Transformable Menu Component for Mobile Device Applications: Working with both Adaptive and Adaptable User Interfaces

V. Glavinic[1], S. Ljubic[2] and M. Kukec[3]
[1] University of Zagreb, Zagreb, Croatia
[2] University of Rijeka, Rijeka, Croatia
[3] College of Applied Sciences, Varazdin, Croatia

***Abstract*—Using a learning system in a mobile environment is not effective if barriers are not overcome in the interaction with targeted users. For that purpose all mobile services, including m-learning ones, demand special attention being paid to interaction with the user. While mobile device applications are becoming more powerful, their development process must utilize the concepts of universal access and universal usability. This paper describes the model of both adaptable and adaptive mobile user interface, through the introduction of a transformable menu component capable to be personalized to each individual user with respect to her/his preferences and interaction style. We discuss the use of customization and adaptation techniques, with the aim to both enhance mobile HCI and to increase user satisfaction, particularly when working with graphically rich m-learning applications.**

***Index Terms*—Adaptive user interfaces, adaptable user interfaces, personalization, mobile human-computer interaction (mobile HCI), mobile learning, universal usability, universal access**

I. INTRODUCTION

Contemporary software systems are recommended to focus on three universal usability challenges: technology variety, user diversity, and gaps in user knowledge. While technology variety requires supporting a broad range of hardware, software, and network access methods, user diversity involves accommodating users with different skills, knowledge, age, gender, disabilities, and so forth [1]. Regarding mobile services, usability must look at the mobile user and surmise what interfaces are appreciated and anticipated by the user [2].

Fast developing mobile computing devices (mobile phones, pocket PCs, Tablet PCs, and personal digital assistants - PDAs) and mobile technology altogether provide end users with new powerful mobile device applications (MDAs). Moreover, improved capabilities of mobile devices allow these applications to run graphical user interfaces with increased complexity. Popup menus, toolbars, icons and dialog boxes are becoming common graphical user interface (GUI) elements within MDAs. The addition of such new features can be advantageous to the mobile device user (regarding enhancements in interaction speed and efficiency), but on the other hand, sophisticated interfaces can foster user diversity. In fact, most users only use a small fraction of the available functions while wading through many unused ones, and different users tend to use different functions even when they are performing similar tasks [3]. Just like in desktop application environment, the MDA development process should emphasize both the user interface and the interaction aspects, the overall goal being to personalize the MDA interface to each individual user and provide easy access to the functions that she/he would use, thus addressing the concepts of universal access [4].

User interface personalization can be implemented through adaptable and adaptive mechanisms. Generally, the difference between adaptable and adaptive systems is presented in [5]: an *adaptable* system is one in which the system performance automation resides in the hands of the user; while conversely in an *adaptive* one the flexibility in automation behavior is controlled by the system. According to the aforementioned, adaptable user interfaces have user-driven customization ability, while adaptive user interfaces have a built-in self-adaptation one. Authors in [6] denote customization and self-adaptation as the only scalable approaches to personalization.

Although adaptation paradigms are device-independent, little work has been done in the area of MDA user interface personalization, as opposed to desktop PC applications. In this respect we discuss in this paper the possibility of implementing a both adaptive and adaptable GUI in MDAs, where personalization is primarily presented and visualized through the model of transformable and movable menu component. We believe that using both interaction techniques could provide optimal time division between working on related tasks and organizing the application interface. Since communication between the user and the system interface will have a direct influence on the user experience of the service provided [7], by so doing we expect bringing user satisfaction at a higher level.

The paper is structured as follows: Section II shortly explains the motivation to develop adaptable and adaptive menu component, Section III describes the personalization capabilities regarding to customization and self-adaptation of the proposed menu model, Section IV shows some layout snapshots of initial menu implementations within different development environments and respective mobile device emulators, while Section V presents a brief conclusion, together with the outline of our future work plan.

II. Motivation

The idea to build a "smart" menu component for an MDA comes as the continuation of our previous research on virtual mobile laboratories [8]. In this work, we have already introduced the model of a system that would enable distance-based laboratory training using wireless interconnected mobile devices (m-Lab). Such a system, which offers much more interactive learning possibilities than commonly used ones (which are for content retrieval only), could completely utilize all the improved capabilities of new mobile devices (e.g. display screens, processor power and memory capacity), by introducing applications with rich GUI elements (Fig. 1). Within the complex process of mobile virtual laboratory implementation, we are now primarily focused on the development of appropriate MDAs, where we expect to fully show the advantages of mobile HCI. In that respect, our motivation objective is to create a user interface model with the ability to (i) implement and run within different graphical MDAs, (ii) provide easy access to numerous application options, and (iii) personalize with respect to each individual user, according to her/his preferences and interaction style, as well as her/his knowledge level.

In present MDAs, finding and selecting desired application option could often be a difficult and time consuming problem. The main reason for this lies in the fact that the user is forced to continuously navigate through built-in multilevel menus, while at the same time she/he is not provided with the information on the menu level position. In such configurations, separate menu levels are usually displayed on the full screen, while the main application content is temporarily hidden. Additionally, this kind of "abrupt interaction" can often result with distraction in user's concentration within the given task.

To improve mobile user interaction with menu navigation, a new model of menu component for graphical MDAs is required. The proposed transformable menu component would have the same form as PC desktop applications menus, as well as suitable size and shape with respect to limitations of mobile device displays. The above mentioned motivation goals (i.e. HCI issues) would be encompassed by using both adaptable and adaptive procedures, thus making the menu component both personalized and easy to use.

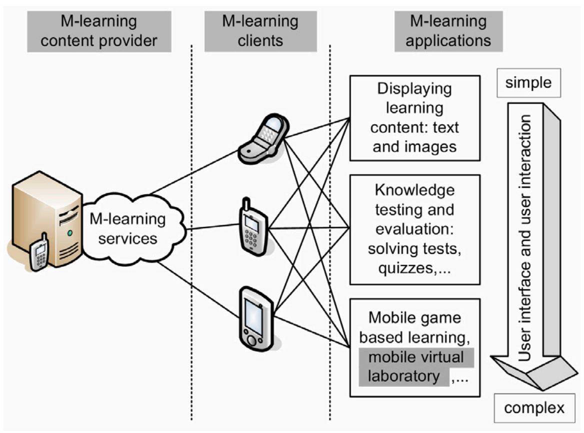
M-learning content provider
M-learning clients
M-learning applications
M-learning services
Displaying learning content: text and images
Knowledge testing and evaluation: solving tests, quizzes,...
Mobile game based learning, mobile virtual laboratory ,...
simple
User interface and user interaction
complex

Figure 1. Improved capabilities of new mobile devices combined with new m-learning services introduce MDAs with complex GUI

III. The Concept

Even though mobile devices are nowadays experiencing rapid improvement, their displays are still not at a level where MDAs could compete with some standard desktop applications. Relatively small screens, being the main restrictive factor, implicate application design with optimal usage of viewing area. In such an environment it must not be the goal to present all available information, but rather some smaller set of substantial data.

A. *Adaptable Menu Component: Customization*

When working with graphical applications (such as those supported in a mobile virtual laboratory), every part of the display area could be significant and/or contain some useful information. Since menu components also occupy a portion of visible space, the user must have customization control over the menu visibility property. In this case, when strong attention must be directed to the application's graphical context (e.g. learning schematics or analyzing graphs), the user can hide the temporarily not required navigation system without trouble (Fig. 2).

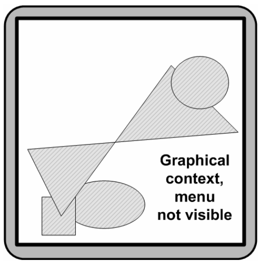

Figure 2. Mobile device screen layout. Running state of MDA is assumed, showing some graphical context, while menu component is hidden

Additionally, menu docking position can also be a considerable issue when interacting with an MDA. Usually a menu component is located at the top of the application user interface. If the predominant part of the essential content is also located at the top of the mobile device display, menu navigation can be less efficient, due to focus elimination and possible repetitive menu actions. This can be avoided by offering the user more customization control, providing her/him the ability to easily change the menu docking position and menu orientation among four supported locations/orientations: horizontal upper, horizontal bottom, vertical left, and vertical right (Fig. 3).

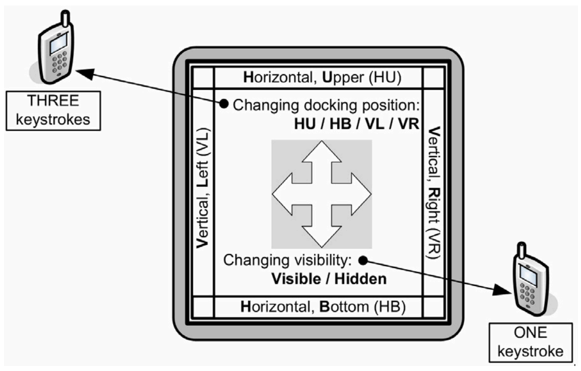

Figure 3. Changing menu visibility and its docking position/orientation must be accessible with just few keystrokes

From the user standpoint adaptable user interface procedures must be designed as simple as possible. The time spent for UI customization must represent the cost of a marginal set of actions within the overall working time in an MDA. If this is not fulfilled and the customization process appears to be time consuming, then either its purpose is rather questionable or the respective adaptable technique is poorly designed. Taking this into consideration, the control of the visibility property in the proposed menu component is enabled through exactly one keystroke. The respective mobile device key is reserved for changing the menu state from visible to hidden and vice-versa. Additionally, changing the docking position is enabled by a three keystrokes sequence: (i) pressing the reserved button to enter the moveable state, (ii) pressing the navigation key (up/down/left/right) to dock the menu to the desired location, and (iii) pressing the same reserved button, this time to leave the movable state.

Examples of different menu docking positions in the active menu state are presented by two sketches in Fig. 4. Numbers from 1 to 9 represent menu headers, which can be text-based or icon-based at runtime. Clearly, icons or abbreviations of header full names represent more suitable options for vertical menu orientation. Therefore, the best approach is to always define shorter header names. Upon selecting the menu header, a set of popup items becomes visible on the screen. As our intention is to prevent hiding all the underlying graphical content, the upper limit of visible popup items must be defined. This has to be a device-dependent factor, depending on the mobile device screen size, which is computed at runtime (a small integer is assumed, being 5 in the figures below). The remaining popup items become hidden but easily accessible via simple navigation. Since the menu header can contain either representative icons or text abbreviations, it is useful to provide full-text information about selected option, thus enabling the user to quickly learn icon or abbreviation meaning.

It is highly probable that different users will use different menu interaction patterns, even when working on very similar tasks. In that respect, a set of frequently used menu items will vary from one person to another. User interaction diversity combined with the fact that navigation to hidden items can be very unattractive (especially in cyclic actions) is the reason for menu component personalization. Rearranging menu items within a popup set could be a very useful customization potential, especially for experienced users working with a familiar set of tasks. It is highly probable that the most frequently used items will be manually replaced to the beginning of the popup set, while non-used items will be moved to the hidden subset. Such a menu component personalization procedure as described above must be presented to the user as a trouble-free process, designed for simple usage. The best way to accomplish these requirements is to take advantage of the existing navigation model for items positioning. A detailed description of the customization process sequence is given in Fig. 5. The user can change item positions using familiar navigation patterns, while the customization mark

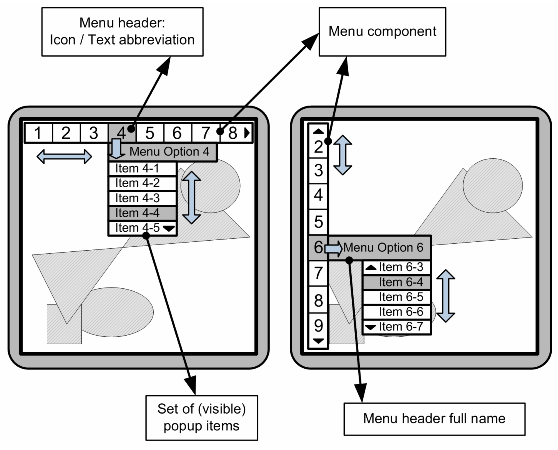

Figure 4. Different menu docking positions and appropriate navigation routes in the active state

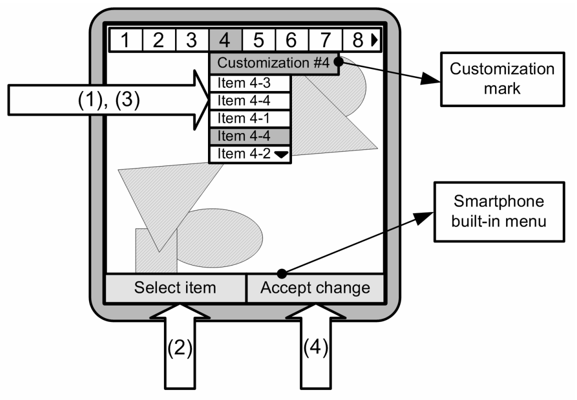

Figure 5. Customization process sequence: (1) Using navigation keys to find required menu item, (2) Using built-in menu option for selecting required item, (3) Using navigation keys to move selected menu item to desired position, (4) Using built-in menu option for accepting changes

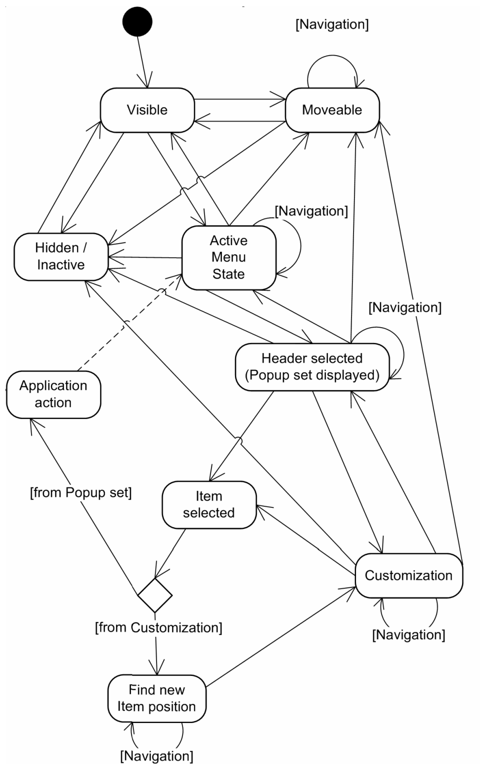

Figure 6. UML state diagram of proposed menu component. All label-free transition arrows represent exactly one keystroke action

(at the place where the header full name usually stands) is used to notify her/him that the customization mode is temporarily active. Combining the mobile device built-in menu, which is usually presented with only two options enabled, with our menu component, could be a supplementary advantage in the interaction process, as shown in Fig. 5.

Following the concepts of universal access and universal usability, personalized menu components must provide easy access to all of the existing menu functions. Within MDA, this is primarily done by simplifying menu state transitions. In our case, all state transitions are available through exactly one keystroke on the mobile device input (see Fig. 6), thus providing a platform for optimal interaction efficiency.

B. *Adaptive Menu Component: System-driven Personalization*

When speaking of MDA user interface automatic adaptation, we must distinguish (i) adaptation to the used terminal characteristics and (ii) adaptation to the user preferences, interaction style and knowledge level. The former feature employs available technologies to enable the development of applications with a unified user interface for different types of handheld devices, most often by using XML-based languages for UI description and specification ([9], [10]). We are however focusing to the latter feature by analyzing the possibilities for implementing automated personalization of the menu with respect to user interaction patterns.

While the user interacts with the MDA, her/his menu actions (navigation and selections) must be actively monitored. This involves installing proper adaptation algorithms which should operate “in the background” and, based on data thus collected, change the momentary menu configuration (Fig. 7). The most suitable and well-known adaptation algorithms are based on the user’s most frequently and recently used items ([3], [11]). While recency-based algorithms are used to dynamically determine and promote the most recently used items to preferable menu position, frequency-based ones are used to promote the most frequently used ones. Because of the user diversity, adaptation algorithms can generate completely different menu configurations for different people.

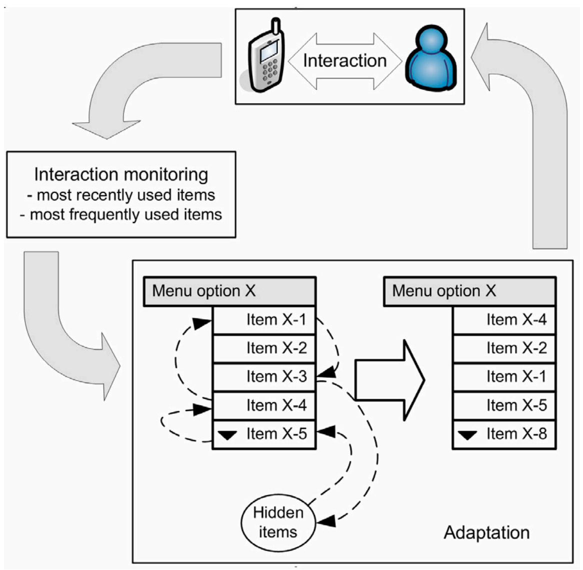

Figure 7. Basic principle for automated menu personalization, using recency-based and/or frequency-based adaptation algorithms

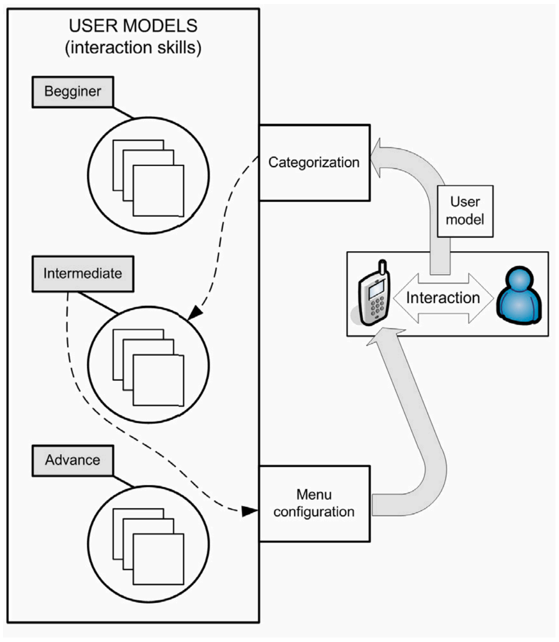

Figure 8. User models in automated menu personalization

The use of the aforesaid background algorithms allows modeling a user behavior profile, which describes the preferred way of user interaction. This profile can also be used in anticipating future user objectives in specific tasks of her/his work process. Such objective anticipation can extend menu usefulness by introducing completely new menu items for multiple action shortcuts. Along with behavior profile modeling, it is also possible to construct user models containing (and updating) information which would help the system to determine the interaction skill category (e.g. beginner, intermediate, advanced), see Fig. 8. Based on this information it would be possible for the system to automatically and quickly adjust the menu configuration. In such a model, the *sparse data* problem (which is related to the *cold start* problem of personalization) can emerge, in situations where there is insufficient information to build user model and categorize users [12]. Using adaptable procedures, i.e. customization, can be used to avoid that kind of problem, since constructing the user model is then explicitly done by performing customization actions. This is yet another benefit of using both adaptable and adaptive techniques (customization and adaptation altogether). User models, initially created by users themselves, can then be altered again, through profile learning algorithms [13].

IV. INITIAL IMPLEMETATIONS

In this section we briefly describe some initial implementations of the proposed menu component within different developing environments and mobile device emulators.

Fig. 9 shows the layout of an MDA with a desktop-like menu component being implemented. It can be seen that the menu contains text-based headers (which are for the moment being represented by characters and not icons).

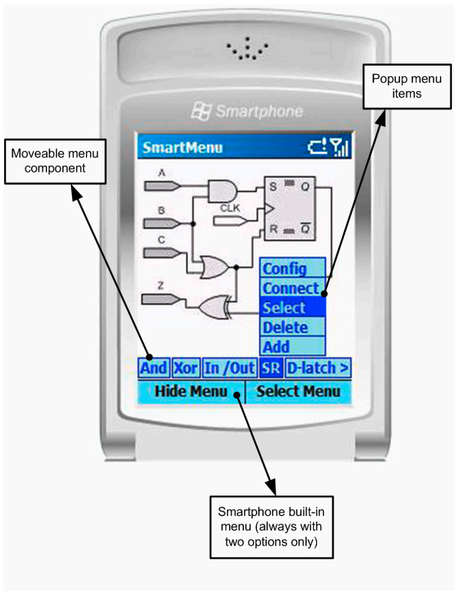

Figure 9. MDA layout with active text-based moveable menu component. Smartphone emulator display size is 176 pixels in

The application is running on the *Smartphone 2003 SE Emulator* (*Microsoft Device Emulator*, a part of *Visual Studio 2005* packet). The smartphone emulator built-in menu options are used to switch the menu visibility state, and to trigger menu navigation (active menu state). The menu component is applicable even on such limited displays (176 pixels in width, and 180 pixels in height). In comparison, many of present smartphones have screen resolutions of 240x320 pixels, while PDA devices have displays with sizes up to 480 pixels in width and 640 pixels in height.

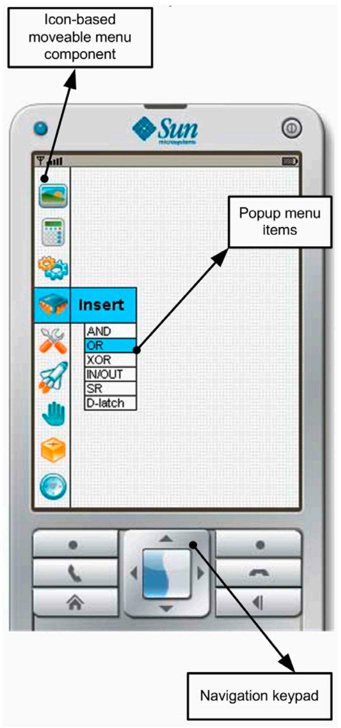

Figure 10. MDA layout with active icon-based moveable menu component. Smartphone emulator display size is 240 pixels in width, and 320 pixels in height

Conversely, Fig. 10 shows the menu component within an MDA which is running on the *Sun Java Wireless Toolkit 2.5* emulator. This example shows the advantage of icon usage for menu headers when running MDAs with vertical menu orientation.

V. CONCLUSION AND FUTURE WORK

M-learning systems will certainly become an additional advantage in the comprehensive process of lifetime learning. Contemporary development of mobile technologies and the associated information-communication infrastructure do make possible the introduction of new and high quality m-learning services and interactive learning contents. At the same time, there already exists a powerful platform for bringing the respective mobile device applications at a higher level. Although present m-learning applications don't take full advantage of this platform (mobile games, satellite navigation, and media streaming are still the only representative applications which completely utilize mobile technology upgrowth), we can suppose that they will soon become very powerful and graphically rich.

When developing these applications special emphasis must be given on the user interface design and the quality of user interaction. The concepts of universal usability, universal access and user experience represent state-of-the-art design guidelines for present day mobile device applications. As opposed to desktop applications, little work is already done for bringing personalized user interfaces to mobile device displays.

This paper proposes a model of such an MDA user interface, where personalization is presented and visualized through a transformable and movable menu component with adaptable and adaptive features. We claim that the usage of these techniques altogether can be a representative model for efficiency enhancement in mobile HCI, and additionally that MDAs integrating the proposed menu component can definitely be considered a significant benefit in the field of m-learning. Developing a proposed full scale menu component (by using the advantages of the object-oriented paradigm) will result with both API extension and component reuse possibility in all likewise applications.

We are continuing our work on the implementation of the presented model by primarily focusing on the development of menu adaptive algorithms. The menu based MDA prototype will be tested in various working environments among the student population. We intend to carry out both usability testing and efficiency measurement (with respect to interaction speed) in controlled experiments, initially run with mobile device emulators. We believe that the obtained results will confirm our expectations, in particular about increasing user satisfaction.

REFERENCES

[1] B. Shneiderman, “Universal Usability: Pushing Human-Computer Interaction Research to Empower Every Citizen,” *ACM Communications*, vol. 43, pp. 85-91, May 2000.

[2] S. Greene and J. Finnegan, “Usability of Mobile Devices and Intelligently Adapting to a User's Needs,” in Proceedings of the 1st International Symposium on Information and Communication Technologies (ISICT'03), *ACM International Conference Proceeding Series*, vol.49, pp.175-180, 2003.

[3] L. Findlater L and J. McGrenere, “A Comparison of Static, Adaptive, and Adaptable Menus,” in Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'04), 2004, pp. 89-96.

[4] C. Stephanidis, “Editorial,” *Int'l J. Universal Access in the Information Society*, vol.1, pp.1-3, June 2001.

[5] R. Oppermann, Ed., *Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software*. Mahwah, NJ, USA: Lawrence Erlbaum Associates, Inc., 1994.

[6] D. S. Weld, C. Anderson, P. Domingos, et al., “Automatically Personalizing User Interfaces,” in Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03), Morgan Kaufmann, 2003, pp. 1613-1619.

[7] B. von Niman, A. Rodriguez-Ascaso, S. Brown, and T. Sund, “User Experience Design Guidelines for Telecare (E-Health) Services,” *Interactions*, vol.14, pp.36-40, October 2007.

[8] V. Glavinic, M. Kukec, and S. Ljubic, “Mobile Virtual Laboratory: Learning Digital Design,” in Proceedings of the 29th International Conference on Information Technology Interfaces (ITI'07), 2007, pp.325-332.

[9] W. Mueller; R. Schaefer, and S. Bleul, “Interactive Multimodal User Interfaces for Mobile Devices,” in Proceedings of the 29th Annual Hawaii International Conference on System Sciences (HICSS'04), 2004, pp.10.

[10] M. Bisignano, G. Di Modica, and O.Tomarchio, “Dynamic User Interface Adaptation for Mobile Computing Devices,” in Proceedings of the 2005 Symposium on Applications and the Internet Workshops (SAINT-W’05), 2005, pp.158-161.

[11] K. Z. Gajos, M. Czerwinski, D. S. Tan, and D. S. Weld, “Exploring the Design Space for Adaptive Graphical User Interfaces,” in Proceedings of the 8th International Working Conference on Advanced Visual Interfaces (AVI'06), 2006, pp. 201-208.

[12] D. Albrecht and I. Zukerman, “Introduction to the special issue on statistical and probabilistic methods for user modeling,” *User Modeling and User-Adapted Interaction*, vol. 17, pp. 1-4, March 2007.

[13] B. Mrohs and S. Steglich, “Architectures of Future Services and Applications for the Mobile User,” in Proceedings of the 2005 Symposium on Applications and the Internet Workshops (SAINT-W’05), 2005, pp. 128-131.

AUTHORS

V. Glavinic is with the Department of Electronics, Microelectronics, Computer and Intelligent Systems (ZEMRIS) at the Faculty of Electrical Engineering and Computing (FER), University of Zagreb, Unska 3, HR-10000 Zagreb, Croatia (e-mail: vlado.glavinic@fer.hr).

S. Ljubic is with the Department of Automation, Electronics and Computing (ZAER) at the Faculty of Engineering (RITEH), University of Rijeka, Vukovarska 58, HR-51000 Rijeka, Croatia (e-mail: sandi.ljubic@riteh.hr).

M. Kukec is with the College of Applied Sciences, Hallerova aleja 5, HR-42000 Varazdin, Croatia (e-mail: mihael.kukec@velv.hr).

Manuscript received 23 April 2008. This paper describes the results of research being carried out within the project 036-0361994-1995 *Universal Middleware Platform for e-Learning Systems*, as well as within the programme *Intelligent Support for Ubiquity of e-Learning Systems*, both funded by the Ministry of Science, Education and Sports of the Republic of Croatia.

Published as submitted by the authors.