Development of an Adaptive Interface for the Electronic School Notebook

Luís Alexandre[1], Salvador Abreu[11]

[1]LabSI[2] / ESTIG, Instituto Politécnico de Beja, Portugal

[11]Universidade de Évora and CENTRIA FCT/UNL, Portugal

luis.alexandre@ipbeja.pt, spa@di.uevora.pt

Abstract. In this article we describe the construction of an adaptive interface for the Electronic School Notebook, with the ability to predict the most likely task that a user is about to perform, based on prior knowledge of actions already made in the system, combined with the action time frame. This adaptive capability will reduce the number of explicit interactions to accomplish a certain task. The system makes use of machine learning algorithms, based on decision trees and Markov chains.

Resumo. Neste artigo descrevemos a construção de uma interface adaptativa para o Caderno Escolar Electrónico, com a capacidade de prever a tarefa mais provável que um utilizador irá realizar no sistema. Esta capacidade de predição baseia-se no conhecimento das tarefas realizadas anteriormente pelo utilizador em conjugação com o espaço temporal das mesmas. Esta interface adaptativa permitirá reduzir o número de interacções explícitas com o sistema, com vista à realização de uma tarefa. O sistema utiliza algoritmos de aprendizagem automática, baseados em árvores de decisão e redes de Markov.

Keywords: Interactive systems, Students with Special Needs, Support Technologies, Human Computer-interaction, Machine Learning, Prediction, Adaptation.

1 Introduction

The Electronic School Notebook (CE-e) [1] in Fig. 1, has become a valuable tool for children with special needs, in the organization of notes in a classroom environment and also in the organization, in a logical manner, of electronic documents related to a given course.

The CE-e together with other assistive technologies, like Eugénio [2] actively support these children, allowing them to perform writing tasks within a very similar time period to the one spent by children without special needs. This tool promotes the

adoption of a methodical process of collecting notes in classroom context, as a result of this process, remarkable benefits can be obtained in terms of academic success [4].

If the system becomes aware of the task at hand by the user, it will be able to better support users, by helping in accomplishing the task.

In some situations it's very difficult to determine the intentions behind a user actions sequence, but in other situations using the domain knowledge [5] or observing the users behavior in similar situations [6], we are able to predict the goal of the user. An obvious case is the opening of a new lesson, of a certain course with the combination of the system time and the information of the student's time table.

Not only chidrem with motor impairments can benefit from these technologies, also students with cognitive impairments can work in a more autonomous manner with the help of these systems. These kind of mecanisms have already been tried in other aplications, like email clients [6], in order to free the user from the execution of routine tasks by delegating the execution responsibility to the system. This approach has also been tried in the development of Eugénio's keyboard assistant [2][7].

The project guiding principle is DWIM (Do What I Mean) [3]. This principle states that the system should behave exactly according to the user objective, instead of behaving according to a miss-executed instruction, not intended by the user. This is an unattainable goal; still it's guiding our intentions.

The main objective of this project is to implement and evaluate an adaptive interface for the Electronic School Notebook that allows students with special needs a higher level of support in performing several tasks.

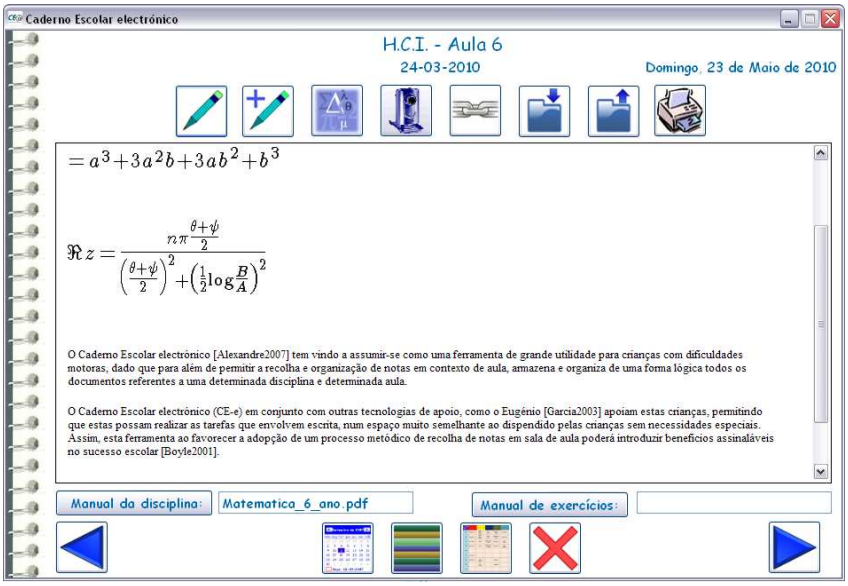

Fig. 1. CE-e notetaking interface

To attain this objective the system must possess application and user models [5] in order to identify the user intentions to provide the necessary assistance in the accomplishment of the task.

In a first moment we gathered the required information to build a knowledge base that contains all the tasks performed by the user on the system, the different actions that lead to the task and finally the context in which they were performed. To characterize the context we gathered a timestamp composed of the date, day of the week, the current time and the associated activities. These records were gathered during normal classes, in the student's personal computers, using CE-e. All the students have cerebral palsy, but this problem only affects them physically. None of these students uses any custom I/O devices, such as switches or adapted mice.

This data and knowledge base was used on the next phase of the project, were we built and evaluate support mechanisms that help users in the accomplishment of tasks, based on the knowledge contained by the knowledge base. This is the main contribution of the project. If the system is able to correctly interpret all the context dimensions, it may evolve from a set of hardcoded rules that provide a limited set of adaptations, to a system that with its continued use will be able to learn and create new adaptations, more appropriate for the user profile. This "intelligence" will only be possible with the use of machine learning algorithms.

On Section 2 of this article we will describe the sate of the art in this subject. The description of the system is made on Section 3 and finally on Section 4 we present a conclusion and propose directions for work to be done in the future.

2 State of the Art

Nowadays we are surrounded by computer systems (e.g. desktops, laptops, PDAs, among many others) and we become computer dependent, all aspects of our lives are determined by decisions or actions that took place on a computer.

However these systems could provide a greater support, if they were able to provide it in an automatic and autonomous form. This support is only possible if the system is able to determine or recognize the context in which both the user and the machine are inserted [8].

Many researchers have reported the importance and benefits that context knowledge can bring to the usability of computer systems thereby increasing and enhancing the Human-Computer Interaction [9]. The main idea behind this knowledge of the context, known as context-awareness, states that computers can feel and react based on the environment. In order to enable this reaction, computer systems should have at their disposal a set of stimuli and prior knowledge that are combined with a set of hardcoded rules. The reaction is the logical result of these calculations [10].

A context-aware system can and should try to make assumptions about the surrounding environment. In [12] the context is defined as "*any information that can be used to characterize a situation of any one entity*".

Context-awareness is often used in satellite tracking systems or in ubiquitous computing. In these systems it's possible to use sensors and other systems that capture with precision part of the context, and thus can, for example, propose and make environmental changes (e.g. environment light or open the windows in buildings, to minimize energy consumption).

Initially the context was understood has being the location, but more recently some researchers proposed that the location coulnd't be understood as the context of a situation, but rather as a component of context [12]. It was further proposed that the context could be used to build adaptive and intelligent interfaces, in which the interaction sould be as implicit as possible.

Many of the studies and attempts of adaption to context have passed through the use of satellite tracking devices. These systems use only a small component of a relatively large universe of possible adaptations to the context. A system that only uses the location to fit the context, can not be considered a context-aware system, and should be considered as location-aware system [13].

In the last years, context information has been combined with time; as result of this many researchers have proposed several types of "intelligent" interfaces (intelligent Human-computer Interaction).

Examples of this new kind of interaction are the "Adaptive and Predictive Interfaces". These interfaces try to minimize the two major problems that affect the interaction between humans and computers: (i) The considerable quantity of information and widgets that populate the interfaces; and (ii) The fact that users must decide the next step or action in a very short time [15].

The idea behind these interfaces is the following: "*Instead of being the user to adapt to the system, the system adapts to the user*". Although based on a known premise, the inherent problems are in large number and complexity. An adaptive interface must be based on cognitive models and on the surrounding environment [15]. These models try to explain the user's level of expertise and experience, through parameters such as: (i) commands made, (ii) error rates, (iii) typing speed, among others. An adaptive interface can make the adjustments in one of two forms:

- The adaptation responsibility is splited between the system and the user, for example, the system makes a small adaptation and waits for its acceptance, if the adaptation is not accepted or not satisfactory, the system makes a new adaptation;
- Alternatively, the system suggests the most adequate adaptation for the current context in a totally independent mode.

The second form is considered by many has the true adaptability. However, this kind of adaptation is very difficult to achieve and it's only possible with the use of machine learning algorithms. The concept behind machine learning can be considered as adaptive, because the algorithms of this type base their decisions on prior information (train and test sets).

However an adaptive interface also has its problems and among others we can mention the fact that the user can not create a mental model of the system, if the appearance of the system often changes. Another potential problem may result from

the loss of control that a user may feel. Finally, and perhaps the biggest problem of adaptive interfaces, turns out to be the cost and complexity of the implementation.

Another paradigm of adaptability and predictability derives from the attention and suggestion. In this case the interfaces are alert and automatically suggest actions to users.

An attentive interface automatically sets priorities and in accordance with these priorities presents to the user the most appropriate information or options regarding the current context and time, in order to optimize the resources of both user and system. With this optimization, the interaction actors never incur in overloading [16]. As in adaptive interfaces, attentive interfaces base their decisions on information and models, based on decisions previously made by the user.

On the other hand we have the suggestive interfaces, which only provide the user with clues about the next possible action, through the enhancement of certain interface components [17].

The interaction with a suggestive interface is very simple, in fact the user only has to choose one of the highlighted options or if none of the options matches the desired one, the user only has to choose the desired option. Usually suggestions are generated by a system often called "suggestions engine", which constantly observes the system state, generating a set of suggestions that matches the patterns closest to the current state. A suggestive interface can be viewed as a gestures interaction system, i.e., instead of updating the system automatically, when it discovered a similar pattern, the interface presents a set of hints and prompts the user to choose one of these.

Suggestive interfaces are an extension of predictive interfaces and are being, viewed by many researchers, as the interfaces of the future, for the vast majority of systems, including WIMP systems (Windows, Icons, Menus and Pointing Devices).

3 Architecture of the Adaptive Interface

The interface of the CE-e prototype was been designed according to a design methodology for interactive systems with focus on the user and, therefore is quite simple and intuitive [1]. However as CE-e intends to provide as much help as possible in tasks of notetaking and information organization, it's intended that the interface can alleviate, as much as possible, the user of routine and repetitive tasks, which may be particularly useful for users with severe motor problems. Thus, based on our experience, in the opinion of technicians and other experts, we decided to implement a suggestive interface on CE-e.

We made this choice because these kind of interfaces are seen as a possible path for the future of interfaces, since they exploit context-awareness in order to reduce the number of explicit commands to be performed by users, required for any given operation [17][25]. The adaptive interface, designed for the CE-e, meets all these requirements, extending the traditional notions of an interface.

cid	name	type	notnull	dflt_value	pk
0	ID	INTEGER	0		1
1	date	NVARCHAR(40)	0		0
2	weekDay	NVARCHAR(20)	0		0
3	time	NVARCHAR(50)	0		0
4	form	NVARCHAR(50)	0		0
5	discipline	NVARCHAR(50)	0		0
6	event	NVARCHAR(50)	0		0

Fig. 2. CE-e log structure

3.1 The Knowledge Base

As previously stated, the system's ability to anticipate or suggest the most probable action for a certain context, can only happen if the system has access to a knowledge base with the knowledge acquired in prior interactions with the system. To build this knowledge base researchers often use a technique called "logging".

For some years, researchers have been working in Augmentative and Alternative Communication (AAC) systems. These efforts are only justified if the impact of these technological advances can be measured. Thus, in order to facilitate the analysis of the collected data; researchers have defined a standard logging format that allows the analysis of data in a systematic way, called Universal Logging Format [18].

The log file structure proposed in [18], has three distinct parts: (i) The head, which specifies the content and format of records; (ii) Body, which is composed of *n* lines, each representative data on a log; (iii) Analysis section (optional), where some statistical analysis of the logs can be referenced.

The logs have a time component, an output or result (action), the type of device used which originated the log, type of communication that was being developed, the context (in case of a AAC tool), which words preceded the log and an indication of, where in the system, the action was triggered.

Under this format, we built a log system with the structure presented in Fig. 2 that contains the following fields: date, day of week, time, system page, active discipline and resulting action.

A log mechanism it's useful if we have a fast, precise and powerful form to extract the information we want. Inspired in [19] and on features of relational DBMS we decide to use SQLite[1]. This relational database management system, besides allowing the construction of queries in standard SQL, doesn't need to have an active server in the system. This log mechanism, adds to the capabilities of SQL, a total independence of the system, since this RDMS is added to the application via a DLL built in C Programming Language.

[1] Project Web site: http://www.sqlite.org/

ID	date	weekDay	time	form	discipline	event
1	26-02-2010	Friday	11:9:58	FormAgenda		iniciar_aplicacao
2	26-02-2010	Friday	11:10:56	FormAgenda		vista_de_livros
3	26-02-2010	Friday	11:11:6	BooksForm		abrir_caderno_Matemática
4	26-02-2010	Friday	11:11:14	NotesForm	Matemática	vista_livros
5	26-02-2010	Friday	11:11:16	BooksForm		abrir_caderno_Língua Portuguesa
6	26-02-2010	Friday	11:11:20	NotesForm	Língua Portuguesa	aceder_manual_disciplina
7	26-02-2010	Friday	11:12:21	NotesForm	Língua Portuguesa	aceder_manual_exercicios
8	26-02-2010	Friday	14:6:56	FormAgenda		iniciar_aplicacao
9	26-02-2010	Friday	14:7:11	FormAgenda		vista_de_livros
10	26-02-2010	Friday	14:7:16	BooksForm		abrir_caderno_Estudo do Meio
11	26-02-2010	Friday	14:7:22	NotesForm	Estudo do Meio	aceder_manual_disciplina
12	26-02-2010	Friday	14:56:21	NotesForm	Estudo do Meio	terminar_aplicacao
13	02-03-2010	Tuesday	9:18:29	FormAgenda		iniciar_aplicacao
14	02-03-2010	Tuesday	9:18:38	FormAgenda		vista_de_livros
15	02-03-2010	Tuesday	9:18:44	BooksForm		abrir_caderno_Matemática
16	02-03-2010	Tuesday	9:18:50	NotesForm	Matemática	nova_nota_de_texto
17	02-03-2010	Tuesday	9:26:49	NotesForm	Matemática	vista_livros
18	02-03-2010	Tuesday	9:26:53	BooksForm		abrir_caderno_Língua Portuguesa

Fig. 3. Extract from a log file

3.2 The Algorithms

For some time now, that researchers in the field of Human-computer Interaction, develop studies on the adaptability of interfaces to users, through learning systems based on machine learning, both for traditional desktop interfaces and for Web interfaces [14][15][20].

Before the implementation of the system, the logs were subjected to machine learning tools, to check which classification algorithm could offer better results.

We can divide machine learning, through classification, in two major groups: Supervised Learning and Unsupervised Learning [21]. The collected logs for the formation of the knowledge base are constituted by the time frame, context and the performed action, i.e. the user's objective. In this scenario we face a case of supervised learning, where the user actions will be the class. Supervised learning can be further divided in classification or regression problems. Since our aim is the prediction of the user most probable action, in a multiclass environment, we have a classification problem.

In the possession of real use records (Fig. 3), it was possible to start working in the "intelligence" component of the system, in order to prove the feasibility of an intelligent interface for the Electronic School Notebook. This type of tests is known as "proof of concept".

The Weka environment [22] was chosen for the knowledge base tests in search of patterns that could justify the system implementation. This environment provides implementations of the majority of machine learning algorithms, like decision trees. Our first choice fell on Decision-trees, since the system will work in real time, making queries to the knowledge base and presenting the hypothesis that best fits the context of the moment.

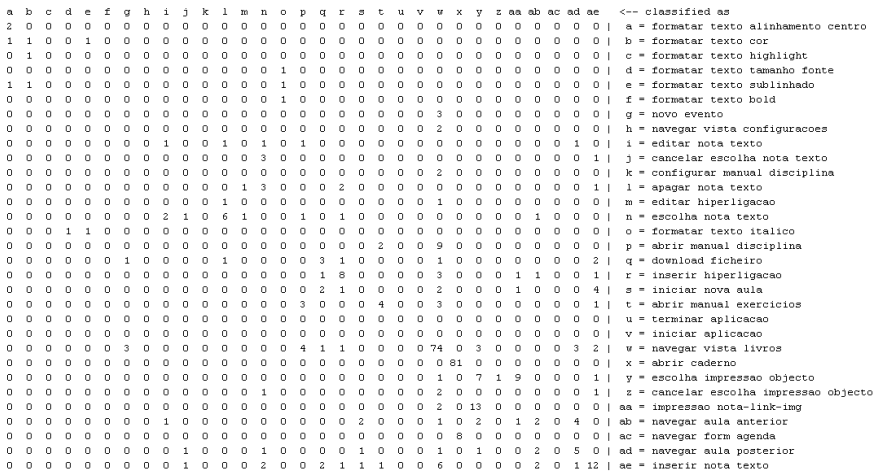

Fig. 4. Confusion matrix generated by Weka with the data from the knowledge base

The first tests carried out, using a C4.5 decision tree [23] showed that this machine learning algorithm can reach a hit rate of 50%, which justifies the development and implementation effort, since the system has forty classes. However these forty options aren't always visible, depending on the screen where the user is located, the system will only present between three and seventeen options.

Other classification algorithms were tested, searching for better results or outcomes that strengthen those obtained using decision trees, particularly Markov chains and Hidden Markov Models [26].

The inference engine of the system will have to manage a small set of attributes, as shown in Fig. 2. The engine will only have four attributes that need to be analyzed, as the fifth attribute is the class attribute, which will serve to train the classifier. However, this classifier will have to deal with a multiclass problem with a large number of classes. The confusion matrix build by Weka (Fig. 4), clearly shows the number of classes. This particularly large number of classes is a possible problem for the classifier, because the resulting tree will have a large number of nodes and leaves, thereby increasing the probability of errors in the suggestion of the most likely action. However the 52% accuracy achieved, ensure that at least one in every two suggestions is correct, thus reducing the work load on the user, necessary to perform tasks that add value to the work of the user in the system.

3.3 The Encoding

The inference engine encoding is being developed with two algorithms, a C5 decision tree, which derives from the C4.5 tree [23] and a Markov chains based algorithm. The decision tree is fastest and more efficient in the process of finding patterns and in the classification of new samples. The implementation with a Markov chains algorithm follows a different methodology. While the decision tree is being implemented with a

DLL based on open-source code, the second is hardcoded and integrated in the source-code of CE-e.

The ultimate objective of the project is the construction of a prototype using the C5 decision tree, but the encoding process of the DLL is delayed due to some problems.

As soon as possible, we wanted a prototype that demonstrates the usefulness of the Electronic School Notebook adaptive interface, so we decided to implement the inference engine of the CE-e with a Markov chains based algorithm. The final version of the system prototype will not be available with this algorithm, because the tests results were poorer, in terms of hits and speed.

After the proof of concept has been held, the DLL based on the decision tree [23], will be developed. This effort is justified by the fact that the DLL is an independent and portable component, allowing and ensuring its reuse in other systems. Another fact that justifies this effort is that the development of new versions of CE-e, with and without an adaptive interface, will be simpler and we don't have the necessity of building two incompatible versions.

With the introduction of the inference engine in the CE-e prototype, shown in Fig. 5, the system had to undergo some changes, so that the various components of the interface could be dynamically changed by the interface inference engine.

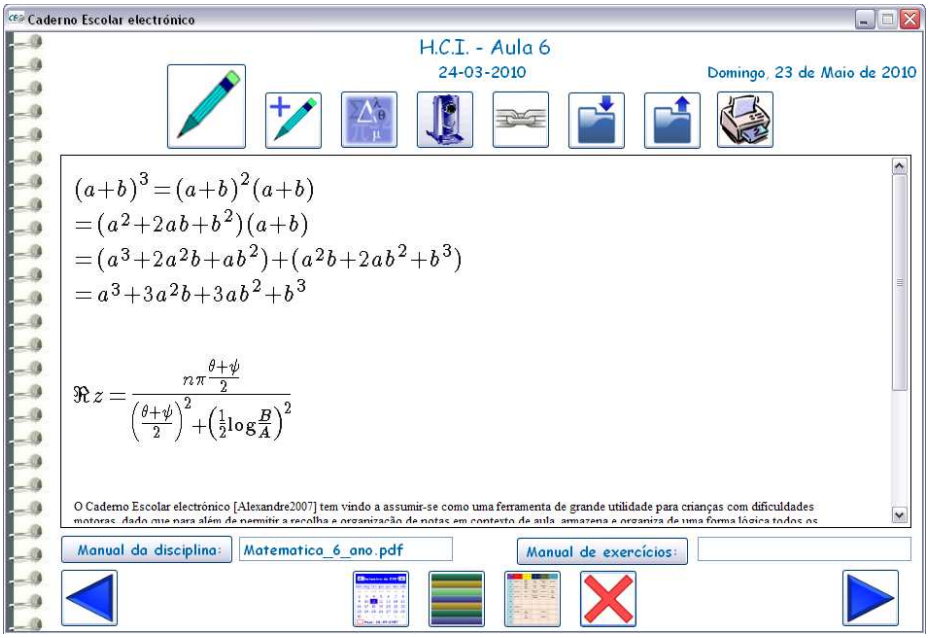

Fig. 5. System prototype with the inference engine

4 Conclusions and Future Work

In this article we propose and present an adaptive interface for a tool that aims to be an alternative to traditional school notebooks, for students with special needs, particularly those who only have physical problems, that restricts the mobility and ability to manipulate traditional I/O mechanisms.

Currently we have a prototype of the CE-e with the adaptive interface in the final stages of development. This initial prototype will only be used for testing and proof of concept, which are ongoing.

The interface makes suggestions to the user using the knowledge base, which contains information about past actions in the system. The analysis is carried out using techniques of machine learning and algorithms based on Markov chains.

With the objective of building a knowledge base that could answer the questions asked during the analysis and decision phase, a relational database log mechanism was built, using the SQLite database manager.

We want a system that continuously learns. This learning process can quickly push the number of lines, of the log file, to a few thousand. As a result, the inference engine can become very slow. A possible solution to this problem is the limitation of the log file, in number of lines or size. This limitation will eliminate automatically the logs with a certain timestamp.

However, the assumption of a temporal limitation for the logs can still change as we are yet analyzing which option will gives a higher percentage of hits. If the system only has access to the logs of the last thirty days of use, will have a temporally limited knowledge, but this knowledge may focus on options that the user has used more frequently. Moreover, and assuming that the logs can grow indefinitely, the system will have an overview of all actions and choices carried out by the user, which could bring higher probabilities of success. This decision can only be taken, after a period of use and testing with a duration not smaller than six months. These tests will determine what is the best structure for the knowledge base, "with forgetfulness" or "without forgetfulness".

A possible solution for the continuous collecting of logs is an offline processing, which will lead to the construction of an index of actions. This technique is quite used in information retrieval systems [24].

Another inherent objective of this project is to provide the system with a set of hardcoded rules, which in case no correspondence between the rule and the result obtained by inference engine, the rule will override the inference engine decision. As an example of such rules we have the timetable that can provide information about the most likely task to execute on the system, according to the time component provided by the operating system.

The suggestion mechanism highlights the most likely option. This enhancement is combined with a keyboard shortcut, to avoid a mouse movement to confirm the option. As earlier mentioned, a suggestive interface can be viewed like a pseudo-gestural interface, because instead of immediately responding to the users input, updating the interface immediately, the system first asks the confirmation after showing multiple suggestions. On CE-e the inference engine will only highlight one

option, and the user can complete the action by choosing the presented option, or can ignore it, by choosing another option, rather than the one presented.

CE-e wants to be an effective alternative to traditional school notebooks. If the introduction of an adaptive interface proves to be an asset, students with physical and motor difficulties, will have at their disposal a tool that will help them further more in the organization of all writing tasks and in the organization of electronic documents.

By now, the adaptive CE-e has an interface that can predict the most likely action for a certain context, which reinforces the lemma of the project: Help and relieve students with special needs of routine tasks, that don't add capital gains to the work that they have to perform, in order to assist in their integration into regular education.

The final testing phase is in progress. These tests will bring the answers regarding the effectiveness of the suggestive interface, designed and built for CE-e.

Acknowledgements

We would like to thank the valuable collaboration, inputs and ideas of the faculty staff of the Polytechnic Institute of Beja assigned to the Information Systems and Interactivity Lab (LabSI^2).

References

1. Alexandre, L., Garcia, L., Bruno, L.: Development of an electronic scholar notebook for students with special needs. In DSAI2007. Vila Real, Portugal (2007).
2. Garcia, L.: Concepção, implementação e teste de um sistema de apoio à comunicação aumentativa e alternativa para o português europeu. Tese de Mestrado. Universidade Técnica de Lisboa, Instituto Superior Técnico (2003).
3. Teitelman, W.: A tour through cedar. Proceedings of the 7th international conference on Software engineering. Pages: 181 – 195. Orlando, Florida, United States (1984).
4. Boyle, J.R., Weishaar, M.: The effects of strategic notetaking on the recall and comprehension of lecture information for high school students with learning disabilities. Learning Disabilities Research & Practice, 16(3), pp. 133–141, (2001).
5. Dix, A., Finlay, F., Abowd, G., Beale, R.: Human Computer Interaction, 3rd Edition. Prentice Hall (2004).
6. Maes, P.: Agents that reduce work and information overload. Communications of the ACM 37(7) (1994).
7. Rodrigues, N.: Desenvolvimento de mecanismo de interacção preditiva para aumentar o desempenho de tarefas. Projecto de Licenciatura. Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Beja (2007).
8. Blum, M.L.: Real-time context recognition. Master's thesis, Department of Information Technology and Electrical Engineering, Swiss Federal Institute of Technology Zurich - ETH (2005).

9. Bradley, N.A., Dunlop, M.D.: Toward a multidisciplinary model of context to support context-aware computing. Human Compute-Interaction, 20:403 – 446, (2005).
10. Schilit, B., Adams, N., Want, R.: Context-aware computing applications. IEEE Workshop on Mobile Computing Systems and Applications - WMCSA'94, 1:89 – 101 (1994).
11. Schilit, B.N., Theimer, M.M.: Disseminating active map information to mobile hosts. IEEE Network, pages 22 – 32 (1994).
12. Dey, A.K.: Understanding and using context. Personal Ubiquitous Computing, 51:4 – 7 (2001).
13. Barnard, L., Yi, J.S., Jacko, J.A., Sears, A.: Capturing the effects of context on human performance in mobile computing systems. Personal and Ubiquitous Computing, Volume 11, Issue 2, Pages: 81 – 96. Springer-Verlag (2006).
14. Langley, L.: Machine Learning for Adaptive User Interfaces. Proceedings of the 21st Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence. Pages: 53 – 62 (1997).
15. Norcio, A.F., Stanley, J.: Adaptive human- computer interfaces: A literature survey and perspective. IEEE Transactions on Systems, Man and Cybernetics, 19, No.2:399 – 408 (1989).
16. Vertegaal. R.: Designing Attentive Interfaces. ETRA'02. N.O., USA (2002).
17. Igarashi, T., Hughes, J.F.: A suggestive interface for 3D drawing**.** Proceedings of the 14th annual ACM symposium on User interface software and technology. Pages: 173 – 181. Orlando, Florida (2001).
18. Lesher, G.W., Moulton, B.J., Rinkus, G., Higginbotham, D.J.: A Universal Logging Format for Augmentative Communication. Enkidu Research, Inc. Lockport, NY (2000).
19. Gonçalves, M.A., Panchanathan, G., Ravindranathan, U., Krowne, A., Fox, E.A.: JCDL'03 Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries (2003).
20. Frias-Martinez, E., Chen, S.Y., Liu, X.: Survey of Data Mining Approaches to User Modeling for Adaptive Hypermedia. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, Volume 36, No. 6 (2006).
21. Alpaydin, E.: Introduction to Machine Learning. MIT Press (2004).
22. Garner, S.R.: Weka: The Waikato environment for knowledge analysis. In Proc New Zealand Computer Science Research Students Conference, pages 57-64, University of Waikato. Hamilton, New Zealand (1995).
23. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers (1993).
24. Manning, C.D., Raghavan, P., Schütze, H. An Introduction to Information Retrieval. Cambridge University Press (2008).
25. Van Dam, A. Post-WIMP User Interfaces, Communications of the ACM, Vol. 40, No. 2, pp. 63-67 (1997).
26. Rabiner, Lawrence R. (February 1989). A tutorial on Hidden Markov Models and selected applications in speech recognition". Proceedings of the IEEE **77** (2): pp. 257–286 (1989).