HCI and Theology: Chalk and Cheese?

by Steve Clough

Introduction: I'm Doing *What*?

Why on earth would I want to consider pursuing a research degree in human-computer interaction (HCI)? That is a good question that I still do not have a good answer to, but my choice seems to come out of my educational and life experience. I started fairly traditionally by getting a computing degree and then going out to work. For the last 20+ years, I have been working in the software development business, in a range of roles and organizations. In a number of these roles, a common theme emerged—that I was writing applications for people who were not always computer literate. It also became clear that many of the computer applications that I was involved in had negative effects on many people—it put them out of a job, it made their job different, or, often, they would be reduced to a computer operative, with many of their skills replaced by a computer. As the computer business changed and more applications moved toward PCs, the situation became worse. As computer applications became far more common, more people who did not understand computers were forced to use them—but the application design, in terms of how they interacted with people, did not change. In fact, I remember one occasion where I was being encouraged to design applications in a way that mirrored mainframe, menu-driven tools, rather than designing a more appropriate interactive interface. These applications made people who knew their jobs perfectly well feel stupid simply because they could not appreciate the computer applications that they were expected to use. And, to be honest, many of the applications were rubbish.

Partly due to my frustration at this inattention to the needs of the user, I began studying again. I followed my interest in theology and eventually obtained a degree in that as well. Because I did this in my own time, there was a necessity to integrate what I was learning about with the world in which I was involved. To a significant degree, this meant that I needed to understand my work designing applications from a theological perspective; I needed to be able to see my work from the view of how it matched my growing theological understanding, especially relating to how God, the world, and people interacted. While this proved a challenge, it was also an interesting challenge that helped me both in my studies and in my work. Most significantly, I learned not to define myself by my work, because who I am is far more than just relating to the job. The job—the career even—is just one aspect of me and needs to appropriately integrate into the other aspects. My theological studies helped me see more about not only God, but also people.

Theology and Computer Science–Where Do They Meet?

The challenge I faced when trying to identify an area of research was to draw the two areas that I have qualifications in (computing and theology) together. At first glance, these are not compatible; one is a philosophical topic, looking at beliefs and ideas, whereas the other is rooted in science, requiring rigorous empirical techniques. While it is true that pursuing a PhD does involve philosophical insight into a topic, was there an area of study that could genuinely combine these areas?

One of the areas of theological study that I found most interesting was relating the Christian message to people in a post-modernist society [4]: how to take a message and relate it to people whose world view was not wholly open to these ideas—how to engage people with the message of Christianity when they are rejecting the metanarratives of the Church. And I realized that examining the interaction of people with computers, especially people who are not especially computer literate, could fit into a similar mold. The challenge was to look at how to take the messages and information that the computer application was producing and present them to the users in a way that was comprehensible to them. In particular, how can you explain to people what the computer application is doing when they neither have nor need any understanding of the inner workings of the application?

From this point, the similarities between theological study and computer application interface design became even clearer to me. I realize that there are huge differences, not the least of which being that this theology has developed over the past 2000 years across nations and cultures, whereas computer science is a relatively modern field, substantially developed in the West. But the chronological and cultural divide that is at the heart of theological study can be matched to the technological divide in computing. Just as most people today do not speak ancient Hebrew or Greek and do not naturally relate to the cultural environment of first century Israel, most people who use computers do not understand the technology that is behind what computers are doing. More significantly, they should not have to. There are those of us who do understand and can work with the technology. It is our role to interpret applications' processing, requirements, and information in a way that is relevant and appropriate to the end-user.

Many of the challenges are also relevant. If the original materials are to be represented to people who do not understand them, it is important that this is done in a way that is not only relevant and appropriate, but also true to the original. People are not entirely objective, whether it comes to explaining biblical messages or interpreting the results of experiments. While a a certain degree of subjectivity is inevitable, we need to acknowledge this bias and try to interpret and understand without letting the bias get in the way. This honesty, openness, and care is important if you are to accept my assistance in understanding the truths of a faith that is as divergent and historically distant as Christianity is. It does not mean that I am always right; in dialogue, we can both learn from each other, and you can learn more about why certain emphases are given. Similarly, for application interface design, you need to trust that the designer is interpreting the information appropriately, that what is displayed actually reflects what is held on the database, that a lack of warnings is an indication that there is nothing you need to be aware of, that when you ask the computer to do something, it will do it, or tell you why it will not, in a way that you understand. There will always be some bias in interface design. This bias is not a bad thing as long as it is acknowledged by the designer and does not prevent full information about the computer's state from being available to the user. A degree of personal customization is also useful so that you, the user of the application, can change the way that information is presented to you, so that it works for you. In addition, as you better understand what the application is doing, you may want to see more details.

What became clear is that the focus from my theological studies on people was also something of major significance for computer applications. If I ignore the personal aspect of my designs, then however well they work functionally, they will not be positive experiences for users, and the applications will devalue people in their working environments. As computers and languages became more and more powerful, the challenges of programming the functional requirements of an application became much more achievable. However, the complex way in which applications interacted with people did not progress to the same degree, although it did become more important. Interestingly, the view of the "user" has not developed that significantly from my early days in the business; users are still often seen as the people who just need to get on with using the application, whether they like it or not.

Software Applications as if People Matter

With all of these similarities, how can I draw a research proposal and approach that is consistent with and, just as importantly, academically acceptable in the computing field? The challenge for me was not simply applying the lessons learned in my theology studies to the field of computing. The real challenge was identifying an area of research in computing to which I could apply the insights of my theology study. I explored the field of HCI to see if I could look at an approach to application design from the starting position of people rather than the application—software design as if people matter.

I wanted to look at software applications in relationship with the people using them. One of the most significant aspects of a relationship is communication; as BT used to advertise, "it's good to talk." Therefore, communication with the user can be considered of prime importance. Communication is not just telling or listening. It is about "body language," including how messages are expressed and how the application looks. It is about how things are said and when things are said, even whether they are said at all. If the communication is right, then the relationship works much easier. If it is wrong—if attention is always being demanded, if things are expressed wrongly and in a distracting way—then the relationship will be strained.

Thus, an application that does not know how to communicate properly, that does not know how to ask for information and pass appropriate comments back, is likely to be seen as hard to use. Communication is complex, involving both listening and talking. From a computer application perspective, the application must not only tell the users what is going on but also handle input from them appropriately. Listening to users, taking some notice of what they do and how they work, is a part of communication. Enabling an interface to be flexible, to adapt (or be adapted) to the way a person wants to work makes it seem more friendly, more usable, and the user will be far more tolerant of issues and problems with the application [1].

Shneiderman says, "It is not as useful to think about what computers can do as about what users can do." [6]. This is a challenge to software developers: to stop being limited by what they perceive the computer to be capable of and be inspired, instead, by what people can do. To ignore the psychological aspect of work environments is to miss out on most of the impact that a software application can have; it should be that the software tools that are used have a positive effect on the working atmosphere, not—as so often is the case—a negative one [2]. The software tools—as they are seen, which means the user interface—should enhance and help the working environment that it is being used in. It should enable you to do a better job. Too often, they fail in this regard. Again, Shneiderman indicates that, "[...]it is time to get angry over the quality of user interfaces" [6]. This applies to all computer software users, whether or not they are computer-literate. So often the response in project teams to UI change requests is, "That is a minor issue. We will try to do it if we have time." Maybe we should be saying, "That is a minor issue, but fixing it will make a big difference to the users. Let's fix it." Seeing end-users as important stakeholders in software applications, not only as the managers who hold the purse strings, would impact the usability of software applications significantly.

Computer applications as if people matter, "technology with a human face" as Schumacher puts it [7], is not just about making applications look nice, pretty, curvy. It is about making them with genuine consideration for the users who will be utilizing them and enabling them to do what they do better, not just in a different way. It is about letting people exercise their own choices, opinions, and views on how they wish to interact with and use an application. It is about humility, admitting that, even as computer professionals, we don't know it all. In fact, when it comes to relating to people, we often know very little.

Work: The Ignored Area

Because of my work in writing e-commerce applications, I was aware of one area that seemed to be missed in most design work yet was one of the parts of the application that was most used. A lot of designers' time and effort would go into the building of the Web site. This is quite right, because this was the shop front. This is what people see of the client. This is the marketing aspect for the client. But the back-end of the application, where the processing of the orders takes place and where the management of the Web site happens, is often ignored.

It is quite understandable in the virtual world as well as the real one that money is first spent on selling items before the usability of the back-end systems. As long as they are functionally correct, anything else is additional, and when budgets and deadlines are tight, additional aspects tend to get removed. From the perspective of application design, however, the back-end applications are likely to be used by staff for a lot of their working day. This means that usability issues that may be frustrating for customers of the Web site can become more than irritations; they can make a system unusable.

It was as I was looking for a starting point for my methodology that I realized the philosopher who could provide some important insight was Marx [3]. Marx was writing about the oppression of workers in the second half of the 19th century, circumstances that are very different from any working conditions today. But the insights he brought about the roles that workers have within organizations, about the commercial pressures and—of crucial importance—the importance of people as people, the importance of valuing people in themselves are still relevant. Possibly the most important aspect of this is the importance of people as producers of value in a working context. Work adds value to raw materials, to make them worth more. In the context of information working, the added value is not the computer reports and information; that is the raw value of the data. The added value that people give is in the interpretation of the data and the selection and accumulation of the information being worked on. People doing their jobs and using computer applications to assist them are adding value to the information that the computer holds. The cleaner the interactions, the better the value that is added. This is in much the same way that good quality implements make the process of producing artifacts easier, thereby improving the quality of the produced artifacts, giving them higher value.

Some of the comments in the previous section relate in general to computer application interaction, so why is work any different in this context than any other interaction? The reason is, as Marx saw, workers have much less choice in their working environment than they may have outside it. The market economy approach dictates that those who produce the best products will succeed, and those who do not, will fail. In other words, only the best survive. However, the problems with this are the definition of "best" and, crucially, who decides what is "best." If I want to go out for a meal, I could, in theory, go to any restaurant that I could reach in time. However, I will restrict this for a variety of reasons: I probably don't want to travel far. I might want Italian food. I will also probably have budgetary constraints. All of this means that I might accept a take-away pizza because that fits my current requirements. It may be the best fit, in terms of what I can afford, with my other restrictions. However, my wife, once I had said that we should eat out, might not accept this as being a valid option. Similarly, when businesses choose software and systems to use, they may have a set of restrictions, such as financial requirements, support contract needs, or performance restrictions. Such decisions are rarely, if ever, made by the people who will be using the software every day. Their "choice" is severely restricted because they have to use it, and they have to do their jobs using it. Complaints are often not heard, because the day-by-day usability is typically not a factor taken into account in purchasing decisions.

Conclusion

My exploration into a computer topic, having developed out of my theological interests, stretched to include Marxist theory too. It all seems like a long way from the scientific background of computing. Yet, the more I worked on it, the more I realized that this was the essence of research study. I was no longer simply exploring the rigorously defined subject, I was looking at my subject from a wider viewpoint. I was applying alternative insights to the discipline, thereby bringing something unique to it, something that was very much defined by who I am and where I have come from.

By looking at HCI from the perspective of people, not computers, and analyzing the genuine usability of existing applications, further evidence can be provided that well-thought-out interfaces and interactions can make a big difference to the genuine usability of software applications. Such interfaces can amplify the degree of fulfillment that people get out of their use of them, or at least the tasks that they are having to perform using them. The work-based focus provides an interesting angle, because the insights of Marx and others who discuss and relate to the whole field of work come into play. But it is also a place where the normal market forces that drive product development are, to an extent, limited and restricted by requirements other than end-user usability.

I hope to continue to gain fresh insights into the odd and murky area of human relationships with computer applications. By studying part-time and working the rest of my time in software development, I am hoping to bring some of my insights into play in the applications I write. At the least, it should keep my feet on the ground so that I do not follow wild flights of fancy that are impractical. The practical aspects of computer applications are what have driven me from the start, and I have no intentions of letting them go now.

References

1. Csikszentmihalyi, M. 2002. *Flow: The Classic Work on How to Achieve Happiness*. Rider & Co. 143ff.
2. Luczak, H. 2006. Task analysis. In *Handbook of Human Factors and Ergonomics*. Salvendy, G., Ed. John Wiley & Sons. 384–385.
3. Marx, K. 1946. *Capital*. J. M. Dent & Sons, Ltd.
4. Newbigin, L. 1977. *The Gospel in a Pluralist Society*. Wm. B. Eerdmans Publishing Co.
5. Shneiderman, B. 2002. *Leonardo's Laptop: Human Needs and the New Computing Technologies*. MIT Press. 63.
6. Shneiderman, B. 2002. *Leonardo's Laptop: Human Needs and the New Computing Technologies*. MIT Press. 31.
7. Schumacher, E. F. 1973. *Small is Beautiful: A Study of Economics as if People Mattered*. Harper. 123.

Biography

Steve Clough (s0113066@glos.ac.uk) is studying part-time for a PhD in human computer interaction at the University of Gloucestershire in England. He works full-time in e-commerce web development at Snow Valley (www.snowvalley.com). He is 45, married, has two teenage boys, and lives in St Albans.

We Want You

to join the *Crossroads* staff

Limited Openings for Associate Editors

Crossroads is looking for qualified Associate Editors to join our talented staff. AEs are involved throughout the editorial process, from reviewing submissions to working with the authors and copy-editing final drafts. Extensive computing experience not required. *Crossroads* is a fairly low-key commitment with all work conducted over the Internet. Opportunities for advancement are available. Positions in marketing, graphics design, and web developers are also open.

Be a part of the student magazine of the first society in computing!

For details, please email crossroads@acm.org

Make a difference...

CHANGING THE WORLD

Sandia National Laboratories is for people with a hunger and desire to make a difference, a passion to change the world. We're one of the top science and engineering laboratories for national security and technology innovation. Be a part of the team helping America secure a peaceful and free world through technology.

We have exciting opportunities for college graduates at the Bachelor's, Master's, and Ph.D. levels in:

- Nuclear Engineering
- Mechanical Engineering
- Electrical Engineering
- Computing Engineering
- Material Sciences
- Information Technologies
- Computer Science
- Chemistry
- Biological Sciences
- Business Administration ... and more

We also offer internship, co-op, and post-doctoral programs.

www.sandia.gov

Sandia is an equal opportunity employer.
We maintain a drug-free workplace.

Sandia National Laboratories
operated by
LOCKHEED MARTIN