# Chapter 3
# Trends and Gaps

**Alan Dix, Benjamin Weyers, Judy Bowen and Philippe Palanque**

**Abstract** This chapter attempts to identify future research directions for formal methods in HCI. It does this using two main approaches. First, we will look at trends within HCI more broadly and the challenges these pose for formal methods. These trends in HCI are often themselves driven by external technical and societal change, for example the growth of maker/hacker culture and the increasing dependence of basic citizenship on digital technology, effectively establishing external requirements for the field. Second, we will look inwards at the FoMHCI literature, the user interaction phenomena it is trying to address and the processes of interaction design it is intended to support. Through this second analysis, we will identify internally generated trends. This does not lead to a single overarching research agenda but does identify a number of critical areas and issues, and hence establishes opportunities for further research to expand the state of the art.

A. Dix (✉)
University of Birmingham, Birmingham, UK
e-mail: alan@hcibook.com

A. Dix
Talis Ltd., Birmingham, UK

B. Weyers
Visual Computing Institute—Virtual Reality & Immersive Visualization, RWTH Aachen University, Aachen, Germany
e-mail: weyers@vr.rwth-aachen.de

J. Bowen
Department of Computer Science, The University of Waikato, Hamilton, New Zealand
e-mail: jbowen@waikato.ac.nz

P. Palanque
IRIT—Interactive Critical Systems Group, University of Toulouse 3—Paul Sabatier, Toulouse, France
e-mail: palanque@irit.fr



B. Weyers et al. (eds.), *The Handbook of Formal Methods in Human-Computer Interaction*, Human-Computer Interaction Series,
DOI 10.1007/978-3-319-51838-1_3

## 3.1 Introduction

This book attempts to capture a snapshot of the state of the art of research in formal methods in human–computer interaction. In this chapter, we ask where it is going and where it could go. Where are the gaps, the opportunities and the challenges for the next decade?

The first part of the chapter looks outward, at broader changes in HCI, how technology and the role of technology is changing, from big data and the Internet of Things to the pivotal role of information technology in modern society.

The second looks more inward at the role of formalism: which aspects of user interaction are being studied and modelled, how formal methods fit into and are addressing different parts of the design and deployment process, and how the techniques and methods within formal methods in HCI are changing.

For each issue, we attempt to identify the challenges this poses for formal methods research in HCI and so create a road map, especially for those starting in the field looking for open questions and avenues for theoretical and practical research.

## 3.2 HCI Trends

The roots of HCI can be traced back many years, indeed the first true HCI paper was Brian Shackel's '*Ergonomics for a Computer*' in (1959); however, the discipline formed properly in the 1980s including the foundation of many of the major international conferences. The study of formal methods in HCI can be traced back to this same period including Reisner's (1981) use of BNF, Sufrin's (1982) Z specification of a display editor and the first PIE paper (Dix and Runciman 1985).

This flowering of HCI research of all kinds was closely aligned to the growth of the personal computer, which moved computing from the domain of a few technical specialists behind sealed doors, to ordinary professionals on their desktops. In the intervening thirty-five years, both technology and the use of technology have changed dramatically. Many of the old concerns are still important today, but there are also new directions in HCI and these create new challenges for the use of formal methods.

In this section, we will trace some of those trends based largely on a recent JVLC article that examined future trends of HCI (Dix 2016). We will then use these general HCI trends to highlight some of the formalisation challenges they present. The trends are divided into three kinds: changing user interaction, changing technology, and changing design and development. Of course, these are not independent; indeed, as noted, the whole discipline of HCI effectively grew out of a particular technological change, the introduction of the desktop computer, and this close interplay between technology and use has continued. While available technology does not necessarily determine the use of that technology, it certainly makes new things possible.

### 3.2.1 Changing User Interaction

The use of computation has evolved from the desktop to virtually every aspect of our day-to-day lives.

***Choice and ubiquity***—In the early days of HCI, computers were largely used as a part of work, your job and employer largely dictated whether you used a computer and, if so, the hardware or software you used. The justifications for usability were therefore essentially about the efficiency of the workforce. The rise of home computing in the 1990s, and particularly the growth of the Internet in the early 2000s, meant that the range of users was far wider, and furthermore, the users were customers and had choice—if applications were not usable and enjoyable, they would be rapidly dumped! This process has continued as prices and form factors have made computing (in some form) available to ever-wider groups worldwide. However, in recent years, this ubiquity has meant that computer access is assumed. In commerce, Internet shopping is not only commonplace, but it is usually the cheapest way to obtain many items; for example, online check-in can be far cheaper than at the airport. In civic society, e-Government services are not only common, but some countries are seeking to make them the only way to access certain services, for example, in the UK, the so-called universal benefits, which integrate many kinds of different welfare payments, can only be accessed via an Internet portal, despite low levels of digital literacy among precisely the social groups who are likely to be claimants (Citizens Advice Bureau 2013; Sherman 2013). That is computer, and in particular Internet, services are beginning to underpin society, so that digital exclusion becomes social exclusion; use is no longer a choice but a necessity for participation in civic society.

*Formal challenges*: The change from optional to necessary use of digital services makes it more important to be able to deal with all kinds of people and also limited digital access. One of the strengths of formal methods is that it can help us analyse and design for situations and people we do not naturally experience or understand. The way this works out for people and for devices and contexts is explored in the next two trends.

***Diverse people***—HCI has always had strands that focus on those who are different from the 'norm': those with varying abilities and disabilities, different cultures or different ages. Of course, this 'norm' has been contested; for example, the tendency to use undergraduate students as the principal subjects in experiments has meant that what is assumed to be universal human behaviour turns out to be very biased towards Western culture (Henrich et al. 2010). Proponents of universal design have long argued that we are all 'disabled' under certain circumstances, for example effectively blind to dials and controls while our visual attention is on driving, and therefore that if design that is good for those with some form of perceptual, motor or cognitive disability is in fact design that is good for all. Although for many years forms of anti-discrimination legislation have made 'design for all' mandatory, it has still remained a marginal area.

This has long been problematic, but now, because computation is becoming essential for day-to-day life, it is impossible to ignore. The necessity of access combined with ageing populations in many countries means that universal access is now essential. This is exacerbated in many countries where ageing populations mean that some levels of perceptual, motor or cognitive impairment are now 'normal' and universally where broadening societal use includes those with low digital literacy and indeed low literacy, including the so-called next billion users in the developing world. Increasingly, we need to consider those at the social, geographic and economic margins, not just the professional 'class A/B' users of the 1980s.

*Formal challenges*: Within the formal methods community, work on multi-modal systems and various forms of model-based interfaces (e.g. Chap. 18; Coutaz 2010; Meixner et al. 2011), go some way to addressing these issues. In professional development, internationalisation is a normal practice for product delivery, and ability checklists are used to tune technology to individual abilities (Dewsbury and Ballard 2014; Whittington and Dogan 2016). The latter are often 'formal' in the sense that they have codified knowledge, but whereas most work on formal methods is based around relatively complex analysis of relatively simple specifications, practical development has relatively large corpora of codified knowledge, but with very simple, tick-box-style reasoning.

There are clear opportunities to create user models that encompass the wide variations in human abilities, and formal technical challenges to combine the kinds of codified knowledge already available with other forms of formal reasoning.

***Diverse devices and contexts***—The mobile-first design philosophy has for some time emphasised that for majority users mobile devices may be their principal, or in the case of the 'next billion' possibly first and only, access to computation. Commercially, the growing range of devices commonly used now includes smartphones, tablets, smart TV, game consoles, and public displays as well as various forms of laptop or desktop computers. Perhaps as important, when we look at need for universal access 'at the margins', we have to consider poor network connectivity, 'last generation' technology and in many areas intermittent or total lack of power.

*Formal challenges*: The challenges of designing for multiple devices are being dealt with fairly well, both in the formal community with work on plasticity and professional practice, notably responsive design. More generally, this suggests we need methods that model both device characteristics, and their environment. Chapter 12 is a good example of the latter, capturing interactions between environmental aspects such as noise, with device modalities, such as audible output. It would be good to see these areas of research expand, in particular to include infrastructure context such as limited networks or power, not just screen size. We often do not even have adequate vocabulary for these: for example, rural networks often experience frequent short glitches, complete drops in connectivity for a few seconds or minutes; with no word or formal metric for these drops, adequate service cannot be specified in the way that bandwidth or latency can. Theoretical work in these areas may have very immediate practical benefits; despite the widespread focus on responsive design, it is common to have failings such as drop-down menus that do not fit on small-screen devices, and even when executed well, responsive

design rarely breaks the mould of simple screen size, rather than more radical modification of the interaction style to fit the device and infrastructure context.

***Physicality and embodiment***—One of the defining features of early user-interface design was the identification of key abstract interaction primitives, not least the windows, icons, menus and pointers of WIMP. Having these abstractions made it possible to easily design applications in the knowledge that while the details of how these primitives appear and behave may vary between specific devices, they can be assumed to exist and in some ways isolate the application from the vagaries of specific devices and even operating systems. On the other hand, as Apple have exploited particularly well, there has always been a close relationship between physical design and software design.

In more recent years, various factors have made the physical, embodied and situated nature of digital technology more significant. Some of this is connected with new interaction modalities such as bodily interaction with Kinect or geo-spatial interaction such as Pokemon Go. Ultrahaptics now means it is even possible to give holodeck-like mid-air haptic feedback (Carter et al. 2013). In addition, as computation has become embedded in everyday objects and the environment, it becomes hard to separate the digital and physical design: this is evident both in research fields such as tangible user interfaces (Ishii 2003) and ubiquitous computing (Weiser 1991), as well as practical design such as screen-based washing machines, or public displays.

*Formal challenges*: The abstraction offered by WIMP has been helpful in formal specification, which has typically been able to operate well above the physical interaction layer of ARCH/Slinky (UIMS 1992; Gram and Cockton 1996). There is some work that deals with the more physical nature of devices including Eslambolchilar's (2006) work on cybernetic modelling of human and device interactions, Thimbleby's (2007) work on physical control layout, physigrams as described in Chap. 9 in this volume, and the use of space syntax and other formalisms for movement in the environment (Fatah gen Schieck et al. 2006; Pallotta et al. 2008). However, compared with more abstracted user-interface specification, this work is still nascent.

***Really invisible***—Weiser's (1991) vision of ubiquitous computing has computers becoming 'invisible'; however, this was in the sense that there are displays everywhere at various scales, but we are so used to them, they fade into the background. This is clearly happening, indeed it is an interesting exercise to walk around your house and count the displays. However, not all computers have obvious displays, and yet this computation embedded into the environment is becoming ever more common (e.g. a modern train has many hundreds of computers in each carriage controlling everything from lighting to toilet doors). Sometimes there is an explicit non-visual user interface, such as body interaction to control a public display or Star Trek-style voice commands. Sometimes there may be more implicit sensing leading to apparent effects, such as a door opening or light coming on. Some sensing may operate to facilitate user interactions in ways that are far less apparent, including the low-attention and incidental interactions described in Chap. 7.

*Formal challenges*: There has been some work in formal methods dealing with the architectural design of this form of environmentally embedded system (e.g.

Bruegger 2011; Wurdel 2011), some (e.g. Chap 7) dealing with non-UI interactions and some including models of the physical environment (e.g. Chaps. 7–9, 12, 15). However, like the related area of physical interactions, this work has nothing like the maturity of more abstracted direct interactions.

***Experience and values***—The shift in the 2000s from the professional to the domestic domain and the associated shift from employer decision to consumer choice meant that 'satisfaction', the oft-ignored lesser sibling of 'effectiveness, efficiency and satisfaction', began to take centre place. The most obvious sign of this was the job title changes from 'usability' and 'interaction design' to 'user experience'. However, this change in job title represented a more fundamental shift in focus towards the aesthetic and emotional aspects of design (Norman 2005). Furthermore, increasing scrutiny of the ways in which user interfaces permeate commercial and societal life has led to an examination of the ways in which values are purposefully or accidentally embodied in designs (Cockton 2004; Harper et al. 2008).

*Formal challenges:* While important trends, it is less clear, given the current state of knowledge, how these issues can be dealt with in a more formal way. One potential path might be to enable forms of annotation and argumentation around designs. Notations such as QOC or gIBIS allow the formalisation of argumentation structures, even though the semantic content of the arguments is entirely captured in textual labels and descriptions. As well as offering potential ways to document and track emotional and value aspects of a design, encouraging the user experience designer to create more formal descriptions could allow automated analysis of more workaday aspects of usability.

***Social and personal use***—Communication has always been a core part of computer use, from simple email to rich collaborative work; however, the growth of social networking has changed the dominant kinds of communication from functional to phatic. Furthermore, individual use of computers is often very personal, not least the collection of data related to health and well-being. From an interaction point of view, the focus is, as in the last issue, more about communicating feelings than information. From a governance point of view, there are increasing worries about the way information and images once shared cannot easily be recalled, the potential for abusive interactions, and the ways in which data analysis can be used by commercial and government bodies in ways which we had never imagined when simply posting a tweet.

*Formal challenges*: Several of the chapters in this book include multiple actors (e.g. Chaps. 13, 15), but dealt with largely in terms of the functional effects of their interactions. There has also been work formalizing collaborations in terms of beliefs (e.g. Ellis 1994) including the way this could be used to make sense of certain artistic installations (Dix et al. 2005). The most extensive formal work on the actual social aspects of interaction is in social network analysis, but to date this is entirely separate from interface-level analysis.

In the area of personal data, the earliest paper on privacy in HCI included simple formalism (Dix 1990), and there have been multiple works on privacy preserving frameworks and, perhaps most relevant, ways of exposing the implications of data sharing (Langheinrich 2002; Hong and Landay 2004). Issues of authentication,

security and provenance are heavily formalized; however, as with the more social aspects, this is currently in ways which are largely disjoint from user-interface specification and analysis.

***Notification-based interaction***—Most social network applications are strongly oriented around streams and notifications. These shift the locus of control away from the user and to the choices that the system makes of what to show and when to make it known. There is concern that this can lead to loss of focus and efficiency; indeed, a survey of American college students found that more than 90% reported that digital technologies cause some distraction from their studies, and 34% more serious distraction (McCoy 2016), and another study found that in-class cell phone use (presumably for texting) led to a drop of a one-third of a grade point (Duncan et al. 2012).

*Formal challenges*: The majority of formal user-interface specification techniques are oriented around explicit user-controlled interaction with only a small amount of work in the formal domain on interruptions (Dix et al. 2004) and dealing with interactions at difference paces (Dix 1992b). However, there is extensive non-formal literature on the impacts of interruptions and multitasking (Adamczyk and Bailey 2004; Czerwinski et al. 2004; Bailey and Konstan 2006) and the opportunity to find ways to match the pace and timing of delivery of notifications to the user's tasks (Dix and Leavesley 2015).

***Basic HCI***—Although we have had over thirty years of 'standard' usability, still we do not get it right! For example, Apple is often seen as the pinnacle of design, yet, when you turn on a MacOS or iOS device, the splash screen invites interaction (password for MacOS, slide to unlock for iOS) well before the system is ready to interpret your actions. Some of this is probably due to the shift of foci towards aesthetic and emotive design; some to do with the very success of user interfaces meaning more and more people are creating web interfaces in particular, but with less intense training and background than was once the case.

*Formal challenges*: Tool support could help this, and indeed, several of the Chaps. 17 and 18 describe tool suites that aid designers (although in some cases quite engineering-savvy ones) to develop and analyse user interfaces. Work clearly needs to be done still to improve both (i) the level and kinds of analysis so that they can truly be used as expert guidance for the novice and (ii) be targeted so that a designer without a formal/mathematical background can use them. The domain-specific modelling notations described in Chaps. 5 and 16 are one approach to achieving this.

### *3.2.2 Changing Technology*

New and emerging technologies pose fundamental challenges for people and society, with corresponding challenges for formalisation.

***Vast assemblies***—Smartphone users may have many dozens, even hundreds of apps, albeit the majority of interaction is with only a few. In the physical world, the

Internet of Things (IoT) promises to fill homes and workplaces with large numbers of potentially interacting small devices. Users will need means to manage and configure these devices, understanding how they can work together to solve specific problems. The larger the collection of devices or apps, the harder this will become. Furthermore, these vast assemblies of small items are likely to suffer from feature interactions, that is where two features, each potentially valuable in their own right, interact badly together. For example, imagine that your kitchen smoke detectors are programmed to increase their sensitivity when they detect the house is empty as no cooking is expected; however, if the Internet-enabled kettle turns itself on a few minutes before you arrive back from work, the steam may well set off the fire alarm.

*Formal challenges*: Dealing with large numbers of simple objects seems like ideal territory for formal methods. On the configuration side, Chap. 16 shows how workflow notations can be used to connect together apps to make larger functionality; this could be combined with techniques to infer or tangibly program connections (Turchi and Malizia 2016). Feature interactions have been studied for many years in telecoms, so there should be knowledge that could be borrowed and modified to deal with other kinds of complex assemblies.

***Big data***—Various forms of big data have been the subject of government funding, popular press and of course extensive commercial interest; this ranges from social networks, as discussed above, to large-scale science, such as at CERN. User interfaces to analysis tools and visualisation have some novel features, but in some ways not so dissimilar to relatively long-standing work in visualisation, data analysis and visual analytics (Thomas and Cook 2005; Keim et al. 2010). However, the vast scale does introduce new issues: how to get an overview of data that is too big to scan; how to track ownership and provenance; and new opportunities, for example the success of recommender systems.

*Formal challenges*: As with the discussion of social network analysis, while the tools for much of this are already formal, it is less clear how, or whether it is valuable, for these to directly connect to user-interface models, or whether supporting big data analysis is 'just' another application area. However, there is certainly great opportunity for big data to be used as part of the formal development process; for example, trace data can be mined to propose common interaction patterns and can be used as part of validation or to drive simulations. Also big data about applications domains could be used as part of knowledge-rich methods (see below).

***Autonomy and complexity***—Large volumes of data have led to a greater focus on complex algorithms to deal with that data including various forms of machine learning, not least 'deep learning' which has recently been used to master Go (Silver et al. 2016). Many problems that used to be thought to require rich symbolic reasoning, such as translation, are now being tackled using shallow but high-volume techniques (Halevy et al. 2009). However, these algorithms are often opaque, an issue that was highlighted concerning the very earliest machine-learning-based user interfaces, warning of the potential for unethical or even illegal discrimination and bias (Dix 1992a). As the use of these algorithms has become more common, these dangers have become more apparent leading to the General

Data Protection Regulation of the Council of the European Union (2016), which mandates that, for certain forms of critical areas, algorithms need to be able to explain their decisions (Goodman and Flaxman 2016).

The successes of machine learning have also led to a general resurgence of interest in the potential and dangers of intelligent algorithms. Sometimes this intelligence is used to aid user-driven interactions and sometimes to act autonomously. The latter have sometimes reached the popular press: for example, when scientists called for a ban on autonomous battlefield robots (Hawking et al. 2015) or when a driver was killed in a self-driving Tesla car (Yadron and Tynan 2016). In the case of Uber, even when there is a driver in the car, the driver's itinerary and fares are driven by computer algorithms, effectively high-level autonomy.

*Formal challenges*: The knowledge needed for algorithms to be both intelligent and explicable will draw on expertise from or similar to that found in HCI. Indeed, the UK funding body EPSRC (2016) has identified human-like computing as an important research area, deliberately drawing on cognitive science and human factors as well as artificial intelligence research. Some of this will be at a different level than the issues usually studied by those looking at formal methods in HCI, effectively providing application semantics for the user interface. However, the shifts of autonomy do need to be taken into account. There has been work on dynamic function allocation in cockpit and control situations (Hildebrandt and Harrison 2003), and a few chapters (e.g. Chaps. 7, 15) deal either explicitly or implicitly with more autonomous action sometimes simply at the level of opaque internal transitions.

### 3.2.3 Changing Design and Development

New ways of creating software and physical artefacts are altering the processes and people involved in user-interface design and hence the notations, tools and analysis techniques needed.

***Maker/hacker culture and mass customisation***—A new digital DIY-culture has emerged, made possible by accessible electronics such as Arduino and RaspberryPi and the availability of digital fabrication from fully equipped FabLabs to hobbyist-budget MakerBots. At an industrial scale, high-budget digital fabrication, such as metal printers, means that the complex and costly spare parts storage and distribution may soon be a thing of the past, replaced by just-in-time printing of everything from spare door handles to gear boxes. Between the two, there is the potential for a new niche of the digital artisan 'modding' consumer products, using open-source 3D-print files, or maybe iTunes-style commercial versions. At both industrial and artisan scale, there will certainly be greater scope for individual configuration and semi-bespoke design at a level beyond even today's ideas of mass customisation.

However, if everyone can make their own TV remote or washing machine facia panel, how do you ensure safety and usability of the resulting interfaces. Furthermore, if things do go wrong, who gets sued? For non-critical devices, it may be that

we see the HCI equivalent of house makeover television programmes and DIY-style how to books aimed at mass-market. However, for both legal and brand protection, products may need to limit acceptable adaptations.

*Formal challenges*: At first it seems that DIY-culture could not feel further from formal methods; but in fact the need for guaranteed properties on highly configurable interfaces is precisely the kind of problem that formal analysis could address. Specification of usability properties goes back to the earliest days of the FoMHCI community (Dix and Runciman 1985; Thimbleby and Harrison 1990; Dix 1991a) and issues of plasticity and model-based design have been studied for many years (e.g. Coutaz 2010) and are represented in this book (Chap. 18).

The level of configuration is different from those that are typically addressed (e.g. changes in physical form) and the design audience is somewhat different. It is likely that at least two kinds of designer-users need to be addressed: those in commercial enterprises or large open-hardware projects determining the properties required and the range of variations that may be possible; and end-users, or those near to the end use (e.g. the digital artisan), who are performing modifications with some sort of tool support. The level of formal expertise needed even to understand the outputs of current tool and reasoning support is still far too high, but both the domain-specific languages in Chap. 16 and the layered approach in Chap. 14 seem like potential approaches.

***Agile and test-driven development***—Although there are domains where monolithic development processes dominate, agile development methods are common in many areas, especially web-based systems. Rather like maker culture, at first glance the apparent 'try it and see' feel of agile systems seems at odds with formal development, but in fact agile methodologies are highly disciplined, typically combining a strong use-case orientation with test-driven development. Furthermore, for massive-scale systems user-interface development is supported by big data, notably A/B testing (Kohavi et al. 2009; Fisher et al. 2012).

*Formal challenges*: Although these are very different cultures, there are clear areas where formal methods could make a greater input into agile development. First is use cases, which may already use UML or similar formalisms, and could easily be integrated into an appropriate task modelling framework. This could help with a problem of agile development that it is often hard to keep track of the 'big picture' when constantly creating and deploying incremental change. On the testing side, while some test-based development includes user interfaces, for example by using headless web browser simulations, this is still a problematic area. Early work has shown that formal property specification could have a place to play, especially in looking at properties that span across individual units of delivery (Bowen and Reeves 2011). Finally, the volume of data available from large-scale traces of user behaviour is perfect input for performance models such as variants of MHP (Card et al. 1980, 1983), as well as other purposes described previously when looking at big data.

To achieve this would not be without theoretical and practical challenges including addressing presenting methods in ways that are accessible to the ordinary

developer, and creating specification methods that can more easily be addressed to facets of an evolving system.

## 3.3 Formalising Interaction: What and How

Having examined the driving forces from changes in HCI, we now turn to formal methods themselves. We look under four headings.

The first two concern the subject of formal methods in HCI, *what* they model. The first of these looks at the various *actors and entities* in the milieu of human interaction with computers, which are currently being modelled and which could be. The second is about different *levels of abstraction*, generalisation and granularity, the kinds of phenomena that we model and reason about.

The third and fourth headings are more about the process and nature of formal modelling. The first of these looks at the development process for interactive systems and asks when in the process our methods are valuable and who in this process is supported. Finally, we look at *how* our models work, the kinds of reasoning and modelling we are currently using and how this may need to change or be augmented, especially if we seek to support practical development.

### *3.3.1 What—Actors and Entities*

The majority of the earliest work on HCI was focused almost entirely on the direct interaction between a single user and desktop computer or other form of computing device (Fig. 3.1). There was always an interest in the varied stakeholders and other context that surrounded such interactions, but most research, and in particular detailed interaction design and engineering, concerned this dyad. Not surprisingly, this has also been an important theme for formal methods in HCI.

Of course this dyadic interaction is important, but not the whole story; even today, students need to be constantly reminded to consider the broader context. Figure 3.2 shows some of the agents and entities in this wider picture. Typically, users do not interact with a single device but several either at different times, or at the same time, for example the mobile phone as 'second screen' while watching

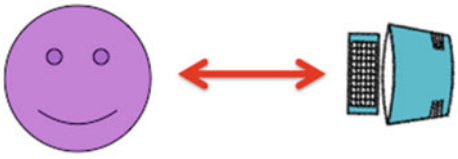

**Fig. 3.1** Human–computer interaction: early days—one person, one computer

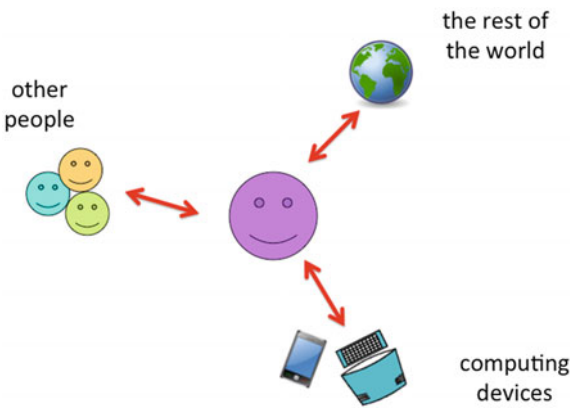


**Fig. 3.2** Human–computer interaction: agents and entities

television. Users also interact with people and with other aspects of the world: from cars and cats to central heating systems and chemical plants; these interactions may simply set the context for direct computer interaction (e.g. noise as in Chap. 12), but may also in various ways concern the computer, for example finding your way around a city using a smartphone map.

As well as direct connections, there are indirect interactions: including sensors and actuators on physical objects such as self-driving cars and pervasive computing; computer-mediated communication and social networks; virtual elements overlaid on the physical world in augmented reality such as Pokémon Go; computational interfaces embedded into physical appliances such as washing machines; and even computational devices embedded in the user or other people.

*Formal challenges*: While there is still a strong concentration on direct interactions, the notations and techniques in the chapters in this book do include different elements in this picture. Several include some element of user or task modelling (Chaps. 8, 11, 13), building on traditions such as MHP (Card et al. 1980, 1983), ICS (Barnard 1985) and CCT (Kieras and Polson 1985), and some include ways to represent multiple actors (Chaps. 13, 15). As noted under *Physicality and embodiment* above, there is some work in modelling direct and indirect interactions with the physical environment (Chaps. 7–9, 15), but these are still rudimentary.

Even the nature of direct interactions has been changing as noted in discussions of notification-based interaction, invisibility and autonomy. There has been a line of work in formal methods dealing with multi-modal interfaces, but now there are fresh challenges, for example dealing with artful interactions such as in crafts or music making and physical movement in the environment.

In general, while formal methods have stepped beyond the single user—single machine dyad, there seems to be substantial room for further work on the richer picture.

### 3.3.2 What—Levels of Abstraction

Computer scientists, and especially formalists, love abstractions, and the formalisms in this book and in the community at large vary over a number of dimensions of abstraction.

***Granularity***—We can view user interaction at a very high/coarse level in terms of task analysis or workflows; the basic unit at this level might be quite substantial, such as interacting with an app (Chap. 19). We can also look at a very low/fine level of basic motor actions and physical interactions, such as Fitts' Law or physigrams as in Chap. 9. Between these, there are many intermediate levels, including the architectural levels in the ARCH/Slinky model. The majority of formal modelling work seems to live in this area where the basic unit of interaction is an abstract button press, or similar action (e.g. Chap. 5).

*Formal challenges*: This suggests two opportunities for future work: first to expand the focus of methods to encompass more of the higher and lower levels; second to fit different levels of models together to enable reasoning across these. This may not be trivial as interactions regarded as atomic at one level have structure at lower levels.

***Continuity and time***—In some ways, the finest level of interaction involves continuous action in continuous time. Despite early work on status–event analysis (Dix 1991a, b; Dix and Abowd 1996) and the TACIT European project (TACIT 1998; Faconti and Massink 2002), this is still an understudied area.

*Formal challenges*: Again there are complex formal issues, but in the broader formal methods community hybrid systems have been studied for many years, so there is clear opportunity to improve coverage. However, as with granularity, the ability to link targeted modelling at different levels seems crucial.

***Level of generality***—In the earliest strands of formal methods in HCI, there was work on notations and methods for specifying specific systems, for example Reisner's (1981) use of BNF and GOMS (Card et al. 1980, 1983); but also work on very abstract models looking at generic properties of all systems, for example the PIE model (Dix and Runciman 1985); and work on properties of undo (Dix 1991a; Mancini 1997). In this book, the majority of work is towards the specification of specific systems, but there is also some work that bridges the extremes of generality including DSL in Chap. 16 and the high-level properties in Chap. 14, which both offer ways to create specifications or properties that apply to a sub-class of systems. While several chapters use variations of generic systems properties such as predictability of actions or visibility of results, there seems to be little current work directed at this generic level.

*Formal challenges*: The lack of work at a generic level may be because there are very limited things one can say at this level of generality and the early work saturated the area. However, the work in this volume that operates generically over a sub-class of systems suggests a potential path that allows formalists to work alongside domain experts to create intermediate notations, formal properties and tools that can then be used by other designers.

***Syntax versus semantics***—Some philosophers, such as Searle (1997), would argue that by definition anything done in a computer is merely syntax, symbol juggling, and never true semantics as meaning cannot be reduced to rules. However, it is clear that some aspects of computation are more 'semantic' than others. Formal methods often talks about the syntax–semantics distinction, but it is noteworthy that both the Seeheim model (Pfaff and Hagen 1985) and ARCH/Slinky model (UIMS 1992; Gram and Cockton 1996) stop at adaptors/wrappers for application semantics. As noted under granularity, the majority of formal work is focused on the dialogue level and, hence, largely about the syntax of interaction, the order of actions and observations, but not their 'meaning'. There are exceptions to this. At a generic level, early analysis of undo included use of category theory to model the meaning of undo parameterised over the state semantics of arbitrary systems. At a more specific level, task analysis, cognitive models and domain modelling capture elements of the social, individual and application meaning. The use of ontologies and OWL in Chap. 12 is particularly interesting as these are languages of 'semantics'.

*Formal challenges*: It is possible that rich models of semantics of systems and the environment could become intractable; indeed, model checking of user-interface specifications already requires some form of abstract interpretation of value domains and faces challenges of combinatorial explosion. However, if formal methods in HCI are to reason about the entire human–computer work system, then there need to be models of each of the elements, so that the physical properties of the world, the human cognition and action, application 'semantics' and dialogue syntax can be verified to all work together to achieve specified goals. Of course, complete models of all of these would be intractable; the greatest challenge will be in determining appropriate levels of detail and abstraction to enable useful results.

### *3.3.3 Who and When (and Why?)*

Just as we looked at the actors and flows involved in interaction itself, we can consider the actors, activities and products involved in the design process for interactive systems (see Fig. 3.3). As noted previously, moves towards agile development methods mean these stages are likely to be highly iterative including deployment itself, where some web-based systems may have many hundreds of releases per week.

Very early, indeed before the start of the design process proper, are the formal experts involved in the formulation of notations, properties and tool creation. As noted, this may also include high-level domain experts helping to create more domain-specific variants.

During the early stages of the design process for a specific system, there are many actors including interaction designers, product designers, user experience specialists; engineers, customers, management, domain experts; and hopefully

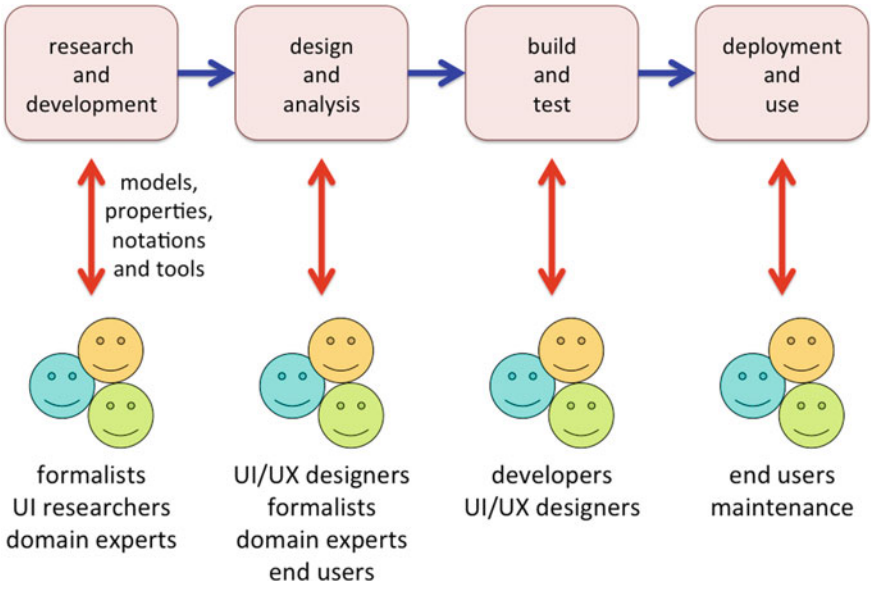


**Fig. 3.3** Design process: processes and people

some users! In theory, appropriate notations and tools should help to clarify the design and communicate between stakeholders. The formal outcomes of this stage are detailed specifications, which potentially may be checked using model checkers or other tools.

As the design is turned into code, the formal specification may be translated into executable code; embedded into in-code verification; or used to create test suites.

Finally during use, executable specifications may actually be running in the code, or the user may use manuals that have been in part created, or verified using formal methods (as in Chap. 11). Traces of use may be collected and used as input for analysis tools for future systems. During this, some systems may be open to end-user modifications or appropriation.

*Formal challenges*: Although there is potential for formal methods to be useful at many stages in this process, in practice there is little real use beyond limited research case studies and some safety critical systems. Indeed, in a 2009 ACM Computer Surveys study of industrial use of formal methods all of the systems surveyed had a critical element and only 10% involved human–computer interaction (Woodcock et al. 2009). The two main barriers are the expertise needed to understand the methods and the time costs of the detailed specification analysis needed even when one has the requisite expertise. Safety critical systems are one of the few areas where the benefits can justify the extreme costs. This leads to three clear challenges: reducing formal expertise, reducing effort and increasing benefit. These are not new, 20 years ago Clarke and Wing (1996) concluded:

> Success in formal specification can be attributed to notations that are accessible to system designers and to new methodologies for applying these notations effectively (Clarke and Wing 1996)

The focus on tool support in several chapters is encouraging as is the use of domain-specific properties and languages (Chaps. 14, 16) as this can help to make the methods more accessible to non-formalists (*reducing formal expertise*). Ensuring any such tools are usable is also critical, as Chap. 17 emphasises (*reducing effort*). Indeed, it is interesting to see work where the design process is the *subject* of formal or structured analysis or modelling, including Chap. 17's use of Norman's model of action to analyse toolset usability and Bowen and Dittmar (2016) semi-formal framework for design spaces.

Using toolsets also means that there is greater opportunity to take the same designer input (in the form of task descriptions, interface specifications, etc.) and use it for different purposes, thus *increasing the benefit*. As several chapters were using ontologies, RDF and OWL, this suggests the potential for offering some form of expert-system-driven guidance (*reducing formal expertise*), perhaps the outcomes of formal analysis could be used to drive more knowledge-rich explanation systems.

As Chap. 6 points out, designers already create many artefacts such as sketches and low-fidelity prototypes; these could be used more effectively in formal methods in the way Denim (Lin et al. 2000) and subsequent systems did for executable prototypes, that is using computational effort to understand the designer's language. These all become even more important when we consider end-user development, although the use of graph transformations for reconfigurable UIs (Chap. 10) may offer one approach to this.

### 3.3.4 How

The above take us back to the kinds of formalisms we apply and the ways these might already be changing, or perhaps should change in the future based on some of the challenges we have seen.

***Types of reasoning***—Many of the chapters in this book use notations and methods that are similar in kind to those found in earlier collections (Thimbleby and Harrison 1990; Palanque and Paterno 1997), albeit used in different ways. There is substantial use of textual notations based on sets, functions, predicates and logics, and also more graphical notations including variants of Petri nets, statecharts and hierarchies for task analysis. A major difference from the early years is that many are subject to some form of automated checking or analysis as well as analysis by hand. Another new development is the use of OWL and similar notations and tools. Unfortunately, as already noted there is still very little use of the mathematics of continuous values or time.

***Knowledge-rich reasoning***—The use of OWL suggests the potential for more knowledge-rich reasoning. Traditional mathematics tends to be based on relatively few rules with relatively complex reasoning, in contrast to expert systems in AI with large rule sets (or knowledge bases) and relatively shallow reasoning. However, user-interface specification, and indeed formal specification in general, tends to already have relatively large specification with relatively simple (possibly automated) analysis. The whole of number theory can be (largely) built from nine Peano axioms, plus a little set theory, even 'toy' interface specifications have more! Furthermore, big data has proved 'unreasonably' effective in using statistical and shallow machine-learning techniques to address problems, such as natural language processing, that had formerly been seen as requiring symbolic artificial intelligence (Halevy et al. 2009). This suggests the potential for using techniques that bring together large data volume knowledge with specifications to address issues such as the inclusion of semantics as well as syntax, and the generation of automated design guidance.

***Flexible levels of detail***—We have seen how different formal notations and techniques operate at different levels of granularity, from workflows on apps to human–motor system analysis. In professional use, different levels will be appropriate for different aspects of a system; indeed, among the conclusions of an analysis of an early successful formal user-interface specification case study (Dix 2002a, b) was the need to be *useful* (address a real problem) and *appropriate* (no more detailed than needed), both of which may vary depending on which aspect of the system is under consideration.

Imagine a simple map-based system that includes buttons to select options, such as satellite imagery versus schematic, but then uses mouse movement and scroll wheel to drag and zoom the map: when the map is clicked a third-party gazetteer application opens in a pop-up showing additional information about the location. Unless the options buttons are particularly unusual, it will be appropriate to deal with them using a dialogue-level notation such as labelled state transitions (Chaps. 11, 15) or ICO (Chap. 17; Palanque 1992). At this level of specification, the map interactions would be abstracted, possibly to separate zoom and scroll functions, or maybe even to a single 'select location'. Similarly, as the gazetteer is pre-existing, we might abstract it to a single 'view gazetteer' operation and not attempt to model its interface even at the level of buttons and dialogue. However, we may also want to use a more detailed analysis of the map interactions themselves, perhaps making use of multi-scale Fitts' Law (Guiard et al. 2001).

There are two separate challenges here. First is dealing with systems at multiple levels of detail. This is already studied, for example, Chap. 6 on combining models and Chaps. 17 and 18, which have toolsets including notations at different levels; however, this may be more challenging when the notations used involve different paradigms (see below). The second, and perhaps more complex, is when different facets of the system are analysed at different levels so that the specification at certain levels of detail is not 'complete'.

***Multiple notations***—Many of the chapters in this book use multiple notations. Sometimes this is to deal with different levels as described above and sometimes because different aspects of the system use different notations, for example user and system models (Chaps. 8, 11) or physical and digital models (Chap. 9). Even when considering the same aspect at the same level of detail, different notations are required for different analysis techniques or tools, a translation process which is usually automated; for example, in Chap. 5 a user-interface model is translated into a form of Petri net for execution and in Chap. 16 domain-specific languages are translated into a form of UML and into linear temporal logic.

If the underlying paradigms are very close, then this may be relatively unproblematic; however, typically languages have different strengths, and information is lost in transforming from one model to another, for example, a single entity in one might correspond to several entities in another, or have no correspondence at all. If one notation is semantically richer than the other, then it may be possible to reason across the two by translation, but even then it can be problematic to translate the outputs of analysis back. A trivial example of this was the early days of Java Server Pages (JSP) before the development of 'Source Map' files (Oracle 2003): compiler messages referred to lines in often unintelligible generated Java code. Verifying the connection between notations also requires some sort of explicit or, more often, implicit shared semantics.

***Generic descriptions and standards***—Whether we are dealing with multiple notations within a single toolset or project, or trying to share artefacts (e.g. UI specifications, traces of user behaviour) between projects, there seems to be a need for different methods, notations and tools to be able to talk to one another. In this book, the nuclear power station case study is specified in some detail in Chap. 4, but how do we know that the formulation of this in other chapters refers to the 'same' thing? In general, as a community do we need some form of standardised means to share?

At a concrete level, this could be ways to specify elements of interaction such as layouts, behaviours or states, so that it is possible to use these to validate whether two specifications really are talking about the 'same' system. Given the differences between notations and models, the likelihood is that this may need to be partial, perhaps instance based: this particular state has this particular visual appearance and after a specific action modifies to a specified new state. Alternatively it may be possible to have a number of agreed ways of sharing more complex behaviours but limited to particular styles of specification.

At a more abstract level, we could seek semantic models that are not necessarily useful for actual specification (e.g. too verbose or complex), but can be used as a basis for giving semantics for other notations, rather like denotational semantics does for programing languages (Stoy 1977). If two notations are given semantics in such a shared semantic model, then it would become possible to assert reliably whether a specification in one is equivalent to one in another, or whether a translation algorithm between the two notations is valid.

One suggestion (Dix 2002a) has been that traces could act as a form of universal semantics, both at a concrete level and as a semantic model, as is used by process

algebras. The advantage of this is that while different notations vary in how they abstract and parameterise behaviour, the actual realised behaviours are observable and shared. Traces are not without issues, notably continuous versus discrete time, and even discrete time at different granularities. A similar argument could be made for visual appearance at least for standard control panels with buttons, drop-downs, etc. Richer system or interface behaviour is more difficult, although a number of notations are enhanced versions of simpler ones such as labelled transition systems, or Petri nets; even if it is not possible to have a single interchange model, it may be possible to have a small selection.

## 3.4 Summary

Tables 3.1 and 3.2 summarise the principal topics and challenges that have emerged in this chapter; they are quite diverse and offer many opportunities for fruitful future research and real societal and economic impact. From these lists, we can bring out a few broader issues.

**Table 3.1** HCI trends: summary and formal challenges

| Trend | Formal challenge |
|---|---|
| *Changing user interaction* | |
| Choice and ubiquity | Importance of diversity (below) |
| Diverse people | Model-based interfaces to take into account varying abilities |
| Diverse devices and contexts | Plastic interfaces beyond screen size |
| Physicality and embodiment | Modelling beyond syntax layer |
| Really invisible | Radically new models of interaction |
| Experience and values | Linking argumentation to formal modelling |
| Social and personal use | Modelling multiple actors, privacy and provenance |
| Notification-based interaction | Modelling when uses not in control—matching pace and timing of notifications to user tasks |
| Basic HCI | Better tool support, especially for non-experts |
| *Changing technology* | |
| Vast assemblies | Application workflows and configuration; feature interactions |
| Big data | Use in formal development, e.g. mining trace data; knowledge-rich methods (below) |
| Autonomy and complexity | Human-like computing; dynamic function allocation; autonomous action beyond internal transitions |
| *Changing design and development* | |
| Maker/hacker culture and mass customisation | Guaranteed properties for configurable systems; tools for near end-users |
| Agile and test-driven development | Formalising use cases; generating tests; trace data |

**Table 3.2** Formalising interaction: summary and formal challenges

| Topic | Formal challenge/issues |
|---|---|
| *What—actors, entities* | |
| Actors and entities | Broadening scope of FoMHCI; modelling users and tasks; physical aspects; artful interactions |
| *What—levels of abstraction* | |
| Granularity | Modelling beyond syntax layer; connecting models at different levels of abstraction |
| Continuity and time | Moving beyond discrete-event dialogue; hybrid systems |
| Level of generality | Domain-specific generic models |
| Syntax versus semantics | Modelling domain semantics |
| *Who and when (and why?)* | |
| Reducing formal expertise | e.g. domain-specific notations, expert-system guidance |
| Reducing effort | e.g. toolset development |
| Increasing benefit | e.g. using formal models for multiple purposes |
| *How* | |
| Types of reasoning | Broad range of notations used; increasing use of automatic analysis; limited knowledge-rich methods; continuity and time still poor |
| Knowledge-rich reasoning | Linking formal specifications and large knowledge bases; application semantics; automated design advice |
| Flexible levels of detail | Dealing with multiple levels of detail; working with incomplete specifications |
| Multiple notations | Translating between notations; verifying connections (shared semantics) |
| Generic descriptions and standards | Interchange models for specifications, physical layout, case studies, etc.; shared semantics (e.g. traces) |

We need to extend the kinds of issues we approach, including: users of different abilities; varying devices and infrastructure; physical and semantic interface issues; and privacy. This may include ways to at least partially connect with hard to formalise areas including cognition, emotion and human values.

We need to be able to deal with systems where the computational part is more intelligent, autonomous and proactive; this includes issues such as notification-based systems, Internet of Things and robotics.

Some of these new or understudied uses may offer ‘easy wins’ for FoMHCI, for example in dealing with the complexities of IoT interactions that are hard for human assessment, or applications to agile methodologies.

We need a range of different levels and kinds of model and the ability to connect these: this includes links between existing modelling approaches in FoMHCI, and other approaches such as formal argumentation or large knowledge bases.

Applications in safety critical domains are likely to remain a core area of study for FoMHCI as these justify societally and economically the costs of extensive analysis. However, there are opportunities for FoMHCI to become more

mainstream both by tackling the ‘easy win’ areas and by seeking ways for formal analysis to become more cost-effective and more accessible to domain experts, developers and end-users.

## References

Adamczyk P, Bailey B (2004) If not now, when? The effects of interruption at different moments within task execution. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI ’04). ACM, New York, pp 271–278. doi:10.1145/985692.985727

Bailey N, Konstan J (2006) On the need for attention-aware systems: measuring effects of interruption on task performance, error rate, and affective state. Comput Hum Behav 22 (4):685–708. doi:10.1016/j.chb.2005.12.009

Barnard P (1985) Interacting cognitive subsystems: a psycholinguistic approach to short-term memory. In: Ellis A (ed) Progress in the psychology of language, volume 2, chapter 6. Lawrence Erlbaum Associates, Hove

Bowen J. Reeves S (2011) UI-driven test-first development of interactive systems. In: Proceedings of the 3rd ACM SIGCHI symposium on engineering interactive computing systems (EICS ’11). ACM, New York, pp 165–174. doi:10.1145/1996461.1996515

Bowen J, Dittmar A (2016) A semi-formal framework for describing interaction design spaces. In: Proceedings of the 8th ACM SIGCHI symposium on engineering interactive computing systems (EICS ’16). ACM, New York, pp 229–238. doi:10.1145/2933242.2933247

Bruegger P (2011). uMove: a wholistic framework to design and implement ubiquitous computing systems supporting user’s activity and situation. PhD Thesis, University of Fribourg, Switzerland. http://doc.rero.ch/record/24442

Card S, Moran T, Newell A (1980) The keystroke-level model for user performance with interactive systems. Commun ACM 23:396–410

Card S, Moran T, Newell A (1983) The psychology of human computer interaction. Lawrence Erlbaum Associates, Hillsdale, New Jersey

Carter T, Seah S, Long B. Drinkwater B, Subramanian S (2013) UltraHaptics: multi-point mid-air haptic feedback for touch surfaces. In: Proceedings of the 26th annual ACM symposium on user interface software and technology (UIST ’13). ACM, New York, pp 505–514. doi:10.1145/2501988.2502018

Citizens Advice Bureau (2013) 22% don’t have basic banking services needed to deal with Universal Credit. http://www.citizensadvice.org.uk/index/pressoffice/press_index/press_office20131105.htm. Accessed 29 Jan 2014

Clarke E, Wing J (1996) Formal methods: state of the art and future directions. ACM Comput Surv 28(4):626–643 (1996). doi:10.1145/242223.242257

Cockton G (2004) Value-centred HCI. In: Proceedings of the third Nordic conference on human-computer interaction (NordiCHI ’04). ACM, New York, pp 149–160. doi:10.1145/1028014.1028038

Council of the European Union (2016) Position of the council on general data protection regulation. http://www.europarl.europa.eu/sed/doc/news/document/CONS_CONS(2016)05418(REV1)_EN.docx. Accessed 8 April 2016

Coutaz J (2010) User interface plasticity: model driven engineering to the limit! In: Engineering interactive computing systems (EICS 2010). ACM, pp 1–8

Czerwinski M, Horvitz E, Wilhite S (2004) A diary study of task switching and interruptions. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI ’04). ACM, New York, pp 175–182. doi:10.1145/985692.985715

Dewsbury G, Ballard D (2014) DTA: the dependability telecare assessment tool—the person-centred telecare assessment. Dewsbury, Bourne, UK. http://www.gdewsbury.com/dta/

Dix A, Runciman C (1985) Abstract models of interactive systems. In: Johnson P, Cook S (eds) People and computers: designing the interface. Cambridge University Press, pp 13–22. http://alandix.com/academic/papers/hci88/

Dix A (1990) Information processing, context and privacy. In: Diaper DGD, Cockton G, Shakel B (eds) Human-computer interaction—INTERACT '90, North-Holland, pp 15–20

Dix A (1991a) Formal methods for interactive systems. Academic Press, London. http://www.hiraeth.com/books/formal/

Dix A (1991b) Status and events: static and dynamic properties of interactive systems. In: Duce DA (ed) Proceedings of the eurographics seminar: formal methods in computer graphics, Marina di Carrara, Italy. http://www.comp.lancs.ac.uk/computing/users/dixa/papers/euro91/euro91.html

Dix A (1992a) Human issues in the use of pattern recognition techniques. In: Beale R, Finlay J (eds) Neural networks and pattern recognition in human computer interaction. Ellis Horwood, pp 429–451. http://alandix.com/academic/papers/neuro92/neuro92.html

Dix A (1992b) Pace and interaction. In: Monk A, Diaper D, Harrison M (eds) Proceedings of HCI '92: people and computers VII. Cambridge University Press, pp 193–207. http://alandix.com/academic/papers/pace/

Dix A, Abowd G (1996) Modelling status and event behaviour of interactive systems. Softw Eng J 11(6):334–346. http://alandix.com/academic/papers/euro91/

Dix A (2002a) Towards a ubiquitous semantics of interaction: phenomenology, scenarios and traces. In: Forbrig P, Limbourg Q, Urban B, Vanderdonckt J (eds) Interactive systems. Design, specification, and verification 9th international workshop, DSV-IS 2002, Rostock, Germany, June 2002. Springer, LNCS 2545, pp 238–252. http://alandix.com/academic/papers/dsvis2002/

Dix A (2002b) Formal methods in HCI: a success story—why it works and how to reproduce it. Lancaster University, UK. http://alandix.com/academic/papers/formal-2002/

Dix A, Ramduny-Ellis D, Wilkinson J (2004) Trigger analysis—understanding broken tasks. In: Diaper D, Stanton N (eds) Chapter 19 in The handbook of task analysis for human-computer interaction. Lawrence Erlbaum Associates, pp 381–400. http://www.hcibook.com/alan/papers/triggers2002

Dix A, Sheridan J, Reeves S, Benford S, O'Malley C (2005) Formalising performative interaction. In: Proceedings of DSVIS '2005 (Newcastle, UK, 13–15 July 2005). Springer, LNCS 3941, pp 15–25. http://www.hcibook.com/alan/papers/DSVIS2005-performance/

Dix A, Leavesley J (2015) Learning analytics for the academic: an action perspective. J Univ Comput Sci (JUCS) 21(1):48–65. http://www.hcibook.com/alan/papers/JUCS-action-analytics-2015/

Dix A (2016) Human computer interaction, foundations and new paradigms. J Vis Lang Comput (in press). doi:10.1016/j.jvlc.2016.04.001

Duncan D, Hoekstra A, Wilcox B (2012) Digital devices, distraction, and student performance: does in-class cell phone use reduce learning? Astron Educ Rev 11(1). doi:10.3847/AER2012011

Ellis C (1994) Goal based workflow systems. Int J Collab Comput 1(1):61–86

EPSRC (2016) Human-like computing report of a workshop held on 17 & 18 February 2016, Bristol, UK. https://www.epsrc.ac.uk/newsevents/pubs/humanlikecomputing/

Eslambolchilar P (2006) Making sense of interaction using a model-based approach. PhD thesis, Hamilton Institute, National University of Ireland, NUIM, Ireland

Faconti G, Massink M (2002) A reference framework for continuous interaction. Int J Univ Access Inf Soc 1(4):237–251. Springer

Fatah gen Schieck A, Kostakos V, Penn A, O'Neill E, Kindberg T, Stanton Fraser D, Jones T (2006) Design tools for pervasive computing in urban environments. In: van Leeuwen JPT, Timmermans HJP (eds) Innovations in design and decision support systems in architecture and urban planning. Springer, Dordrecht, Netherlands, pp 467–486

Fisher D, DeLine R, Czerwinski M, Drucker S (2012) Interactions with big data analytics. Interactions 19(3):50–59. doi:10.1145/2168931.2168943

Goodman B, Flaxman S (2016) EU regulations on algorithmic decision-making and a "right to explanation". In: Presented at 2016 ICML workshop on human interpretability in machine learning (WHI 2016), New York, NY. http://arxiv.org/abs/1606.08813v1

Gram C, Cockton G (eds) (1996) Design principles for interactive software. Chapman & Hall, London

Guiard Y, Bourgeois F, Mottet D, Beaudouin-Lafon M (2001) Beyond the 10-bit barrier: Fitts' law in multi-scale electronic worlds. In: People and computers XV—interaction without frontiers: joint proceedings of HCI 2001 and IHM 2001. Springer, London, pp 573–587. doi:10.1007/978-1-4471-0353-0_36

Halevy A, Norvig P, Pereira F (2009) The unreasonable effectiveness of data. IEEE Intell Syst 24 (2):8–12. doi:10.1109/MIS.2009.36

Harper R, Rodden T, Rogers Y, Sellen A (eds) (2008) Being human: human-computer interaction in the year 2020. Microsoft Res. http://research.microsoft.com/en-us/um/cambridge/projects/hci2020/

Hawking S, Musk E, Wozniak S et al (2015) Autonomous weapons: an open letter from AI & robotics researchers. Future of Life Institute. http://futureoflife.org/AI/open_letter_autonomous_weapons

Henrich J, Heine S, Norenzayan A (2010) The weirdest people in the world? Behav Brain Sci 33 (2–3):61–83. doi:10.1017/S0140525X0999152X

Hildebrandt M, Harrison M (2003) Putting time (back) into dynamic function allocation. In: Proceedings of the 47th annual meeting of the human factors and ergonomics society

Ishii H (2003) Tangible bits: designing the seamless interface between people, bits, and atoms. In: Proceedings of the 8th international conference on intelligent user interfaces (IUI '03). ACM, New York, NY, USA, pp 3–3. doi:10.1145/604045.604048

Hong J, Landay J (2004) An architecture for privacy-sensitive ubiquitous computing. In: Proceedings of the 2nd international conference on mobile systems, applications, and services (MobiSys '04). ACM, New York, NY, USA, pp 177–189. doi:10.1145/990064.990087

Keim D, Kohlhammer J, Ellis G, Mansmann F (2010) Mastering the information age-solving problems with visual analytics. Eurographics Association, Goslar, Germany. http://www.vismaster.eu/wp-content/uploads/2010/11/VisMaster-book-lowres.pdf

Kieras D, Polson P (1985) An approach to the formal analysis of user complexity. Int J Man Mach Stud 22:365–394

Kohavi R, Longbotham R, Sommerfield D, Henne R (2009) Controlled experiments on the web: survey and practical guide. Data Min Knowl Discov 18(1):140–181. doi:10.1007/s10618-008-0114-1

Langheinrich M (2002) A privacy awareness system for ubiquitous computing environments. In: Borriello G, Holmquist LE (eds) Proceedings of the 4th international conference on ubiquitous computing (UbiComp '02). Springer, London, UK, pp 237–245

Lin J, Newman M, Hong J, Landay J (2000) DENIM: finding a tighter fit between tools and practice for web site design. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI '00). ACM, New York, USA, pp 510–517. doi:10.1145/332040.332486

Mancini R (1997) Modelling interactive computing by exploiting the undo. Dottorato di Ricerca in Informatica, IX-97-5, Università degli Studi di Roma "La Sapienza". http://www.hcibook.net/people/Roberta/

McCoy B (2016) Digital distractions in the classroom phase II: student classroom use of digital devices for non-class related purposes. J Media Educ 7(1):5–32. http://en.calameo.com/read/000091789l5b8f5b352ba

Meixner G, Paternó F, Vanderdonckt J (2011) Past, present, and future of model-based user interface development. i-com 10(3):2–11

Norman D (2005) Emotional design. Basic Books

Oracle (2003) JSR-045: debugging support for other languages. JSRs: Java Specification Requests. https://jcp.org/en/jsr/detail?id=45

Palanque P (1992) Modélisation par Objets Coopératifs Interactifs d'interfaces homme-machine dirigées par l'utilisateur. PhD, Toulouse I

Palanque P, Paterno F (eds) (1997) Formal methods in human-computer interaction. Springer, London

Pallotta V, Bruegger P, Hirsbrunner B (2008) Kinetic user interfaces: physical embodied interaction with mobile ubiquitous computing systems. In: Kouadri-Mostéfaoui S, Maamar M, Giaglis P (eds) Advances in ubiquitous computing: future paradigms and directions. IGI Global Publishing, pp 201–228. ISBN: 978-1-599-04840-6

Pfaff G, Hagen P (eds) (1985) Seeheim workshop on user interface management systems. Springer, Berlin

Reisner P (1981) Formal grammar and human factors design of an interactive graphics system. IEEE Trans Softw Eng 7(2):229–240. doi:10.1109/TSE.1981.234520

Searle J (1997) The mystery of consciousness. Granta Books, London

Shackel B (1959) Ergonomics for a computer design 120:36–39. http://www.idemployee.id.tue.nl/g.w.m.rauterberg/presentations/shackel-1959.PDF

Sherman J (2013) Half of benefit claimants lack skill to complete online forms. The Times, London. http://www.thetimes.co.uk/tto/news/uk/article3914355.ece

Silver D et al (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529:484–489. doi:10.1038/nature16961

Stoy J (1977) Denotational semantics: the Scott-Strachey approach to programming language semantics. MIT Press, Cambridge, Massachusetts

Sufrin B (1982) Formal specification of a display-oriented text editor. Sci Comput Program 1:157–202. North Holland Publishing Co.

TACIT (1998–2002) Theory and applications of continuous interaction techniques. EU TMR Network Contract: ERB FMRX CT97 0133. http://www.webalice.it/giorgio.faconti/TACIT/TACITweb/doclist.html

Thimbleby H, Harrison M (eds) (1990) Formal methods in human-computer interaction. Cambridge University Press

Thimbleby H (2007) Using the Fitts law with state transition systems to find optimal task timings. In: Proceedings of second international workshop on formal methods for interactive systems, FMIS2007. http://www.dcs.qmul.ac.uk/research/imc/hum/fmis2007/preproceedings/FMIS2007preproceedings.pdf

Thomas J, Cook K (2005) Illuminating the path: research and development agenda for visual analytics. IEEE Press

Turchi T, Malizia A (2016) A human-centred tangible approach to learning computational thinking. EAI Endorsed Trans Ambient Syst 16(9):e6. doi:10.4108/eai.23-8-2016.151641

UIMS (1992) A metamodel for the runtime architecture of an interactive system. The UIMS developers workshop. SIGCHI Bull 24(1):32–37. ACM. doi:10.1145/142394.142401

Weiser M (1991) The computer of the 21st century. Sci Am 265(3):66–75

Whittington P, Dogan H (2016) A smart disability framework: enhancing user interaction. In: Proceedings of the 2016 British HCI conference

Woodcock J, Larsen P, Bicarregui J, Fitzgerald J (2009) Formal methods: practice and experience. ACM Comput Surv 41(4):19. doi:10.1145/1592434.1592436

Wurdel M (2011) An integrated formal task specification method for smart environments. University of Rostock. ISBN: 978-3-8325-2948-2

Yadron D, Tynan D (2016) Tesla driver dies in first fatal crash while using autopilot mode. Guardian