

This item was submitted to [Loughborough's Research Repository](#) by the author.
Items in Figshare are protected by copyright, with all rights reserved, unless otherwise indicated.

Making accessibility accessible

PLEASE CITE THE PUBLISHED VERSION

PUBLISHER

© Matthew Tylee Atkinson, Colin H.C. Machin, David Sloan

LICENCE

CC BY-NC-ND 4.0

REPOSITORY RECORD

Atkinson, Matthew T., Colin H.C. Machin, and David Sloan. 2019. "Making Accessibility Accessible". figshare.
<https://hdl.handle.net/2134/4463>.

This item was submitted to Loughborough's Institutional Repository (<https://dspace.lboro.ac.uk/>) by the author and is made available under the following Creative Commons Licence conditions.



For the full text of this licence, please go to:
<http://creativecommons.org/licenses/by-nc-nd/2.5/>

Making Accessibility Accessible*

Matthew Tylee Atkinson

Colin H. C. Machin

David Sloan

Presented at Accessible Design in the Digital World (ADDW), 24th September 2008

Abstract

In recent years, the need for adaptive design in information systems has come to the fore. Both industry and academia have begun to respond to these problems. There now exist recognised baseline standards for content accessibility and assistive technologies (ATs) are available on many platforms. Many exemplary research projects have found powerful and sometimes highly adaptive solutions to a range of accessibility problems.

However, there are significant difficulties with the current state of accessibility as a whole. For current ATs to be of most use, there is a responsibility on the user to be aware of their access limitations and implement the most appropriate accessibility solution for these needs—indeed, many accessible design guidelines for ICT developers are predicated on the user having the most appropriate access solution for their needs. However, it cannot be assumed that a user will have the most appropriate access technology, or even be aware that they need one, given the gradual rate of acquisition of an impairment, the dynamic nature of the impact of the impairment, or—more likely—impairments, allied with a lack of awareness of available technical solutions.

Other issues include: content and software developers seeing accessibility as a niche and, therefore, prohibitively expensive to implement given the expected gain in market share. Also, most research projects—though providing technically adept solutions to these problems—may not be possible for developers to use due to the wide variety of technical requirements of these disparate solutions. In this paper, we discuss these problems in the context of current literature and make high-level proposals as to how these problems may be addressed.

1 Introduction

Are computer-based assistive technologies¹ accessible? Who should they be accessible to—groups of people who are considered disabled; those in ex-

treme environments; “normal” computer users; system developers?

Should the term “accessibility” even be used, as it seems to be ignored by the general public and misunderstood by developers, who believe they must do extra work for those with “special needs” when, in reality, we *all* have special needs.

Invariably the steps needed to achieve the above goals are both social and technical in nature, with the social needs driving appropriate technical innovation². There is currently a great amount of research interest in accessibility—both as new technologies are created with inherent barriers to users and we realise that technologies originally developed to help “those with disabilities” could be of use to many people. The evolution of technology has generally lead to enhanced multimedia capability, combined modalities of information provision and more dynamicity of web content, which creates new accessibility challenges not present when we dealt only with static text documents and simple forms. Assistive technology that benefits others—this includes predictive text for mobile telephones, cassette tape recorders and captions for television in public spaces [5, pp.11–15].

This paper discusses some of the positive achievements in the field and what the authors perceive as some key challenges—which are addressed by a series of proposals for ongoing and future work.

2 What Has Gone Well?

“Accessibility” is now a well-established topic in both academia and industry and has interests ranging from the web content adaptation and browsers, through business applications to computer games and mobile computing. This section discusses the positive aspects of what has been achieved and is currently visible to users (contemporary research projects are discussed in the context of the problems they aim to solve later).

*Copyright 2008 Matthew Tylee Atkinson, Colin H. C. Machin, David Sloan

¹Though there are assistive technologies in many other areas, such as captions and audio description for television, we only discuss computer-related AT here.

²One of the problems, discussed in section 3.4, is that we are still unaware of the social needs to a high enough resolution.

2.1 Assistive Technology is Available—and Sometimes Integrated

On the desktop, most major platforms³ now provide at least one GUI framework for developing applications that automatically exposes information required by assistive technologies (ATs). The standards employed are different for each GUI toolkit, though there have been some efforts to bridge these gaps⁴. Table 1 shows a brief overview of adaptations supported out-of-the-box and available as third-party add-ons in common OSes.

Similar standards exist—and are used—for allowing ATs to access the document object model (DOM) of web content in three popular browsers: Firefox (Mozilla), Safari (Apple) and Internet Explorer (Microsoft). This is important because it allows disabled people—with the appropriate ATs—to use the latest browsers, including support for relatively new web standards aimed at making dynamic web applications⁵ accessible [26].

In the smartphone and PDA arena, both screen readers and magnifiers exist for devices running Symbian 60 and Microsoft Windows Mobile (at extra cost).

2.2 Standards are Available and (Increasingly) Used

A number of standards have been developed to enable the development of ATs.

Most popular GUI toolkits expose, as discussed, accessibility information to ATs. There are many standards for this; some open and some proprietary. Recently Microsoft has proposed a new standard, “UI Automation” to replace the previous “Microsoft Active Accessibility (MSAA)”, though currently no commercial AT for Windows supports it.

Direct access to web content for ATs is provided in most popular browsers by bridges to (standard protocols for ATs to interrogate) the DOM.

Web Content itself may be created in line with recommendations issued by the W3C [27] which have helped to raise awareness of accessibility issues [6]. Particularly at a grass-roots level, accessibility has become embraced as part of a wider philosophy of standards

based design, where the goal is to make digital content available and usable on the widest possible range of platforms and devices, including assistive technologies—see for example the Web Standards Project⁶.

Game accessibility guidelines have been created by groups such as IGDA and DIGRA and independent researchers as part of an awareness-raising effort in the computer games industry—sometimes presented in engaging ways to developers [14].

Laws have been introduced. In some countries, legislation promoting the rights of disabled people has been introduced that either directly or indirectly references technical requirements that define lawfully accessible technology, a prime example being the Section 508 standard in the US, referred to in Section 508 of the Rehabilitation Act Section 508 Standards⁷.

3 What is Still Not Right?

In comparison to today’s world of complex graphical user interfaces, multimedia and multimodal interaction, and dynamic web applications, accessibility challenges to using computers have been less significant, particularly for blind and visually impaired people [7].

3.1 Assistive Technology is a Second-Class Citizen

The concept of inclusive design, variously known as universal design or design for all [5] promotes the development of technology and technological resources that can be used without significant extra effort by people with sensory, motor and cognitive impairments. A significant source of confusion which can inhibit the route towards optimally inclusive technology lies in the complex relationship of objects involved in a typical technological interaction, and the resultant definition of roles and responsibilities towards accessibility amongst each object. The key components are as follows.

The information a user wishes to access and interact with (a web page, or word processor document, or spreadsheet, for example). The user has a large degree of control over the information they wish to access and use.

The software being used to render and interact with the information. The user may or may not have a choice over which software they

³The “GNOME” GUI environment on UNIX-like OSes; Mac OS X and Windows

⁴e.g. the effort to bridge Microsoft’s new “UI Automation” standard to that used by GNOME—http://www.mono-project.com/Accessibility:_Architecture (all links accessed on 29/08/2008).

⁵i.e. AJAX, or “Web 2.0” applications

⁶<http://www.webstandards.org>

⁷<http://www.section508.gov/index.cfm?FuseAction=content&ID=12>

Table 1: Comparison of accessibility features in common OSes (latest versions; including bundled applications)

Feature	Mac OS X	GNOME	Windows
Full-screen magnification	Integrated	Integrated	Commercial add-on
Colour deficit support	Greyscale	High-contrast themes	Commercial add-on
Resolution and text size	Integrated	Integrated	Integrated
Screen reader	Integrated	Integrated	Free/commercial add-on
Read specific text	Integrated	Free add-on	Commercial add-on
On-screen keyboard	Commercial	Integrated	Commercial add-on

use. The software may include some options to enhance accessibility, for example configuring display or behaviour characteristics.

The assistive technology is additional software or hardware that works with the software to improve the rendering of and interaction with the information. Ideally, the user will have the most appropriate assistive technology for their access requirements, but in practise this cannot be assumed.

The operating system is the technical framework on which the software and assistive technology resides. The operating system should also include a range of accessibility options. The user in theory has a choice of operating system, but in practise most users will not consider changing operating system.

For many people with access requirements, a separate assistive technology is not required. Instead, it is sufficient for adjustments to be made at operating system or software level to, for example, input device behaviour or display. For others, a separate assistive technology is necessary.

For the information provider, the key is to develop information in a way that allows assistive technologies and accessibility options provided by the software or operating system to work as expected. This is the objective of guidelines for content accessibility as discussed in section 2.2.

However, particularly on the Web, a number of factors have combined to blur these responsibilities, with a trend towards information authors to go beyond accessible design guidelines and take on responsibility for providing accessibility features that should be the responsibility of the software or the operating system. Thus, on the Web we see for example text-to-speech solutions⁸, text resizing widgets, and colour pickers, all aimed at supporting customisation of page display. The benefits of such features centre on exposing accessibility features that help people who otherwise may be unaware that (a) it is possible to adjust the appearance of a

web page and (b) they may find it easier to read—or listen to—a page with these changes having been made. Given that studies have found that many web users, particularly older web users, who may benefit from accessibility features are not aware of their existence [10, 21, 22], there is a sizeable audience that would immediately, and positively benefit.

The arguments against this approach focus around the “teach a man to fish” philosophy [18]—in that by providing such features on a web page, a user may then expect to find them consistently provided on all other web pages they visit, which will not be the case. Instead, the user is encouraged to make adjustments to their browser, or in the case of text-to-speech using an assistive technology, which can then be applied to all pages they view. Accessibility features provided “beyond the browser” add complexity to the information provided, and promote a distorted view of the responsibility for accessibility as being more on the information provider than on the user—analogueous to television presenters being told to shout in order to accommodate listeners who are hard of hearing.

The result is that there is a lack of consistency, both from the developer perspective and from the user perspective over where accessible design ends and assistive technology begins. The above argument is further blurred by the concept of dynamic diversity—where a user’s needs change over time—as discussed in the following section.

While legal obligations exist, where the responsibilities of organisations providing technologies to ensure they are optimally accessible remain unclear, there may be an unwillingness to take action—particularly if there are perceived to be no business benefits in improving accessibility.

3.2 Commercial Assistive Technology is Undiscoverable and Immobile

Typical commercial AT software can cost a great deal—from £300 to £600 (\$550 to \$1,000)—and this doesn’t include very expensive hardware such as Braille displays (typically £100/\$180 per indi-

⁸e.g. Browsealoud, <http://www.browsealoud.com>

vidual cell/character). This is most unfortunate because many disabled people can be considered as living in poverty⁹.

The same expenses may be significant barriers for developers who are interested in incorporating accessibility into their products. Though open source adaptations exist they are not popular; possibly due to their relative infancy and a lack of coordinated marketing—though projects such as OATS aim to amend this¹⁰. Additionally, while open source AT can provide a good basis for accessibility testing, developers who want to test for and address the idiosyncrasies between and within different commercial ATs will also be faced with significant expenditure.

It is also important to consider how discoverable and available a given AT is to the user it is intended to support. Currently, a user must go through several steps: (a) awareness of an AT; (b) awareness of their need for it; (c) procurement; (d) installation and configuration and (d) usage (including any adjustments in configuration). Some of these problems may be addressed in domain-specific solutions [2]. These issues are dealt with further in section 5.

Although table 1 shows that ATs or at least provisions for them are available in most major OSES, this doesn't tell the full story: the ATs supported are aimed at people with more severe impairments and not those with minor-moderate and sometimes transient disabilities discussed here. Table 2 highlights the lack of support for this large group of people.

3.3 Low Recognition of Dynamic Diversity

An inappropriate model of accessibility support assumes a user's access needs are static, and that a single AT, once provided and configured, solves all of the user's problems. In reality, though, there is ample evidence to indicate that capabilities are dynamic—sensory, physical and mobility impairments decline over time as part of the ageing process, but may fluctuate over shorter timeframes. Additional factors, such as the availability of aids such as reading glasses, or medication, also lead to changes in capability that may not always be predictable, yet may impact on a user's ability to use a computer—even with assistive technology present. This is the concept of dynamic diversity [15] and can also include large fluctuations over a period of years for people from many different backgrounds [4].

As will be discussed later, research into more adaptive systems is beginning to gain traction. We

submit that in order for accessibility techniques to penetrate into the mainstream—where they would undoubtedly be of great use to even more people than they currently are—then considering accessibility as providing adaptations as and when required is an important step.

3.4 Limited Data on Capability Change

We know from IGDA [1] and others [10] that there are many disabled people, in a number of different groups, and many others that could benefit from AT. There also exist standard models of human capabilities [9], including some data on how these capabilities are affected by disabilities [3].

There is currently a dearth of detailed and *long-term* studies covering the needs of both disabled and non-disabled users who may suffer minor-moderate difficulties (sometimes many simultaneously)—i.e. into the effects of dynamic diversity on AT needs as discussed above.

3.5 The Legal “Threat”

That there are legal obligations on organisations not to unjustifiably discriminate against disabled people is relatively well accepted; however there appears to be significantly less awareness over how these responsibilities translate in a technical context. The situation is complex, given that organisations may have obligations to: (a) provide disabled employees with the most appropriate assistive technologies for their access needs and (b) provide information and services in an accessible format, to employees and to customers. The situation is further clouded by the fact that implementation of accessibility may only be required “where reasonable”¹¹

Paradoxically, the relative lack of legal action in the area indicates that there is not yet a culture of settling issues of perceived discrimination in this way in court, at least not in the UK. This may betray a lack of knowledge amongst disabled people over the existence and availability of suitable assistive technology, and in the responsibilities for digital information providers to provide their information in an accessible way such that assistive technology can effectively present it.

3.6 Poor Transfer from Research to Industry

As stated, the assistive technology industry is small, and by its nature, highly dependent on trends and

⁹Results of a study into the economic status of disabled people: <http://www.jrf.org.uk/knowledge/findings/socialpolicy/060.asp>

¹⁰<http://www.oatsoft.org/>

¹¹http://www.rnib.org.uk/xpedio/groups/public/documents/publicwebsite/public_rnib003561.hcsp#P38_3742.

Table 2: Comparison of adaptive accessibility features in common OSes (latest versions; including bundled applications)

Feature	Mac OS X	GNOME	Windows
Sharing user’s settings across machines	Yes	Yes	Yes*
Apply user’s preferred settings automatically on log-in	Yes	Yes	Yes
Suggesting appropriate assistive features to user	No	No	No
Checking for appropriateness of enabled assistive features	No	No	No
API for informing applications of users’ needs	No	No	No
API for applying adaptations to applications’ content	No	No	No

* Sharing AT on Windows would usually require that the low-level components of the AT are installed on any machine the user may use. This can cause incompatibilities with certain graphical applications, affecting other users.

developments in popular operating systems (overwhelmingly Microsoft Windows) with which the assistive technology will have to work. Thus when academia presents more innovative approaches to solving accessibility problems, assistive technology vendors may be reluctant to invest in integrating these approaches if they do not align with business constraints—if a solution does not fit into the established model (accessibility APIs in the OS, or the internal structure of the AT itself), then the time required to implement it may be considered a prohibitive expense.

By way of example: a research project into adaptivity may use a given system for storing and maintaining users’ preferences, which in turn uses one of a myriad different machine-learning algorithms. There will also be a user model, possibly quite specific to the adaptation being developed—and every coder has their favourite language. It is clear how it can sometimes be very hard to turn these proofs-of-concept into mainstream products.

4 Related Work

As discussed, accessibility, despite the legal and charitable aspects, often comes down to being a business decision on the part of application and content developers—the delivery of accessibility can never rest solely on the shoulders of AT vendors¹². In very simple terms, achieving mainstream in-

dustry adoption of accessibility techniques may be considered as fulfilling two conditions: (a) demonstrating sufficient return on investment for software developers to implement accessibility features or comply with the relevant standards and (b) decreasing the cost to them of implementing these features or standards.

4.1 Adaptivity as a Means to Mainstream Accessibility

One key problem is that of providing incentive for developers and content creators to make their work accessible. This is partly addressed by recognising that accessibility concerns apply to many more people than most initially believe—including many adults [10, 19] and older users [20, 15]. The argument for better support of older people is that they often experience capability fluctuations and decline due to the ageing process—and may also experience multiple disabilities at the same time. Under this much wider and more realistic definition of who is affected by accessibility, it is in content providers’ interests to make their work accessible, particularly as a growing proportion of society will benefit [13].

Further, yet more people may experience “disabilities” brought about by limitations in the devices in use (e.g. small-screen PDAs), or imposed by the environments they find themselves in (e.g. particularly bright or noisy places). These effects, as with many of those brought about by the ageing process, may well be minor and temporary, but they do exist and could, potentially, be addressed using the same types of technology developed for those with more severe impairments—e.g. text-to-speech, key debouncing, content filtering, magnification. The difference between this example and the rather static

¹²It *cannot* because even with the inclusion of “scripts” and “maps” to improve application accessibility, third-party software cannot fundamentally change the behaviour or presentation of most common applications. It *should* not because accessibility is also a social concern that affects many people and, as such, should be an expectation of society as a whole.

nature in which current commercial ATs are used is in the *extent* to which these techniques would need to be applied and the *duration* they would be required for.

Due to the fact that our perceptions of an interface and the situations in which we use it are variable, both between users and over time for each user, it is clear that no single interface could satisfy the needs of all users—though attempting to design interfaces that have reasonable defaults and can satisfy most users “out-of-the-box” is a laudable goal (this notion is sometimes known as “Universal Design” [23]). At a fundamental level, some of the content and other standards discussed above do provide reasonable baseline values for users’ capabilities [27, 9]. Adopting these standards increases the likelihood that a given accessibility adaptation will be able to render any given content accessible for a user with differing capabilities because standard methods for adaptation may be used (such as the DOM [20]).

As one of our goals is to reduce the effort on the industry-based software developer, let us consider that ideally their involvement after designing their application and interface (ideally using a technique not dissimilar to their current one), should be over. However, there is clearly a lot of work necessary to provide the dynamic adaptive behaviour discussed above.

On the one hand we require a range of adaptations: from those for motor-impaired users [12] to people with colour deficit [17]. It is also necessary to detect problems that the user may be having, so that these adaptations may be brought in only when needed—this could be when the user enters a noisy environment (simple to detect on most portable hardware) or it could be in relation to disability, such as motor control problems [16]. Finally, the configuration of the adaptations and the system controlling them is an important and complex issue. Trade-offs exist between the aggressiveness of the system in employing adaptations and the user’s sense of control, as well as issues surrounding the privacy of configuration information, which could be used to infer personal and possibly identifying details about the user [24].

Though it has been hinted at in some of the literature, there has been little practical work in the area of applying adaptations (a) system-wide—i.e. across a range of applications rather than in one particular application¹³, allowing lessons learnt about the user’s needs in one application (such as contrast ratios and text size) to benefit others (b) to turn express GUIs in a wide range of other modalities.

¹³application-specific usage/preferences data could be held by the OS in much the same way that settings for individual window display are now

In a system designed to cope with fluctuating user capabilities and device and environmental constraints, there is a possibility that two conflicting adaptations may be required at the same time. Though sometimes there are solutions to this (such as swapping full-screen magnification for GUI widget enlargement in systems designed for vision-impaired and motor-impaired users [11, fig. 5.4, p. 113]), there has been little work on what the general case—and solution—may be.

There are a number of existing systems designed to provide information and interfaces in an accessible way to users, such as AVANTI [8]. These projects lay a lot of the groundwork for a more general system (for which an architecture is proposed below) but are generally quite domain-specific in nature so cannot be applied widely as-is.

4.2 Encouraging Industry Adoption

Even large companies seem to have to put a great deal of effort into including accessibility features (or simply standards and hooks for third-party adaptations). In some cases they can be a success, allowing people to use a system that it was not possible to use before¹⁴ but equally there are examples of large companies failing to meet accessibility demands¹⁵

If accessibility was an automatic side-effect of using existing development tools, it is unlikely anyone would avoid it. Though our field is a long way from this goal, recent work has sought to begin bridging that gap.

Several standards exist for abstract (XML-based) mark-up of user interfaces, though only some may be suitable for use in adaptive systems [25]¹⁶. Another type of abstract user interface is used in decision-theoretic interfaces [11, chapter 3], in which an abstract UI is transformed into a concrete interface for rendering (this could be with the aid of one of the XML-based systems—this work also proposes that plug-ins for contemporary development tools should be produced that allow the computer to infer the abstract UI specification from the actions of a developer designing the GUI. Currently, adopting XML-based UI mark-up would require considerable re-training for developers, so is unlikely to happen until the technology matures and offers further benefits.

Any industry with established working practises will be averse to change. It is clear, therefore,

¹⁴VoiceOver, a screen reader from Apple, allows access to the Mac <http://www.apple.com/voiceover/>.

¹⁵e.g. many have found font sizes on the Apple iPhone too small: <http://discussions.apple.com/thread.jspa?messageID=7606002>; <http://code.google.com/p/iphonefrotz/issues/detail?id=27>.

¹⁶This paper does not discuss the XUL standard from Mozilla or the XAML standard from Microsoft, but both of these are specifically GUI-focused and a truly adaptive interface may not be graphical at all times.

that either: research into AT (and HCI) proposes new development techniques that are different than those of today but incorporate other significant advantages as well as accessibility or, most likely in the short-medium term, we develop ways to integrate accessibility into contemporary development environments, as above.

The challenge of attempting to support adaptivity in current applications is great and should not be overlooked.

5 Proposals

There are many large challenges facing accessibility as a field as it tries to gain true mainstream recognition. This section proposes a small number of ideas as to how we may continue this journey, building on existing work, and how we may begin to encourage developers as to the advantages of adaptivity.

It should be borne in mind that these are very high-level proposals, partly due to space restrictions but mainly due to the fact that they are suggested directions for the field to take—and, though they don't claim to solve all problems, we submit that they will help us tackle a range of related matters.

5.1 From Monolithic ATs to a Climate of Adaptations

Contemporary ATs are, as discussed, expensive and static in nature. Due to the way they are often implemented, they can preclude the use of other ATs concurrently¹⁷. Let us assume that, due to the potentially large market for adaptive systems, developers would be amenable to using a framework for adaptivity if using it required little effort on their part.

It is proposed that the OS should include a lightweight library that is transparently linked to all applications (much the same as MSAA or AT-SPI are now) to allow them to communicate with individual *adaptations*. These adaptations may be interface renderers, allowing widgets to be presented in a range of modalities, or input handlers that may perform operations such as key debouncing.

There already exist a wide range of *preference* and machine-learning systems that could be used/enhanced to track user preferences over time and across different applications [11, 8]. Given a suitable *user model* and *problem detection* framework (see below), difficulties that the user experiences could be tracked in a similar way, to allow future decisions on adaptations to be more accurately calculated.

¹⁷This is particularly true on Windows, where most ATs have to embed themselves deeply into the system, using “interceptor” drivers to recognise the presence of custom widgets, for example.

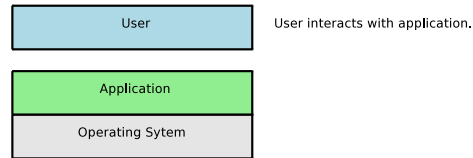


Figure 1: Architecture of system with no AT installed.

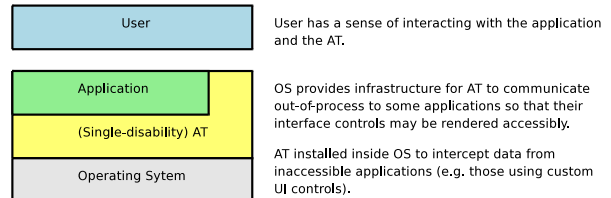


Figure 2: Current monolithic AT arrangement.

Each individual adaptation would be very simple; capable of only one style of rendering—such as text-to-speech—and would have the ability to take parameters controlling the adaptation. The system would need to determine which adaptations must be brought in at any given time; this could be achieved by way of a *controller* process (provided with the OS/adaptivity library), which would respond to predicted or detected problems the user faces, as well as environmental constraints. Multiple adaptations may be usable concurrently on a given interface (or parts of it) so, for example, the user whose eyes tire when reading a large amount may have certain types of content spoken aloud but toolbars and menus simply enlarged.

Figures 1, 2 and 3 shows an example of how such a system may be structured and how it compares to current systems, both with and without AT. It should be noted that the lightweight adaptation framework could easily be developed separately from and later installed into the OS (this would be required during the proof-of-concept stage), though it should ideally be included with an OS.

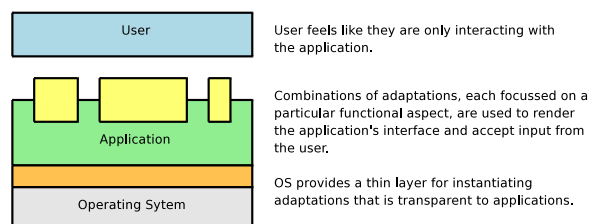


Figure 3: Proposed adaption-based system (bottom).

5.2 Problem Detection and Control of Adaptations

The controller discussed in the previous section would be responsible for coordinating which adaptations are employed and to what extent, based on predicted or detected problems. The controller would react to device and environment constraints, detected and predicted user impairments and known user preferences.

With regards to user models and problem prediction and detection, there are two contrasting possibilities: either the user model used to detect problems (and possibly the problem-detecting code itself) is included with the OS, or it is bundled as separate adaptation-like plug-ins tailored for different user types. In the former case, the user would have no work to do before the system could be monitoring their progress and able to offer help. Given that we are all human and a range of user models exist, even some that even take into account the effects of disabilities, it may be possible to adopt a standard user model.

Alternatively, in the latter case, the system would only be able to monitor the user after the appropriate monitoring plug-ins had been installed. This invites the problems associated with lack of discoverability, but does mean that the models and detection algorithms may be developed separately to the OS and low-level adaptivity framework. In reality, it is likely that a hybrid approach, where very simple detectors are included with the OS, will be adopted. This is similar to the current approach used by Windows, for which the user is required to purchase ATs if needed: a number of cut-down ATs are provided with the intention that they are just enough to help the user get the system installed and running.

The issue of bootstrapping—arriving at a reasonable starting profile and set of adaptations for a given user—has been raised in other work [20, 24] and clearly needs further work. Perhaps when the social challenge of encouraging people to *expect* adaptivity features has been overcome, this will cease to be a problem as systems will include user models.

Another issue is that of processing power—some devices are not able to support such user modelling. In this case, such devices can be designed to use web services or “cloud” computing to offload the work onto more powerful systems.

5.3 Infrastructure for Adaptation Discovery

It is not proposed that the OS must include all conceivable adaptations—rather that problems are detected and adaptation plug-ins be downloaded when required (possibly including a purchasing stage).

Adaptations could be advertised via a range of directory services, possibly run by the OS vendor and funded by revenue from adaptation purchases.

5.4 Key Challenges

We have discussed the potential of accessibility and access technologies to help a range of people and suggested a high-level system architecture that brings together related work in order to achieve this. A number of key challenges exist before such a system could be put to use in the mainstream.

The benefits of adaptivity in the mainstream

should be demonstrated in order to raise awareness and promote adoption of the techniques already developed and—considerably more importantly—*enforce the expectation in users that adaptive features must be included in products*. Once this social step is taken, it will no longer be necessary to continually point out that improved accessibility (adaptivity) could help users of each new technology.

This will require collaboration on a large scale and include the current AT and mainstream software industries. Achieving mainstream adoption may be considered as increasing both return on the investment of developers and decreasing the cost of implementing accessibility.

Academic research must continue producing and refining accessibility techniques and conducting studies into the nature, requirements and effectiveness of these techniques *regardless of consideration for the industry applications*. This ensures that radical new research may still take place and is still likely to contribute to the goal of increasing return for industry-based developers.

A technique for creating truly adaptive interfaces

based on multiple small and concurrent adaptations needs to be developed. This will provide a technique for choosing the appropriate modalities for presentation of interface elements and any required adaptations (analogous to current ATs, though much less obtrusive), based on user and device capabilities, as well as any constraints imposed by the environment. Furthermore, the interfaces generated must be able to adapt to these changing needs *at run-time*.

Work by the authors is ongoing in this area.

Easier abstract interface specification tools

must be created, so that developers are more likely to use tools such as the above to give their products flexible interfaces. A two-pronged approach may be used for this:

creating tools for developers that promote the creation of accessible UIs (such as those proposed in the literature, discussed above); also tools may be developed to help at least semi-automatically patch existing applications in order to make their interfaces more flexible.

Integration of Adaptivity Services into mainstream OSes and devices would enable a well-known method for commercial, open-source and research adaptations to plug into and thus be deployed widely with ease, as and when users require them. Discovery of appropriate adaptations could be made possible if they were made available in a known location. In the interests of preserving forwards compatibility, the mechanism would have to be lightweight and easily upgradeable.

Work towards this goal will be carried out as part of the Sus-IT New Dynamics of Ageing (NDA) project.

Similar approaches for content adaptation will need to be developed. When content creators adopt standards such as the DOM, it is possible for many successful adaptations to be made [20]. Similarly, adopting domain-specific best-practises, e.g. appropriate markup of learning object meta-data, many intelligent decisions on content appropriateness, filtering and rendering requirements can be made by a computer system [8].

Capability Change and Adaption Usage Data needs to be collected in order to provide empirical data to tune existing models. This will also be carried out as part of the Sus-IT project.

6 Conclusions and Future Work

We have discussed a small number of the many positive achievements of accessibility research and the AT industry. We have also discussed some serious social and technical challenges that must be tackled before we believe the full potential of current research and industry developments may be realised—and have proposed important areas for future research and collaboration, both internally and with industry.

A number of the challenges are being addressed by the authors' ongoing work and will be reported on in due course.

7 Acknowledgements

This work is part of an ongoing project and has been partially funded by the Sus-IT network of the New Dynamics of Ageing programme and the Grundy Educational Trust.

References

- [1] K. Pierre, J. Chetwynd, B. Ellis, D. M. Hinn, S. Ludi, and T. Westin. Game not over: Accessibility issues in video games. In *Proc. of the 3rd International Conference on Universal Access in Human-Computer Interaction*. Lawrence Erlbaum, 2005.
- [2] Jeffrey P. Bigham, Craig M. Prince, and Richard E. Ladner. WebAnywhere: a screen reader on-the-go. In *W4A '08: Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A)*, pages 73–82, New York, NY, USA, 2008. ACM.
- [3] Pradipta Biswas, T. Metin Sezgin, and Peter Robinson. Perception model for people with visual impairments. In *Proceedings of the 10th International Conference on Visual Information Systems (VISUAL 2008)*, LNCS 5188, pages 279–290, 2008.
- [4] T. Burchardt. The dynamics of being disabled. *Journal of Social Policy*, 29(4):645–668, 2000.
- [5] P. John Clarkson and Simeon Keates. *Countering Design Exclusion: An introduction to inclusive design*. Springer-Verlag UK, March 2003.
- [6] Jon Dodd. Accessibility from the front line: a uk industry perspective of web accessibility. *SIGACCESS Access. Comput.*, (82):25–29, 2005.
- [7] Alistair D. N. Edwards. The rise of the graphical user interface. *Libr. Hi Tech*, 14(1):46–50, 1996.
- [8] Josef Fink, Alfred Kobsa, and Andreas Nill. Adaptable and adaptive information provision for all users, including disabled and elderly people. *New review of hypermedia and multimedia*, 4:163–188, 1998.
- [9] Edwin A. Fleishman, Marilyn K. Quaintance, and Laurie A. Broedling. *Taxonomies of human performance: The description of human tasks*. Academic Press (Orlando Fla.), 1984.
- [10] Forrester Research & Microsoft Corporation. New Research Study Shows 57 Percent of Adult Computer Users Can Benefit

- From Accessible Technology. <http://www.microsoft.com/presspass/press/2004/feb04/02-02AdultUserBenefitsPR.mspx>, 2003.
- [11] Krzysztof Z. Gajos. *Automatically Generating Personalized User Interfaces*. PhD thesis, 2008.
 - [12] Krzysztof Z. Gajos, Jacob O. Wobbrock, and Daniel S. Weld. Automatically generating user interfaces adapted to users' motor and vision capabilities. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 231–240, New York, NY, USA, 2007. ACM.
 - [13] L.A. Gavrilov and P. Heuveline. Aging of population. In Paul Demeny and Geoffrey McNicoll, editors, *The Encyclopedia of Population*, volume 1, pages 32–37. Macmillan Reference USA, 2003.
 - [14] Dimitris Grammenos. Game over: learning by dying. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1443–1452, New York, NY, USA, 2008. ACM.
 - [15] Peter Gregor, Alan F. Newell, and Mary Zajicek. Designing for dynamic diversity: interfaces for older people. In *Assets '02: Proceedings of the fifth international ACM conference on Assistive technologies*, pages 151–156, New York, NY, USA, 2002. ACM.
 - [16] Amy Hurst, Scott E. Hudson, Jennifer Mankoff, and Sherri Trewin. Automatically detecting pointing performance. In *Proceedings of the ACM International Conference on Intelligent User Interfaces (IUI) 2008*, January 2008.
 - [17] Luke Jefferson and Richard Harvey. An interface to support color blind computer users. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1535–1538, New York, NY, USA, 2007. ACM.
 - [18] R. Johansson. Scrap text resize widgets and teach people how to resize text. http://www.456bereastreet.com/archive/200709/scrap_text_resize_widgets_and_teach_people_how_to_resize_text/, 2007.
 - [19] S. Keates and P. J. Clarkson. Countering design exclusion: bridging the gap between usability and accessibility. *Universal Access in the Information Society*, 2:215–225, Oct 2003.
 - [20] John T. Richards and Vicki L. Hanson. Web accessibility: a broader view. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 72–79, New York, NY, USA, 2004. ACM.
 - [21] D. Sloan, A. Dickinson, N. McIlroy, and L. Gibson. Evaluating the usability of on-line accessibility information. Technical report, 2006.
 - [22] A. Syme, A. Dickinson, R. Eisma, and P. Gregor. Looking for help? supporting older adults' use of computer systems. In M. Menozzi M. Rauterberg and J. Wesson, editors, *Human -Computer Interaction, INTERACT*, pages 924–931, Zurich, Switzerland, September 2003.
 - [23] Trace Centre. General concepts, universal design principles and guidelines, 1971.
 - [24] Shari Trewin. Configuration agents, control and privacy. In *CUU '00: Proceedings on the 2000 conference on Universal Usability*, pages 9–16, New York, NY, USA, 2000. ACM.
 - [25] Shari Trewin, Gottfried Zimmermann, and Gregg Vanderheiden. Abstract user interface representations: how well do they support universal access? In *CUU '03: Proceedings of the 2003 conference on Universal usability*, pages 77–84, New York, NY, USA, 2003. ACM.
 - [26] W3C. Accessible Rich Internet Applications (WAI-ARIA) Version 1.0 Working Draft, August 2008.
 - [27] Web Accessibility Initiative. Web Content Accessibility Guidelines (WCAG) 1.0, May 1999.