Implementation bias in HCI – and escaping it

Harold Thimbleby, Swansea University, Wales

HCI, whether for research or for specific usability case studies, tries to align the user task with the user interface design, with its computer implementation. For research, HCI tries to identify the principles; for specific case studies, HCI tries to improve the user experience or the product delivered. In all approaches, the user task is central, yet the user task may be an artifact of previous implementations (it may suffer *implementation bias*), and users — and user studies — may therefore accidentally focus on problems with their current task as implemented rather than opportunities for a new HCI approach. In this paper, we use presentations (i.e., giving talks or lectures) as a detailed stand-alone case study, and we show that implementation bias is a serious impediment to thinking clearly about improving the task of delivering quality presentations. We then argue, using a very different example (healthcare IT) that implementation bias is, in fact, a common and serious problem across HCI generally.

Key Words: HCI, Implementation bias, healthcare IT, presentations.

DOI: 10.24982/jois.1717017.004

1 THE HCI OF PRESENTATIONS AND HEALTHCARE

When a presentation or lecture starts, how often do we peek inside the speaker's laptop and see their whole digital life? We often see all of their slides as they struggle to get the presentation set up, or we see their private desktop clutter, and we watch painfully as they try to find their presentation, and then struggle how to get into presentation mode. That is, assuming the projector even works.

Most talks we listen to and most we give involve computer technology, most noticeably in presentation tools like PowerPoint, Keynote, PDF viewers, or Prezi. Typically, we use a laptop and connect to a projector — another computer. Often this is where human-computer problems become visible: a stressed speaker trying to start a talk cannot find out how to get the computer or projector to work or talk to each other. The computer might have HDMI but the projector has VGA, or the projector has DVI and the laptop has USB-C. Or somebody has lost the remote control. Aargh! Sometime somebody in the audience knows what to do, because they are familiar with the computer and, crucially, they are not stressed.

Often *hindsight bias* makes the presenter feel worse. Once their problem is sorted out, it seems easy. (Was it only a matter of clicking on a box in the control panel?) How stupid they feel for not knowing something so simple!

When presenters send their talk in early for a presentation so a technician can set up the computer and video systems, there are different problems. The PC the speaker has to use may be unfamiliar, and at a conference there may be many talks that the speaker has to sort out which is theirs. Then they discover — part way through the talk — that some fonts are missing, or the video they expected to use was not copied onto the talk they sent to the conference, or that the sound is awful. Discovering a problem part way through a talk is a good way of getting even more stressed!

During the presentation, somebody in the audience asks a question and the speaker cannot work out how to skip out the next slide or change the order of their presentation to respond to the question. Or if the question is about the previous slide, going back to

show it gets all of the slide builds wrong (because they were animated to go forwards in a talk, not go backwards). And it is not possible to annotate a slide without dropping out the software's presentation mode.

Then things like software updates or other announcements and warnings interfere with the presentation. Or the speaker's laptop battery starts to go flat.

Developers have focused on automating the easy bit: editing and presenting a talk. Conventional human-computer interaction (HCI) will help make this part of the job easier, and also help make it more enjoyable (with UX, user experience work). There is a tendency to push presentations towards professional production standards, so the number of features and options has exploded, and the usability of the presentation software has certainly become a serious HCI issue.

The learning curves of presentation software are steep, and speakers get locked into software they have spent considerable time learning. This exacerbates the problems of delivering presentations: there are more incompatible software and hardware problems.

Developers build features into presentation software and test it in controlled environments. One imagines that developers have wonderful resources — nice rooms, expensive projectors, WIFI that works, and so on. Certainly, their *technical* resources must be good. They are relaxed as they test their latest animations. Mess up, and they will go back to their office and fix the bug ...

In contrast, a user might be applying for a job. They are stressed, as they *have* to get the presentation to work first time in an unfamiliar environment, and their career depends on this one presentation. Mess up and they won't get a job offer.

Developers often do and attend very structured presentations, and their software reflects this. Real speakers often give talks at conferences that are running late, and the chairman changes the schedule and timings at the last minute. How does a speak rearrange their talk in front of the audience, without revealing their whole story line?

The big picture of using computer technology to help prepare and give presentations has turned into a "tunnel vision" (concentration to the exclusion of distractions outside the area of focus) of having fancy presentation software that makes a very small part of the whole task amazing. Speakers end up focusing on the presentation *software*, not on the presentation or the audience who engage with it; developers ignore the whole task lifecycle, since it is not their job to do anything other than develop cool features for the presentation software.

Typical HCI techniques, such as User Centred Design (UCD), A/B comparisons and more focus on the user and their existing tasks. UCD may explore what users want — but they will want better presentation software! A/B comparisons will help find better design features (comparing A against B) — for presentation software. Even getting embedded with users preparing presentations and delivering presentations will find out issues that users want improving. Of course, the real "users" are the audience, and their perspective will be very different.

But what if we have the task wrong? However much the task is improved, whether by pure software engineers making cool features, by HCI professionals making features that better fit what users want, or UX professionals making features that are more satisfying, we just improve the task *as perceived* better, rather than improving the task.

1.1 Implementation bias and the Double Diamond

A common problem in software engineering is that programs are developed that pay too much attention to the particular hardware or implementation they are to run on. Software that runs fantastically on an iPhone but not on an Android exhibits implementation bias.

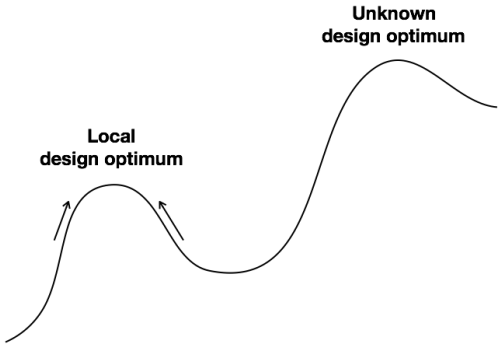

Fig. 1. Hill climbing aims upwards from where ever it finds itself, and thus gets to the top of the current hill, but may not get to the top of the highest hill. Although hill climbing gives the impression of improvement, time spent chosing the right hill to climb is a good investment — for an alternative visualisation, see the Design Council's Double Diamond, shown in figure 2.

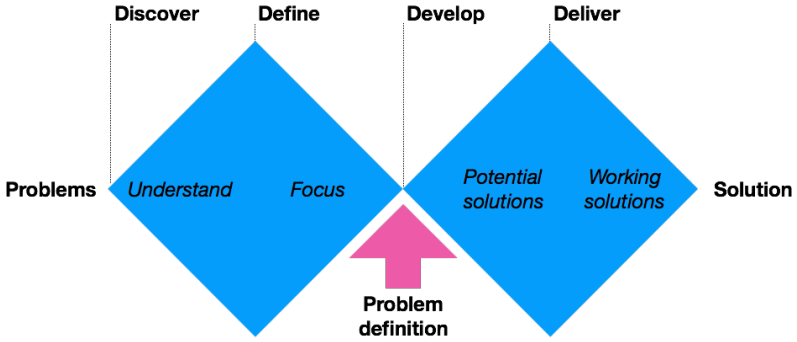

Fig. 2. The UK Design Council's Double Diamond (redrawn from [1]). The first diamond emphasises finding the right problem to solve, and hence avoids the common problem of finding the right answer to the wrong question.

And as developers "improve" something for one platform, the chances that it works well on another diminish. Worse, when Apple improves their iPhone, it is likely that the original iPhone implementation bias not only made the software not just dependent on iPhones but dependent on a particular model or operating system version.

Implementation bias is seductive; it presents a soluble problem, but by focussing on that problem developers create a bigger problem: dependency on a particular implementation.

Likewise, HCI broadly suffers from the temptation of implementation bias. We see the implementation of a task in a particular system, and then recruit HCI to improve that system. In giving presentations, the temptation is to see the system being PowerPoint (or whatever) then recruiting HCI to improve that system. No doubt we can improve that system, and HCI provides many techniques (eye tracking, for instance) that are very satisfying to use and will deliver measurable improvements. But we are implementation biased. We are not improving presentations, but particular software.

HCI is not about making computer systems better for people because the existing computer systems may be starting in the wrong place. If we improve the current system, we are hill climbing, a well-known limited approach to problem solving (figure 1). A more mature approach is to use the Design Council's Double Diamond approach [1] — the first diamond is a process that ensures we are solving the right problem, rather than optimising the wrong problem (figure 2).

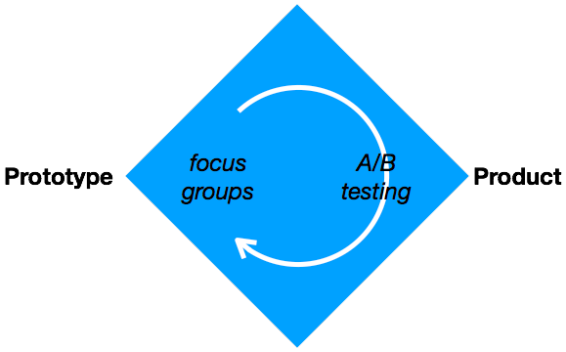

Fig. 3. Conventional HCI starts from the assumption that some computer-based prototype or sketch must be refined and optimised, perhaps through creative processes such as focus groups with users, then convergent processes such as A/B testing, and iterating until the desired performance is achieved. The conventional approach to HCI is only one of the two diamonds shown in figure 2.

1.2 The case of healthcare IT and HCI

Healthcare IT is an area that reflects these tensions well. Healthcare, in the modern sense of hospitals and professionalised medical care, has matured over centuries and has a deeply entrenched culture. For example, while "patient care" might be the avowed goal, surgeons, nurses and physicians — to name a few — live in different worlds. Computerising healthcare immediately leads to numerous problems, such as lack of interoperability (e.g., physiotherapists and radiographers have systems that won't talk to each other); yet it seems self-evident that computerising with modern technologies such as tablet computers should deliver benefits we all experience in the rest of our lives — such as social media, ecommerce, teleconferencing and more. Some countries are driving their healthcare systems to go paperless, but this can be achieved by scanning in paper documents and turning them into JPEGs — which goes paperless, but makes the information harder to manage. Piles of paper are a symptom of a problem, and "computerising the paper away" does not solve the underlying task problem; it just hides it.

Using HCI in the healthcare enviroment is unlikely to tackle the cultural re-engineering that is required to improve healthcare. Indeed, hospitals might be better off keeping patients healthy and out of them; making hospitals more efficient by computerisation merely perpetuates the out-moded model of the patient as a passive recipient of care *after* a medical problem. Prevention is better than cure, but HCI would need resources and a powerful conviction to improve healthcare so broadly conceived.

It is unlikely we can recruit HCI to improve healthcare in a short article; we can point out the problem of myopic tunnel vision on *apparent* inefficiencies, where clinicians complain and demand improvements, but the task is bigger than what any users are actually doing. A cardiologist wants better user interfaces for their defibrillators, and while it is worth improving them, the real HCI issue is that patients don't want heart problems in the first place. In fact, patients don't want to be patients.

Further discussion about healthcare IT and HCI concerns can be found in [2–5]; we now turn to presentation HCI.

1.3 The case of presentation IT and HCI

Healthcare, then, is a massive problem, and conventional HCI working at it will fall into the trap of hill-climbing, rather than finding a better solution for computers to help with

- How do you get the projector to work with the PC?
- How do you display the right screen? (Often there are two screens.)
- How do you not reveal the entire presentation while setting up?
- How do you connect the presentation controller? (Worse if it is wifi based.)
- How do you go backwards and forwards to answer questions, without rebuilding slides?
- How do you skip slides?
- How do you sort out a problem without losing the audience's screen picture?
- How do you stop virus and other upgrade interruptions?

Fig. 4. A small sampling of soluble presentation HCI problems. All of these are serious problems for the lecturer and will undermine their confidence in front of an audience, yet no features in current presentation tools address these issues. There are others, for example when creating presentations — when the lecturer is less stressed — the tools could provide a lot more help and training, for instance: "here is a feature you haven't used yet and which will be useful…" This figure continues in figure 5.

the actual task.

In contrast, we (almost) all do presentations, and we are pretty much free to do them as we like. We have analogous cultural problems: most of us have sat through poor lectures, and few of us have experienced outstanding presentations. And when we do, being able to deliver memorable talks with humour, presence, confidence, and so on, seems beyond us. If we think about it, we would like PowerPoint to be more reliable loading media files — but improving PowerPoint may not be improving the task.

The companion paper [6] analyses and explores the task of preparing and delivering quality presentations. Crucial to our approach is that we are analysing quality presentations; we are not analysing what most speakers do, or what most speakers want, or what most presentation software provides. We are interested in the underlying task, and how to improve it, without implementation bias (i.e., without prematurely focusing on any particular technology).

Virtually none of the analysis in the companion paper [6] depends on PowerPoint or its competitors — in fact, we will expose how such tools distract a presenter from thinking clearly about their presentation. Or, rather, presentation tools make the speaker think too much about content *in* the tool, rather than delivering content effectively *to* an audience. These are different tasks, and we will argue that HCI has made the wrong task easier — and PowerPoint (and its equivalents) have become scapegoated, thus, ironically, increasing pressure to use HCI to improve *them* and not the task!

2 THE SYMPTOM OF PERSISTENT HCI PROBLEMS

Presentation tools have been around for decades, since before Aldus Persuasion 1.0 was released in 1988 to a ready market [7]. They have been developed by leading programmers, leading usability groups, and by major international corporations. Yet they still have usability problems (a sample is listed in figure 4). If so many easy-to-fix problems remain after decades, it implies that the market (that is, the presenters and their budgets) have been

- Why isn't there any sound?
- I've overrunning, how do I skip to the end?
- In an interactive talk, how do I type audience comments onto the slide?
- The projector is a different aspect ratio; how do I check the talk? (without showing the entire talk to the audience!)
- My battery is going flat. What should I do?

Fig. 5. Continued from figure 4. A small sampling of soluble presentation HCI problems, continued.

misdirected to look in the wrong place — people believe that the solution to their problem is a *product* and, once bought into that product, that solutions are upgrades to that product. Indeed, there is a self-fulfilling ratchet effect, "Aren't you using the latest version? How can you possibly project my version if you haven't upgraded?" as consumers vie with each other to lead the early adopter pack. And once a lecture theatre succumbs to upgrading, everybody who uses it has to upgrade.

While we are misdirected into thinking solutions to presentation problems are *entirely* to be bought in products, we will fail to address the full scope of the task. Training, voice skills, design skills, handouts, writing, planning and more. The companion paper explores some of these issues, and the critical role they play in quality presentations.

It is perhaps fine for lecturers to be misdirected; they are just consumers in a lively marketplace, after all. But the problem for HCI is that we, the HCI practitioners and the software developers too, also live in that other world where we are not professional HCI experts, but simply consumers of stuff — in the case here, stuff for giving presentations. We have to be extremely self-aware to be able to stand back and see HCI issues for what they are. It may seem obvious that the consumer products we use are wonderful — but unless we do careful experiments, we simply do not know. We are susceptible to cognitive dissonance, attribute substitution, denial and rationalisation just like everyone else.

3 CONCLUSIONS

HCI is not about making computer systems better for people; it is, at root, about the harder task of finding out how to make tasks better for people (and not necessarily the tasks we started with, but finding, evaluating and creating better tasks that achieve the higher goals) and only using computers if we must.

The case of giving talks and presentations was used as a case study (full details in the companion paper, [6]) to show that better talks are created and delivered by considering much higher-level goals than are apparent *within* the conventional HCI perspective of interactive computer systems such as PowerPoint. Using HCI to improve PowerPoint risks making PowerPoint (or whatever system is fixated on) even more of a trap that will further divert speakers away from performing their task with the audience well. Seductive, powerful user experiences may not actually be what users need to perform their tasks effectively and enjoyably. Perhaps we have been misdirected into consumerism — an HCI where it is all about products — rather than an HCI that is all about empowering users. Like improving presentation software, "improving" healthcare IT systems and healthcare HCI diverts attention from the higher goals of healthcare. Most of us will be patients (if we don't die first), most of us (reading this paper) will give presenations.

When our professional HCI life overlaps with our personal life, as it does, we have to be very careful to be aware of the HCI issues that remain. Both as patients and as presenters we end up in a sort-of learned helplessness; this how it is.

Fortunately, we can all easily work to improve our presentations regardless of the interactive platforms we use for presentations; but the task of healthcare professionals — and HCI experts and developers who support them — which was touched on in section 1.2, is far more important and far more constrained — health is at risk and lives are at risk. Here, we *have* to think more clearly about HCI. When we consider healthcare IT and how HCI can help, we must lift the focus of improving IT systems to improving healthcare (a longer paper may have explored HCI in other safety critical domains, like aviation and diving).

Perhaps people reading this paper will, I hope, go on to try to make better presentations about improving healthcare. If so, then everybody will benefit.

ACKNOWLEDGEMENTS

This work was supported by EPSRC grant no [EP/L019272/1] and by See Change (M&RA-P), Scotland. Author's address: H. Thimbleby, Swansea University, Wales. Author's URL: `www.harold.thimbleby.net`

References

[1] Conway, R., Masters, J. & Thorold, J. *From Design Thinking to Systems Change* (Royal Society of Arts, 2017).

[2] Thimbleby, H., Lewis, A. & Williams, J. Making healthcare safer by understanding, designing and buying better IT. *Clinical Medicine* **15**, 258–262 (2015).

[3] Thimbleby, H. Improve IT, improve health. *IEEE Computer* **50**, 86–91 (2017).

[4] Thimbleby, H. Trust me, I'm a computer. *Future Healthcare Journal* **4**, 105–108 (2017).

[5] Ovretveit, J. *et al.* Using and choosing digital health technologies: A communications science perspective. *Journal of Health Organization and Management* **31** (2017).

[6] Thimbleby, H. Pirate talks. *Journal of Interaction Science* (2017).

[7] Miller, F. P., Vandome, A. F. & McBrewster, J. *Adobe Persuasion* (Alpha Press, 2010).