Finding Design Qualities in a Tangible Programming Space

Ylva Fernaeus & Jakob Tholander
Department of Computer and Systems Sciences, Stockholm University
Forum 100, 164 40 Kista, Sweden
{ylva, jakobth}@dsv.su.se

ABSTRACT
We reflect upon the process of developing a tangible space for children's collaborative construction of screen-based systems. As in all design work, the design process involved continual refinements of initial ideas and their practical realisation. We discuss how some widely held qualities often put forward with tangible interfaces were given up in favour of reaching overall goals of interaction. In particular our design involved a shift from a focus on persistent representation and readability of tangible code structures, to instead focus on achieving reusability of programming resources. On a general level, our results illustrate a view on tangibles as resources for action instead of only as alternative forms of data representation. Importantly, this view includes action directed towards the computer as well as off-line socially oriented action conducted with the tangible artefacts.

Author Keywords
Tangible programming, TUI, embodied interaction

ACM Classification Keywords
H.5.2: User Interfaces.

INTRODUCTION
There is much current research with the overall goal to develop a richer understanding of how new programming tools could be designed for, and used by, children [9, 12, 14, 23, 29, 31, 36]. One specific challenge in this area is to support children's *collaborative construction of screen-based systems*. While a computer screen allows for objects to be copied, to animate, and to respond to actions, a physical space affords hands-on manipulation, physical sharing of resources, and parallel activities with everyday artefacts and materials. Thus, this work relates to a general challenge often addressed in tangible interaction, where the goal is to bridge the "gap" between physical and virtual forms of expression [13, 32].

CHI 2006, April 22–27, 2006, Montréal, Québec, Canada.

In parallel with these efforts, research in human-computer interaction has recently become conceptualised as a design and practice oriented field of study [15]. Designing with digital media has been characterised as dealing with a "split world" of physical sketching and manipulation on the one hand, and screen-based interactions on the other [20], and in addressing this a large number of methodological developments have been pursued. Examples of work in this area includes computationally enhanced physical cards to use with recorded video material, wizard of oz studies, body storming, low-fidelity prototyping, and role-playing games.

We contribute to this work by providing a methodological example of developing a tangible programming space, allowing children (age 6-12) to collaboratively create their own dynamic play-worlds to run on a computer screen (see Figure 1). The system has been evaluated through workshop activities in an art gallery [12] highlighting how the physical resources were appropriated for developing a sense of a shared and focused activity, including social play as well as construction of dynamic and interactive systems. The space is currently being set up at the Swedish Museum of Natural History in Stockholm, where it is planned to be taken into broader use by visiting school classes.

Through examples and sketches, we discuss how the system was given its resulting form and functionality, and consider some specific analytical outcomes that could be derived from this process. Thereafter we analyse our findings in terms of four concepts often discussed in tangible user interfaces: "coupling", "manipulation", the

Figure 1. The tangible programming space in use

notions of "input" and "output", and physical artefacts as representations of digital information. Our analysis specifically emphasises the relation between physical manipulation and digital representation, and how to make efficient use of these two in combination. We point to the significance of considering this relationship and its consequences in terms of what users can accomplish through using the system.

BACKGROUND

In designing interactive systems for children a significant challenge is to address ways of integrating technology with children's social, cultural and physical circumstances [8]. A common way of addressing these issues is through novel technologies that allow for a richer range of social interactions than is possible in ordinary PC settings [5, 28]. Examples of systems include rooms for collaborative storytelling activities (such as StoryMat [28], KidStory [23], and Pogo [6]), and tools for manipulation and control of robotic or programmable toys (e.g. CurlyBot [14], System Blocks [36], and Topobo [26]). This trend is also reflected in a range of commercial systems targeted at children, such as interactive dance mats and robotic construction kits.

In new interfaces for programming, collaborative aspects have been addressed through tools designed for groups working together using networked computer stations [31], as well as through a range of tangible forms of interaction [9, 22, 23, 29, 35]. With tangible programming tools, users are able to define computational actions by manipulating physical objects that represent functions, objects and relations in program. An important focus in this area is thus to explore how programming structures and operations can be represented in tangible form [22, 26]. Only a subset of all approaches to tangible programming [e.g.29] concerns tools to support interaction with on-screen materials.

The work presented here is practically grounded in several years of research where children make use of extended versions of the ToonTalk programming environment. The interface of ToonTalk comprises an on-screen "world" in which a number of resources are provided to support children in designing their own interactive systems. This includes tools for copying, moving or changing the size of pictures, and libraries of programming behaviours. Such behaviours could be added to pictures and other objects to make them perform specific actions in a game or dynamic illustration, for instance to move in a direction, to play a sound, or to disappear upon collision with other objects.

In our studies of children using ToonTalk to build their own games, the children usually developed the visual content and much of the narratives of their games away from the computer setting. This allowed them to make use of a range of design materials and tools that they were already familiar with, such as textiles, plastics, paper, and modelling clay. The objects produced in such sessions were then photographed, trimmed, and imported to the visual programming environment, in which dynamic and interactive properties could be created. [10, 30]

Analyses of children interacting in these and similar settings have identified a need to bridge the characteristics of offline design activities with programming oriented activities performed on the computer. Activities around the physical materials often become less structured, but with more children involved at the same time and with more discussions regarding the overall functionality of the systems, while the activities "on" the computer more have a characteristic of turn-taking and negotiations regarding detailed interface manipulations [30].

Moreover, the very different characteristics of current programming tools and low-fidelity material make it difficult for children to move back and forth between the two kinds of resources. Our previous efforts have concerned the development of resources and activities for children to more actively interlink programming and offline design activities. Examples are to encourage the children to iteratively go back to the low-fidelity material to create more objects, to design physical representations of programming resources, and to perform activities where groups of children collaboratively enact ongoing programming projects offline. The purpose with such activities is to support children in grounding their ideas in the available resources for programming, and in doing this increasing the chances of turning their efforts into working games and simulations.

Based on these observations we find it essential to design new tools that more naturally link the offline and online design activities that children engage in.

DESIGN PROCESS

The research process accounted for here was based on qualitative analyses of design prototypes and of staged activities with children using these. This is in many ways similar to participatory design approaches to research, which in recent years have become dominant also in studies within the field of interaction design and children [8]. However, instead of involving the children explicitly as informers to the design process, our approach to user involvement is to stage activities [16] aiming to engage children in productive use of technology. From critical reflection of patterns of use and of the artefacts developed, we have further developed and refined our designs. In our analytical work we have extensively used interaction analysis based on video recorded material.

The development process will be reflected upon and analysed through the following activities (1) specifying the overall goals for interaction and activity, (2) analysing

staged activities with low-fidelity prototypes, (3) exploration of hi-fi prototypes built with tangible technologies, and (4) analysis of the final design of the system.

Overall goals for interaction and activity

Our initial ambition for the activity aimed at was that designing, building and playing should be performed together and with a shared set of resources, in a similar fashion as when children design and play with more conventional lo-fidelity materials. Important properties of such settings are that they allow for action and interaction to be performed concurrently, and that physical manipulation of resources can be conducted jointly as well as individually. This balance between individual and collective activity in collaboration around technology has been emphasised for instance by [17].

Another important design consideration was to build on children's experience and interest in screen-based systems. Much can happen on a computer screen that is not possible in the physical space; things can deny the laws of physics by animating in irrational ways, objects may change shape and size, they may be copied, deleted, modifications can be undone, and much more. Thus, the constructions that the children create should be dynamic in a sense that is not possible to achieve with more linear forms of media, e.g. sequential video or animation.

The programming model aimed at was the mode of construction employed when children make programs using behaviours in ToonTalk. This is similar to the object-oriented approach to programming, where visual objects can be made to inherit properties of other objects and where readily prepared classes of actions can be dynamically combined. Core to this idea was to allow for object-instance relations between physical and virtual resources, and in doing this moving the programming activity away from the PC setting.

The system should include a construction mode and an execution mode. In execution mode the objects on the screen should start acting according to behaviours and properties attached with them through manipulations of physical tools in construction mode. It should also be possible to load and to save existing game configurations. Further, since our ambition was for the system to be usable in everyday settings such as in homes and in schools, we wanted its physical parts to be mobile and possible to store away when not in use.

Staged activities with low-fidelity materials

In working towards identifying the tangible properties that the system would entail, a range of materials related to programming and design were explored, including paper prototyping, modelling clay, specific play tools built by us, Lego RCX robots, and traditional Lego kits. This work was conducted primarily through workshops with groups of children, but also through hands on physical sketching in our research group.

With help from children engaged in programming in ToonTalk, a variety of approaches to tangible programming was explored, all illustrating different aspects of how a system could be represented in tangible form. The tangible resources were modelled after the set of behaviours, properties and other tools that the children had available when programming on the computer. Figure 2 shows a selection of the materials including dice, bottles, cups, plastic cards, sticks, etc., developed and used with groups of children.

These resources were taken into use in activities for debugging and role-playing existing programs or systems that they were currently being developed on the computer. In these activities the children collaboratively act out the workings of the different parts of a system as a way of discussing its functionality. This is very similar to standard techniques of low-fidelity prototyping [27], only that the focus was here explicitly on programming and implementation, rather than on general issues related to interface design. The activities normally start out from games and systems running on the computer, and during the activities, the systems often get "reprogrammed" in several variations [11].

The key question explored through these activities was which of the programming constructs that should take physical form in the tangible system. As has been noted by Stanton et. al. [28], apart from being useful resources

Figure 2. Tangible sketches for exploring and building computational systems (far left) and staged activities with these.

for collaboration amongst children, tangibles normally also slow down the overall pace of interaction. It was therefore important to consider what in the interaction that the children would focus upon. Central to this was therefore to not only find a set of constructs that *can* be given a physical form, but more importantly to find a level of computational detail that can be manipulated and understood by a group of children building dynamic and interactive systems. Thus these explorations concerned physical representation of higher level constructs, such as adding of behaviours, to lower-level issues such as message passing, changes of variables, iteration, and selection.

Design considerations based on the activities

Children's interaction with the different kinds of resource showed how tangible tools made to represent lower-level functionality led the children into discussions concerning the inner workings of computers and computational systems. On the other hand, collaborative and collective construction was more prominent in the activity when the physical objects represented higher-level relations and functionality, allowing for discussions directly connected to observable properties of the systems that the children were working with. Our analyses also emphasised how the shape of physical objects such as cards and other artefacts, as well as the social setting and physical positioning became important resources in children's sense making practices.

The activities had been performed in different settings, and we saw several important qualities of the activity especially when taking place on a floor surface. When working on a tabletop surface, the situation becomes structured towards people sitting on chairs, with a consequence that positions often become static. A floor on the other hand affords people to rearrange themselves more freely, thus allowing for spreading of the activity in more dynamic ways. This was important also since we saw that the possibility for bodily engagement contributed to increased social interactions. As noted by many others [e.g.3, 8], a central quality of situating an activity on the floor is that it is a common arena for children's everyday play, thus allowing the activities to work alongside other activities. Moreover, such a setting provides a room-sized interaction surface allowing for a large group of children to be involved in the activity.

Based on this analysis we decided to aim for designing a physical space where the activity should take place on an *interactive surface on the floor* where children could build their own dynamic fantasy worlds. The physical resources for programming should be based on higher level constructs for programming pictures on the screen. Thus, the physical space should provide physical objects representing *pictures* on screen, as well as objects representing *behaviours and properties*. By combining these on the floor surface, it should be possible to program animated systems to run on the screen.

Explorations of hi-fi prototypes

A challenge when designing tangible interfaces is to identify the specific hardware technologies for the system to be based upon. The system should ideally be built with existing component-based technology, which had proven reliable in other settings. Therefore, a number of existing tangible and sensor technologies were reviewed and tested. We primarily explored different kinds of wireless identification technologies (iButtons[1], RFID[2], Barcodes) in combination with technologies such as multiple mice and mobile phones. Below is a short description of the specific design considerations arising from the process of working hands on with building and testing of high-fidelity tangible prototypes.

One early experiment concerned how to make use of iButton technology to design the interaction for associating behaviours to pictures on the screen. IButtons are small identification devices that can be physically attached to slots on an object connected to a computer (see rightmost picture in Figure 3). A solution based on this technology appeared promising, especially since it could be grounded on the principles of tokens+constraints [33], which is a well documented approach that had

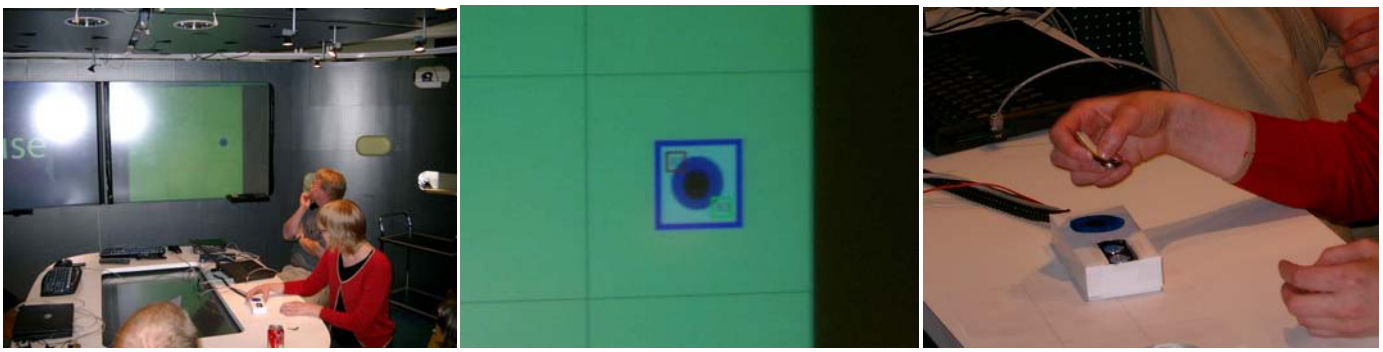

Figure 3. A tangible prototype explored in laboratory setting.

proven useful in many other tangible applications (e.g. in Algoblocks [29], and Bishop's marble answering machine). In the specific design that we explored, each picture on the screen would have a corresponding tangible object in the physical space, equipped with a number of slots on its top, into which physical tokens (iButtons) would fit (see Figure 3). Each token corresponds to a programming behaviour, so that when the programmed system is running, the pictures on the screen acts according to the tokens physically connected to its corresponding physical object. The mode of construction possible with this setup seemed intuitive in terms understanding how to manipulate its physical parts. A programmed picture would thereby also have a concrete persistence through its physical manifestation, and could for instance be stored or physically brought to other contexts of interaction.

An alternative approach based on RFID-technology, was to group physical objects on a grid of RFID-readers (see Figure 4). This idea was illustrated through the use of plastic pockets on the floor surface where tagged pictures and behaviour cards could be stacked together. Users would pick picture cards and place them in a pocket corresponding to where they should be on the screen. They then place properties or behaviour cards on or under the elements already added. The on-screen representation would continuously show which cards have been placed where on the floor, meaning that while you may only see the topmost card in the physical space, the screen displays a visual indication of all the cards (picture + behaviours) on each position. Similar to the first approach, an important property of this design was that it allows users to *"read" the code* of a programmed system through inspecting the physical space as well as through looking at the on-screen representation.

Analysis of tangible prototypes

Through experimenting hands-on with the tangible prototypes we found that the close mapping between the physical and virtual display made it easy for a user to see how the two sets of representations were connected and

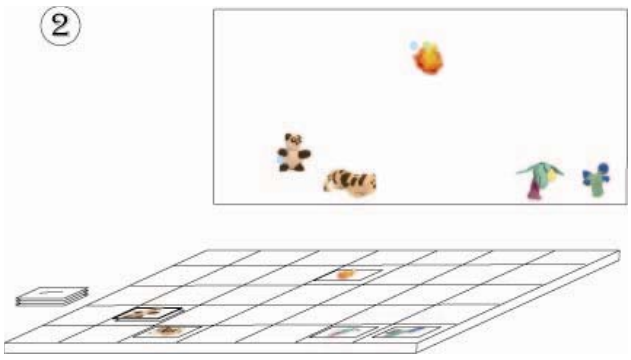

Figure 4. A prototype sketch with a grid of RFID-readers.

related. However, the ease of understanding came at a cost of what could be built with the systems. In terms of computational expression, these systems provided for no more "programming" than early attempts, for instance the Zowie [25], or digital desk [34] systems, where the digital display is used more as a computationally enhanced version of configurations in the physical space.

As an example, neither of these approaches allowed for the basic action of making several copies of the same computational object. In a computational system, one would expect to be able to make for instance a forest with many trees, though there would be just one picture of a tree. It is also likely that users want to add a specific programming behaviour to many different pictures. Though this could be addressed by making many copies of each physical object, building systems with large amounts of objects and behaviours would require an unmanageable number of tangible resources. Moreover, even though the screen would reflect the state of the tangible tools when in "programming" mode, the layout of the physical and the on screen representations would instantly diverge at run-time.

This was a critical drawback of both these first two design alternative, since maintaining rich computational expressiveness was one of the primary reasons for allowing users to build systems to run on the screen. Limitations of physical space should not constrain what users could build on the screen. Moreover, the system should be for programming and modelling, which we realised is something beyond having a dynamic representation on screen of what statically exists in another form of representation. If code structures are to take form in the physical space, they will simply not be possible to use more than once in the same system.

A direct design consequence of this analysis was to find a way of *making the physical resources reusable*, meaning that each physical object must be allowed to become logically connected to several screen positions at the same time. This specific issue was initially addressed through approaches that let everything placed on the floor become automatically added to the display, so that moving a physical object to a new position on the floor would make a new copy of the object on the display. To remove something from the screen, one would have to use a physical "eraser" object at the position of the object to be removed. However, since no representation of the constructions on the screen would be physically available on the floor surface, this would require the elements added to the screen display to be somehow displayed onto the floor surface. While this could become a possible solution, we saw many problematic issues in interaction with interfaces displayed on floor surfaces such as shadowing and mobility. We were also concerned that such a setup would confuse the users regarding the

differences between physical and digital surfaces and resources. Moreover, since the physical objects would be sensed by the sensors anywhere on the floor surface, resources not used would need to be placed off the mat. We saw this as a major drawback of interaction, especially since for the physical resources to be ready-to-hand for children using the system these should ideally be available for use also on floor surface upon which the children would sit.

A consequence of this was that we had to give up some of the qualities afforded by the physical material, including persistency of code, readability of code in the physical space, and mapping between physical and virtual representations. Instead of the physical and the virtual attempting to work as "mirrors", it did in this case become more important that the two forms of representations complement one another through the different kinds of resources and information that is possible to provide in each of them.

THE TANGIBLE PROGRAMMING SPACE

The design presented here should not be understood as *the* solution to our initial design challenge, but one that has shown to successfully meet the criteria that we had set up for the activity, and which is also directly grounded on the results from our previous design activities.

A core feature of this setup was the addition of an extra layer of interaction as intermediary between the mat and the tangible objects. The purpose with this was to allow for reusing of physical resources, and also to make it more explicit to users how the virtual and the physical forms of representations are separated. Instead of placing physical pictures and behaviour objects directly on the mat, these would now be placed on top of a wireless RFID-reader that in turn is located on the floor surface. Visual mapping is then between the readers and the screen, rather than directly between card-screen. To add a picture to the screen, a reader first needs to be placed on the mat and thereafter used together with a physical picture.

Figure 5 shows the physical setup of the system, consisting of (1) a large white plastic mat with a grid of identifiable position tags underneath, (2) a set of plastic programming cards, (3) several tangible creator blocks with an RFID-reader that is wirelessly connected to the software on the computer, and (4) a visual display showing the system that is being built.

The *creator blocks* connect the physical system with its virtual representation. When users interact with the system, they add objects and behaviours to the on-the screen representation by placing cards on top of the creator blocks. A rectangle moves on the screen in correspondence to how the creator block is moved on the mat. The position of the creator blocks and the id of the

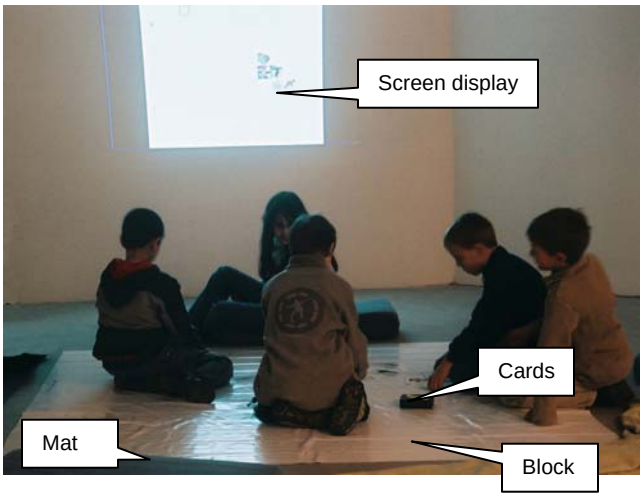

Figure 7. The setup of the programming space.

Figure 7. Programming cards: Picture and behaviour cards placed on the creator bocks on top of the mat.

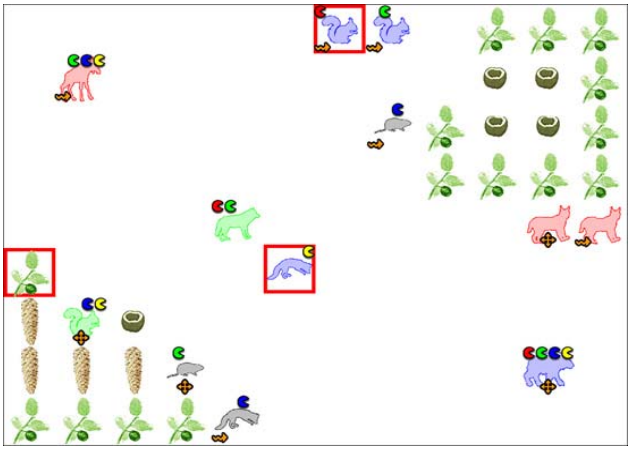

Figure 7. The display when in programming mode. Red squares indicate positions of creator blocks.

cards placed on them are wirelessly communicated to the software running on the host computer.

There are two kinds of *programming cards*: pictures and behaviour cards. Picture cards are used to place new objects at specific locations on the screen, while the behaviour cards are used to specify the functionality of objects that have already been added. To add a new

picture at a specific location, a picture card is placed on top of the creator block (Figure 7). Behaviours are added to existing objects by first placing the creator block at a position where there is an object and then putting a *behaviour card* on top of it. Behaviour cards consist of a set of behaviours for movement, collisions, user interaction, and for changing properties of objects.

Constructions made with the system would then only be stored in digital form. Even though a physical item can be used to logically store the code of a specific construction, the code itself can only be accessed through a computer screen.

DISCUSSION

Below we analyse our findings in terms of four concepts often discussed in tangible computing. The concepts that we will discuss are coupling, manipulation, the notions of "input" and "output", and physical artefacts as "representing" digital information. The reason for this is to show how the design process has shaped our understanding of these concepts, which we hope contributes to the discussion around these ideas.

Coupling: mapping between the physical and the digital

A common way of framing the challenge of interaction design is through the notion of coupling, where the ideal situation is to mimic how people interact with everyday objects. A common example is the blind man and his cane, where interaction and response from the interaction ultimately happens in the same object [e.g. 7]. Based on Fishkin's [13] analysis of current directions in tangible computing, this is also what appears as the most promising use of tangibles for interaction.

In our final design, a programmed system will be present only on the screen, and will never get a representation in the physical space. Throughout the construction activity, the children will need to look up at the screen to identify what has been programmed, and look down on the floor to locate, arrange, and discuss the resources for extending the system. Placing a physical picture or behaviour object directly on the mat surface does not have any effects on the computational system whatsoever. Further the interaction model used in the system implies that all actions for completing different commands are basically the same. Adding a "move to the left"-behaviour would require the same physical action as adding any behaviour or picture to the system. In this sense, the system provides a *limited physical coupling* between performed physical action and its effects on the screen.

However, when analysing the system in the context of our earlier proposals and design activities, we find that the final system allowed for increased flexibility and expressiveness regarding what could be accomplished through the programming activity at large. The designs that initially seemed to provide a closer mapping between physical and virtual representations were in effect much more limiting in terms of programming.

A possibly trivial conclusion is that good coupling between the physical and the screen does not mean that all digital representations should take physical form and neither that all actions with the physical resources should take effect on the screen. Instead, coupling physical with virtual actions is required only at certain points in the interaction. A central function of the creator blocks is for instance that they may work as "sights", allowing children to locate a position on the screen before adding something there. These sights are visible on the screen also when not actively using the creator block, making it possible to locate a creator block on the floor by looking at the screen surface. The close coupling between the focus point on the screen and the device may also work as an aid in accounting for ones actions in the larger social setting.

We also found that the separation between the locus of interaction and feedback not always became a disadvantage in the design. Instead the gap between the physical and the virtual could be understood as a way of supporting the children in the activity of programming, for instance by making a clearer distinction between programming of a system and its execution. This particular distinction is something that many children often have difficulties with when programming using screen based systems, i.e. separating actions describing a system's behaviour from when the system runs.

Manipulation: sensory experience of interaction

In many ways the affordances of physical objects are less ambiguous than virtual ones, and thus more predictable to manipulate. We touch, we move around, and we make constant use of the physical artefacts around us. Much of the argumentation for tangible user interfaces has also concerned the increased sensory experience of manual actions such as pulling, twisting, shaking, and squeezing.

What we see through our different designs is that the final version had less physical affordances in terms of how to manipulate its physical parts. While our earlier designs attempted to structure the interaction through physical constraints, the interaction devices of the final system take the shape of plain surfaces upon which different objects can be placed. This naturally limits the range of possible manipulations, and could also be seen as enforcing additional work of repositioning a creator block each time an action should be recorded.

On the other hand, this set-up meant that the creator blocks could be used for tangible representation of actions beyond adding pictures and behaviours to specific positions on the screen. For instance, a number of "control" cards were created, which let the user perform global action to the system, such as playing, stopping, and

saving the system that has been built. Another example is how the creator blocks could be made to work as interactive resources not only when building a system, but also as devices for interaction with the system that has been created.

Moreover, the clean design of the creator blocks, without any buttons, knobs, levers or displays, allows them to blend into the background of the activity. This helps leading the attention to other parts of the system, such as the screen, the cards, and the surface on the floor, which in turn supports users in maintaining a focus on the shared aspects of the activity. So, even though our design initially attempted to achieve rich physical affordances and manipulations, our overall interaction scenario led us to put stronger emphasis on the role of the tangibles in the larger social setting. The reason for developing interactive resources in tangible form in our case became more directed towards broader issues such as the possibility for people to act individually as well as collectively, to be able to arrange and to hand over physical resources to one another, to draw someone's attention to something through physically relocating oneself, and to account for ones action in the group.

In this sense, what could be *accomplished* through the manipulations became more important to the design than how the sensory experience of the *manipulation* would be perceived.

Relation between input and output of the system

A mainstream perception of tangible computing is the endeavour of designing systems where user input and system output may occur in the same artefact [32]. This is especially since the physical parts of a tangible system often come to replace parts that would otherwise be displayed on a screen, and also that arrangements in the physical space are sometimes the only way that the system displays its actual state.

In our design, much of what a user will experience of the system is present only in the physical space. This became especially evident through the extra layer of interaction, which makes it possible to have resources spread across the surface of the mat without having them recorded into the system. An effect of this is that it allows for many possible interactions with the resources without actually interacting with the computational system. This could include handing over a programming card to a friend, placing a set of cards in a stack for later use, and to arrange and rearrange the physical workplace. In our user studies, we find that such forms of manipulations often become core to the interactive situation. Thus, the mat works as a confined surface around which not only programming actions take place, but also a range of other related activities, such as planning and testing of ideas, selecting and locating behaviours and pictures to use, decorating the floor surface, and for social play. While such offline and parallel activities have been noted as important qualities of user's interaction with technology [see e.g.1, 4, 24], they do not fit naturally with the idea of systems as designed only for responding to user input.

Similarly, from a perspective of design and craftsmanship, McCullough [21] emphasises how designing with digital tools always, just as with any other design material, has to do with the joint achievements of multiple senses, such as vision and touch, and of bodily experience and manual skill in different physical contexts. Central to such a conception is that it seeks to avoid dualistic perspectives on human action, such as distinguishing between "doing" and "thinking", and separating bodily actions from cognitive ones. Throughout our design process, the distinction between internal and external properties of the "system" appeared increasingly artificial to us. In working towards the goal of supporting children to collaboratively create dynamic play worlds to run on a computer screen, we instead had to put more focus on an integrated view on the activity as a whole, where offline activities play as much part in the user interaction as do actions with more direct effects to the system displayed on the screen.

Tangibles as representation of digital information

A more conceptual discussion concerns how to position the development of tangible interfaces as traditional psychological notions from an information-processing paradigm get replaced with descriptions of collaborative and bodily forms of interaction. An example of this is Ullmer and Ishii's conceptualisation of tangible user interfaces, were they describe their work as an attempt to bridge the gap between the physical and the virtual by making digital "information" tangible [32]. As a contrast to this viewpoint several authors have emphasised how the form of interactive resources changes the meaning of symbolic and computational manipulation. It has for instance been suggested how the essence of tangible interaction may be better captured, and more naturally placed within the current discourse of human computer interaction through notions borrowed from ethno-methodology [7], activity theory [18], and design practice [2]. This means that the initial definitions of TUIs have been accompanied with conceptualisations that focus on human *action*, rather than on representation and transformation of *information*. This shift in perspective guided our design process as well as our analysis of it.

Note that in our early design activities and the resulting prototypes, the physical objects for pictures and behaviours worked as representations of what can be *available* on the screen, while in the final version the cards work more straightforwardly as resources for making *actions* happen on the screen. Placing a card on a creator block would mean "display picture on the selected

position on screen" or "add behaviour to the picture at selected position on screen", not just "picture" or "behaviour". We find this an important distinction in terms of how to understand and conceptualise interaction with the system that we designed.

Through the final design of the system, we made the explicit design choice of making a stricter separation between programming tools and the system being built with these. The physical space became used for providing the *resources* for building a computational system, while the screen display is used for *representing* what has been built so far, and for running the system. The purpose was thus not to represent what is in the physical space as accurately as possible on the screen, but to combine important qualities of physical and digital design.

LESSONS LEARNED

The overall goal with this work was to through experimental designs develop a richer understanding of the specific properties of new programming materials. In reflecting upon this process, we have identified three issues that contributed to this.

The first issue concerns the correlation between physical persistence of programming code and reusability of programming resources. This had important consequences in how one should interpret and understand the mapping between the physical and the tangible resources. This also includes issues of *persistency* and *readability* of programming code. It may seem a trivial fact that resources for possible commands in any system need to be available in the physical space. However, it may seem less obvious that *actions taken* with these resources do not necessarily need to be represented within the resources themselves. A direct mapping between physical resources and on-screen representations meant that the possibilities to scale up the system with larger amounts of objects and behaviours were restricted. Turning programming code into physical form became not just another way of representing digital information, as is commonly conceptualised in the design of visual representations of text based programming languages [see 19].

A second issue concerns the importance of appreciating the difference between a complete scenario of interaction from the actual manipulation of tangible resources. Our earlier prototypes were in many ways richer than the final system in terms of physical manipulation, and also utilised more of the specific affordances of the physical materials. However, our final approach was richer regarding the computational actions that became possible to perform. This emphasises the need of finding a balance in the use of the specific properties of the physical and digital resources, and not just treat the tangible resources as input/output channels that can be analysed and understood on their own.

A third and more general issue that we would like to put forward is how our design process illustrates a move from a data-centric view of interaction, to a view focusing on representational forms as resources for action. Through such a perspective, is it not "information" that is moved between people and devices, neither logically nor actually. Instead the physical artefacts must be understood as having deeper social and personal purposes in the shared, collaborative space of physical and bodily sense-making activity that users engage in.

ACKNOWLEDGEMENTS

Thanks to everyone involved in development of the technology Johan Mattson, Martin Jonsson, Jesper Holmberg, Christopher Balanikas, Korai Duhbaci, Manu Gupta, and of course all the children we have worked with in various projects. We also would like to thank our "Writing club" (Kristina Höök, Petra Sundström, Jarmo Laaksolahti, Åsa Rudström) for putting up with all our weird drafts. We also would like to thank Ken Kahn for discussing and critiquing our design ideas, Kevin McGee for last minute feedback on our writing, and finally all the five anonymous reviewers for their wonderful comments.

REFERENCES

1. Benford, S., H. Schnadelbach, B. Koleva, B. Gaver, A. Schmidt, A. Boucher, A. Steed, R. Anastasi, C. Greenhalgh, T. Rodden, and H. Gellersen, Sensible, sensable and desirable: a framework for designing physical interfaces, Technical Report EQUATOR-03-003, School of Computer Science & IT, Nottingham University, 2003.
2. Buur, J., M.V. Jensen, and T. Djajadiningrat. Please touch tangible UIs: Hands-only scenarios and video action walls: novel methods for tangible user interaction design. *Proc. DIS*. ACM Press (2004), 185 -192.
3. Cassell, J., K. Ryokai, and C. MIT Media Laboratory, MA, USA. Making Space for Voice: Technologies to Support Children's Fantasy and Storytelling. *Personal and Ubiquitous Computing*. 5, (2001), 169–190.
4. Cohen, J., M. Withgott, and P. Piernot. Logjam: a tangible multi-person interface for video logging. *Proc. SIGCHI conference on Human factors in computing systems* (1999), 128-135.
5. Crook, C. Children as Computer Users: the Case of Collaborative Learning. *Computers and Education*. 30, 3/4, (1997), 237-247.
6. Decortis, F.o. and A. Rizzo. New Active Tools for Supporting Narrative Structures. *Personal and Ubiquitous Computing*. 6, 5-6, (2002), 416 - 429.
7. Dourish, P., *Where the action is. The foundations of Embodied Interaction*. 2001, Cambride, MA: MIT Press.

8. Druin, A. and C. Fast. The Child as Learner, Critic, Inventor, and Technology Design Partner: An Analysis of Three Years of Swedish Student Journals. *The International Journal for Technology and Design Education*. 12, 3, (2002), 189-213.
9. Eisenberg, M., A. Eisenberg, M. Gross, K. Kaowthumrong, N. Lee, and W. Lovett. Computationally-Enhanced Construction Kits for Children: Prototype and Principle. *Proc. International Conference of the Learning Sciences* (2002), 79-85.
10. Fernaeus, Y., C. Aderklou, and J. Tholander. Computational Literacy at Work, Children's interaction with computational media. *Proc. CELDA 2004*. IADIS Press (2004), 181-188.
11. Fernaeus, Y. and J. Tholander. Collaborative computation on the floor. *Proc. CSCL 2003* (2003),
12. Fernaeus, Y. and J. Tholander. "Looking At the Computer but Doing It on Land": Children's Interactions in a Tangible Programming Space. *Proc. HCI2005*. Springer (2005), 3-18.
13. Fishkin, K.P. A Taxonomy for and analysis of tangible interfaces. *Personal and Ubiquitous Computing*. 8, (2004), 247-358.
14. Frei, P., V. Su, B. Mikhak, and H. Ishii. Curlybot: designing a new class of computational toys. *Proc. SIGCHI conference on Human factors in computing systems*. ACM Press (2000), 129-136.
15. Fällman, D. Design-oriented Human-Computer Interaction. *Proc. CHI2003*. ACM Press (2003), 225 - 232.
16. Iacucci, G., C. Iacucci, and K. Kuutti. Imagining and experiencing in design, the role of performances. *Proc. NordiCHI2002*. ACM Press (2002), 167-176.
17. Kaptelinin, V. and M. Cole, *Individual and Collective Activities in Educational Computer Game Playing*, in *CSCL2: Carrying Forward the Conversation*, T. Koschman, R. Hall, and N. Miyake, Editors. 2002, Lawrence Erlbaum.
18. Kuutti, K., *Activity Theory as a Potential Framework for Human-Computer Interaction Research*, in *Context and Conciousness*, B.A. Nardi, Editor. 1996, MIT Press: Cambridge, Massachusetts.
19. Lieberman, H., ed. *Your wish is my command. Programming by Example*. 2001, Morgan Kaufmann Publishers: San Fransisco, CA.
20. Löwgren, J. and E. Stolterman, *Thoughtful Interaction Design*. 2004: MIT Press.
21. McCullough, M., *Abstracting Craft : The Practiced Digital Hand*. 1997: MIT Press. 250.
22. McNerny, T.S. From turtles to Tangible Programming Bricks: explorations in physical language design. *Personal and Ubiquitous Computing*. 8, (2004), 326-337.
23. Montemayor, J., A. Druin, A. Farber, S. Simms, W. Churaman, and A. D'Amour. Physical programming: designing tools for children to create physical interactive environments. *Proc. CHI2002*. ACM Press (2002), 299-306.
24. Patten, J. and H. Ishii. A comparison of spatial organization strategies in graphical and tangible user interfaces. *Proc. DARE 2000*. ACM Press (2000), 41-50.
25. Phillippe, P., *Zowie*. 2005.
26. Raffle, H.S., A.J. Parkes, and H. Ishii. Topobo: a constructive assembly system with kinetic memory. *Proc. SIGCHI conference on Human factors in computing systems*. ACM Press (2004), 647 - 654.
27. Rettig, M., *Prototyping for tiny fingers*, in *Communications of the ACM*. 1994. p. 21-27.
28. Stanton, D., V. Bayon, H. Neale, A. Ghali, S. Benford, S. Cobb, R. Ingram, C. O'Malley, J. Wilson, and T. Pridmore. Classroom Collaboration in the Design of Tangible Interfaces for Storytelling. *Proc. CHI2001*. ACM Press (2001), 482-489.
29. Suzuki, H. and H. Kato. Interaction-Level Support for Collaborative Learning: AlgoBlock-An Open Programming Language. *Proc. CSCL* (1995), 349-355.
30. Tholander, J., K. Kahn, and C.-G. Jansson. Real Programming of an Adventure Game by an 8 year old. *Proc. ICLS 2002*. Lawrence Erlbaum Associates (2002), 473-480.
31. Tisue, S. and U. Wilensky. NetLogo: A Simple Environment for Modeling Complexity. *Proc. International Conference on Complex Systems* (2004),
32. Ullmer, B. and H. Ishii, *Emerging Frameworks for Tangible User Interfaces*, in *Human-Computer Interaction in the New Millenium*, J.M. Carrol, Editor. 2001, Addison-Wesley. p. 579-601.
33. Ullmer, B., H. Ishii, and R.J.K. Jacob. Token+constraint systems for tangible interaction with digital information. *ACM Transactions on Computer-Human Interaction*. 12, 1, (2005), 81 - 118.
34. Wellner, P., *Interacting with paper on the DigitalDesk*, in *Communications of the ACM*. 1993. p. 87 - 96.
35. Wyeth, P. and H.C. Purchase. Using Developmental Theories to Inform the Design of Technology for Children. *Proc. Interaction Design and Children*. ACM Press (2003), 93-100.
36. Zuckerman, O., S. Arida, and M. Resnick. Extending tangible interfaces for education: digital montessori-inspired manipulatives. *Proc. SIGCHI conference on Human factors in computing systems*. ACM Press (2005), 859 - 868.