Design by Physical Composition for Complex Tangible User Interfaces

Tanja Döring
Pervasive Computing and User Interface Engineering Group
University of Duisburg-Essen
45117 Essen, Germany
tanja.doering@uni-due.de

Bastian Pfleging
Pervasive Computing and User Interface Engineering Group
University of Duisburg-Essen
45117 Essen, Germany
bastian.pfleging@uni-due.de

Christian Kray
School of Computing Science and Informatics Research Institute
Newcastle University
Newcastle Upon Tyne NE1 7RU, UK
c.kray@ncl.ac.uk

Albrecht Schmidt
Pervasive Computing and User Interface Engineering Group
University of Duisburg-Essen
45117 Essen, Germany
albrecht.schmidt@uni-due.de

CHI 2010, April 10–15, 2010, Atlanta, Georgia, USA.
ACM 978-1-60558-930-5/10/04.

Abstract
In this paper, we present a novel approach to create devices with tangible user interfaces by physical composition. While the separation of the user interface from the application logic has a long tradition in software engineering, for products with tangible user interfaces there is no equivalent approach that realizes a true separation and flexible combination of interface components, underlying technology, and software parts. We propose a novel concept that is based on an inner Core for the basic technical and software platform of a product and an outer Shell that builds a flexible and exchangeable tangible user interface from passive components. Using vision-based tracking, we can realize a clear separation between the components. No wiring is necessary. This paper introduces our novel approach and presents a first working prototype as well as initial results from its application in a design workshop.

Keywords
Tangible user interfaces, physical interfaces, evolutionary prototyping, physical prototyping, paper prototyping, 3D printing, dialog independence, smart products

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation: User Interfaces— Input devices and strategies, Standardiza-

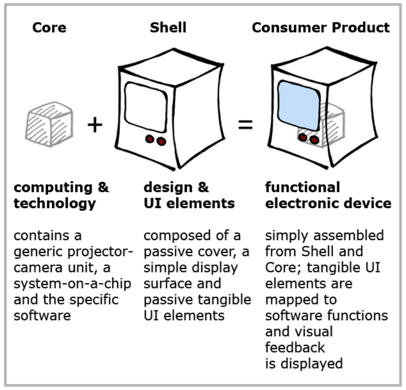

Figure 1: Composition of a working device with tangible user interface, e.g. an alarm clock or a media player, by assembling Core and Shell.

tion, Prototyping; J.7 Computer in Other Systems – Consumer Products

General Terms
Design, Human Factors

Introduction
In Software Engineering, the separation of the user interface (UI) from the application logic ("dialog independence") has a long tradition [3], and has facilitated the design and customization of software-based UIs (e.g. exchanging skins, colors, or profiles). In addition, this approach has enabled non-experts to more easily change UIs (e.g. via a graphical interface toolkit) according to their needs and preferences without having to change the functional portion of the software, i.e. without having to make modifications to the program/source of code of an application. The separation of the UI from the application logic can also improve the design process as UIs can be changed easily in response to user feedback at all stages of the development.

While this approach benefits the design of graphical user interfaces (GUIs), it does not directly translate to the domain of tangible user interfaces (TUIs). This is due to a number of factors such as the physicality of interface elements and the resulting limitations, e.g. the need for electrical connections or the difficulties of changing the shape and mapping of a physical control. Currently available toolkits in this area focus more on prototyping aspects rather than on the creation and adaptation of real, working products. In a sense, they therefore require programming and expert knowledge in order to adapt the UI, and it can be difficult to change the TUI once it has been 'built'.

In this paper, we present a novel approach for creating products with tangible user interfaces of arbitrary shapes and materials. It is based on a thorough separation of physical interface elements from the functional part of a device: an inner *Core* provides all the basic technical and software infrastructure of a smart product, and an outer *Shell* carries all the physical interface elements such as buttons and sliders (see Figure 1). The placement and type of all tangible interface elements on the Shell can be changed easily while the product as a whole remains fully functional. This approach enables customization, personalization and end user design. It provides flexibility and support for the design process of tangible UIs, which are on par with what is currently available for GUIs.

In the following, we present the concepts underlying our approach, a first implementation and initial results from a design workshop, where we evaluated the approach with end users and designers. The current system uses vision-based tracking to realize a clear separation between the components without wiring. The 15 participants of the workshop successfully used the system to design a simple device with tangible user interface (i.e. an alarm clock). Their overall reaction to the proposed approach was positive.

Related Work
Within the trend of ubiquitous computing, where computer devices are more and more integrated into the daily environment and thus into arbitrary objects of diverse shapes and materials, it is increasingly important to discuss the product design and development strategies in the area of tangible user interfaces [5]. The research community in tangible user interaction has already put a lot of effort into building toolkits that

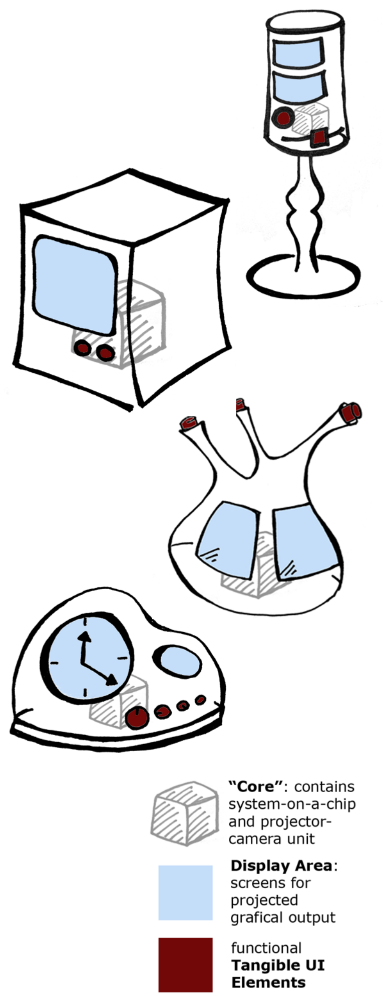

Figure 2: Sketched examples for possible products with tangible user interfaces, built by physical composition. All objects contain a generic Core and a separate Shell with display areas and functional tangible UI elements.

facilitate the creation of applications with tangible user interfaces (one example is phidgets [2]). As hardware devices are still more challenging to design than software components, toolkits for physical user interfaces are of great value. Thus, an important topic is the decoupling of hardware and software components. This has, for example, been addressed by Sankaran et al. [9] who presented "Blades and Tiles", a hardware toolkit for interaction researchers that offers means to integrate user-developed interaction modalities. Toolkits like "Blades and Tiles" realize a quite powerful tool for engineers, but also require at least some knowledge in wiring and programming. Another approach is BOXES [4], a toolkit that combines free exploration of form with interactive function. As such, it primarily addresses designers, who can use everyday non-interactive objects and shape their prototypes from cardboard or foam. To make the objects interactive, touch sensors need to be attached and the hardware has to be connected via a USB cable to a PC. Afterwards, it can be connected as input device to any GUI-application by mapping its input events to mouse events in certain on-screen areas. This toolkit provides great support for rapid and free form hardware prototyping. It is aimed at interaction designers rather than typical end-users and requires at least some knowledge in wiring and programming. Another shortcoming, which limits it to a prototyping toolkit, is the absence of a display on the hardware and the need to work with a separate computer instead of directly with the device. Approaches without wiring typically use vision-based tracking and example toolkits are PaperMaché [8] and DisplayObjects [1]. They mark great steps towards decoupling physical interfaces and underlying functions as well as the integration of both parts without the need for wiring. However, designers still need to setup a prototyping environment with the necessary infrastructure, e.g. cameras and a computer. As the implementation of real vision processing is still challenging, fiducial-based systems present a very useful approach to facilitate vision-based tracking, especially in prototyping environments. Among these, AR-Toolkit [7] and ReacTiVision [6] are two widely used toolkits, which can easily be included into any application. Projections for graphical output have been integrated into many tangible user interfaces prototypes, for example, as used within the DisplayObjects workbench [1].

Approach

In this section, we will explain the key aspects of our approach to enable end-users to design complex tangible user interfaces by physical composition in further detail. The key ideas underlying it are:

- Decoupling the functional Core and the user interface Shell
- Providing a generic technical Core that can be used to build any device, for which software exists that implements its functionality
- Allowing a flexible design and ad-hoc exchange of the UI Shell with arbitrary forms and materials
- Providing physical interface elements (e.g. knobs, sliders, wheels) as 3D models for self-printing or already fabricated with 3D printing technology
- Providing a variety of software modules for different kinds of devices or functional ranges

The central idea of our approach is a separation of the active, functional electronic component of an interactive device from its passive physical user interface. A ge-

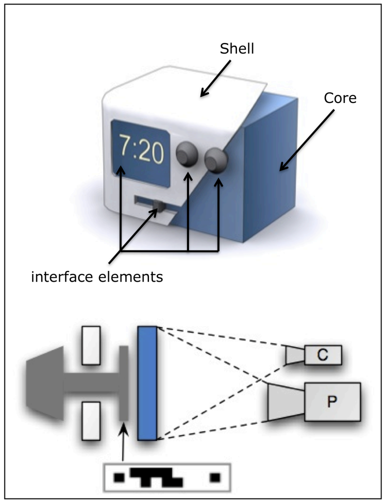

Figure 3: Example device with white cut-out Shell and blue Core (top), and side view (bottom) with dark-grey control, attached visual code, camera (C) and projector (P).

Figure 4: A first proof-of-concept toolkit to build an alarm clock with tangible user interface.

neric Core provides the actual functionality of the device. It consists of hardware and software components. The hardware includes a "system on a chip" and a camera-projector unit. We assume that in the near future, especially when specifically fabricated as one Core unit, the hardware will be suitable for even very small devices. Moreover, our concept envisions a camera-projector system with a functional radius of 180 degrees that defines the potential active area of the device. The micro projector uses laser technology in order to work properly on varying and even very short distances. This generic Core delivers the basic technology platform and can be (re-)used for unlimited numbers of devices. Besides the hardware technology it contains a visual tracking software that handles the mapping of interface components as well as the device-specific software. In order to create a fully functional device, a UI Shell has to be slipped onto the Core. The Shell defines the outer appearance of the device. It is passive and consists of an arbitrarily shaped carrier form as well as the physical interface controls that operate the device. The coupling between physical interface controls and various functionalities of the device is created through visual tracking, and therefore does not require any wiring or programming besides using a simple visual mapping configuration software which is running on the system itself. As the Shell functions as a projection screen, no other equipment or computer system is necessary to build, configure, design or run the device. Figure 2 shows schematically how exemplary products, built with our approach, could look like.

Our concept of decoupling the functional Core and the passive user interface Shell for products with tangible user interfaces leads to four basic implications:

(1) *Simplified hardware (ex-)changeability:* Construction kits with varying sets of UI elements could be delivered. In the future, UI elements could also be provided in form of software as 3D models and be (self-)fabricated with 3D printers (which are assumed to be cheaper and easier available in the future). This constitutes a step towards making hardware as easy (ex–)changeable as software.

(2) *Design by the consumer:* Consumers can easily choose, create, or redesign their preferred consumer device, regarding functionality, appearance and user interface. This concept gives users the options for easy customization and personalization of consumer devices, which so far has not been possible to this degree. Moreover, consumers could also transform one device into a different one through exchanging the Core software and newly coupling or changing their UI elements.

(3) *Sustainability:* The passive physical parts of a user interface of a device are exchangeable and furthermore reusable in different configurations. Moreover, also simple materials like cardboard or paper and other biodegradable materials can be used. This could lead to a better sustainability of consumer products, which so far has not yet been addressed properly.

(4) *Manufacturing:* Due to the separation into a generic technical Core, the software modules for different devices, and the physical UI components, each part could be produced independently. The generic technical Core could be mass-produced and benefit from an economy of scale, while many (small) companies could specialize in the production of specific software and interface components.

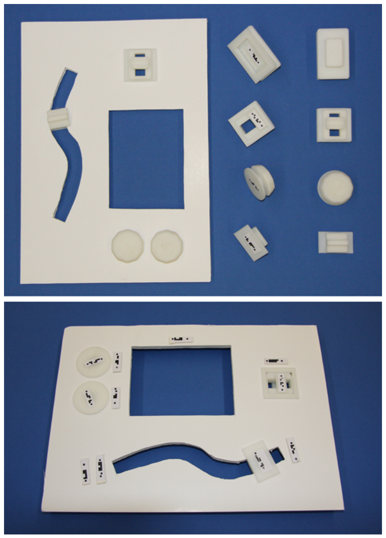

Figure 5: *Top*: Example of a designed front panel made of foam board and 3D printed controls.
Bottom: Backside of the front panel - markers either represent controls or functions.

Implementation

Figure 3 illustrates how we implemented the proof-of-concept version of our approach. The top part shows its main components: a (white) slide-on Shell with cut-outs to enable displays, where the (blue) Core is visible, and to attach (dark grey) physical controls. The bottom part of the figure shows a side view of the front panel. Physical controls are attached to the Shell by attaching a back part to the front part via a rod. The back part sits between the Shell and the Core and has a visual code attached to it. A camera (C) inside the Core can track this visual code and therefore detects when a user moves or turns a control. The reacTIVision framework [6], extended by a new marker detection engine, was used to track those markers visually. The new engine detects small-sized fiducials (17.5 mm x 3 mm) as shown in Figure 3 while still using a low camera resolution (640x480 pixels). The Core also houses a projector, which projects an image on the back plane of the housing of the Core and thereby displays the content onto cut out areas in the Shell.

For the initial toolkit version we built a square wooden box (see Figure 4), which contained a standard webcam and small projector. A removable front panel (made of 5mm thick foam board) served as a simplified version of the Shell. A transparent acrylic panel with a layer of tracing paper served as projection screen. Both camera and projector were connected to an external computer, which provided the required functionality. Later, all computing elements should be included in the system itself. We designed different physical controls (buttons, sliders, dials, and switches) and produced them using a 3D printer. Back and front part of controls could be physically clicked together and could be attached to the foam board (Shell) at any position. In our first version, parts of the board had to be cut out in order to stick UI elements to it and to design a projection area (see Figure 5, top). We used self-adhesive markers that could be stuck to UI elements to mark the position, orientation, and type of each interface control as well as to assign a certain function to it (see Figure 5, bottom). Thus, the toolkit could map controls to functions by using a proximity mapping: if a marker representing a desired function of the product was placed next to the corresponding interface control (identified by its marker), a link between the interface control and the function could be established by looking for adjacent pairs of function markers and interface control fiducials. As soon as the state (position/orientation) of an interface control changed, the interface toolkit generated an event describing the current state of the interface control and the assigned function. By sending this event to each registered module the different implemented functions could be executed. This facilitates the development and distribution of software modules, as the event handling is the only interface between a module and the toolkit. Thus, software modules can, e.g., be provided as downloads from the web along with a printable sheet of the different markers that are used by the specific module.

Initial Evaluation

In order to evaluate our approach with users, we organized a design workshop. We were interested in whether the approach was seen as useful and if designers and end users understood the underlying concept and could use the proposed method to create a working device. We recruited 15 participants aged between 20 and 60, males and females, including both people with and without a background in design. We wanted to find out, whether either group could deal with the concept

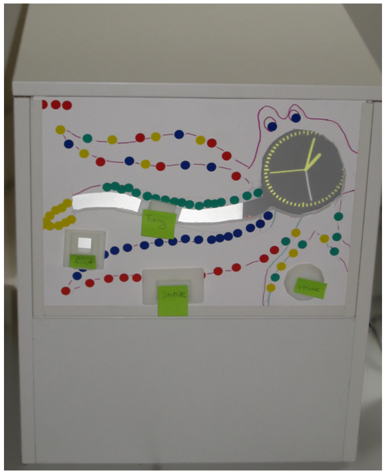

Figure 6: Example of a functional prototype of an alarm clock designed by end-users. It was built by physical composition with the proof-of-concept toolkit set. Future toolkit versions can enable end-users to build real and more complex products with tangible UI elements just by physical composition.

and whether our approach would enable a new type of dialogue between the two groups. Divided into four groups, each group was given the task to create different prototypes of alarm clocks with the functionality and appearance they like. Each participant was able to grasp the proposed concept and to use the system to assemble physical interfaces. Both designers and potential end-users were able to use the approach to build working prototypes. We also learned that the Shell material and the physical assembly process need to be easier to handle (i.e. the 5mm foam board used for the Shell was difficult to cut). Overall, the majority of the participants preferred the physical prototyping method for designing the alarm clock over pure paper prototyping. The main reasons given for this decision were that participants liked that they could 'see it work' immediately and that they 'got a feel for it'. Figure 6 shows an example of the created physical prototypes.

Conclusion

In this paper, we introduced an approach for the separation of an active and functional Core component and a passive Shell. This clear separation offers a set of interesting properties with regard to the design process, manufacturing, and sustainability. A prototypical system that uses a camera/projector system as the Core and 3D-printouts for the Shell demonstrates the feasibility and utility of the approach. The initial evaluation with designers and end-users in the context of a workshop shows that the approach is understandable and can provide additional value during the design phase. Future work will include the further development of software components (e.g., core modules with functionalities for different devices, further facilitation of graphical output customization), the identification and construction of fundamental physical building blocks, the exploration of form factors and different materials as well as the application of our approach in future design workshops.

References

[1] Akaoka, E., Ginn, T., and Vertegaal, R. DisplayObjects: Prototyping functional physical interfaces on 3D styrofoam, paper or cardboard models. In *Proc. of TEI '10*. ACM (2010), 49-56.

[2] Greenberg, S. and Fitchett, C. Phidgets: easy development of physical interfaces through physical widgets. In *Proc. of UIST '01*. ACM (2001), 209–218.

[3] Hill, R. D. The abstraction-link-view paradigm: using constraints to connect user interfaces to applications. In *Proc. of CHI '92*. ACM (1992), 335-342.

[4] Hudson, S. E. and Mankoff, J. Rapid construction of functioning physical interfaces from cardboard, In *Proc. of UIST '06*. ACM (2006), 289–298.

[5] Ishii, H. and Ullmer, B. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proc. of CHI '97*. ACM (1997), 234–241.

[6] Kaltenbrunner, M. and Bencina, R. reacTIVision: a computer-vision framework for table-based tangible interaction. In *Proc. of TEI '07*. ACM (2007), 69–74.

[7] Kato, H., Billinghurst, M. Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In *Proc. of IWAR '99*. San Francisco, USA (1999).

[8] Klemmer, S. R., Li, J., Lin, J., and Landay, J. A. Papier-Mache: toolkit support for tangible input. In *Proc. of CHI '04*. ACM (2004), 399–406.

[9] Sankaran, R., Ullmer, B., Ramanujam, J., Kallakuri, K., Jandhyala. S., Toole, C., and Laan, C. Decoupling interaction hardware design using libraries of reusable electronics. In *Proc. of TEI '09*. ACM (2009), 331–337.