User Capability in an Adaptive World

ABSTRACT

General computing devices are becoming increasingly ubiquitous, personal, and mobile; and bring expectations of multimedia delivery with them that are traditionally the domain of desktop computing. Given their small form factors with restricted interaction modalities, optimizing interaction between user and device becomes critical to the usability and accessibility of the device. This paper considers how such devices may adapt effectively to the capabilities of a user given the specific context of the device's use. We present simple but powerful models of capability, capacity, and preference that allow for a wholly adaptive user experience, where the models both drive selection and configuration of appropriate interaction modalities, and themselves adapt their settings in order to reflect both changes in the environment, and the history of user behaviour. In order to achieve this, user profiles are no longer collections of purely static values, but may also contain functionally dependent properties that are changeable in response to external events.

Categories and Subject Descriptors

D.2.2 [**Design Tools and Techniques**]: user interfaces.
K.4.2 [**Social Issues**]: Assistive Technologies for persons with disabilities.

General Terms

Measurement, Design, Human Factors.

Keywords

Abstract User Interface, Hypertext, Hypermedia, Dexter Model.

1. USER BASE

Starting with the goal of maximizing the potential user base of a specific product, the first question we need to answer is:

Who precisely are our users, and what do they require of us as designers, in order to use our product successfully?

How that information is used in product design is dependent upon the design approach selected, but any approach is dependent upon the answer to the question posed.

It is the latter part of the question that is most evident in user interface design. If we consider the computer programs of common devices such as mobile phones, they generally provide some means to personalize the product in terms of the modalities used when interacting with users. In each case, it is user preference that is recorded rather than real understanding of the needs or capabilities of the user. Those preferences may be grouped, for example into "accessibility options", with some guidance to support user choice, but essentially they remain preferences. Does this matter? After all, if we know what the user wants, then we can adapt the product to match their wishes. In this paper we argue that it does fundamentally matter, and that knowing preferred settings for particular modalities is insufficient to maximize the potential user base, and flawed in terms of the maintenance and portability of a user's personal profile over time. Instead, what is proposed is a functional model of user capability that is used to describe a user's individual capacity to interact with user interfaces, that taken together with their own preferences and understanding of their capabilities, fully describes their requirements for successful interaction with a product.

2. ACCESS FOR ALL

In addition to the many sets of highly-specific and non-porTable preference options observable in commercial products, significant work has been done in the education field to create application-independent Personal Needs and Preference profiles, culminating in the creation of International Standard ISO/IEC 24751 [1] which aims to describe, functionally, a user's needs and preferences in a given operating context. This standard builds on the work of the IMS Global Learning Consortium's Access For All Meta-data Specification [2] which describes user needs and preferences in a standardised way using the World Wide Web's XML notation [3]. The ISO standard makes it clear in section 5.2 that the XML formatted profile is expected to be generated by a software tool that is holding the user's information rather than something created and maintained by hand, but in practical terms, it is the generated XML document that forms the user's profile, and it is the XML document that the user presents to applications and devices in order to construct or adapt content presentation.

The underlying Access for All meta-data model is a hierarchy of containers that describe accessibility, so for example accessibility is defined as a collection of resource descriptions; resource descriptions are defined as a primary resource description and a

collection of equivalent resource descriptions; primary resource descriptions are defined as "EARL" descriptions [4] of transformability and flexibility and a collection of equivalent resources; and so on. A practical example of this hierarchy taken from a live project at Teesside University [5], and using the Access for All XML notation, is shown in Figure 1. For economy of space, only a fragment of the complete profile is given, with some sections, attributes and values removed or truncated.

```
<accessForAll... >

<context identifier = "ID000002" lang = "en">
   <!-- Blind user using screen reader and Braille output
        device in a classroom
   -->
<display>
  <screenReader>
    <screenReaderGeneric>
      <link usage = "required" value = "speakLink"/>
      <speechRate usage = "preferred" value = "180"/>
    </screenReaderGeneric>
    <application name = "Jaws" ... >
    </application>
  </screenReader>
  <braille>
    <brailleGeneric>
      <numDots usage = "preferred" value = "8"/>
    </brailleGeneric>
    <application name = "Duxbury..." version = "10.6"
      priority = "2">
    </application>
  </braille>
</display>

<control>
  <keyboardEnhanced>
    <keyboardEnhancedGeneric>
      <stickyKeys usage = "required" value = "true">
        <playSound usage = "required" value = "true"/>
      </stickyKeys>
    </keyboardEnhancedGeneric>
  </keyboardEnhanced>
</control>

<content>
  <alternativesToVisual>
    <audioDescription ... lang = "en" type = "expanded"/>
  </alternativesToVisual>
</content>

</context>
</accessForAll>
```

Figure 1 – Example Access for All profile

The XML in Figure 1 was generated by a user profiling tool, but remains human-readable, with the container hierarchy evident.

Figure 1 represents a blind user, using a screen reader application and a Braille output device, with the user's needs and preferences organized by section: display; control; and content. This is a common means of grouping user interface information in software engineering, exemplified by the Model-View-Controller pattern [7]. The pattern separates content to be presented from the means of presentation, and also from the means of interaction during presentation. That separation allows for alternative means of presentation and interaction to be described for the same content. In this case, two means of presentation are defined: a screen reader; and a Braille device. Access for All goes beyond a simple separation of concerns by allowing for alternative content structures in addition to the alternative means of presentation. To give a web analogy, multiple views of the same web page are possible by changing the Cascading Style Sheet (CSS) [6] related to that page, but changing content on the page requires the core HTML of the page itself to be re-written; Access for All allows for reference to those HTML changes to be made indirectly.

3. FRED IS LIKE JIM EXCEPT…

What is evident in Figure 1, is that Access for All is predominantly a hierarchy of settings and options that are, with the exception of the <content> element, literal values. There are no references to other contexts, or to elements within a particular context. There is no equivalent of "Fred is like Jim except…" anywhere within the XML, or indeed within the underlying information model. Consequently, were the context to vary (perhaps the blind student is on a fieldtrip) resulting in, say, different pitch and volume levels better suited to a noisy external environment, then every single element of the profile would be repeated to express even these minor changes.

A similar problem occurs with the <application> element. What happens if the student uses a different application when studying at home? Everything else may remain the same, there may even be no different configuration parameters between the applications, for example a screen magnifier, yet the Access for All model requires a different context, or a re-definition of context from "in a classroom" to, say, "in a quiet room" so that both applications are defined within the same context. In practical terms this makes management of a profile covering multiple contexts cumbersome; it also explains the emphasis on generation of the profiles using a software tool, rather than maintenance by hand. However, this in itself raises questions about the approach:

Assuming that the tool generating the Access for All profiles is well written, so that the same information is held only once and referred to as necessary, why is that data model not the user profile rather than a cumbersome container format that is potentially full of duplication?

Only the authors, IMS, can fully answer that question, but experience of following this alternative strategy, and which is the basis of our models of capability, capacity, and preference, does provide clues. Most notably, there is a loss of clarity when reading our version of XML (our model is also expressible in XML) compared to Access for All, and is an entirely impractical format for manual construction and maintenance. Formats easier to read and to maintain were considered, but the final decision was that the XML was an intermediate format used purely to allow easy transmission across the web, written by a program, and the format was optimised to reflect this. For simple profiles with only one or two contexts, the Access for All XML format is by comparison, capable of manual construction and maintenance. So, the context of the profile's use does fundamentally affect how it is described. Access for All is good for small simple profiles with a minimal number of contexts; the alternative presented in this paper targets the more complex general case. In particular, we consider the complexities of mobile contexts where user need varies dramatically depending upon the environmental context, and upon the physical properties of the device being used.

4. USER NEED

User need is referred to in section 5.2 of the draft ISO/IEC standard [1]:

"The information collected as an Access For All Personal Needs and Preferences (PNP) statement is associated with the user's

functional abilities and the assistive technology or other non-standard technology in use as well as other user preferences (a functional approach), rather than with the name and other details of a human impairment (a medical approach)".

But what exactly does the user need? Returning to the example in Figure 1, we have a blind user who requires a screen reader, Braille output, and a modified keyboard. Does the user need a screen reader, or does she simply wish to use one? Isn't it more likely that the user needs access to text and an understanding of the menus she is navigating on the computer, and that her preferred (and perhaps only available) choice is to use a screen reader to achieve this? Such a need is still functional in nature, not medical, but how that need would be met is quite different. Certainly with Access for All, users may express their preference, or requirement, for screen readers, and indentify their preferred settings for particular products, but that does not necessarily describe need. There is something missing.

5. CAPABILITY

What is missing from the Access for All model, is a description of user capacity to interact with particular modalities. This is specifically excluded by the ISO specification, in the explicit rejection of the "medical model" of user impairment in favour of a functional approach, but such rejection confuses impairment and capability. That difference is best described by example.

There are many ways of describing colour-blindness. It is possible, for example, to take an etiological approach as shown in Table 1, which describes a hierarchy of medical terms that fully describe most forms of colour-blindness.

Table 1 – Etiological model of colour-blindness

Property Name	Values	Parent	Description
sight	FULL PARTIAL NONE	None	Top –level property for colour-blindness. Remaining template properties only of interest for PARTIAL sight.
monochromacy	TRUE FALSE	sight	User has no colour perception
dichromacy	TRUE FALSE	sight	User has no colour perception in one of low, medium, high sets of colour receptors.
trichromacy	TRUE FALSE	sight	User's spectral sensitivity is altered on one of low, medium, high sets of colour receptors.
protanopia	TRUE FALSE	dichromacy	User has a complete absence of red retinal photoreceptors.
deuteranopia	FULL PARTIAL NONE	dichromacy	The user's green retinal photoreceptors are absent or partially absent, moderately affecting red-green hue discrimination.
tritanopia	FULL PARTIAL NONE	dichromacy	The user's blue retinal photoreceptors are absent or partially absent.
protanomaly	TRUE FALSE	trichromacy	The user has an altered spectral sensitivity in their red retinal receptors (closer to green receptor response).
deuteranomaly	TRUE FALSE	trichromacy	The user has an altered spectral sensitivity in their green retinal receptors.
tritanomaly	TRUE FALSE	trichromacy	The user has impaired blue-yellow hue discrimination.

A functional approach, from the future work section of the IMS specification [2] identifies a number of "functional" elements related to colour. One, "colorAvoidance", requires that colour is not used as the sole discriminator for information; all of the others directly reflect some form of colour-blindness. Of these, only "colorAvoidance" appears to have made it into the ISO specification. The full list is: avoidGreenYellow; colorAvoidance; useMaximumContrastMonochrome; avoidRed; avoidRedBlack; avoidGreen; avoidBlueYellow; avoidOrange; avoidPurpleGray.

Such a list is certainly more appealing to a web designer than trying to remember the meaning of protanopia and how to compensate for the impairment when designing a web page. The list however is not complete, as not all colour-blind people are wholly colour-blind. Many, such as the main author of this paper, experience forms of mild colour-blindness that shifts the neutral point within the high, medium, or low frequency ranges. In my case, this results in mild deuteranomalia, which affects colour perception in the green-yellow-red section of the spectrum, but without the dimming that can occur with, for example protanopia. Clearly, the list of functional solutions to apply for colour-blindness in the IMS specification does not perfectly address such a condition. This is the fundamental problem with the functional approach: a model of specific solutions for specific conditions is unwieldy and unquantifiable.

Table 2 – Capability model of colour-blindness

Property Name	Values	Parent	Description
sight	FULL PARTIAL NONE	None	Top –level property for colour-blindness. Remaining template properties only of interest for PARTIAL sight.
colorLow	Percentage	Sight	The effective low frequency colour perception of the user. 100% would be no impairment. 0% would suggest some form of colour blindness. A mid-value of 50% would suggest a mild form of colour blindness.
colorMedium	Percentage	Sight	The effective medium frequency colour perception of the user. 100% would be no impairment. 0% would suggest some form of colour blindness. A mid-value of 50% would suggest a mild form of colour blindness.
colorHigh	Percentage	Sight	The effective high frequency colour perception of the user. 100% would be no impairment. 0% would suggest some form of colour blindness. A mid-value of 50% would suggest a mild form of colour blindness.
intensityLow	Percentage	Sight	The effective low frequency intensity perception of the user. 100% would be no impairment. Non-zero would suggest some form of colour blindness.
intensityMedium	Percentage	Sight	The effective medium frequency intensity perception of the user. 100% would be no impairment. Non-zero would suggest some form of colour blindness.
intensityHigh	Percentage	Sight	The effective high frequency intensity perception of the user. 100% would be no impairment. Non-zero would suggest some form of colour blindness.

A capability approach to colour-blindness is shown in Table 2. In this approach, it is the effective capability to perceive particular parts of the colour spectrum that is modelled. So it is less a model of impairment, and more a model of capacity to interact with the visual design space, expressing colour-blindness within the ontology of colour perception, not the ontology of medical diagnosis. It is what the user can do, not why she cannot.

It is this subtle difference between capability and impairment that defines our approach to user profiling; we are interested in what a user can do within each of Nesbitt's physical design spaces [8].

Similar capability properties are possible for vision more generally. Some example properties are shown in Table 3. Note that while they do reflect some underlying medical condition, they describe particular user capabilities of interest to a user interface designer. The example capability set is based upon the real-life experiences of a person with Multiple Sclerosis.

Table 3 – Example capability model of vision

Property Name	Values	Parent	Description
sight	FULL PARTIAL NONE	None	Top –level property for vision. Remaining properties only of interest for PARTIAL sight.
stereo	FULL PARTIAL NONE	sight	Stereo vision.
focus	FULL PARTIAL NONE	sight	Can the user focus on a point? PARTIAL would suggest blurred/double vision. Example of NONE would be a user who can distinguish light and dark, but not images.
focusDuration	Time in minutes	focus	Length of time user can continue to focus on a point (not necessarily the same point) before experiencing fatigue.
tracking	FULL PARTIAL NONE	focus	Can the user visually track a moving item? This is not a measure of focus (the image may be blurred for instance) but it is related: identifying and tracking an image.
trackingDuration	Time in minutes	tracking	Length of time user can continue to track an image visually before experiencing fatigue. Assuming that tracking a moving image is a greater cognitive load than simply watching static images, this value should be less than focusDuration.
viewRectangle	x, y, w, h in pixels	sight	A viewing rectangle within the user's field of vision. Nominally a rectangle within a 1024x768 pixel screen on a 15" laptop mounted at a normal viewing distance from the user. Anything less than 1024x768 would typically suggest tunnel vision.
nonViewRectangle	x, y, w, h in pixels	sight	A rectangle within the user's field of vision not readable by the user. Nominally a rectangle within a 1024x768 pixel screen on a 15" laptop mounted at a normal viewing distance from the user. Any such centrally placed rectangle would suggest either poor or no central vision, perhaps only peripheral vision

Property groupings are also identifiable for the sonic and haptic design spaces, and it is possible to imagine other groupings, not related to specific design spaces, with use of language one obvious candidate. A small edited fragment of such a language-based grouping is shown in Table 4. That there can be multiple views of the same properties, each relevant to the different tasks and design spaces that a user encounters, and which appear to have some hierarchy of importance embedded within them, suggests a radically different approach to representing context than in a purely hierarchical set of containers such as the information model of Access for All.

The "minReadFontSizeForFont" property in Table 4 is of particular interest. Font size and face are both frequently offered configuration options, often at great levels of detail as is the case with CSS driven web pages. Font size is also a property in Access for All. The problem with this approach is that usually there is only one setting allowed per property, yet properties such as font size are functionally dependent on context; text on a noisy background may need to be larger than the same text on a high contrast, plain background to be readable. Further, the physical stability of the screen also plays a part, so that a person with hand tremors may find that the readable size of text depends on whether the screen is placed on a Table, or is held in their hand. This is resolved in Access for All by the use of multiple contexts, but at a cost of potentially multiplying the number of contexts. Similarly, the "minInterWordGap" property may vary between noisy and quiet environments, and between different qualities of speech synthesis.

Table 4 – Example language based properties

Property Name	Values	Parent	Description
language	FULL PARTIAL NONE	None	Can the user understand language (in any medium)?
hapticLanguageSet	Braille, HapticMap	Language	Tactile based languages understood by the user.
readSignText	FULL PARTIAL NONE	sight + signLanguageSet	Can the user read (and see) sign
writeFontSet	CURSIVE, BLOCK, SELECT	fontLanguageSet	Modes of writing text. SELECT means some form of technology e.g. keyboard, scanning, eye tracking etc.
minReadFontSizeForFont	Font size in points + font name	readFontText	Minimum readable font size for user defined in points when presented on a 1024x768 pixel 15" screen.
minInterWordGap	Time in milliseconds	readAudioText	Minimum required gap in milliseconds between words required for the user to understand the spoken word.

6. MODELLING USER CAPABILITY

One alternative modelling approach to Access for All's hierarchy of containers, with the problems of duplication and multiplication as described above, is to banish hierarchy entirely, replacing it with data-base like views of properties and property settings. To this end, two Shlaer-Mellor information models are presented, one (Figure 2) describing the organization of capability properties, and one (Figure 3) describing the organization of a specific user's settings for different contexts. The relationship of such settings to user preference was modelled separately, and is described later.

Describing an object model purely in terms of the associations between elements on the model is not new, and has existed as part of the Shlaer-Mellor method [9] since 1988, and exists today as ExecuTable UML [10].

The Capability Model scopes properties first by subject ontologies such as visual, sonic, and haptic. Subject ontologies are disjoint, so individual properties exist in exactly one ontology.

A Property, whatever it represents, is characterized by the data type associated with its related settings. Five intrinsic data types are suggested in Figure 2, covering Boolean values, numbers, numeric ranges, text, and discrete lists of alternative values (e.g. FULL, PARTIAL, NONE in Table 4). In practice, numbers, ranges, and text require further sub-typing.

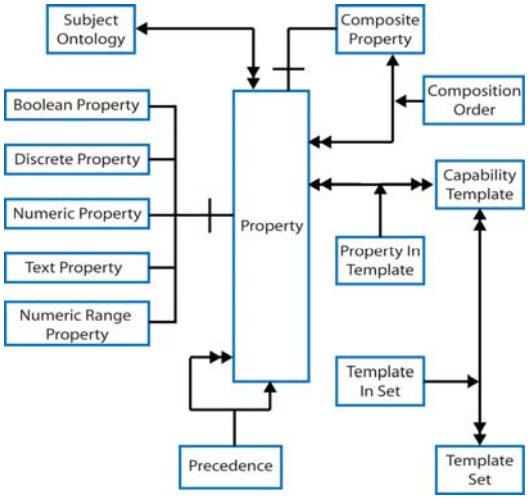

Figure 2 – Capability Model

A property may also be a CompositeProperty. This deals with properties such as the usable audio frequency range for a user, which may be described as a collection of numeric ranges measured in Hertz, with gaps between the ranges. Formalization by the CompositionOrder element allows for a natural order to be applied to the composition, for example ordering the usable frequency ranges from lowest to highest.

Properties are assumed to have a natural hierarchy of importance described by the Precedence element. That hierarchy relates to the "parent" columns in Tables 1 to 4, and describes the logical order to acquire user settings, for example it makes no sense to acquire a setting for "minReadFontSizeForFont" if the user has no sight. As is clear from the "parent columns", Properties may sometimes appear in multiple precedence trees.

Properties may be grouped into Capability Templates. These templates represent views of Properties that reflect grouping such as those of Tables 1 to 4. The same Property may exist in many templates, again reflecting the overlaps of Tables 1 to 4. This mechanism aims to help in the acquisition of user settings, grouping properties for presentation to the user in accessibility wizards, whilst reflecting the fact that the same settings may need to be presented in different combinations for different users. One example of such overlaps would be between basic and expert setup options in an accessibility wizard.

CapabilityTemplates are themselves grouped into TemplateSets. This reflects the fact that there may be several related CapabilityTemplates, for example vision may have been described in terms of colour, layout, and animation. Again, this ability to group is targeted at acquisition and maintenance of user settings.

The Capacity Model describes settings for multiple contexts. Regardless of the its logical meaning,, a Setting is characterized by the data type it holds. Consequently there is a direct mapping between, for example, Boolean Setting and Boolean Property.

Settings themselves refine the characteristics of an Entity. An entity in terms of a user profile is either a user, or a group of users. This reflects that, in some circumstances, more than one user may be represented within a single user profile. This would happen for example when users collaborate to use the same computer, say a group of students in a group exercise. In this case users require "highest common denominator" settings for the interface to allow them all full access, but they may still require personalized settings for certain activities such as text input. Access for All solution for this scenario provides a single group profile without the ability to also identify the individuals.

Settings are organized into SettingGroups, where an instance of a SettingGroup is the peer of a CapabilityTemplate. There may be many SettingGroups for a template. A SettingGroup performs the same role as the <context> element in Access for All, although a single <context> may be described by more than one SettingGroup depending upon the chosen grouping of Properties into CapabilityTemplates. The key difference between <context> and SettingGroup is that the same settings may appear in more than one group. So one setting of say, "writeFontSet" in Table 4, may appear in multiple contexts without duplication; the individual Setting is referenced in every case. Note that whilst a SettingGroup is the peer of a CapabilityTemplate, there is no requirement for there to be a Setting for every Property in the CapabilityTemplate; this reflects the fact that not all Properties are necessarily relevant to a specific user in every context; this helps remove redundancy from the contexts.

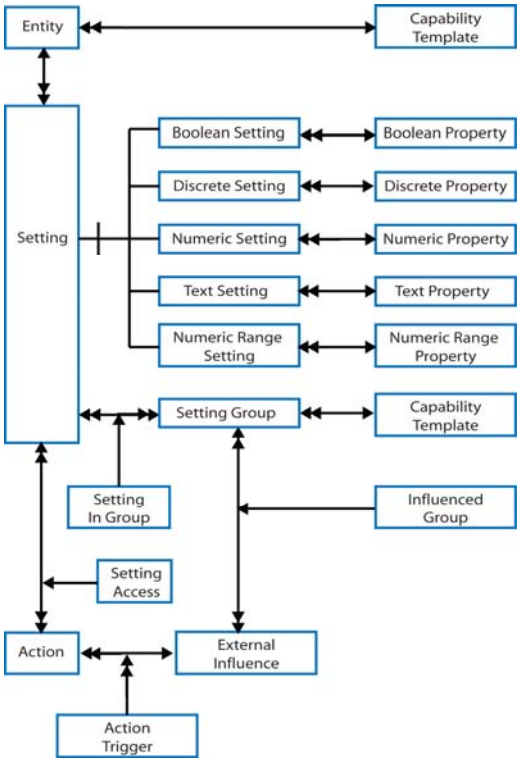

Figure 3 – Capacity Model

Where the Capacity Model steps well beyond Access for All, is in the provision of functionally dependent Settings. These aim to represent situations such as users having a mobility impairment that affects their visual capacity for particular devices or under certain environmental conditions; it is particularly handheld mobile devices that have driven this particular modelling choice.

Functional dependency is expressed through Actions. An Action is a mini program that can read and write Settings. A single action

may access many Settings. Actions trigger as a result of ExternalInfluences. An trigger may be the initial construction or maintenance of a SettingGroup for example, or it may be the result of user behaviour. This latter case allows for fully adaptive user interfaces, where the user interface, or underlying program, monitors user behaviour to detect problems or errors in utilizing particular interaction modalities, and adjusts the user's settings to better reflect their capacity. One example would be a word processor noticing that the user regularly presses the letter "d" and then corrects it to an "e", suggesting that the user's finger is slipping on a keyboard ("e" is above "d" on a QWERTY keyboard). Haptic properties related to the landing zone of keys, and use of a software "guard rail" between keys could then be updated in the user's profile. The ExternalInfluence in this latter case would be the running application.

7. USER PREFERENCE

User preference, in the general sense, is personal intervention in the choice of alternative outcomes for a given activity. That choice may be guided by personal capability and the context of intervention, but it remains an arbitrary personal selection. Those choices may not appear sensible to others, nor may it always be possible to satisfy what could be highly contradictory requests.

Within the narrow confines of user interfaces, user preference manifests itself in the choice of configuration settings over a diverse range of subject matter, with the Windows control panel a very good example; covering everything from "accessibility" options, to user account management, to printer installation.

The distinction between capability and preference is similar to the definition of an abstract user interface. Just as an abstract user interface separates what is to be represented from the manner of its representation, so we choose to separate the model of the user's physical and cognitive capabilities from their arbitrary personal intervention into how those models are applied.

The effect of that distinction on existing profiling techniques is illuminating. When inspecting the Access For All model, it is difficult to see anything other than preferences expressed in the model. But for the comment in Figure 1, "Blind user using screen reader and braille output device in a classroom", it would be difficult to identify specific capabilities at all. There is an echo of capability in the "usage" attribute that identifies whether an element is required or merely preferred, but "required" is detached from why the element is required.

Given the decision to fully separate capability from preference, it is then necessary to ask where preference modelling lies within the overall approach to user modelling. The answer must be that it exists wherever the user needs to express their preference for particular configuration settings, or in direct selection and of interaction modalities, or selection of alternative computer programs. However, it may well be that not all configuration settings need to be user configurable. This is resolved in our approach through a context-independent model of preference shown in Figure 4, and use of Shlaer-Mellor bridges to relate preferences and settings scattered throughout the disparate problem domains that make up a user interface; this implies that there is a missing abstract model of the underlying application.

Bridges provide a mechanism to describe relationships between elements in different models, for example ConfigurationSetting in the Preference Model and SettingGroup in the Capacity Model; and to Property in the Capability Model if the user can define or modify the general capability set. Overall, this provides a powerful and flexible approach to the management of user preference. If we consider bridges to "belong" to users, rather than to the "system", then it is possible to describe different User Preferences for different categories of user; such as administrators and standard users. The ability to express preference for the manner of construction of the user interface becomes context sensitive in the same way as capability profiles.

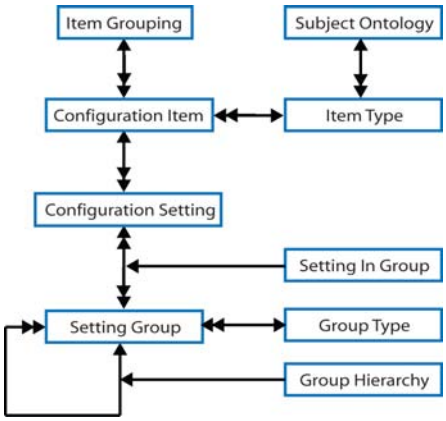

Figure 4 – User Preference Model

The model of Preference centres on ConfigurationItems which are instantiated as ConfigurationSettings, and grouped to make cognitive sense to the user. Such Setting Groups may reflect groupings found in the target domains, for example SettingGroup in the Capacity Model, but not necessarily. It may be that not all settings in a Profile SettingGroup should be offered for configuration, for example in a split between novice and expert user; in this case it may be that the settings in the SettingGroup are a subset of those provided by the Capacity Model.

Note that the Preference Model is quite different from the Capacity Model in that the Preference Model is an organization of content, not a container of values; ConfigurationSetting in the Preference Model refers to a value, it does not hold any value.

8. VERSIONING OF PROFILES

One objection to the Access for All model, is the inflexibility of its container-based model to describing small differences between contexts. The proposed solution replaces containers with an information model describing the associations between properties, settings, and groupings; in doing so, it removes duplication within the model and allowing more flexibility in group definition.

Less evident in the models of Capability, Capacity, and Preference is how the models are initially populated, maintained over time, and delivered to the device that requires them. The approach taken was to re-use a model of adaptation created for the CISNA model of adapTable hypermedia [12], which forms part of the same over-arching project to create self-adapting user interfaces.

The Adaptation Model in Figure 5 describes how Instances of data models are combined to create a single view of data; "Instance" in this case corresponds to a version of data populating Capability, Capacity, or Preference models. Each element of each model is considered to be a Table of entries for that element, with

columns of the Table representing attributes of the element. Some attributes will be explicit values; others will be references to attributes of other Tables recording the relationships between the elements.

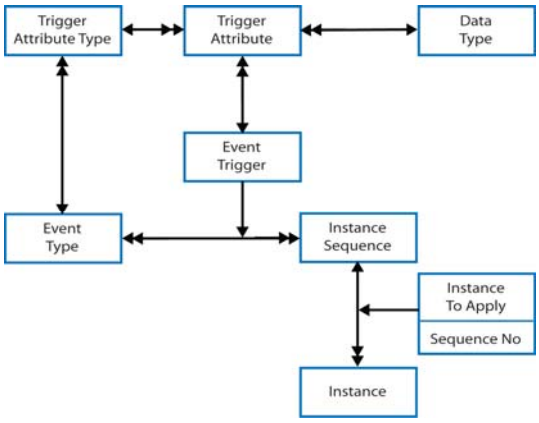

Figure 5 - Adaptation Model

Each instance adds, modifies, or deletes rows in the Tables. The concept followed here is that of a database transaction where a number of changes are defined and grouped together with a specific meaning, for example updating a bank account record. Using this approach, it is possible for a number of versions of the final merged system to be described. For example, a default model of a user interface may be modified to change the language used; or restructured to make content read better when using a screen reader; or input modalities modified to support a user with mobility impairment; or a combination of them all. Which InstanceApplication is valid (the merged instances) is dependent upon external EventTriggers, for example “New User”.

The practical consequence of this approach is that it is possible to say “Fred is like Jim except...”, and starting with Jim’s profile (which is an Instance of each of the Capability, Capacity, and Preference models) create new Instances describing only the differences between the users. Fred’s profile would contain only those Settings that are different to Jim. Whilst variations of a specific user are not the most common way of defining user profiles, variations from default templates are. This mechanism supports standard templates for blind, deaf, or indeed any other recognizable stereotype, that can then be modified accordingly. Note that the mechanism also supports smaller incremental templates, for example modifying visual settings for a user with tunnel vision, and this leads to effective support for users with spiky profiles, such as users with Multiple Sclerosis who experience varied and multiple impairments. The template system is one of the ways to initialize a user profile with data.

The Adaptation Model also provides a mechanism for automated construction and maintenance of group profiles, where an InstanceApplication can be defined to hold a group Entity; that Entity is created in the Capacity Model. The settings of the group Entity are created as functionally dependent upon user settings, with a resolution to identify the “highest common denominator” settings defined in Actions expressing dependence. An InstanceAction then defines a sequence that merges the users’ profiles, and then finally adds the new group Instance.

9. DISCUSSION

The capability modelling approach to expressing user profiles falls somewhere between a purely medical model of impairment characterized by the etiological model of colour-blindness, and the purely functional model of preference for particular interaction modalities and applications characterized by the Access for All standard. It is perhaps best described as a model of how impairment presents itself, not to doctors, but to user interface designers; giving a paradigm shift from the ontology of medicine to the ontology of design.

The argument given in favour of the purely functional approach defined in Access for All, is stated in section 5.1 of the ISO specification: “If the structure were based on information about users' impairments it would still need to address their functional abilities at some stage, as it is this information that is needed by learning systems to adapt content and navigation.” Quite, but there is a distinction between describing ability, and identifying and configuring specific modalities that are effective for the user; the two concepts are related, but they are not equivalent. We would argue that capability modelling is a much better fit to section 5.1 than the ISO specification in this respect.

9.1 Workflows

There is, however, merit in the argument that at some point it is necessary to conFigure the device or program to the user’s needs. If the underlying assumption is that this is achieved through the use of third party applications and equipment, as is also stated in section 5.1, then configuration of that technology becomes central to rendering content to the user. Such configuration issues are entirely absent from the Capability and Capacity models presented here, and it is necessary to question why that might be. The answer appears to be associated with the workflow by which the configuration settings acquire their values. The implication of a capability-based model, is that the target system utilizes a description of the user, given in terms of design ontology, to conFigure itself through some rule-based intelligence to match user need; selection and configuration of appropriate applications becomes local to the target system.

In comparison, in the Access for All functional model the user profile is equipped with pre-selected accessibility solutions garnered with their configuration settings. It is in construction of the XML that all decisions are made. Given that the Access for All XML is expected to be generated by a tool, it is this off-line tool that makes all accessibility decisions, not the target environment. This gives us a distinction between an off-line static model of user accessibility, and an on-line dynamic model with decisions about rendering made on-demand by the target system.

Whether the off-line or on-line approach is the most appropriate to managing adaptation of a user interface to user need is dependent upon the use-cases considered. In a highly controlled, highly managed environment, such as school or college, with large number of individuals operating within a defined environment, an offline system may well appeal. This is essentially the use-case from which Access for All evolved. In the more general case of adults and children interacting within the broader social context of work and play, where the range of equipment and applications is almost unquantifiable, and where the number of environmental factors is broad, the on-line dynamic approach of capability modelling may well be the only effective approach.

9.2 Adaptivity

When adaptivity based on experience of user behaviour is introduced into the argument, the workflow by which feedback from the device or application is received is critical. Clearly, the localized on-line approach is likely to be more responsive to user behaviour than an approach that requires the feedback to an off-line tool for regeneration of a profile that is then transported back to the device or program where the user is working. In practical terms, this means that only the on-line model is suiTable for adaptive systems.

The functional dependencies that exist in user capability, for example the impact on vision of environmental conditions, and potentially of mobility impairments such as hand tremors when using mobile devices, also impacts upon the choice between the purely functional and the capability approach. With Access for All's approach, functional dependency within a profile is handled by separation into multiple contexts where all dependencies have been resolved, but what is not discussed is how selection between potentially significant numbers of contexts is made in the target environment. The Access for All model only provides for a single context code for each context, with no human-readable equivalent, which means either that in reality there can be only one context defined in each Access for All XML profile, or that the target (or presumably the user) understands all possible codes and either selects the appropriate context, or offers a choice to the user; none of which appear to be ideal once the number of contexts expands beyond a trivial number. In comparison, the Actions built into the Capacity Model, resolved automatically and dynamically as the profile is accessed by the target system appears to be a more effective and scalable solution.

10. CONCLUSION

This research has raised question regarding the acquisition and transport of user settings in general. Where exactly are the properties and settings that are found within a user profile created? How are they stored? How are they maintained and updated? Where exactly is the profile located? Is it in the information stored in some off-line tool; or is it the content stored in the file presented to the target environment? These questions raise more general questions regarding workflow, and the impact that a particular choice of workflow has on the character of the properties chosen to profile. In this paper we have discussed two potential workflows: the static off-line model represented by Access for All, and a dynamic on-line model represented by our capability model. Which is the most appropriate for a particular situation appears to depend on the scalability required in terms of the number of contexts to be handled; how much user behaviour is required to feed back into the profile, and how quickly that must happen; and the degree to which the target environment can be expected to infer application choice and to derive appropriate configuration settings. Access for All, and our proposed capability models exist at opposite ends of the spectrum for answers to those questions.

Looking purely at the capability modelling approach, the most interesting and exciting aspects of the approach relate to the support for functionally dependent settings in the Capacity Model. This allows for significantly more sophisticated user support, for example: dynamic adjustment of colour, intensity and font size in response to changing environmental conditions without user intervention to select a different context; describing changes in settings over time to account for user fatigue, and for that adjustment to be calibrated by feedback from the program or device on current user performance; or describing the impact on one setting caused by the values of others, such as the impact of stability of a display.

Taken together, this functionality describes a new generation of user profile that transforms the static data structures of existing approaches, and replaces it with the concept of an autonomous agent acting on behalf of the user that is able to adjust its settings over time to reflect both user experience and the passage of time during a user's session on a specific device or program.

This concept of agent itself leads to further consideration of how the decisions on rendering the user interface may be made. If the user's capabilities may be represented by an autonomous agent, can the same not also be said of the capabilities of the device in use, and the capabilities of the environment to support the user? Given that the demands of the user and of the device; and the constraints of the environment; may be in competition to drive selection of interaction modality, then perhaps standard Game Theory can be applied to resolve the competition issues. This leads to the fascinating possibility that the definition of accessibility (through adaptation) lies within the mathematical formalism of computer science.

11. REFERENCES

[1] ISO, ISO Access for All Standard ISO/IEC 24751-2:2008

[2] IMS, IMS AccessForAll Meta-data Overview, Version 1.0,http://www.imsglobal.org/accessibility/accmdv1p0/imsaccmd_oviewv1p0.html, accessed 24 June 2009.

[3] W3C, Extensible Markup Language (XML), Fourth Edition http://www.w3.org/TR/REC-xml/

[4] W3C, Evaluation and Report Language (EARL) 1.0 Schema, Working Draft 28 April 2009, http://www.w3.org/TR/2009/WD-EARL10-Schema-20090428/

[5] Gkatzidou, G. and Pearson E., The potential for adapTable accessible learning objects: A case study in accessible vodcasting, In *AJet*, http://www.ascilite.org.au/ajet/ajet25/gkatzidou.html

[6] W3C, Cascading Style Sheets, level 2 CSS2 Specification, Fourth Edition, http://www.w3.org/TR/2008/REC-CSS2-20080411/

[7] Gamma E. et al, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1994.

[8] Nesbitt K.V., Modeling the Multi-Sensory Design Space, In *Australian symposium on Information visualization*, - *Volume* 9 (CRPITS'01), Australian Computer Society, 2001

[9] Shlaer S. and Mellor S.J., *Object-oriented Systems Analysis – Modeling the World in Data*, Yourdon Press, NJ, 1988

[10] Mellor S.J. and Balcer M.J., *ExecuTable UML: a foundation for model-driven architecture*, Addison-Wesley, Reading, MA, 2002.

[11] Dodd, R., et al 2008. The CISNA model of accessible adaptive hypermedia. In *Proceedings of the 2008 international Cross-Disciplinary Conference on Web Accessibility (W4a)*. W4A '08, vol. 317. ACM, New York, NY, 27-36. DOI= http://doi.acm.org/10.1145/1368044.1368052