Measuring Website Usability for Visually Impaired People - A Modified GOMS Analysis

Henrik Tonn-Eichstädt
Institut für Medien- und Kommunikationswissenschaft
Technische Universität Ilmenau
Am Eichicht 1
98 693 Ilmenau, Germany
henrik.eichstaedt@tu-ilmenau.de

ABSTRACT

Web designers regularly wonder which version of a design would suit best their target groups' needs. This becomes even more complicated if the design is to comply with accessibility rules. This paper describes an interaction model of blind users' interaction strategies. This model is based on GOMS (Goals, Operators, Methods, Selection rules) and can be used to measure aspects of website usability for blind users. The model evolved from findings of user observations and field studies. It can be applied to specific layouts in order to find the 'best' alternative.

'Classic' GOMS models lack functions which are necessary for the presented GOMS model. Thus, new structures to extend the classic GOMS notation are proposed. Finally, an example GOMS analysis is run on a modified version of the ASSETS '06 web page.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics—*Performance measures*; D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User interfaces*; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human factors*; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*Evaluation/methodology*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Ergonomics and Evaluation/methodology and Interaction styles and User-centered design*; H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia—*User issues*

General Terms

Design, Human factors, Measurement, Performance

Keywords

Accessibility, Blind users, Braille, Speech, GOMS, Screen reader, Usability, Visually impaired users, Evaluation

ASSETS'06, October 22–25, 2006, Portland, Oregon, USA.

1. INTRODUCTION

The design of accessible web pages is not only a process in which general checkpoints must be kept in mind: 'Accessibility means maximizing the number of people who can use computer systems by taking into account [...] varying physical and sensory capabilities of users. By this definition, accessibility is simply a category of usability' [20]. The Web Accessibility Initiative mentions key usability features describing web accessibility: 'Web accessibility means that people with disabilities can perceive, understand, navigate, and interact with the Web, and that they can contribute to the Web' [27]. ISO standards soften these views, classifying accessibility as merely 'strongly related to the concept of usability' [9]. This paper subscribes to the view that accessibility is an occurrence of usability. Thus, usability criteria such as effectiveness, efficiency, and satisfaction [5] can be used to measure accessibility.

To evaluate the accessibility of web pages, designers can rely on guidelines and software tools that were created to simplify the evaluation process. However, the established Web Content Accessibility Guidelines, version 1.0 (WCAG 1, [25]) have proved to be incomplete and incomprehensible for practical work (e.g. [19]).

Software tools[1] can only test checkpoints which can be evaluated automatically. Especially technical issues can be rated with such a tool. Furtheron, they can merely give hints at weak points where designers should have a closer look at. These points are related to content and the basic concept of the web page. But both are a broad subset of all WCAG checkpoints.

Reviewing these tools, Abascal et al. [1] evaluate them as being too narrow and sparsely flexible. Ivory et al. [11] categorize and evaluate a lot of these tools. As a conclusion, automated tools are rated as helpful for designers. But they are also reviewed to be incomplete. Considering blind users, the authors state that even with 'issues facing blind users, there are still many open questions'.

Fukuda et al. [7] point out that most evaluation tools only focus on guidelines. Time-relevant issues are not taken into account. The authors describe the 'Accessibility Designer' (aDesigner[2]), a tool that evaluates the usability of a webpage for the visually impaired ([21]). One measure used is

[1] Just to mention a few online tools: WebXACT (http://webxact3.watchfire.com/); WAVE Accessibility Tool (http://www.wave.webaim.org/) or Cynthia Says (http://www.contentquality.com/)

[2] http://www.alphaworks.ibm.com/tech/adesigner/

'reaching time', which is seen as key measure for the 'navigability' of a page. Reaching time is an efficiency measure in the sense of usability criteria.

Calculating reaching time, 'time estimation is [currently] based on speech rate and text length'[3]. Only the use of skip links and headings is considered. As there are several possible time estimates, the shortest time necessary for reaching an element is used for evaluation. Takagi et al. [21] look forward to more flexible calculation models to integrate into aDesigner.

With the aid of GOMS [4], the execution time to complete concrete tasks can be estimated. Regarding the interaction of blind users, execution time is the same as reaching time and can be used as one possible measure of efficiency. By calculating the execution time, one quality of page layout can be assessed.

1.1 Users' profile

In order to perform a GOMS analysis, it is necessary to know the user and his/her interaction options and habits. Theofanos and Redish [22] describe the challenges and potentials for users employing a screen reader to interact with web pages. A report initiated by the British Broadcasting Corporation (BBC) highlights aspects of various people with disabilities [3]. Online tutorials[4] and works of the Web Accessibility Initiative ('WAI', [26]) use user archetypes to describe different types of limited access to web pages.

Although those works give an overview (but by no means a complete one) of the 'web life' of the disabled, they are either not yet completed or limited to certain regions (the US and/or Great Britain). Further aspects, the differences in user customs and habits, user abilities, cultural peculiarities and assistive technologies in use must also be considered.

In order to broaden this view, user observations were conducted in summer 2005. Additionally, two field studies have been run. Their findings build an additional basis for generating a GOMS model.

14 blind users were observed while solving simple tasks (finding content, acquiring content, activating element) on specially set up, accessible test pages. As the tasks changed from very specific ('Click the link 'politics'!') to more unspecific ones ('Find the paragraph about sports!'), users could use different strategies depending on the task. Participants explored the pages using headings and skip links, used the find feature, or relied on the links list to click a link. After having finished the tasks, an interview followed to question actions, habits and observed peculiarities.

Two field studies followed. There, the users' skills were surveyed in a broader participants group. 270 participants answered questions about the technologies they used and their habits on the internet ([23] presents the report in German language). 150 users participated in a second field study and evaluated different example layouts which could be used to create a HTML layout.

Key findings of the observations and the studies are:

- Users access different assistive technologies with a broad variety of functions. These functions have to be taken into account evaluating page structures and accessibility features.
- Even if the users know only few of the screen readers' features, they use the known features very confidently and instinctively. Users generally do not have to think about which key to press in order to execute a known function.
- Experienced users rely to a set of different exploration and navigation strategies (e.g. using heading list, links list or the find feature). If one strategy fails, another could be applied. In order to support the preferred strategy, each strategy has to be considered during page design.
- Although being experts with their screen reader, users will be novices of a newly visited site. If this site shows good accessibility (e.g. presenting a well defined and logical structure, skip links and/or other accessibility features), users will quickly learn about the page structure and layout and are able to interact confidently within the site.
- Users configure their screen reader individually. Among the few settings which most users have changed, speech rate is the mostly adjusted one. This and individual Braille reading rates make it difficult to run a time-based analysis.
- Before navigating in a website, blind users must navigate through a single page's elements and contents. Relevant information has to be detected; irrelevant content has to be skipped. Navigating on a page will be called 'microscopic navigation' to differentiate site-wide and page-wide navigation.
- Verification of a focused-on element is extremely important in order to use the screen reader efficiently on a website. Normally, after skipping to an element, available meta information and the content of the element are checked before the next navigation step is executed.

2. GOMS

GOMS is an acronym for 'Goals, Operators, Methods, Selection Rules,' which serve as the foundation for this engineering model. GOMS makes it possible to calculate execution times for performing tasks in user interfaces. Execution times can be used to compare design alternatives and determine the most usable solution.

Card et al. introduced GOMS in the early 1980s [4]. John (CPM-GOMS, [12, 13]) and Kieras (NGOMSL, [17]) developed variations of GOMS which were better-suited for specific analysis objectives. In order to run a successful analysis, four limitations of GOMS must be taken into consideration [4, 14, 15]:

1. *The analysis is suitable for expert users only.* According to the findings of the observation, users can be classified as experts, despite the fact that they will probably be unfamiliar with advanced features of their screen readers. Learning times can be estimated using NGOMSL [17].
2. *It can be applied to linear tasks only.* Screen readers transform visual interfaces into readable, linear output. Additionally, the interaction of users working

[3]Cited from a personal mail written by Shin Sato, co-author of the mentioned papers.

[4]e.g. http://diveintoaccessibility.org/

with screen readers is sequential, since the focus of interaction is always identical with the focus of perception. Thus, parallel task execution is nearly impossible (except for elements that are simultaneously displayed on a Braille display). Complex, parallel tasks can be analyzed using CPM-GOMS [12, 13].

3. *GOMS assumes error-free execution.* This is an idealized prerequisite. Errors of course do occur. Using the GOMS analysis, we assume that competing designs are well-created and equally usable. They are thus likely to produce the same number of errors, which will compensate each another during analysis.

4. *GOMS assumes closed tasks.* Time-intensive memory recall is not covered in GOMS analysis. The tasks which were the basis for the observations will be used as a foundation for the GOMS analysis. Working with a learned and concrete layout, a task will always be closed.

GOMS has proved to be applicable in a number of scenarios. Ivory and Hearst [10] assess GOMS suitable for graphical interfaces ('WIMP' - windows, icons, mouse and pointer). For interaction with screen readers, where the process of reading a web page is segmented into several sequential tasks, running a GOMS analysis seems appropriate.

Classical GOMS has no defined structure to be used for elaborating a GOMS model. Some examples where annotation could be derived from are given in [4]. For the following model, NGOMSL structures will be used.

NGOMSL models are written in program form, include selection rules, and allow operators at the keystroke level, which are elementary in the described model [14]. Thus it has enough complexity to describe screen reader interactions but is upright enough to be easily readable and manageable by humans.

Although NGOMSL is a 'top down' method in which methods and operators are created starting with the top-level task [14], the following approach will first focus on screen reader-specific details in order to create a general model. Later, this general model can be used in a top-level analysis. An earlier version of a GOMS approach has proved to be incomplete and inappropriate in a number of ways [6]. The model described in this paper will include the lessons learned from this previous model.

Due to screen reader flaws, characteristics and differing functions, additional interaction can be necessary. Flaws and functions are highly dependent on the product and mostly prevent using an alternative strategy (e.g. skipping between headings). For these screen readers, not all analyzed strategies are applicable.

For more detailed information about GOMS, please refer to further reading (starting with [14, 15]). Due to space limitations, a brief introduction to the writing of NGOMSL had to be left out here. But as the notation is not to intricately, the presented model is understandable enough without this introduction.

For the presented GOMS model, new structures are introduced. These structures enhance classic GOMS and do not affect existing elements in a negative way. The new structures were necessary, because some contexts could not be described with classical GOMS alone.

2.1 Basic operators' execution times

To calculate a task's execution time, knowledge of basic operator times is essential. Various publications mention times which have been determined empirically (e.g. [4, 18]). But for users using assistive technology, only one study could be found in which specific operator times for Braille are given [8]. We tried to extract basic operator times from the video taped observations. Unfortunately, the test setup was not suitable for this. Thus concrete operator times have been neglected up to this point - the model is qualitative in nature up to now. Concrete times should be determined in further research when the model has proved to be applicable.

2.2 GOMS model for screen reader use

Building a GOMS model from scratch is a difficult project which requires high expertise by the analyst. In order to provide a model which can be used by developers or an automated tool, GOMS templates will be created which can be arranged corresponding to a concrete task. The use of templates in GOMS analysis is introduced in [24].

Please note that added to the NGOMSL-structure, steps which describe operators are classified according to the keystroke level as mental (`M`), key press (`K`), homing (`H`) or system response / wait (`W`[5]). The basic operators button press (`B`), pointing (`P`) and draw (`D`) are not used so far.

2.2.1 Verification is crucial

As mentioned above, users spend considerable time verifying screen reader output. According to the guidelines, content should begin meaningfully, be written unmistakably, and be coded in correct markup in order to present usable pagelets[6] [22]. Markup will help users who know the appropriate screen reader function to process the page more quickly.

The decision which is made during verification is represented by a mental operator. The result of verification can be successful (positive) or unsuccessful (negative). As long as the user does not decide on positive or negative, verification continues. What is important: a negative decision mostly can be reached more quickly than a positive decision. This is indicated by separate `M`-operators for each decision.

Method for goal: Verify object
Step 1: Accomplish goal: acquire object fragment
Step 2: Decide: irrelevant object then return 'neg', return with goal accomplished **M**
Step 3: Decide: relevant object then return 'pos', return with goal accomplished **M**
Step 4: Goto 1

In order to consider the verification result, the goal 'Verify object' must be called in a slightly different manner than classic goals. The statement must provide an exit for the case that a negative (or positive) decision was made. Returning with a positive (or negative) decision, the parent method would proceed.

...

[5]Originally, Card et al. used '`R`' (response) to describe this operator [4]. '`W`' (Wait) was introduced by Kieras to 'make clear, that what is relevant is how long the user must wait [...]' [18].

[6]A 'pagelet' is a self-contained logical region within a page [2]

Step ...: Accomplish goal: Verify element /neg: Goto 1
Step ...: ...

2.2.2 *Microscopic Navigation*

In order to focus on an element within a page, the user must navigate to that element. He/she would use the most familiar screen reader feature. Based on the observations, three options are possible:

Quick key: next element. The user will use his/her most familiar quick key to skip elements. It is possible that the user will focus on irrelevant elements and skip those elements without further interest. Skipping an element consists of four steps:

Method for goal: browse to element
Step 1: Press Navigation quick key **K**
Step 2: Accomplish goal: Verify element type /neg: Goto 1
Step 3: Accomplish goal: Verify content /neg: Goto 1
Step 4: Return with goal accomplished

The difference between steps 2 and 3 is that the user may expect a certain element type and skip an element of another type before the screen reader begins speaking the element's content. If the user focused on an expected element type, he/she must acquire the element's content in order to decide.

List mode. Using a list mode[7] is very similar to the use of quick keys. Additionally, activating and closing the list must be done. While browsing for the element, the user now only needs to verify the element's content because he/she has already decided to focus on a certain element type which the list displays:

Method for goal: use list mode and focus on element
Step 1: Open List mode **KK**
Step 2: Wait for mode being ready **W**
Step 3: Focus next element **K**
Step 4: Accomplish goal: Verify Content /neg: Goto 3
Step 5: Close List mode and return **K**
Step 6: Return with goal accomplished

Find feature. It is difficult to describe the find feature using GOMS. Too many degrees of freedom are associated with this process. Some restrictions must be made to generate a usable approach: assuming that the guidelines with regard to page design are followed and theory can be converted into practical design, the user has predicted a correct and suitable fragment to use as a search term[8]. Using this fragment, the correct element will be among the results.

If the first search result did not focus on the expected element, the search must be repeated or the shortcut 'skip to next result' must be used.

Method for goal: use search mode and focus on element
Step 1: Open Search Mode **KK**
Step 2: Wait for mode being ready **W**
Step 3: Accomplish goal: enter search fragment
Step 4: Start (resume) search **K**
Step 5: Accomplish goal: Verify search result /neg: Step 4
Step 6: Return with goal accomplished

[7]List modes list all elements of one specific type (e.g. headings, links) occurring on the page and allow sorting of and tabbing through these elements.

[8]To avoid null results due to case sensitiveness of the find feature, blind users often type word fragments which did not contain at least the word's first character.

2.2.3 *Choices*

Considering three alternatives for navigation: which approach would a user prefer? GOMS provides selection rules for choosing one of many options. But these selection rules are based on a rational decision, which can not be assumed in light of observed screen reader use. Some users know how to search, others will use the general skip key, and a third group will use the list mode. If one option fails, users will try another one. But basically, all options are equal and must be considered during analysis. As selection rules are not appropriate in some cases, 'choices' were created. A choice is a set of equal alternatives for accomplishing a goal.

Choice set for goal: find element
* Accomplish goal: browse to element
* Accomplish goal: use list mode, focus on element
* Accomplish goal: use search mode, focus on element

2.2.4 *Acquiring content*

The actions of reading or listening to content have different characteristics. One is an active process (reading Braille is like reading a printed text), the other a passive one (comparable to listening to an audio book of that printed text). As was the case with navigation, a choice set must be defined for content acquisition:

Choice set for goal: acquire content
* Accomplish goal: listen to content
* Accomplish goal: read content

Speech output. Creating the speech output, the screen reader adds meta information to the elements' content ('heading level one...', 'Link...', 'Graphic...'). This additional information must be considered when calculating execution time:

Method for goal: listen to content
Step 1: wait for speech to start **W**
Step 2: listen to meta information $\mathbf{W}[speed * length\ of\ information]$
Step 3: listen to speech output $\mathbf{W}[speed * length\ of\ content]$
Step 4: return with goal accomplished

Step 1 was inserted here, because this step describes delay caused by the speech synthesis when entering the page's content. This step has to be taken into account every time a page is loaded or the focus returns to the page after exiting a special mode. Probably, inserting this step at other places (page introduction, list mode and find feature) would be more accurate. We will leave it here for the moment to have it in the correct context (speech) rather than at the correct position.

The system's response time depends on the amount of content and the screen reader's speech rate which every user sets to his or her preferred speed. With regard to a specific analysis in a defined context, these times are constants and should be set individually for every analysis.

Braille display. Reading Braille is composed of two components: one is reading the Braille symbols; the other is scrolling the text lines. The frequency of scrolling depends on the display size and can be calculated as n/s, where n is the number of characters and s is the size of the display. Scrolling itself is done by pressing a Braille key.

Method for goal: read content
Step 1: read content **n*Br**(*aille operator*) + **(n/s)*K**
Step 2: return with goal accomplished

The mentioned Braille operator is derived from [16]. It comprises the action of reading one Braille symbol.

2.2.5 Activating elements

If an interactive element (e.g. a link or a form element) is focused on, the user can activate this element. Usually this is done by pressing a keyboard key or a routing key on the Braille display. Pressing a key is already a keystroke-level operator. Thus, no special method will be created.

Activate element **K**

2.2.6 Homing

While interacting with screen readers and using a Braille display, the user is sometimes forced to change hand position from keyboard to Braille display or vice versa, a process referred to as 'homing.' Including homing operators for users working with screen readers is nearly as complex as inserting mental operators [4, 18]. Users who prefer Braille will seldom home to the keyboard. Users who rely on speech output are less likely to home to the Braille display. Even when a Braille user homed to the keyboard, observations showed that one hand stayed on the Braille display, which makes considering a homing operator superfluous. Homing operators should be added manually to the complete analysis if it seems appropriate.

Homing **H**

2.2.7 Text entry

Typically, with screen readers (the find feature) and on web pages (online forms) text entry is reduced to sequences of a few characters.

Method for goal: enter text
Step 1: Retrieve text from memory **M**
Step 2: Type text **n*K**
Step 3: Return with goal accomplished

2.2.8 General features

With respect to interacting with accessible web pages using screen readers, other general aspects must be taken into consideration in addition to those mentioned so far.

Individual starting point. Because an individual starting point can not be predicted in advance, the analysis should consider the possibility of some elements being focused on more quickly if the user starts at for example the end of the page as an easy-to-reach starting point. An alternative analysis considering for example starting at the end of the page should be executed.

Skip links. If the page contains helpful skip links, some users will use them to proceed more quickly through a task. To consider this, it should be kept in mind that if the verify operator is applied to a helpful skip link, the analysis continues with the next element after the skip link's target. The operator 'activate element' must be added for every helpful skip link within the page.

Page introduction. After a page has loaded, some screen readers read the page title and give a short overview of the page's structure (such as 'Page contains 3 headings and 27 links'). The user will continue to listen to it as long as it gives new information. For this, no special method is necessary to define. Acquiring the page introduction is not more than acquiring content:

Method for goal: perceive page intro
Step 1: Accomplish goal: verify page title
Step 2: Accomplish goal: verify page intro
Step 3: return with goal accomplished

2.3 GOMS Summary

Based on the templates described above, a concrete GOMS analysis can be started. Although some elements are missing (forms and tables are not considered so far), the templates can be used as presented and completed using more general, 'classic' GOMS methods and operators if necessary. An example analysis will be shown in the next section.

3. EXAMPLE ANALYSIS

The reason for generating a general GOMS model was to establish the basis for a software tool which could automatically calculate efficiency as an usability measure. This means that the structure of a page could be analyzed for users working with screen readers. Concurrent designs could be compared to identify the better solution. As one possible design alternative to be compared, the ASSETS '06 home page (http://www.acm.org/sigaccess/assets06/) will be subjected to a sample GOMS analysis.

3.1 Operator execution times

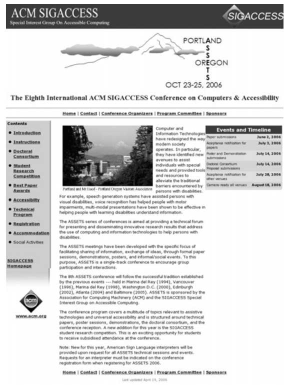
To complete a GOMS analysis, overall execution times must be calculated. Since the concrete times for each operator are not available considering users working with screen readers, the figures proposed in [18] will be used for this example run:

H(oming)	0.4 seconds
K(eystroke)	
(0.12 - 1.2 seconds)	0.28 recommended for most users
M(ental)	
(0.6 - 1.35 seconds)	use 1.2 seconds

Additionally, a time for the newly introduced Braille operator is taken from [16]:

Br(aille)	
(0.17 - 0.19 seconds)	average: 0.18 seconds

This estimate seems to be too high compared with observations. Due to the lack of a more accurate figure, 0.18 seconds will work well at the moment. In due course, it



(Title) ASSETS two thousand six dash ...
(Intro) **Page has one heading and twenty-two links**
Graphic SIGACCESS Banner
Graphic SIGACCESS Logo
Graphic ASSETS two thousand six conf...
Heading level one The Eighth International ACM SIGACCESS Conf...
Link Home
(Special chracter) vertical bar
Link Contact
(Special chracter) vertical bar
Link Conference Organizers
(Special chracter) vertical bar
Link Program Committee
(Special chracter) vertical bar
Link Sponsors
Heading Level two Contents
List of ten items
bullet Link Introduction
bullet Link Instructions
...

Figure 1: The ASSETS '06 home page has some accessibility flaws. Assuming that headlines are coded properly, the text as it is shown on the right will be generated by a screen reader. Spoken meta information is printed in bold and underlined. Some content is preceded by an italicized comment in parenthesis. Tables within the page are detected as layout tables and thus are not displayed by the screen reader.

should probably be differentiated between reading text and scanning text using Braille.

Speech rate is assumed to be 240 words per minute (0.25 seconds per word). The time for a mode to be ready for use will be assumed to be 1.5 seconds. The time for speech output to begin will be set to 0.75 seconds which is due to the observation that speech output was at least three words behind Braille display. Please note that speech rate, Braille reading times and system response times vary broadly. So the analysis should be run several times using varied time estimates to be able to compare the layouts on different time levels.

3.2 Example page

The ASSETS '06 home page does not comply with WCAG accessibility guidelines and is somewhat poor in structure. Headlines for example are not coded as heading, list items of the same level are each coded in a separate list. Relying on HTML 4.01 transitional, the page contains 17 validation errors. Some are related to unexpected end tags, others to missing start tags. Assuming a validating page and a slightly corrected page structure, the page would be accessible enough to serve as a test subject (Figure 1).

3.3 GOMS analysis

A complete analysis should calculate execution times for all important elements within the page. To limit this demo to a manageable size, the analyzed task will be to click the link 'Instructions'. An author should click this link to get information about writing a paper for ASSETS '06. The author guesses the link label correct and would use the fragment 'nstruct' (leaving out the 'I' character) for the find feature.

As the targeted element is a link, both speech output and Braille display will allow the user to rely on meta information in order to filter irrelevant elements. Speech output indicates a link by speaking 'link'. On a Braille display, links are displayed differently compared to normal text.

Figure 2 contains a compacted task analysis which shows which contents are focused using each of the four different find strategies 'next line', 'next link', 'find feature' and 'links list'. Additionally, Figure 2 gives information about the amount of text that has to be acquired by the user.

Table 1: Total execution times of the example GOMS analysis.

(sec.)	next line	next link	find feature	links list
Speech	68.36	29.34	19.43	28.49
Braille	85.04	41.68	24.50	42.56

Running the analysis, the methods are executed top-down for each strategy. During this process, some methods may be used multiple times. This is indicated by giving the number of executions in Figure 3. Additionally, for each step which is an operator, the 'external operator time' is given. Here, one difference compared to classical NGOMSL has to be mentioned: the external operator time for wait operators due to text acquisition ('listen to meta information', 'listen to content' and 'read content') can not be calculated multiplying operator time by number of executions. Regarding content acquisition, the amount of acquired text multiplied by the time for each acquisition has to be taken into account.

Kieras gives a simple equation for calculating the overall execution time for a task considering the number of GOMS statements and all operator times [18]. The results for the example analysis are shown in the 'totals' line in Figure 3 and are repeated for comparison in Table 1.

To be able to compare different code alternatives, this analysis has to be done for every (or at least every important - depending on the site concept) element within the layout and the equivalent elements within any concurrent layout. Comparing the total execution times, some layout will be more suitable for one kind of task; another will probably be perfect for a different task. Based on this, designers are able to decide which design would be best to optimize. It is unlikely to expect one (good) layout to be better for all tasks in all strategies - remember that accessible and reasonable layouts are prerequisites for this analysis.

4. SUMMARY / CONCLUSION

This paper shows an extension of the GOMS model to

		strategoy used (amount of content using speech - words / using braille - characters)			
spoken output	braille output	next line	next link	find feature	links list
ASSETS two thousand six dash[.../]	ASSETS 2006 -(.../)	5 / 13	5 / 13	5 / 13	5 / 13
Page has one heading and twenty-two links		8	8	8	8
	Page has 1 heading and 22 links	**/ 31**	**/ 31**	**/ 31**	**/ 31**
(... opening and using mode ...)		-	-	*	*
Graphic[.../]	SIGAC(.../)	1 / 5	-	-	-
Graphic[.../]	SIGAC(.../)	1 / 5	-	-	-
Graphic[.../]	ASSET(.../)	1 / 5	-	-	-
Heading[.../]	The E(.../)	1 / 5	-	-	-
Link Home	Home	2 / 4	2 / 4	-	1 / 4
vertical bar	|	2 / 1	-	-	-
Link Contact	Contact	2 / 7	2 / 7	-	1 / 7
vertical bar	|	2 / 1	-	-	-
Link Conference Organiz[.../]	Conference Organiz(.../)	3 / 18	3 / 18	-	2 / 18
vertical bar	|	2 / 1	-	-	-
Link Program[(.../)]	Program(.../)	2 / 7	2 / 7	-	1 / 7
vertical bar	|	2 / 1	-	-	-
Link Sponsors	Sponsors	2 / 8	2 / 8	-	1 / 8
Heading[.../]	Conte(.../)	1 / 5	-	-	-
List[.../]	**List**(.../)	1 / 4	-	-	-
bullet Link Introduction	- Introduc(.../)	3 / 10	2 / 10	-	1 / 8
bullet Link Instructions	- Instructions	3 / 15	2 / 15	2 / 15	1 / 13
...					
total amount of content to be acquired (words / characters)		**44 / 146**	**28 / 113**	**15 / 59**	**19 / 109**

Figure 2: Each navigation strategy causes different contents to be focused by a blind user. The table shows on the left which contents are spoken or printed in Braille. Parenthesis (speech) and square brackets (Braille) indicate at which point a user probably would skip to the next element due to irrelevant content. Additional meta information generated by the screen reader is printed bold. Please note that speech synthesis of course speaks punctuation marks (vertical bar, parenthesis, bullets and dashes) as one word whereas Braille displays them as one character.

Strategy used:	**next line**	**next link**	**find feature**	**links list**
NGOMSL Statements *Eot = External Operator time [sec.]*	*Executions / EOt*	*Executions / EOt*	*Executions / Eot*	*Executions / EOt*
Method for goal: find and activate link	1	1	1	1
Accomplish goal: find link	1	1	1	1
Activate element	1 0.28	1 0.28	1 0.28	**(done in list mode)**
Choice set for goal: find element				
* Accomplish goal: browse to element	1	1		
* Accomplish goal: use list mode and focus on element				1
* Accomplish goal: use search mode and focus on element			1	
Method for goal: browse to element	1	1		
Step 1: Press Navigation quick key	16 (16 * 0.28)	7 (7 * 0.28)		
Step 2: Accomplish goal: Verify element type /neg: Goto 1	13			
Step 3: Accomplish goal: Verify Content /neg: Goto 1	11	7		
Step 4: Return with goal accomplished	1	1		
Method for goal: use list mode and focus on element				1
Step 1: Open List mode				1 0.56
Step 2: Wait for mode being ready				1 1.5
Step 3: Focus next element				6 (6 * 0.28)
Step 4: Accomplish goal: Verify Content /neg: Goto 3				7
Step 5: Close List mode and return				1
Step 6: Return with goal accomplished				1
Method for goal: use search mode and focus on element			1	
Step 1: Open Search Mode			1 0.56	
Step 2: Wait for mode being ready			1 1.5	
Step 3: Accomplish goal: enter search fragment			1	
Step 4: Start (resume) search			1 0.28	
Step 5: Accomplish goal: Verify search result /neg: Step 4			1	
Step 6: Return with goal accomplished			1	
Method for goal: Verify object	26	9	3	9
Step 1: Accomplish goal: acquire object fragment	26	9	3	9
Step 2: Decide: irrelevant object then return 'neg', return with goal accomplished	25 (25 * 1.2)	8 (8 * 1.2)	2 (2 * 1.2)	8 (8 * 1.2)
Step 3: Decide: relevant object then return 'pos', return with goal accomplished	1 1.2	1 1.2	1 1.2	1 1.2
Step 4: Goto 1	**(generally, first acquired fragment will do for making a decision)**			
Method for goal: enter text			1	
Step 1: Retrieve text			1 1.2	
Step 2: Type text			7 (7 * 0.28)	
Step 3: Return with goal accomplished			1	
Choice set for goal: acquire content	26	9	2	9
* Accomplish goal: listen to content	**(this is either listen or read)**			
* Accomplish goal: read content				
(either:) Method for goal: listen to content	26	9	2	9
Step 1: wait for speech to start	1 0.75	1 0.75	2 (2 * 0.75)	1 0.75
Step 2: listen to meta information	14 (23 * 0.25)	8 (15 * 0.25)	2 (10 * 0.25)	1 (8 * 0.25)
Step 3: listen to speech output	12 (21 * 0.25)	8 (13 * 0.25	2 (6 * 0.25)	9 (13 * 0.25)
Step 4: return with goal accomplished	1	1	1	1
(or:) Method for goal: read content	*26*	*2*	*3*	*9*
Step 1: read content	*26 (146 * 0.18)*	*2 (113 * 0.18)*	*3 (59 * 0.18)*	*9 (109 * 0.18)*
Step 2: return with goal accomplished	*26*	*2*	*3*	*9*
Total execution time (speech / braille):				
number of statements	204 / 228	83 / 83	43 / 42	77 / 84
* 0.1 sec.	20.40 / 22.8	8.30 / 8.30	4.30 / 4.20	7.70 / 8.40
sum EOt [sec.]	47.96 / 62.24	21.04 / 33.38	15.13 / 20.30	20.79 / 34.16
Total time [sec.] (speech / braille)	**68.36 / 85.04**	**29.34 / 41.68**	**19.43 / 24.50**	**28.49 / 42.56**

Figure 3: The GOMS analyses compares four different navigation strategies, shows which GOMS-steps are used regarding each strategy and what the times for executing each step are. Calculating the overall execution time, speech output as well as Braille is considered (parted with a slash).

calculate the time to execute a task on a web page. This method seems to be applicable and could potentially enhance a time estimate as it is used in aDesigner. Considering execution time, the usability criterion 'efficiency' is used to evaluate web page design.

Still, some questions stay open. The model has not been verified so far. Calculated times should be opposed to values from user testing. Furthermore, the impact of different speech rates and Braille reading times has been put aside. Where and to what extend do these aspects influence the model's outcome?

Finally, the verified model has to be built into an automatic tool to facilitate analysis for designers and developers. It is important to keep in mind that accessibility only can be accomplished by addressing various factors, i.e. regarding technology, concept, and content must work together.

5. REFERENCES

[1] J. Abascal, M. Arrue, I. Fajardo, N. Garay, and J. Tomas. The use of guidelines to automatically verify web accessibility. *Universal Access in the Information Society*, 3(1):71–79, Mar. 2004.

[2] Z. Bar-Yossef and S. Rajagopalan. Template detection via data mining and its applications. In *Proceedings of the 11th international conference on World Wide Web (Honolulu, Hawaii, USA)*, pages 580–591, New York, NY, USA, 2002. ACM Press.

[3] BBC. *Accessibility study of BBCi: Problems faced by users with disabilities*. BBC, 2002. http://www.bbc.co.uk/commissioning/bbci/pdf/BBCi_Accessibility_Study_7-10-02.pdf (cited 2005-10-18)

[4] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction*. Erlbaum Associates, Hillsdale, New Jersey, USA, 1983.

[5] CEN. *EN ISO 9241-11 - Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11: Guidance on usability*. CEN, Geneva, 1998.

[6] H. Eichstädt. Strategien blinder Nutzer bei der Bedienung linearisierter Oberflächen. In *Mensch & Computer 2005: Kunst und Wissenschaft - Grenzüberschreitung der dritten ART*, pages 61–70, München, 2005. Oldenbourg Verlag.

[7] K. Fukuda, S. Saito, H. Takagi, and C. Asakawa. Proposing new metrics to evaluate web usability for the blind. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1387–1390, New York, NY, USA, 2005. ACM Press.

[8] H. Horbach. Taktiles lesen. *Der Blindenfreund. Zeitschrift für das Blindenwesen*, 71(3), 1951.

[9] ISO. *ISO/TS 16071 (Technical Specification) - Ergonomics of human-system interactions - Guidance on accessibility for human-computer interfaces*. ISO, Geneva, 2003.

[10] M. Y. Ivory and M. A. Hearst. The state of the art in automating usability evaluation of user interfaces. *ACM Comput. Surv.*, 33(4):470–516, 2001.

[11] M. Y. Ivory, J. Mankoff, and A. Le. Using automated tools to improve web site usage by users with diverse abilities. *IT&Society*, 1(3):195–236, 2003.

[12] B. E. John. Extensions of goms analyses to expert performance requiring perception of dynamic visual and auditory information. In *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 107–116, New York, NY, USA, 1990. ACM Press.

[13] B. E. John and W. D. Gray. CPM-GOMS: an analysis method for tasks with parallel activities. In *CHI '95: Conference companion on Human factors in computing systems*, pages 393–394, New York, NY, USA, 1995. ACM Press.

[14] B. E. John and D. E. Kieras. The GOMS family of user interface analysis techniques: comparison and contrast. *ACM Trans. Comput.-Hum. Interact.*, 3(4):320–351, 1996.

[15] B. E. John and D. E. Kieras. Using GOMS for user interface design and evaluation: which technique? *ACM Trans. Comput.-Hum. Interact.*, 3(4):287–319, 1996.

[16] T. Kahlisch. *Software-ergonomische Aspekte der Studienumgebung blinder Menschen*. PhD thesis, University of Dresden, 1998.

[17] D. E. Kieras. A guide to GOMS model usability evaluation using ngomsl. In M. Helander, T. K. Landauer, and P. V. Prabhu, editors, *The handbook of human-computer interaction. (Second Edition)*, pages 81–97. Elsevier, Amsterdam, North-Holland, 1997.

[18] D. E. Kieras. *Using the Keystroke-Level Model to Estimate Execution Times*. The University of Michigan. 2001. ftp://www.eecs.umich.edu/people/kieras/GOMS/KLM.pdf (cited 2005-10-18)

[19] J. Lazar, A. Dudley-Sponaugle, and K.-D. Greenidge. Improving web accessibility: A study of webmaster perceptions. *Computers and Human Behavior*, 20(2):269–288, Mar. 2004.

[20] P. Reed and P. Billingsley. Software ergonomics come of age: The ANSI/HFES-200 standard. In *Proceedings of the Human Factors and Ergonomics Society*, pages 323–327, Philadelphia, 1996. HFES.

[21] H. Takagi, C. Asakawa, K. Fukuda, and J. Maeda. Accessibility designer: visualizing usability for the blind. In *Assets '04: Proceedings of the 6th international ACM SIGACCESS conference on Computers and accessibility*, pages 177–184, New York, NY, USA, 2004. ACM Press.

[22] M. F. Theofanos and J. Redish. Bridging the gap: between accessibility and usability. *interactions*, 10(6):36–51, Nov.-Dec. 2003.

[23] H. Tonn-Eichstädt. Fallstudie Technolgien und Nutzer. Jul. 2006. http://studien.eichstaedt.org/Fallstudie-Technolgi.52.0.html.

[24] A. H. Vera, B. E. John, R. Remington, M. Matessa, and M. A. Freed. Automating human-performance modeling at the millisecond level. *Human-Computer Interaction*, 20(3):225–265, 2005.

[25] WAI. *Web Content Accessibility Guidelines, Version 1.0. W3C Recommendation 5-May-1999*. 1999.

[26] WAI. *How people with Disabilities Use the Web. W3C Working Draft, 10 Dec. 2004*. http://www.w3.org/WAI/EO/Drafts/PWD-Use-Web/

[27] WAI. *What is Web Accessibility, 2005*. http://www.w3.org/WAI/intro/accessibility.php