Designing for Appropriation

Alan Dix
Computing Department, InfoLab21
Lancaster University, Lancaster, LA1 4WA, UK
+44 1524 510 319
alan@hcibook.com
http://www.hcibook.com/alan/papers/HCI2007-appropriation/

ABSTRACT

Ethnographies often show that users appropriate and adapt technology in ways never envisaged by the designers, or even deliberately subverting the designers' intentions. As design can never be complete, such appropriation is regarded as an important and positive phenomenon. However designing for appropriation is often seen as an oxymoron; it appears impossible to design for the unexpected. In this paper we present some guidelines for appropriation based on our own experience and published literature and demonstrate their use in two case studies. You may not be able to design for the unexpected, but you can design to allow the unexpected.

Categories and Subject Descriptors

H.5.5 [Information Interfaces and Presentation]: HCI

General Terms

Design, Human Factors.

Keywords

appropriation, hackability, tailorability, guidelines.

1. INTRODUCTION

Appropriation is a common theme in ethnographies of new technology use and is often seen as an important sign of users' acceptance of technology. However, while much has been written about the *importance* of appropriation, it is far harder to find practical advice on how to *design* for appropriation.

Certainly reading accounts of appropriation can sensitize a designer to the issue, as do more theoretical works emphasizing the ecological fit of technology and the importance of technology being embedded into users' real work practices. However, even in Dourish's "Where the action is", probably one of the most well known texts in the area, the advice on '"Moving towards Design" is, quite reasonably, broad [9].

This paper was born out of the need to capture some of this knowledge more explicitly, particularly for my own students of HCI and because I was faced with writing about the issue for the next edition of a textbook [6]. So I wanted a form, which whilst still open, gives directed design guidance. Being more specific is of course dangerous as one can be more wrong! So, these design guidelines are not presented to foreclose debate, but to present an incomplete but I hope useful practical contribution and also a departure point for ongoing discussion.

The next section will present a short description of appropriation for readers for whom it is not a familiar topic and also discuss the reasons why it is important, followed by a brief overview of some literature in the area. A set of design principles is then proposed drawn from the literature and my own experience. These are then illustrated by micro case studies showing how they have been applied in real designs.

2. ABOUT APPROPRIATION

2.1 What is Appropriation?

In ethnographies and field studies a frequent observation is that people do not 'play to the rules': they adapt and adopt the technology around them in ways the designers never envisaged. Think to your own experience: perhaps you have used a screwdriver to open a paint tin, or heavy textbook to prop open a door … or tried to open a bottle of wine without a corkscrew.

Improvisation is critical to 'getting things done'. Sometimes we have exactly the right tool to hand, but often the particular circumstances are not totally foreseen and we need to work with what we have to hand.

We see the same process of appropriation with digital technologies. Email is intended as a way to communicate with remote colleagues, but some people email themselves web links whilst browsing instead of using a bookmark, 'communicating' with themselves; others use email attachments as a way to share files with a colleague on the next desk.

These improvisations and adaptations around technology are not a sign of failure, things the designer forgot, but show that the technology has been domesticated, that the users understand and are comfortable enough with the technology to use it in their own ways. At this point we know the technology has become the users' own not simply what the designer gave to them. This is *appropriation*.

Appropriation may occur where there is no existing tool for the task, for example, the users mailing themselves a web link because bookmarks and email folders are distinct and they want to organise them together. It may also occur where there is an alternative method, but the appropriation is easier either at the moment or because of learning time, for example, using email for sharing files instead of configuring shared network folders.

2.2 Advantages of Appropriation

Appropriation is important for several reasons:

situatedness – The end point of design has been described in terms of *intervention* [6], not just an artefact or even the artefact and its immediate ways of interaction with it, but the way in which it changes the environment in which it is set. While each word processor may be the same, each office, home or laptop on a train is a different environment. We cannot expect to be able to understand each environment fully and to meet every possible task or need.

dynamics – Environments and needs change. Suppose we designed specifically for a particular work group in a particular office, and covered all eventualities for them. A month later, a year later, new people would have joined the group, the external business environment may have changed their focus, there may be additional software or furniture that changes the digital and physical workspace. Design for use must be design for change.

ownership – With appropriation comes a sense of ownership. This may simply be a feeling of control, users feeling they are doing things their own way. It may also be explicit: often people proudly show you the ways they use software and technology to achieve their purposes. These positive feelings can be as important as the things that are achieved.

Sometimes appropriation can be a form of *subversion*, deliberately using something in a way it was not intended, not just because of something the designer didn't think about, but in order to thwart its intentions. For example, in the days before mobile phones were ubiquitous, people often avoided paying the charge on a public payphone by saying something like: "when I'm ready to be picked up I'll ring twice on the phone and then hang up". Whether this is an advantage or disadvantage of appropriation depends on who you are!

This form of subversion is often seen in work contexts: a salesman might deliberately create a 'phantom order' and later withdraw it in order to ensure there is stock available for a loyal customer [1]. In this kind of setting the subversion is against the formal rules in the system, but may be working towards the same ultimate goal of making the best sales for the company.

2.3 Related Work

As noted, appropriation is a common theme in ethnographies of the workplace and the home. In their study of mobile phone use by teenagers, Carroll et al. proposed a framework, the *Technology Appropriation Cycle*, for understanding the *process* of technology appropriation [2]. They distinguish technology-as-designed (as provided by the designer) and technology-in-use (as embedded into the lives of users). These are then linked through a process of appropriation whereby technology is either never seriously considered (non-appropriation) or taken on board selected and adapted by users (appropriation), but even if appropriated may at some stage be rejected (dis-appropriation). Focus on process is also evident in other writings, for example several papers at the CHI2005 workshop on community appropriation [13].

In a DIS2004 keynote Tom Moran suggested several "trends supporting design": open standards, web architecture, portalization, freeform technologies [14]. These are focused more on the ability of end-users to hack or modify systems and are based on a long tradition of user-tailorable systems such as the Xerox Buttons interface [12]. His keynote prefigured the explosion in Web2.0 mashups, which have themselves triggered a fresh focus on 'hackable' systems [11].

3. GUIDELINES FOR APPROPRIATION

The idea of designing for appropriation almost seems like an oxymoron: "plan for the unexpected". However, whilst you cannot design for the unexpected, you can design so that people are more likely to be able to use what you produce for the unexpected – they do the final 'design' when the need arises.

The fact that design for appropriation is possible is made most clear by realising that some sorts of design make appropriation difficulty or impossible. Consider an espresso machine. It is so special purpose it is hard to imagine any alternative use. (Although human ingenuity is such that I expect a rush of emails telling me about alternative uses for espresso machines!) This also shows that design for appropriation is not always what is desired, the espresso machine does one thing very well indeed – why do any more with it.

However, explicit design guidance to allow appropriation is less clear. Here are some principles, but they should be taken as ways to encourage reflection, not a tick list to verify.

allow interpretation – Don't make everything in the system or product have a fixed meaning, but include elements where users can add their own meanings. For example, in MacOS you can associate colours with file, but there is no fixed meaning to a red file (maybe urgent, or problem) – it is the user who provides the interpretation. Similarly, in MacOS folders and Windows desktop you can position file icons freely. *Because* location does not mean anything to the system, the user is free to group files; perhaps have one corner of a folder window means "finished with". Similarly in a database system, a simple free text comment field allows users to add details the designer never considered, just like writing comments on a paper form.

provide visibility – Make the functioning of the system obvious to the users so that they can know the likely effects of actions and so make the system do what they would like. This is particularly important when the effects of actions are distant or at different time, for example in a collaborative application. Often systems, particularly networks, try to cover up or hide the underlying 'details'. This is fine so long as it is totally and permanently hidden, but often the details leak out (e.g., at the limits of wireless coverage). Users find their way round these problems if they are made clear (signal level bars on a mobile phone allow you to find better signal). The notion of 'seamful' design [3] deliberately exposes these 'seams' in coverage and connectivity in order to create games and applications. Note the common usability heuristic 'visibility of system status' usually refers to the *relevant* state; whereas it is often the *irrelevant* state and *internal process* that can be appropriated.

expose intentions – While appropriation can be very powerful, we have also seen that it can be used to subvert systems. Rather than trying to prevent such subversion the designer can deliberately aim to expose the intention behind the system. This means that (cooperative) users can choose appropriations that may subvert the rules of the system and yet still preserve the intent. For example, if logging into a system is a slow process, then one member of a work group may login once at the beginning of the day and then everyone else use the logged-in system. If the purpose of the login is security, then this may be fine so long as the machine is never left unattended, however if the purpose is to adapt the system to individual users then this appropriation would be inappropriate. Exposing the intentions behind a system can be a frightening thing – you cannot hide behind "well that's the way it works". However, doing this can make the assumptions explicit and if they are wrong then they need to be re-examined. In the login example, if the purpose is personalisation would it be possible to allow a single secure login but have a facility to quickly swap between users.

support not control – As a designer you want to do things right, to make them as efficient and optimal as possible. However, if you optimise for one task you typically make others more difficult. In some situations, such as very repetitive tasks, then designing explicitly for the task may be the correct thing to do, perhaps taking the user step by step through the activities. However, more often the tasks description is incomplete and approximate, in particular ignoring exceptions. Instead of designing a system *to do* the task you can instead design a system so that the task *can be done*. That is you provide the necessary functions so that the user can achieve the task, but not drive the user through the steps. Dourish describes this as "informal assemblage of steps rather than rote procedure driven by the system" [9]. Of course you still want the common tasks done efficiently and so you may provide fast paths, or wizards to perform frequent activities using the basic tools and operations … remembering of course visibility, making sure the user understands what is being done.

plugability and configuration – Related closely to the idea of support is to create systems where the parts can be plugged together in different ways by the user. Quoting from Dourish again "Users, not designers manage coupling" [9]. This is most obvious in programmable or scriptable systems and there is considerable work in making these more accessible to the user. This ability to plug-and-play components becomes a critical issue in ubiquitous computing and Newman et al. discuss the idea of recombinant computing [15] were systems are created bottom-up from small end-user combinable components. In more a traditional interface MacOS Automator allows the user to chain together small actions from different applications, so that, for example, you can create a workflow that takes a collection of images, sepia tints them, and then mails them all to a group of people from your address book. Similarly Yahoo! Pipes (pipes.yahoo.com) allow users to create and share mashups of RSS feeds and search results.

encourage sharing – People are proud of their appropriations of technology, so let them tell others about it! If one user learns a good trick for using an application or device, then this may be useful to others as well. Documentation can be enhanced by end-user contributed material; many web sites offer tips and advice on different software, and this can be designed as an integral part of a system, perhaps a 'tips' button that allows users to annotate functions with their favourite tricks. This could function across institutions making use of the communities of practice to which the user belongs. This sharing is even more important in the case of programmable systems. Even if the configuration or scripting is designed to be 'easy' it will still be only a small subset of users who actively script. However, if you make it easy for more confident or more technically adroit users to share with others then your product grows all on its own. The success of Xerox Buttons was an early example of this [12]. More recently, the MacOS Dashboard has this shareable quality, as does the Firefox architecture. Both allow the creation of plugins using a combination of small XML and HTML files and Javascript; importantly both have web sites dedicated to sharing these. As an online application, the Yahoo! Pipes interface not only allows sharing of complete pipes, but also makes it easy to see the graphical 'code' of pipes and hence copy and adapt them.

learn from appropriation – After a while one old, but broad bladed screwdriver becomes 'the' paint-tin opener. What was once a temporary use of a tool has become specialised. This crystalising of appropriation leads to a new tool and the entrepreneur might spot this, notice the particular kinds of screwdriver that made good paint-tin openers and then design a special purpose tool just for the job. By observing the ways in which technology has been appropriated, we may then redesign the technology to better support the newly discovered uses. This is a form of co-design where the users are considered an integral part of the design process. This closing of the *Technology Appropriation Cycle* has been called *design from appropriation* [2]. Of course any redesign should also take into account potential further appropriation. This learning from appropriation is particularly easy in some web applications (e.g. blogging or photosharing sites) where the results of users appropriation of the application are easily visible to the designers.

A common feature to these principles is openness, making things that allow themselves to be used in unexpected ways, echoing in some ways the idea in literature of an 'open' or 'writerly' text [4]. It is also to some extent about humility, knowing that you do not understand completely what will happen in real use, no matter how good your user-centred design process has been.

4. IN PRACTICE

We will now look at two micro case studies of how these principles work in practice. The first is a post hoc analysis of learning from appropriation and the latter an example of deliberate design for appropriation.

4.1 OnCue – Learn from Appropriation

OnCue is a desktop tool available during the dot.com years [5]. It appeared as a small floating tool palette and when the user copied or cut any text onCue examined it using primitive 'intelligence' to decide what kind of thing was in the clipboard and offered various web and desktop services that could use this kind of data; for example, a name might trigger a phone lookup.

An early and enthusiastic adopter was interviewed about his use of onCue. During the discussion it turned out that sometimes he had a name, postcode, or something that he wanted to use onCue to lookup, but it was not in a document to copy. So he opened an empty document in a text editor, typed into it and then copied the text. Only at that point did onCue fire up and offer suggestions. It was interesting that he did not consider this laborious process a major 'problem' partly because onCue was giving him sufficient value and partly, because (as noted earlier) he was rather proud of his workaround.

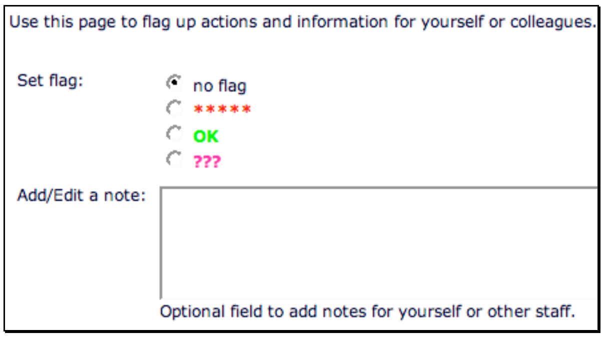
At the risk of hurting his pride (!) a redesign was suggested and in the second release of onCue clicking the onCue icon would open up a small search-engine-like type-in box beneath the onCue tool palette. We should note that this use of onCue had been entirely unforeseen by the designers (although in retrospect it seems obvious), and the redesign was entirely due to following the principle of '***learn from appropriation***'.

4.2 eCommerce Design for Appropriation

The second mini case study concerns a small eCommerce web site, although for commercial confidentiality some of the details have been altered. The system gathered online orders, but if the order included books, where stock levels varied, payment was delayed until the book administrator could verify that all the books were in stock and only then charge the customer's card.

A few months after initial deployment the book administrator asked whether it was possible to make a modification to the administration system. When an order was not completed for some reason she wished to be able to indicate the reason explicitly, rather than leave it simply uncompleted. This was partly for her benefit, and partly to flag up such transactions for the finance staff. She suggested a simple 'incomplete' flag.

The requirement was straightforward; however, following the '***support not control***' principle it was decided not to create an interface feature that directly addressed the expressed need. Instead a more generic status flag with a number of different icons was added: stars, question mark, etc. In addition a notes field was added. Following the principle '***allow interpretation***' we did not prescribe a particular use of either of these, but left it to users to make their own use of the elements (see Fig 1.).


Use this page to flag up actions and information for yourself or colleagues.

Set flag: [✓ no flag] ***** OK ???

Add/Edit a note:

Optional field to add notes for yourself or other staff.

Figure 1. Fields for user interpretation

At first these were used only for the initial purpose, but over time we saw the use change. In particular, in early use, orders that were incomplete due to an out-of-stock book were simply labeled generically "book out of stock", whereas later these were marked with a different icon from other orders and the name of the book put in the notes field, acting as a sort of 'to do' showing which books needed reordering – the user had appropriated the system in a way that would have been impossible with a more task-focused, but closed design. Due to staff turn-over we were unable to discuss this, but is clear that the design was successful in allowing appropriation..

5. DISCUSSION

Arguably the design features suggested by the appropriation principles seem trivial. In fact, this is often the case, designs that are closed are often more apparently sophisticated, because they may do more for the user, but ultimately do not allow the users to do more for themselves. Good design for appropriation in practice however is clearly *not* trivial given the lack of suitable features in many systems and the results of innumerable ethnographies, hence the importance of *explicit* advice as well as case studies and vignettes.

It is evident from the case studies that design for appropriation (a) is possible, (b) can work and (c) is not so complex as it may appear. Clearly all systems have boundaries, but we can design them so that the space of possibilities for design by users in the context of use is expanded. In some situations, (e.g. safety critical systems) it is important for users to operate systems exactly as designed, hence the importance of '**expose intentions**' as users will inevitable bypass controls (as in Chernobyl) if they do not understand why they are there.

The guidelines are not new in that most can be found in some form embedded in the literature, for example, variants of the first two can be found (albeit buried somewhat) in [7]. However, I believe this is the first attempt to systematically extract this knowledge and to present it in an applicable form.

Validating design principles is hard as simple post-hoc evaluation is methodologically unsound [10]. However, a more thorough theoretical framework or model of appropriation would be valuable to both validate these principles and suggest future directions of study.

6. REFERENCES

[1] Ainger, A. and Maher, R. The 'Salesman's' Promise (CSCW in Sales). Chapter 5 in *Remote cooperation: CSCW issues for mobile and tele-workers*. A. Dix and R. Beale (eds.), Springer Verlag. 1996.

[2] Carroll, Jennie., Howard, S., Vetere, F., Peck, J. and Murphy, J. Identity, power and fragmentation in cyberspace: technology appropriation by young people. In *Proc. of Australian Conf. on Information Systems*, 2001.

[3] Chalmers, M. and Galani, A. Seamful Interweaving: Heterogeneity in the Theory and Design of Interactive Systems. In *Proc. DIS 2004*, ACM Press, 2004, 243-252.

[4] Cuddon, J.. *The Penguin Dictionary of Literary Terms and Literary Theory*, 4th ed. Penguin, 1998.

[5] Dix, A., Beale, R. and Wood, A. Architectures to make simple visualisations using simple systems. In *Proc. AVI2000*, ACM Press, 2001, 51-60.

[6] Dix, A, Finlay, J., Abowd, G., and Beale, R.. Chapter 5. Interaction design basics. *Human-Computer Interaction, 3rd ed.*. Prentice Hall, 2004.

[7] Dourish, P. The appropriation of interactive technologies: some lessons from placeless documents. *Computer Supported Cooperative Work* 12, 4 (Sep. 2003), 465-490

[8] Dourish, P. *Where the Action Is: The Foundations of Embodied Interaction*. Cambridge: MIT Press, 2001.

[9] Dourish, P. Implications for Design. In *Proc. CHI 2006* (Montreal, Canada), ACM Press, 2006, 541-550.

[10] Ellis, G. and Dix, A. 2006. An explorative analysis of user evaluation studies in information visualisation. In *Proc. of BELIV '06*. ACM Press, New York, NY, 1-7.

[11] Galloway, A., Brucker-Cohen, J., Gaye, L., Goodman, E., and Hill, D. 2004. Design for hackability. In *Proc. DIS'04*. ACM Press, New York, NY, 2004, 363-366.

[12] MacLean, A., Carter, K., Lövstrand, L., and Moran, T.. User-tailorable systems: pressing the issues with buttons. In *Proc. CHI '90*. ACM Press, 1990, 175-182.

[13] March, W., Jacobs, M., and Salvador, T. Designing technology for community appropriation. In *Proc. CHI '05 Extended Abstracts*, ACM Press, 2005, 2126-2127

[14] Moran, T., Everyday Adaptive Design (keynote). *DIS'02*. http://www.sigchi.org/dis2002/

[15] Newman, M., Sedivy, J., Neuwirth, C., Edwards, W., Hong, J.., Izadi, S., Marcelo, K., and Smith, T. Designing for serendipity: supporting end-user configuration of ubiquitous computing environments. In *Proc. DIS'02*. ACM Press, 2002, 147-156.