Designer driven interaction toolkits

Walter A. Aprile
IO-Studiolab
Industrial Design Engineering
Delft University of Technology
Landbergstraat 15
2628CE Delft
Netherlands
w.a.aprile@tudelft.nl

Aadjan van der Helm
IO-Studiolab
Industrial Design Engineering
Delft University of Technology
Landbergstraat 15
2628CE Delft
Netherlands
a.j.c.vanderhelm@tudelft.nl

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

Abstract
Most interaction toolkits, historically, have been designed by technologists for designers. In this workshop paper we explore how designers could design such a toolkit for themselves, following their own methods and focusing on their disciplinary concerns.

Keywords
Interaction, design, tangible interaction, user-driven design, hacking, prototyping, Arduino, Max/MSP

ACM Classification Keywords
D.2.2 Design Tools and Techniques: *Evolutionary prototyping*, *User Interfaces*

General Terms
Workshop paper, Work in progress

Introduction
Interaction toolkits are software and hardware toolkits that enable designer to prototype and develop interactive systems. Notable examples include Arduino[8], a C-based microcontroller development system, Processing [11] and [12], a Java derived development environment focused on interactive graphics and Max/MSP, a commercial visual programming language for multimedia with over twenty years of evolution behind itself. Like Arduino, Processing and MAX, most interaction toolkits have

been designed and developed either by technologists working in close collaboration with designers, or by researchers/practitioners with a background that spans computers and design. Moreover, the developers of these toolkits are frequently engaged in designer education. We wanted to experiment with an interaction toolkit for designers and *by* designers. Design MSc students with no particular expertise in the field have been let loose into what was, for them, unexplored land. The results are certainly different from technology (and technologist) driven interaction toolkits.

Context

The course "Interactive Technology Design" introduces design students to interactive prototyping and its tools. Sketches and prototypes are used initially for inspiration and, towards the end of the course, for involving users. The prototypes are exhibited at the end of the course for a combination of show and user testing under fairly realistic condition.

Two student teams were given as a brief "*explore, on the basis of your experience with interactive sketching tools, a designer-driven design for an interactive sketching tool*". Students were given a minimal bibliography including [4], , roBlocks [13] , [7] (shared Phidgets), iStuff [10] and [1] on sketching user experiences. It was suggested to think of toolkits in the context of practices [14] which, in this case, were the design students own practices.

Students were asked to proceed through five iteration steps, each one of which offered the chance to broaden the field of inquiry. The steps took up progressively more time, as the students committed to certain elements of their design and decided which directions were most promising. Moreover the students were also encouraged to explore toolkits that do not particularly focus on interaction, like the ones made by Bosch, Lego, Fischer Technik. Students were encouraged to think of two intentionally vague use cases: designing a lamp and designing an alarm clock. The students were also introduced in detail to the Arduino system and to Max/MSP.

Process

The context of this work is the field of Design, not HCI or Computer Science. Rigor in Design takes different forms than in science and engineering: fundamental concerns differ. For example, Turing-completeness would be an important concern from a computer science point of view, while from a design point of view it could just be a desirable feature. Again, from an HCI perspective, extensive user testing and design validation is key, while design projects frequently lack resources or –to be honest- focus and interest towards this type of activities.

On the other hand, Design is always very concerned about the appearance of its manifestations. Choices of palette and typography, material feel and texture, and more generally "form" get a great deal of attention. Such choices, speaking again in a very general way, can hardly be said to be a key concern of practitioners of CS or HCI.

Moreover, and we are aware that this is a very broad generalization, Design practitioners tend to focus on complete systems (as opposed to specific technologies or interface elements or techniques). Complete systems of the scale we develop tend to be very difficult to

study comparatively. Such systems tend to be evaluated on the merits of their aesthetic consistence, their innovativeness for the design field and their conceptual strength.

The expression "designerly way of knowing" is due to Nigel Cross [2] and [3], and it is sustained by the assumption that there is a third way of knowing, different from the way of the natural sciences and the way of the humanities. The process followed in the design of Atreyu and Sketchonary has been thoroughly designerly, in that it has aimed to broaden the field of inquiry to human, cultural and technological issues through aggressive iteration, converging finally on designs that cannot be justified scientifically but can be defended in the design domain.

Results

We describe here the two systems we have designed. They stand in almost direct contrast one to one, as Atreyu is a purely physical TUI without any element of GUI, while Sketchonary uses video-tracking and retro-projection to augment physical tokens with computational behaviors.

Atreyu

The Atreyu system appears as a grid of equilateral triangles. The individual triangle can be connected an all sides to its neighbors. Each triangle forms a node of an improvised network that transfers power and data. Triangles can host IO modules (white) or act as simple passive bridges (black). One special red triangle injects power in the network and stores the current configuration.

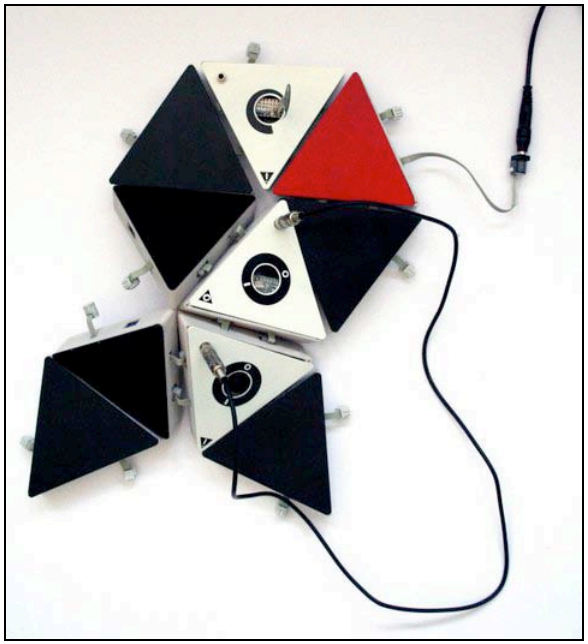

Figure 1: A small Atreyu network. The black wire temporarily connects two IO modules to establish a relationship between them. Black modules are passive.

Relationships between inputs and outputs are established by temporarily connecting two modules with a patch cable inserted into a socket on the top face of the triangle. Each black triangle contains an Arduino Pro Mini, running the I2C protocol for connectivity. The Arduino is mounted on a custom PCB contained in a custom, 3D printed case.

In the current design, the Atreyu system is limited to simple proportional relationships between triangles: turn a knob more, and the light goes up, shake harder the vibration sensor and an LED blinks faster. Although

the design specifies also processing and control modules, such as timers and adders, these modules have not been realized during the course.

Difficulties were encountered at the physical level. In particular, choosing a reliable, cheap and fast connector system proved very challenging. The chosen system, due to its ease of assembly and low price was the RJ14 4-wire telephony jack.

Atreyu shares with [9] on siftables the interest for "[exploring] how sensor network technologies might be combined with features from the GUI and TUI in order to further increase the directness of a user's interaction with digital information or media". In the Siftables system, though, the individual interaction element is a rather powerful computer with advanced communication and computation capabilities, while in Atreyu the basic triangle is a rather simple object. Additionally there is a strong difference in purpose: Siftable objects stand for (or are?) information objects (photographs are given as examples) to be sorted and organized, while Atreyu objects are computational network and sensor objects for exploring interaction.

A design experience similar to Atreyu, although on a very different physical scale, is Tanaka and Kaito's I/O-CRATE, box-shaped modular units featuring embedded microprocessors, sensors, actuators and batteries. I/O-CRATEs are modular, stackable and built out of beer crates for solidity. Their purpose is to build public interactive experiences for social locations or events. The programming language shown in [15] is also limited to simple action-reaction patterns, where something happens when a sensor reading satisfies certain instantaneous conditions.

While we can safely say that Atreyu fell short of the brief –it will take a lot of conceptual and technology work to make it go beyond simple input-output proportionality- it points in a direction that gives us hope for the future from a design perspective.

Positive user feedback during limited testing with designers and non-designers at the course final exhibition allows us to think that further work on this – or a similar - system could be justified.

Sketchonary

Sketchonary is a tabletop interface that enables multiuser sketching and augments it with a layer of information, real time rendering of input/output interaction. Sketchonary can compile sketches to Arduino code and upload them immediately to a board.

Sketchonary shares with Fritzing [5] a strong focus on designers as main users who are not necessarily familiar with engineering notation and thinking. The declared objective of Sketchonary is to enable the exploration of ideas and reducing slowdowns due to lack of knowledge by the designer. The system follows Buxton's definition of sketching as questioning, exploring and suggesting what could be, rather than refining what should be [1]. For this reason, it was decided that the user should remain at a high level of abstraction and not engage with relatively low-level tools (for example, a C source code editor). In this vein, Sketchonary focuses on playful interaction without error messages but with happy accidents and multiuser interaction.

The interface of Sketchonary consists of a table surface where electronic components, drawing materials and

fiducials can be stored. The table surface contains a back-projected sketching surface where the fiducials can be placed.

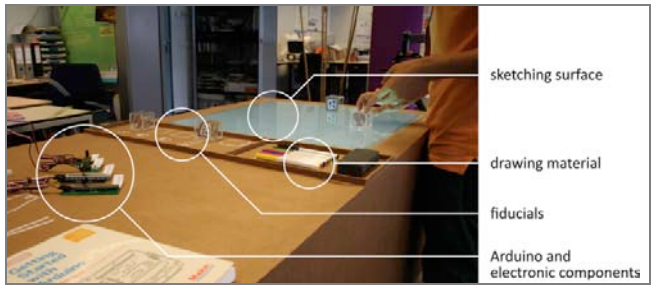

Figure 2: the Sketchonary table

Fiducial symbols applied on transparent tiles are used for most of the interaction. The set of Input fiducials comprises a manipulator box, and four input tiles; horizontal movement, vertical movement, proximity, rotation. A Timer fiducial contains the typical functions of a wake-up alarm. The output is currently limited to an RGB LED that can be controlled along five dimensions, namely RGB balance, brightness and blinking frequency.

Sketchonary can be either in Dictionary mode, where each fiducial triggers the appearance of contextually relevant on-screen documentation, or in Sketching mode, where the user defines interactive behaviors. The Mode fiducial toggles the interface between Sketching mode and Dictionary mode. Users are encouraged to sketch on the screen with whiteboard markers and place fiducials representing input and output on their sketch.

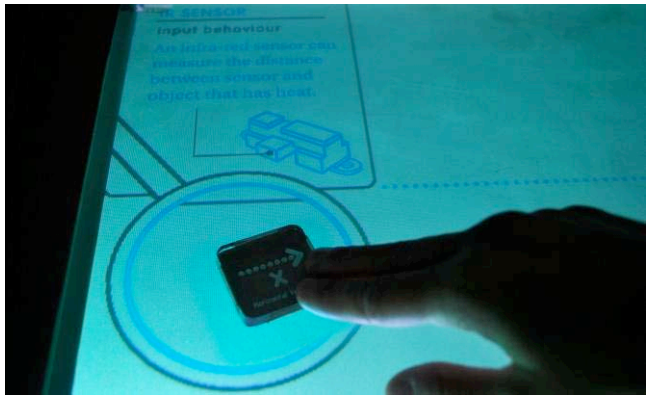

Figure 3: detail of Sketchonary in Dictionary mode.

Once the fiducials are on the screen, projected attachment points on their sides allow the user to visually snap them together and decide what input controls what output. To simulate input variation on screen, a special manipulator box is used.

Landay's pioneering work [6] is an influence on Sketchionary, mostly through its insistence on rough improvisational sketching as an early phase of design, although it remains to be said that Landay's tool, and it commercial successor SketchFlow, concentrate on GUI design while the proposed use cases for the projects we discuss in this paper are more in the physical domain.

Conclusion

Conclusions, for works in progress, are inescapably provisional. Both the Atreyu and Sketchonary team chose to forgo generality and narrowly focus on one use case. The interfaces have been developed to a level that allows limited and informal user testing, enough to decide how to take the projects on.

From an engineering and scientific point of view it can be argued that, due to the design teams near total lack of previous experience with interactive prototyping or electronics, the designers retreated on the safer ground of detailing, materials and texture, while developing insufficiently the computational framework and, more in general, being a bit vague with the technology.

From a design point of view, many things were "gotten right": aesthetic consistency, a fluid interaction style, style, desirability and some irony in the presentation of the projects are something to be encouraged.

The question of how to apply user-driven design when the users are designers is still open. We want to take this work further, either by developing Sketchonary and Atreyu or by giving the same brief to more students.

Acknowledgements

We thank the following students for their significant contribution to this work: Ainhoa Ostolaza, Thijs Waardenburg, Palma Fontana, Alice Mela, Robert Paauwe, Simone Rebaudengo, Karla Rosales, Enrica Masi, Myeongsoo Shin and Kim Sohyun.

Bibliography

[1] Buxton, B. (2007) *Sketching user experiences: getting the design right and the right design*. San Francisco, CA, Morgan Kaufmann.

[2] Cross, N. (1982) Designerly ways of knowing. *Design studies*; 3(4), pp. 221-227.

[3] Cross, N. (2007) Forty years of design research. *Design studies*; 28(1), pp. 1-4.

[4] Greenberg, S. (2007) Toolkits and interface creativity. *Multimedia Tools and Applications*; 32(2), pp. 139-159.

[5] Knörig, A., Wettach, R. and Cohen, J. (2009) Fritzing: a tool for advancing electronic prototyping for designers.

[6] Landay, J. A. (1996) SILK: sketching interfaces like krazy. Conference companion on Human factors in computing systems: common ground. Vancouver, British Columbia, Canada, ACM.

[7] Marquardt, N.Greenberg, S. (2007) Shared Phidgets: A Toolkit for Rapidly Prototyping Distributed Physical User Interfaces. *Proceedings of Tangible and Embedded Interaction (TEI)*, pp. 13-20.

[8] Mellis, D. A., Banzi, M., Cuartielles, D. and Igoe, T. (2007) Arduino: An Open Electronics Prototyping Platform. *CHI 2007*, San Josè, USA, April 28 - May 3, 2007.

[9] Merrill, D., Kalanithi, J. and Maes, P. (2007) Siftables: towards sensor network user interfaces. Proceedings of the 1st international conference on Tangible and embedded interaction. Baton Rouge, Louisiana, ACM.

[10] Rafael, B., Meredith, R., Maureen, S. and Jan, B. (2003) iStuff: a physical user interface toolkit for ubiquitous computing environments. Proceedings of the SIGCHI conference on Human factors in computing systems. Ft. Lauderdale, Florida, USA, ACM.

[11] Reas, C.Fry, B. (2003) Processing: a learning environment for creating interactive Web graphics.

[12] Reas, C.Fry, B. (2007) *Processing: a programming handbook for visual designers and artists*, The MIT Press.

[13] Schweikardt, E.Gross, M. D. (2008) The robot is the program: interacting with roBlocks. TEI 2008: Second International Conference on Tangible and Embedded Interaction. Bonn, Germany, ACM; pp. 167-168.

[14] Shove, E., Watson, M., Hand, M. and Ingram, J. (2007) *The Design of Everyday Life*. Oxford, Berg.

[15] Tanaka, H.Kaito, S. (2007) Universal modular kit for temporal interactive place in public spaces.