or not, can be exploited for building suitable, state-of-the art research prototypes. By their usage we come to interesting new appliances for the exploration of novel interaction techniques for human computer interaction.

As shown with the example [16] above, we also execute the technological step ahead from e.g. wired devices like mouse and keyboards to wireless devices like the Nokia 770. The improvement through general technological advances strengthens the suggested approach.

An additional argument for using such available devices is the completed design process and the evaluation undertaken by the companies before their market launch. The knowledge in those devices ensures ideally for their utility and usability.

Often it appears the cost for a custom development is lower than reusing existing hardware. To our knowledge, many people who argue this way do this based on the price for the single components and do not include in the time their researchers, staff, and students spend in the

development that is not directly related to their research goals.

5 Conclusions

We argued that restriction, modification and extenstion of existing commercially off-the-shelf systems is suitable for deriving state-of-the-art basic components for systems in the field of ubiquitous and pervasive computing. The resulting systems consist of already well-designed building blocks, contributing to improved utility and usability of the overall to-be-built system. They lower the overall time and cost of such systems giving way for more deployed installations in the real world. Also, much systems, like the Nokia 770, consist of an agglomeration of parts that cannot be bought or built for an equal amount of money while additionally lowering the amount of low-level knowledge needed to actually built and use similiar components in the own projects.

6 Future Work

We are currently analyzing recently presented and deployed infrastrucutres and systems in ubiquitous and pervasive computing to derive a heurisitc on when it is more promising to use and modify existing systems than to start development from scratch.

We hope that we can provide a well-founded heuristic together with a list of requirements and desicion support. This would not also help to cut time and costs but also, by reducing the amount of low-level knowledge like electrically wiring small sensors, enable researchers from other domains with the possiblity to build systems with more multidisciplinary backgrounds and thereby make them more user- and application-oriented and bring more systems out of the lab into the real world where users can benefit from them.

7 Acknowledgements

The work has been conducted in the context of the research project Embedded Interaction ('Eingebettete Interaktion') and was funded by the DFG ('Deutsche Forschungsgemeinschaft').

We thank Nokia Corporation for providing us with three Nokia 770 Table PCs for our research.

We thank Particle Computer GmbH for providing us with rudimentary source code for the RCores.

References

[1] K. Andersen. Hacking wireless game-pads. Toolkit Support for Interaction in the Physical World on Pervasive 2004, April 2003.

[2] M. Beigl, A. Krohn, T. Zimmer, C. Decker, and P. Robinson. AwareCon: Situation Aware Context Communication. In *Proceedings of Ubicomp 2003*, Seattle, USA, October 2003.

[3] W. Burleson and T. Selker. Canopy climb: a rope interface. In *GRAPH '03: Proceedings of the SIGGRAPH 2003 conference on Sketches & applications*, pages 1–1, New York, NY, USA, 2003. ACM Press.

[4] W. Buxton. Living in augmented reality: Ubiquitous media and reactive environments. In *Video-Mediated Communication*, pages 363–384. Erlbaum, 1997.

[5] ChipCon. ChipCon 1010 Product Page. `http://www.chipcon.com/index.cfm?kat_id=2&subkat_id=12&dok_id=55`, visited March 2006, March 2006.

[6] C. Decker, A. Krohn, M. Beigl, and T. Zimmer. The particle computer system. In *IPSN Track on Sensor Platform, Tools and Design Methods for Networked Embedded Systems (SPOTS), Proceedings of the ACM/IEEE Fourth International Conference on Information Processing in Sensor Networks*, April 2005.

[7] Electronic Assembly. LCD-Module eDIP Product Page. `http://www.lcd-module.de/eng/dip/edip.htm`, visited March 2006, March 2006.

[8] Equator. Interdisciplinary research collaboration (irc). `www.equator.ac.uk`, 2006.

[9] H.-W. Gellersen, G. Kortuem, M. Beigl, and A. Schmidt. Physical Prototyping with Smart-Its. *IEEE Pervasive Computing Magazine*, 3(3):74–82, July–September 2004.

[10] M. Kranz, L. Moesenlechner, and D. Blank. Research Group Embedded Interaction - Project Page Nokia 770. `http://www.hcilab.org/projects/nokia770/nokia770.htm`, visited January 2006, January 2006.

[11] Lantronix. Lantronix WiPort Product Page. `http://www.lantronix.com/device-networking/embedded-device-servers/wiport.html`, visited March 2006, March 2006.

[12] Maemo. Maemo for Nokia 770 Table PC. `http://www.maemo.org`, visited January 2006.

[13] Particle Computer GmbH. RCore Product Page. `http://www.particle-computer.de/index.php?article_id=35&clang=0&ctype=1`, visited March 2006, March 2006.

[14] T. Prante, C. Röcker, N. Streitz, R. Stenzel, C. Magerkurth, D. van Alphen, and D. Plewe. Hello. Wall - Beyond Ambient Displays. Video and Adjunct Proceedings of UbiComp 2003, October 2003.

[15] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, and W. V. de Velde. Advanced interaction in context. In *First international conference on Handheld and Ubiquitous Computing HUC99*, pages 89–101. Springer LNCS, 1999.

[16] Siemens Forum (AR). Demonstration Aula Regia on the CybernariumDays. http://www.cybernarium.de/download/CDaysMuenchenKatalog.pdf, November 2004.

[17] Small Devices C Compiler. SDCC Homepage. http://sdcc.sourceforge.net/, visited March 2006, March 2006.