Univ Access Inf Soc (2006) 5: 51–72
DOI 10.1007/s10209-006-0024-1

LONG PAPER

Anthony Savidis · Dimitris Grammenos
Constantine Stephanidis

Developing inclusive e-learning systems

Published online: 20 April 2006

Abstract The requirement for e-inclusion relates to the employment of appropriate development strategies for software applications and services so as to accomplish accessibility and increased interaction quality at deployment time. Inclusive e-learning is the outcome from the application of e-inclusion design and implementation methods in the context of e-learning systems. This paper reports consolidated development experience from the construction of the following e-learning systems: (a) training applications for hand-motor impaired users and for people with cognitive disabilities; (b) learner-adapted courseware and (c) a universally accessible educational computer game. In this context, the primary emphasis is placed on the reporting of the design and implementation aspects to accommodate the inclusive system characteristics, rather than on the typical e-learning software engineering approaches.

Keywords Inclusive e-learning · Accessible training applications · Learner-adapted courseware · Universally accessible computer games

1 Introduction

The development of e-learning systems is targeted towards supporting the overall learning process with software instruments enabling (a) learners to easily and effectively assimilate the learning material; (b) tutors to carry out more productive and effective learning processes and (c) supervisors to organize, execute, monitor and evaluate the on-line learning process. In this context, e-learning systems are basically "information systems", but not always structured and developed following the particular software engineering methods of the information-systems field. For instance, edutainment software employs multimedia interaction methods to deliver cartoon-like, mostly drill-and-practice sessions in a "learning by playing" fashion. In such systems, the learning content is mostly fused with the multimedia presentation logic, usually not being modelled and stored as structured content in a database. Additionally, training systems targeted at assigning trainees very close to real working tasks and activities place primary emphasis on the implementation of "learning by doing" methods. However, there are situations where the delivery of traditional electronic courseware is still needed, as a means of supporting distributed on-line courses, in processes that could be characterized as mainly involving "learning by studying" interactions. In the latter case, the learner is usually faced with interactive material that has to be explicitly studied, whereas the e-learning system emphasizes content delivery rather than the delivery of a new teaching style.

While e-learning systems exist for numerous thematic topics, employing teaching approaches such as those mentioned above, there is relatively limited reported experience in applying "design for all" and "universal access" methods in the context of e-learning system development. Recently, in the context of "user interfaces for all" [17], the need to produce software applications and services accessible and optimally usable by potentially everyone has been emphasized. In some development cases it is practically unavoidable to design and deliver customized system versions for specific target user groups, while in other cases the pursuit of systems capable to adapt to individual end-users is proved to be a more cost-effective goal. The "user interfaces for all" objective represents a tangible technical objective for

A. Savidis (✉) · D. Grammenos · C. Stephanidis
Institute of Computer Science, Foundation for Research and Technology – Hellas (FORTH), 70013 Heraklion, Crete, Greece
E-mail: as@ics.forth.gr
Tel.: +30-2810-391741
Fax: +30-2810-391740

C. Stephanidis
Department of Computer Science, University of Crete, Crete, Greece

e-inclusion, where inclusive e-learning normally designates the specific instantiation of the e-inclusion target in the e-learning domain. This paper reports the design, implementation, evaluation and deployment experience regarding specific inclusive e-learning systems. Each of the reported systems addresses a different learning domain and target user group. In particular:

- *Vocational training applications for people with disabilities*. In this context, two applications are presented, adopting a didactic style which borrows elements from both "learning by doing" and "learning by studying" approaches:
  - An accessible "canteen manager" application, training hand-motor and people with cognitive disabilities for the cashier management of a typical "canteen". In this case, the training application was also the real-life application system, where the training process was carried out with a supervisor either present in the field or indirectly monitoring through a camera.
  - A multimedia "sew tutorial" application, training people with cognitive disabilities in typical sewing tasks. In this case, the trainees are put in front of the real sewing devices, while the training process is based on two strategies: (a) giving each student a personal computer, requiring that they manually operate the multimedia tutorial application; and (b) putting a single large projected display and loud speakers, as a coordinated training session guided and operated by a tutor-supervisor.
- *Software engineering method for learner-adapted courseware*. This method is the outcome of recent work targeted at providing an appropriate specialization of the unified user interface development paradigm [14], the latter addressing the construction of interfaces automatically adapted to individual uses, to courseware that could be adapted on-the-fly to individual students. In this context, the architectural structure of a web-based learner-adapted digital library for e-learning has been produced. The presented approach has been mainly designed for e-learning systems emphasizing both "learning by studying" and "learning by doing" (i.e. interactive exercises) teaching methods.
- *Universally accessible games*. A web-based (multiplatform) chess-game accessible by able-bodied, blind and hand-motor impaired users has been developed. Until recently, the vast majority of developments towards universal access had practically neglected the entertainment application domain. The critical importance of games for disabled people has been acknowledged through the recent establishment of the special interest group on Game Accessibility of the International Game Developers Association (see [7]). In particular, games such as chess, which primarily challenge and sharpen thinking and intellectual skills, are also considered to fall in the e-learning domain, reflecting a "learning by playing" didactic methodology.

1.1 About the applications

While all the applications presented in this paper fall in the general domain of inclusive e-learning, each specific application concerns a different sub-domain, practically requiring distinctive design, implementation and evaluation perspectives. As depicted in Fig. 1, there is no evident direct overlapping among the four basic application case studies, whereas their sub-domains are very much interrelated. For example, while the two basic vocational training applications, i.e. the sew trainer and the canteen manager, primarily emphasize accessibility-oriented features, it is expected that vocational training applications may largely benefit from the employment of learner-adapted courseware delivery techniques. Similarly, while the universally accessible chess game does not incorporate structured stored content, and also does not support vocational training demands, arguably it provides added value to construct accessible games to accommodate vocational training requirements; further, such games may internally encapsulate structured content with learner-oriented automatic delivery. Please note that not all the various e-learning sub-domains are listed within Fig. 1.

1.2 Key contributions

The overall contribution of this paper concerns the reporting of consolidated experience in the course of real-life developments for a range of related applications and methods within the general domain of inclusive e-learning systems. The key contributions reported in this paper are summarized as follows:

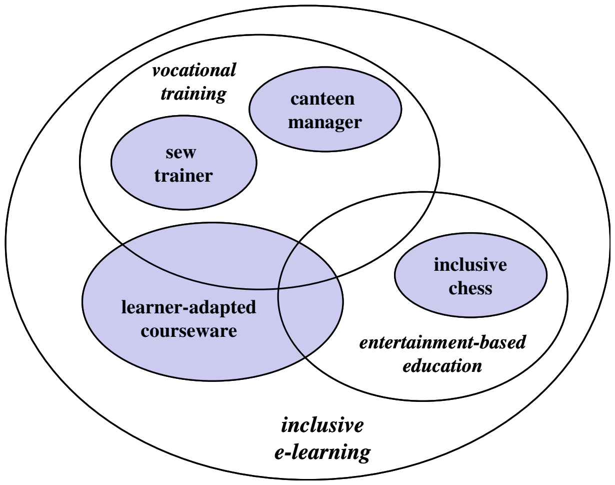


Fig. 1 Relationships among the applications to be discussed, their particular sub-domain and the overall inclusive e-learning field

- The most prominent design and implementation challenges in developing real-life inclusive vocational training applications, placing particular emphasis on the embedding of hierarchical scanning techniques for users with severe impairments of the upper limbs; today, there is very limited reported experience for genuinely accessible (i.e. not relying on third-party assistive technologies) vocational training applications.
- The design and deployment experience in constructing vocational tutoring applications for people with cognitive disabilities, placing emphasis on group-oriented computer-augmented tutorial sessions.
- The software implementation architecture for learner-adapted courseware, by presenting the necessary elements for augmenting existing learning systems with appropriate meta-data categories and intelligent processing kernels to accommodate learner-adapted behaviour.
- The design and implementation of an inclusive web-based chess game, one of the finalists of the European 2004 "Design for All" awards, that is accessible by able-bodied, motor-impaired and blind users.

2 Related work

The e-learning domain encompasses a wide range of software systems, with a variety of design methodologies, interaction features, didactic styles, content organization approaches and target audience. The technical focus of the reported work is on accessibility, individualization and entertainment, while application-oriented emphasis is on vocational training, course teaching and learning-oriented strategy games.

2.1 e-Learning and games

Play is a fundamental activity of children that allows them to acquire the foundations of self-reflection and abstract thinking, develop complex communication and meta-communication skills, learn to manage their emotions and explore the roles and rules of functioning in adult society [20]. Although there have been several debates on the effects of computer games on children, it is indisputable that they can allow for experimentation and promote active learning by shifting players into the participant role [1], being in line with modern active, learner-centred learning approaches (e.g. constructivism) in contrast to the more formal, teaching-based paradigms followed in the past. Strategy games (such as chess) can serve both active learners, who usually understand information actively working with it, and reflective learners, who prefer to think before acting. Additionally, computer games are not only for educating the young. Although learning through play is mostly associated with children, there are several examples of computer games used for adult education. For example, the military field has traditionally used simulations as a training medium. In the past few years, corporate learning (e.g. www.games2train.com, learningware.com) has shown an increasing interest in employing computer games to substitute or enhance learning techniques that employees find boring and are unwilling to follow, and even university departments have started introducing computer games in their curricula to support alternative learning styles, attract student interest and help reinforcing learning objectives [3, 5].

2.2 Games and accessibility

Usually, computer games are quite demanding in terms of motor and sensor abilities needed for interaction control, a fact that renders them virtually inaccessible to people with disabilities, and in particular to blind people and to those with severe motor impairments of the upper limbs. From a technical point of view, two main approaches have been adopted in order to address the issue of computer games accessibility in general:

- Games are developed to be compatible with the use of assistive technologies, such as screen readers, mouse emulators or virtual keyboards (a solution that can practically work only with non-action games, which do not rely upon fast reflexes and user reactions).
- Special-purpose games are developed which are optimally designed for people with disabilities, like audio-based games for the blind, switch-based games for the motor-impaired, etc.

The first approach typically suffers from low interaction quality, and achieves limited accessibility. The second approach, though being the most promising from the quality point of view, has two key drawbacks: (a) the cost of developing high quality games is prohibitive when the potential target group is limited; and (b) there is an evident hazard of segregation between able and disabled game users, leading to potential social exclusion.

2.3 e-Learning and accessibility

Currently, there are a number of relevant important standardization efforts for e-learning content. Most of those concern architecture, learner modelling and learning objects' modelling. Among the most promising efforts are the work of the IEEE Learning Technologies Standards Committee (LTSC—[6]) and the IMS (Global Learning Consortium—[8]) Project, as well as the ISO–IEC JTC1 SC36 working group [9] on setting up standards for learning technologies. The IEEE LTSC working groups most relevant to the work reported in this paper are: (a) LTSA, Learning Technology Systems Architecture; (b) LOM, Learning Object Metadata and (c) PAPI, Public and Private Information for Learners.

Finally, the CEN/ISSS workshop [2] has the objective to encourage the effective development and use of relevant and appropriate standards for learning technologies for Europe. The Learning Technologies Workshop decided, as a matter of principle, not to duplicate work already being done elsewhere, but to ensure that diverse European requirements are properly addressed by global initiatives.

Until now, such standardization work has been mainly targeted towards producing standard models for the computational and storage aspects of e-learning systems, not putting emphasis on accessibility-related issues. Paradoxically, accessibility is technically considered to be only an interaction-related concern that could be addressed later through appropriate add-ons. However, content modelling is severely affected when considering end-user needs; for instance, audio captioning is required for deaf users, while image descriptions are needed for the blind. In this context, the work of the W3C/WAI (Web Accessibility Initiative) has been targeted at the production of web-design guidelines to ensure that web-pages would look more "friendly" to assistive technologies typically used by disabled people, such as screen readers or keyboard emulators. But even in this case, it is clear that the primary objective is just to push traditional applications a little forward, which are not developed to be genuinely accessible by design, so that they incorporate some features that enable third-party assistive software to be more effective.

In contrast to the lack of standardization work, there are a large number of commercially available applications targeted to special education, mainly addressing general-purpose instructional needs for specific categories of learning difficulties, such as reading, writing, cause and effect associations, basic life skills, hand-eye co-ordination, dyslexia etc. However, there is very little reported experience in inclusive vocational training for applications specifically designed for real-life work tasks. Moreover, as it has been previously mentioned, the tremendous potential from the "fusion" of entertainment computing, education software and inclusive interaction has been just recently acknowledged with the formation of the IGDA WG on accessible games.

3 Vocational training applications

3.1 "Canteen manager" for motor-impaired people

The canteen manager application has been developed with a twofold objective: (a) serving as a training tool; and (b) being the real canteen-cashier application. The target groups for this training application were: (a) users with severe disabilities in the upper limbs; (b) people with non-severe mental impairments; and (c) able-bodied users. Additionally, it has been necessary to support accessibility for people with both mental and motor impairments (i.e. combined disabilities).

In Fig. 2, the overall system infrastructure is shown, focusing on the special switch input devices employed for scanning interaction. The type of switches shown is only indicative, as the "switch adapter box" included in the system allows attaching various types of switches, typically connected through a serial port. The discussion is followed by a brief description of the key user interface design decisions, continuing with a presentation of specific prominent system development issues; then, the findings of the usability evaluation process are discussed.

3.1.1 Design objectives and requirements

The two prominent design requirements of the cashier training application were: (a) the need to support accessibility for people with severe disabilities of the upper limbs, and motor-impaired users in general; and (b) the necessity to serve both as the initial training instrument and as the target application to be eventually deployed in real-life, with the following set-up:

- The cashier application operator (the disabled user) interacts with the customer to receive the orders and then issues the receipt that is given to the customer.
- Another member of the staff (an able-bodied user) gets the printed receipt from the customer, together with the payment, deposits the cash and then supplies the corresponding products.

3.1.2 Interface design

Following the key design requirements, the training session for the deployment phase has been eventually organized in two different ways (see Fig. 3): (a) direct

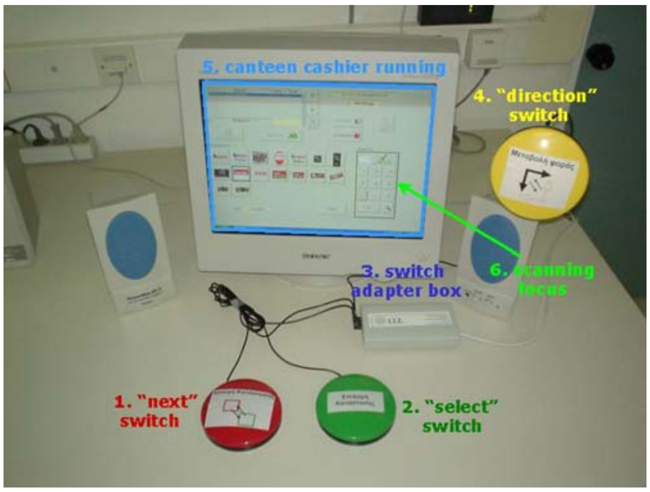


Fig. 2 A snapshot of the "Canteen Manager" cashier application running, illustrating the peripheral devices and their interaction-specific role; direct input is allowed through the use of a touch screen, while the keyboard is not used

supervised mode (initially, upon the early training sessions); and (b) indirect supervised mode (after the user has gained adequate experience of use). The user interface had to be easy to use, comprehensible and accessible. In this context, it was decided to minimise the use of verbal explanations (text) as much as possible, by providing iconic representations, in most cases exact photographs of the canteen products, together with digitised speech-audio.

This design style is different from the one deployed in common cashier applications used in restaurants, cafes or canteens, all of which largely employ textual descriptions. The initial screen of the cashier training application is provided in Fig. 4. The product categories are provided with iconic representations (not photographs), which have been carefully chosen among alternatives after an initial interview and evaluation round with end-users, scoring the appropriateness of each alternative icon regarding comprehensibility. After a specific product category is chosen, a menu consisting of the product photographs is displayed (see Fig. 5). The trainee may either select a product, in which case the focus moves automatically to the dialogue box for specifying the number of purchased items (this is the scenario shown in Fig. 5), or the trainee returns back to the product categories menu. When the number of items to be ordered is verified (i.e. the dialogue box of Fig. 5), the order is automatically added to the total (see upper left part of Fig. 6).

The design decision to emphasize the use of iconic menus through exact photographic representations of products has proved to be suitable for the specific type of training topic and target groups of trainees, as it allowed to quickly shift to unsupervised sessions (actually indirectly supervised as the sessions were always monitored through a camera). It has been observed that trainees were capable of easily handling the sessions without any external support, a fact that allowed them to quickly increase their knowledge regarding the cashier system, while also developing their confidence and sense of independence in being able to effectively accomplish the assigned task.

The design of the user interface to support accessibility by hand-motor impaired users employs switch-based hierarchical scanning techniques, and in particular the method originally reported in [15] for augmenting with scanning capabilities the basic windows object library. Using this technique, the graphical direct-manipulation dialogue is decomposed in a dialogue structure involving sequences of only two basic actions: *next* and *select*. Typically, each of those is associated with a particular binary switch, thus enabling the thorough use of a graphical application without the mouse and the keyboard, which are not accessible to motor-impaired users. To accomplish this goal, it is imperative to appropriately reproduce the dialogue for every type of user interface object into sub-dialogues of *next* and *select* actions. The way this has been achieved for the cashier application is shown in Fig. 7, providing the dialogue automata for the key graphical user interface objects (i.e. container objects, composite objects and iconic buttons).

During interaction, the particular interaction object owning the scanning dialogue focus is indicated with a green bounding rectangle (called the ‘‘scanning highlighter’’). As shown in Fig. 7, the dialogue with a button object is decomposed as follows:

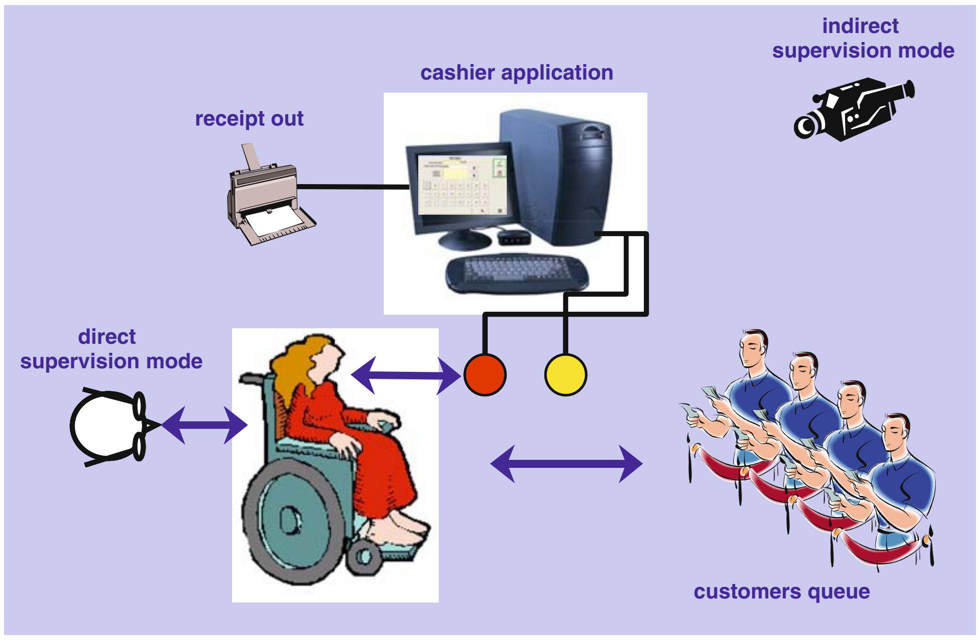


Fig. 3 The organized training sessions with the cashier application in supervised and non-supervised modes; multiple real-life scenarios for customer arrival pace were defined and applied

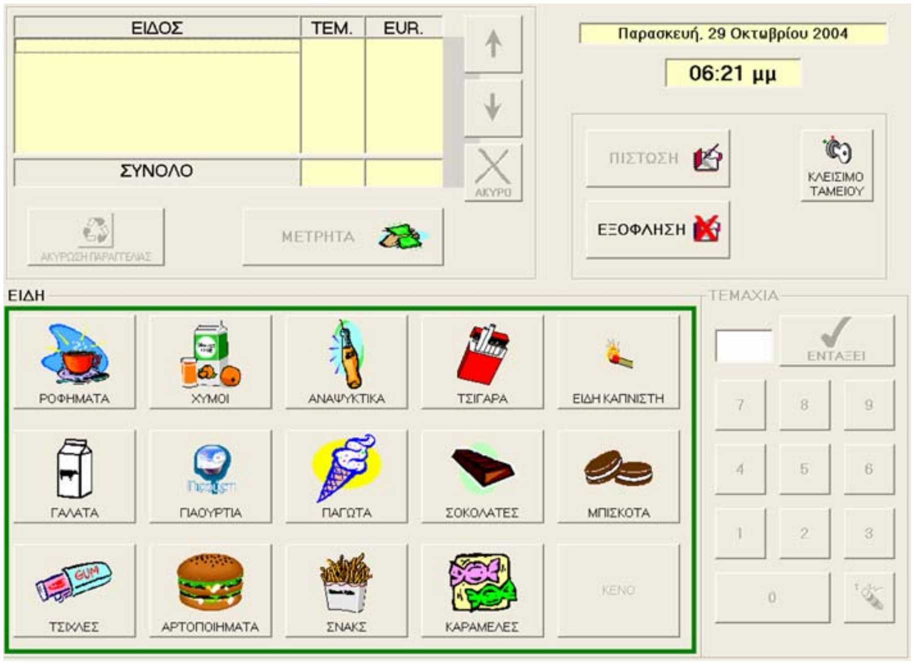


Fig. 4 Snapshot of the main screen of the cashier application running (in Greek). The *green rectangle* plays a twofold role: **a** shows the only group of options (product categories) that is selectable; and **b** it is used for hierarchical switch-based scanning of options

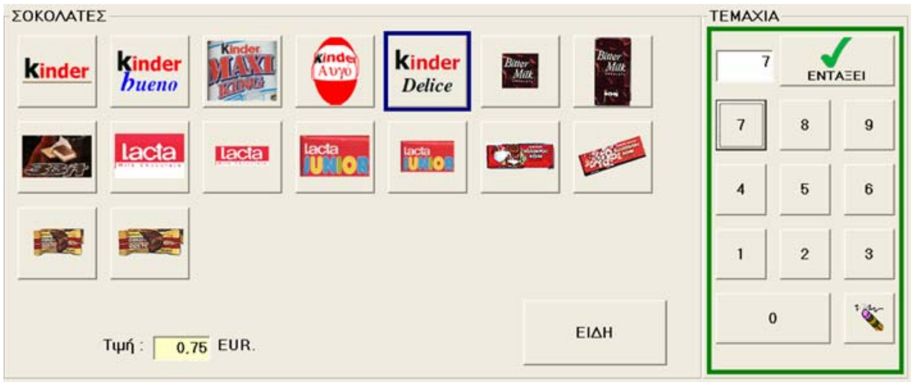


Fig. 5 Product selection menu (photographs), and dialogue box to provide the number of purchased items

- First, the scanning highlighter is drawn on the graphical content of the button object to indicate it has the focus.
- If the switch being associated with the *next* action is pressed, the scanning dialogue moves to the next interface object (at the same level of the object hierarchy), and the current button object effectively loses dialogue focus.
- If the switch being associated to the *select* action is pressed, the button is considered to be pressed, in which case all internal (to the button object) actions are performed, so that a "button press" notification is eventually posted to the owner application (i.e. the cashier).

In the case of container or composite objects, the dialogue is more complex to allow the user control if the dialogue with such an object is to be initiated (the "entry" mode, see Fig. 7, indicated with a green highlighter), or is to be skipped (including all contained/constituent objects), enabling to speed up the scanning dialogue (the "exit" mode, see Fig. 7 and left part of Fig. 8, indicated with a red highlighter).

3.1.3 System development

The cashier trainer application has been developed using Microsoft foundation classes (MFC), running under Windows. The scanning dialogue capabilities have been implemented as derivations of the original MFC, incorporating hierarchical scanning by implementing the dialogue automata shown in Fig. 7; this constituted an implementation of the interaction-object augmentation software pattern according to the technique discussed in [13]. The switches have been driven (only for input) at the level of the serial port interface. The menus of product categories and products, including the photographs, graphical illustrations and prices, were all defined in a configuration file (in Extensible Mark-up

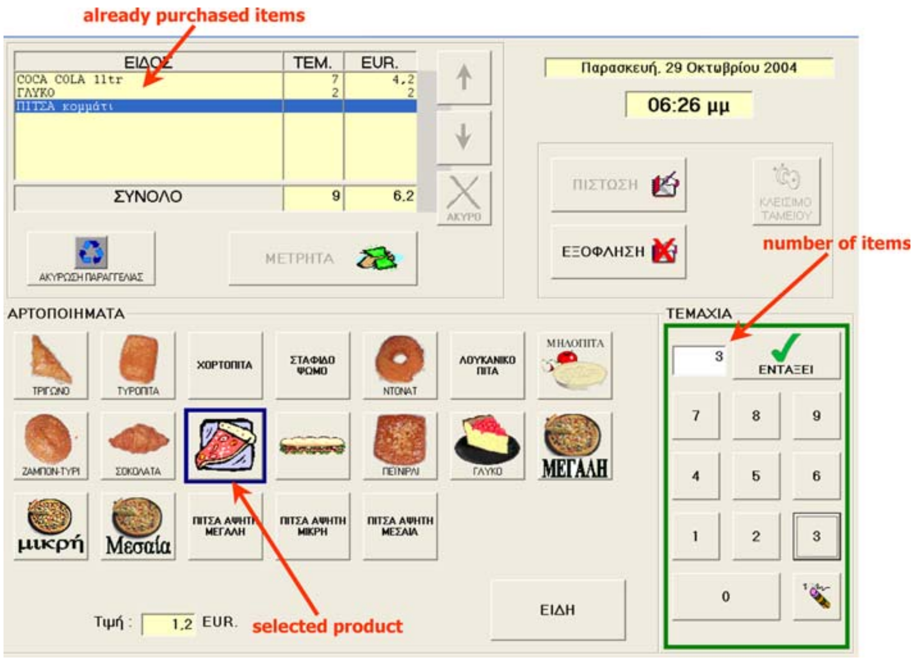


Fig. 6 The view of the cashier application once the trainer has entered two items and is in the process of entering customer's order for the number of the currently selected product (which is a slice of pizza)

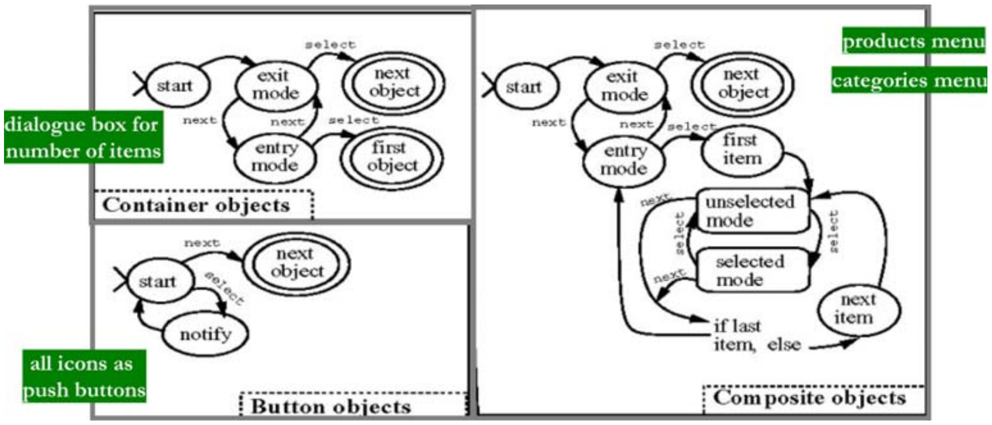


Fig. 7 The dialogue automata for the three types of interface objects in the cashier application, showing the dialogue decomposition into two fundamental input actions: next (first switch) and select (second switch)

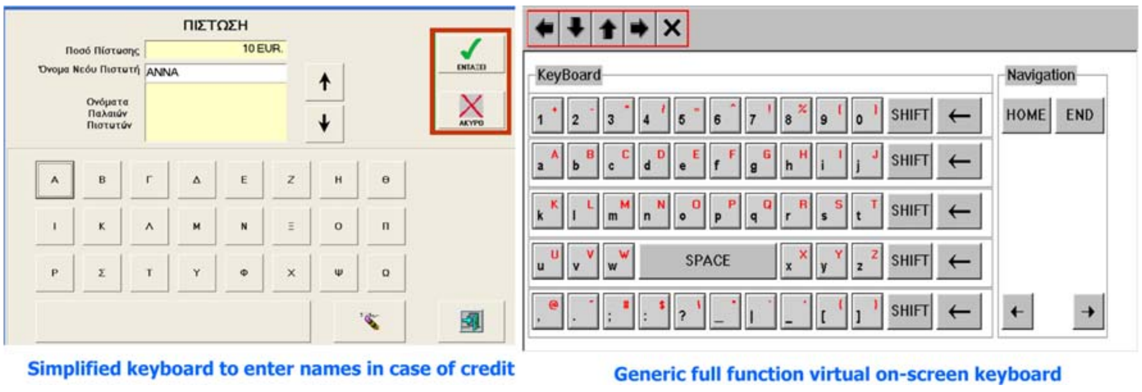


Fig. 8 The full-screen virtual keyboard supporting switch-based scanning for the cashier-training application (left) and a full-function on-screen keyboard that proved to be excessively detailed for the cashier application (from [18])

Language—XML). Initially, a "raw" configuration data in an INI file was used; subsequently, it was decided to employ more structured content descriptions with XML definitions (XML is largely employed for structured configuration data). Alternatively, all product-related information can be stored directly in a relational database.

Finally, it has been necessary to allow the focus object to be changed using direct pointing (the cashier application uses a touch screen) also when the scanning dialogue is used. This was necessary in all cases where direct intervention from an able person was required. For instance, during the training sessions in direct supervision mode, the supervisor could intervene and select the correct items from the screen using touch pointing. Clearly, since such an action effectively modifies the focus object of the scanning dialogue (i.e. where the scanning highlighter is), it had to be internally mapped to the augmented dialogue implementation, so that the graphically selected focus object would also become the scanning focus.

3.2 "Sew trainer" for people with cognitive disabilities

3.2.1 Design objectives and requirements

This application had a purpose similar to the cashier-trainer application, in the sense that on the one hand it is focused on a very specific job, while on the other the training application is intended for direct deployment during field trials. The only difference was that while the cashier trainer had a dual role, being both the training application and the real application in the field, the "sew trainer" played primarily the role of a multimedia tutor. However, the deployment approach for the sew trainer application for real-life training sessions also had to support different tutoring scenarios, as it was derived from the requirements of the particular target user group.

During the initial design–discussion sessions with experts in special education, coming mainly from the specific organization where the sew trainer would be eventually deployed, it came out that it was critical to support both (see also Fig. 9) (a) individualized and group training (i.e. each trainee or group of trainees sitting in front of a PC running the sew trainer application); and (b) tutoring classroom sessions, where the system is in use by a trainer, projected into a large screen, while all trainees sit in front of the actual sewing machines. The following sections briefly introduce the key aspects of the sew trainer application.

3.2.2 Interface design

The user interface design had to be particularly simple, delivering minimal functionality while emphasizing iconic illustrations and pictures. Initially, as the target user group consisted of language-literate adults, satisfactory reading capability was assumed. However, it was quickly understood that this was not necessarily the general case, as users had varying reading capabilities or deficiencies. Hence, it became necessary to introduce digitised speech of a relatively slow speech rate (it was proved quickly that the use of speech synthesis in the Greek language was far less than satisfactory).

Another small design detail concerned with the rendering of the selectable objects in the sew trainer user interface: the default behaviour of the software library that has been employed for interface implementation was to draw un-selectable icons or options in the so-called "grey style". Since this was not well understood by users with cognitive disabilities, as concluded during the initial evaluation sessions and interviews, the technique finally implemented was to simply draw selectable items (or groups of items) with green bounding rectangles and non-selectable items with red bounding rectangles (see Fig. 10).

Additionally, in the first prototype, audio feedback was included for all buttons, in case the mouse focus entered an option, e.g. audio messages for "stop sound", "exit" etc. However, if such a mouse focus event occurs while audio explanations are provided, the playback needs to be temporarily suspended to play the button title. This was more confusing than helpful, so in the final design the audio playback of button titles was dropped.

In Fig. 11, one of the main interface screens is shown, through which trainees may select the parts of the sewing machine for which they need explanation/training

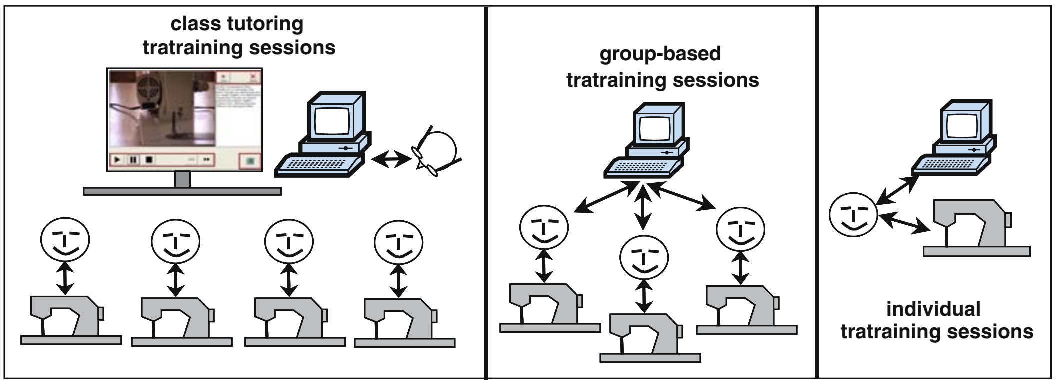
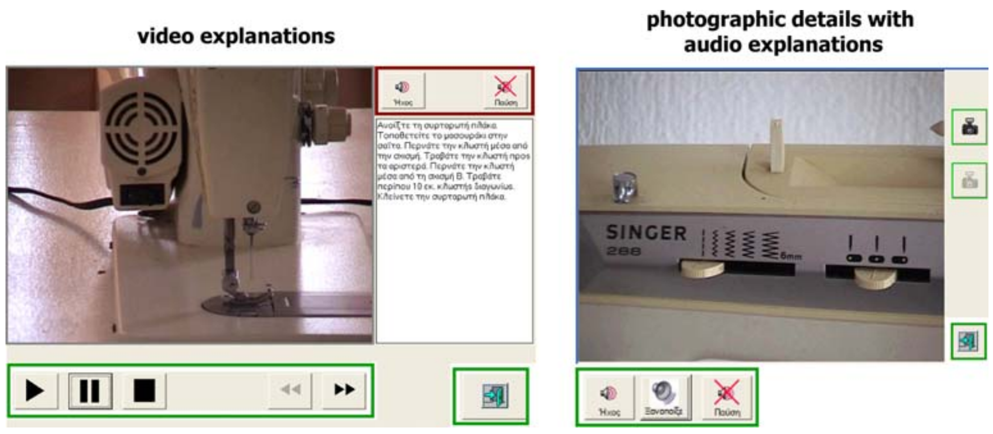


Fig. 9 The different types of training sessions foreseen for the sew trainer application



Ανοίξτε τη συρταρωτή πλάκα. Τοποθετείτε το μασουράκι στην σαΐτα. Περνάτε την κλωστή μέσα από την σχισμή. Τραβάτε την κλωστή προς τα αριστερά. Περνάτε την κλωστή μέσα από τη σχισμή Β. Τραβάτε περίπου 10 εκ. κλωστής διαγωνίως. Κλείνετε την συρταρωτή πλάκα.

Fig. 10 Snapshot of the sew trainer application showing video explanations (left side) and photographic illustrations of sewing machine details (right side)

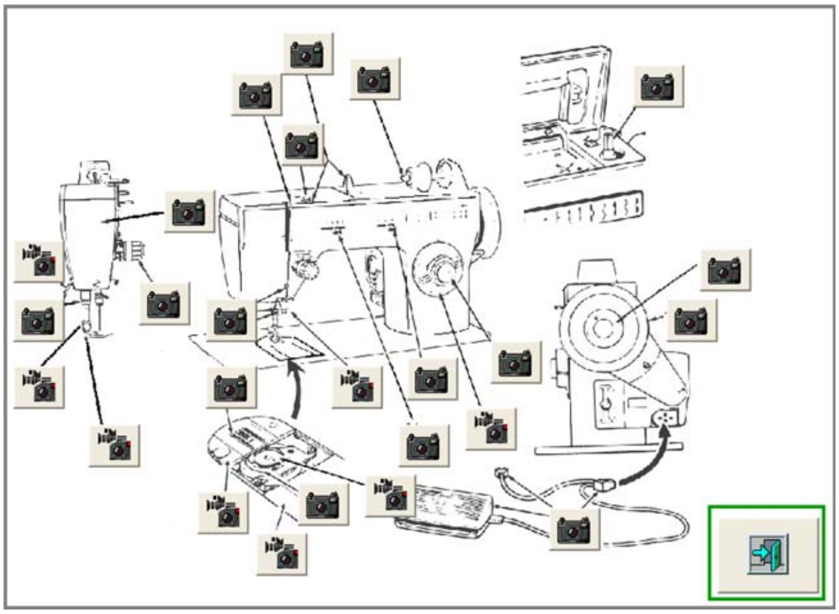


Fig. 11 One of the main screens of the sew trainer application showing the details of the sewing machine for which information is provided, together with the form of the explanation (i.e. pictures or video)

material. In this case, typical active areas over the graphical image were defined, providing audio feedback regarding the title of the particular machine accessory and/or sewing procedure. Selection can be carried out by either clicking on the active area or by pressing the associated push button (with the photographic or video camera icon).

3.2.3 System development

The sew trainer application has been developed using MFC, including the Media Player control for video and audio play back. The video material has been stored in MPEG format (including audio), and the separate audio explanations in WAV files. Finally, the location of files for graphical icons, photographs, video and audio explanations were defined within a Windows INI configuration file.

3.3 Usability evaluation

3.3.1 Approach

The approach employed for the valuation of the training applications focussed on obtaining information concerning the overall interface quality. Subjective evaluation aims to provide tools and techniques for collecting, analysing and measuring the subjective opinion of users when using an interactive software system. Subjective measures are quantified and standardised indicators of psychological processes and states as perceived by the individual user. Typically, subjective evaluation, if it is to deliver maximum benefits, requires a fairly stable prototype of the interactive system. It is a type of usability evaluation, which tries to capture the satisfaction of end users when using a particular system. It therefore does not rely upon an expert's opinion or the opinion of any other intermediary evaluation actor. Some of the benefits of subjective evaluation versus other engineering techniques for usability evaluation include the following:

- Subjective evaluation measures the end-user opinion that is frequently dismissed by other engineering approaches to usability evaluation.
- Subjective evaluation techniques are very reliable, valid as well as efficient and effective in comparison with available alternatives.

When conducting subjective usability evaluation, the analyst should carefully select the sample users to be representative of the target user group. There are several techniques available for subjective evaluation. These include interviews which may be structured or unstructured, the use of diary studies as well as talk about methods. Various questionnaire techniques have been successfully introduced and widely used in subjective evaluations. Questionnaires may be factual, attitude-based or survey type. Amongst the most popular questionnaires are the IBM Computer Usability Satisfaction Questionnaires [11], which has been adopted as the instrument of this study, and the SUMI questionnaire [10].

3.3.2 About the questionnaires

The IBM questionnaires constitute an instrument for measuring the users' subjective opinion in a scenario-based situation. Two types of questionnaires are typically used; the first, After Scenario Questionnaire (ASQ), is filled in by each participant at the end of each scenario (so it may be used several times during an evaluation session), while the other one, Computer System Usability Questionnaire (CSUQ), is filled in at the end of the evaluation (one questionnaire per participant). A 7-point scaling system is used, with low scores being more positive than high scores. In the conducted experiment, non-discrete values were allowed as well. The result of the subjective evaluation with the IBM CUSQ is a set of psychometric metrics, which can be summarised as follows:

- ASQ score, for a participant's satisfaction with the system for a given scenario.
- OVERALL metric, providing an indication of the overall satisfaction score.
- SYSUSE metric, providing an indication of the system's usefulness.
- INFOQUAL metric, providing the score for information quality.
- INTERQUAL metric, providing the score for interface quality.

3.3.3 Process

It is important to mention that Subjective Usability Evaluation using the IBM questionnaires requires a scenario-based procedure. To this effect, a comprehensive scenario has been developed per training application to facilitate the evaluation process. After performing the scenario, the end-users were requested to fill in the ASQ questionnaire and the CSUQ questionnaire; for the specific target user groups, this process has been carried out with close co-operation with the therapists who had to question end-users and then fill-in the questionnaires. As the training applications were targeted to supporting specific real-life tasks, the scenarios included real-situations that end-user had to cope with. For instance, as part of one scenario for the cashier-trainer application, the end-user was asked to handle "an order of two pieces of pizza and an orange juice, charging those via credit" while in other cases there were "customers" issuing such requests directly.

3.3.4 Subjects

Two independent groups were set up, one per training application. The cashier application user group consisted of 12 people as follows:

- Five hand-motor impaired users.
- Six users with cognitive disabilities.
- One motor impaired person also having cognitive disabilities (not having hand-motor impairments).

The user group for the sew trainer application consisted of seven people as follows:

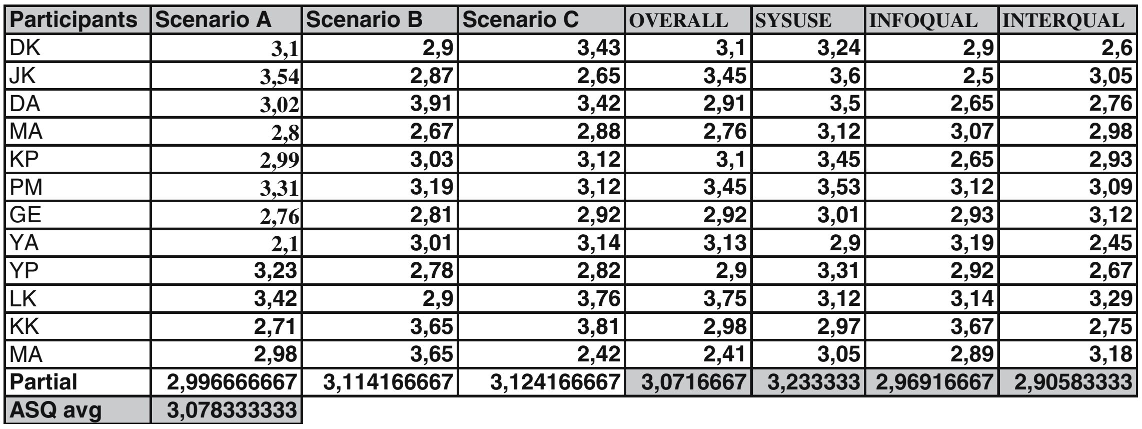
- Three female adults (below 30 years).
- One male teenager (15 years old).
- Three male adults (below 30 years).

Before conducting the study, a couple of days were dedicated to introduce the participants (including their therapists) through on-line demonstrations of the training applications, in which they were allowed to comment in a "thinking aloud" process, being given all the necessary explanations and clarifications. Then they were introduced to the usage scenario, with some time spent to give the appropriate guidance as to what exactly should be done.

3.3.5 Results

In the scoring of the IBM questionnaire, the therapists were allowed to provide scores with fractional parts, in case they felt the discrete values could not convey accurately the desirable scoring. This has resulted in an excessive use of scores with fractional parts, which had an inherent effect on the evaluation results.

In Fig. 12, the results for the cashier training system evaluation are presented. The overall conclusion is that the subjective opinion of users regarding the cashier training application is "good", as indicated by the fact that the overall score (OVERALL, 3.07) passes very successfully the acceptability test by taking scores far less than 4. The best scores are observed in the interface



Participants	Scenario A	Scenario B	Scenario C	OVERALL	SYSUSE	INFOQUAL	INTERQUAL
DK	3,1	2,9	3,43	3,1	3,24	2,9	2,6
JK	3,54	2,87	2,65	3,45	3,6	2,5	3,05
DA	3,02	3,91	3,42	2,91	3,5	2,65	2,76
MA	2,8	2,67	2,88	2,76	3,12	3,07	2,98
KP	2,99	3,03	3,12	3,1	3,45	2,65	2,93
PM	3,31	3,19	3,12	3,45	3,53	3,12	3,09
GE	2,76	2,81	2,92	2,92	3,01	2,93	3,12
YA	2,1	3,01	3,14	3,13	2,9	3,19	2,45
YP	3,23	2,78	2,82	2,9	3,31	2,92	2,67
LK	3,42	2,9	3,76	3,75	3,12	3,14	3,29
KK	2,71	3,65	3,81	2,98	2,97	3,67	2,75
MA	2,98	3,65	2,42	2,41	3,05	2,89	3,18
Partial	**2,996666667**	**3,114166667**	**3,124166667**	**3,0716667**	**3,233333**	**2,96916667**	**2,90583333**
ASQ avg	**3,078333333**						

Fig. 12 The spreadsheet file with the summary of the usability evaluation results for the cashier-trainer application; the averages are shown in *shaded mode*

quality metric (INTERQUAL); this was more or less anticipated, mainly due to the largely tested and verified quality of the scanning interaction techniques that have been injected in the trainer application (the five motor-impaired users gave the highest INTERQUAL score). The results of Fig. 13 revealed also some interesting aspects about the sew trainer application. While the overall scenario-based experience with the system has been considered as satisfactory (ASQ values 3.26 and 3.28, respectively) for the two basic scenarios, there has been a small deviation among the scores on information quality (INFOQUAL, 2.94) and interface quality (INTERQUAL, 3.2). The latter indicates that although trainees found effectively all the information they needed for the purposes of the scenarios, they mainly considered that the delivery could be further enhanced.

4 Learner-adapted courseware

4.1 Overview

4.1.1 Key objectives

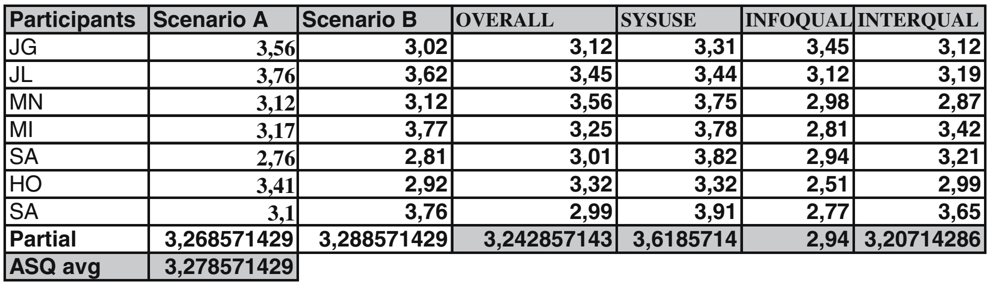
Work in this domain has been targeted towards identifying an appropriate software architecture for digital libraries of learner-adapted courseware, and has been carried out in the context of the DELOS NoE (Network of Excellence) in Digital Libraries (see Acknowledgements). Those findings are currently applied in the development of a learner-adapted e-learning system, reflecting the marriage and consolidation of techniques from two different fields: (a) automatic User Interface personalization; and (b) e-learning information systems. In this context, we will introduce the key design and implementation aspects towards learner-oriented adaptation, rather than detailing the overall software engineering approach for the e-learning system under implementation. Some of the primary functional requirements driving the reported work are:

- To support *adaptation of the user interface* to learner computer knowledge—work in this context is reported in [14].
- To support adaptation of the course content to learner profile.

In this context, as shown in Fig. 14, *content instances*, constituting the "raw material" which is to be appropriately selected and dynamically assembled in a learner-adapted course, is logically separated from *course instances*, providing the semantic aspects of the overall course structure.

This logical split enables a course to be delivered in different forms, by controlling various semantic parameters, such as level of detail, type of explanation, complexity of examples, etc (the specific set of such semantic parameters will be elaborated upon later on in this paper).

Fig. 13 The spreadsheet file with the summary of the usability evaluation results for the sew train trainer application



Participants	Scenario A	Scenario B	OVERALL	SYSUSE	INFOQUAL	INTERQUAL
JG	3,56	3,02	3,12	3,31	3,45	3,12
JL	3,76	3,62	3,45	3,44	3,12	3,19
MN	3,12	3,12	3,56	3,75	2,98	2,87
MI	3,17	3,77	3,25	3,78	2,81	3,42
SA	2,76	2,81	3,01	3,82	2,94	3,21
HO	3,41	2,92	3,32	3,32	2,51	2,99
SA	3,1	3,76	2,99	3,91	2,77	3,65
Partial	**3,268571429**	**3,288571429**	**3,242857143**	**3,6185714**	**2,94**	**3,20714286**
ASQ avg	**3,278571429**					

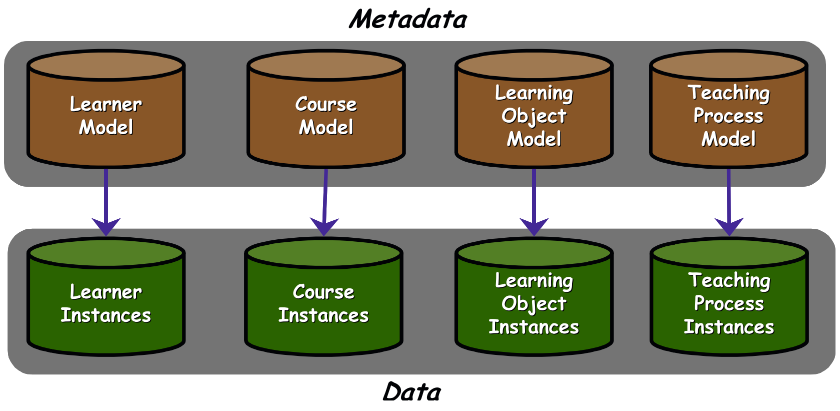


Fig. 14 The basic categories of meta-data and data in adapted interactive courseware

4.2 Metadata design

4.2.1 Learner modelling

Information regarding learners needs to be based on a learner meta-model, which captures all the information considered important regarding the engagement of learners in electronic training courses. The model must be general enough to address various categories of training processes, such as formal classroom based, casual informal or commercial worker training. For instance, in classroom-based training, evaluation records may need to be maintained, while in worker training, work-role specific information will need to be managed. An example showing two different learner-model templates is shown in Fig. 15. The learner model needs to encompass appropriate abstract attributes, enabling concrete learner models to be instantiated for various application domains and training processes.

4.2.2 Course modelling

The structure of a particular course may vary depending on the subject domain. Alternative meta-models need to be also defined, as variations to be employed on a per-domain basis. Currently, it has been decided to deliver the following predefined course templates, each addressing a particular set of domains:

- Interactive book template.
- Guided tutorial template.
- Lecture session template.
- Training session template.

A course template provides the logical structure of a course, on the basis of a hierarchical organisation model, in which the most primitive constituent components are the *learning objects* (to be discussed). The learning objects themselves still follow specific models, meaning they exhibit their own structure and attributes. An excerpt from a book course template is shown in Fig. 16.

4.2.3 Learning objects modelling

Learning objects are the basic constituents of a course (i.e. an instantiation of a course template). They are linked directly to content, and they encompass course-related semantic attributes. Learning objects may have alternative forms, depending on the values of the semantic attributes. In other words, learning objects can be *polymorphic content* entities. This property introduces a small extension to the basic IEEE LTSC standard, in which learning objects have a singular mapping to content resource and delivery format items (i.e. content entities). Additionally, even though learning objects link to content, they need not be physically attached to content; this is accomplished by associating content via resource identifiers, as opposed to embedding content directly within learning objects. This separation is a key ingredient for adapted courseware, as it allows learning objects to be completely decoupled from content form, enabling adaptively existent alternative forms for differing learner requirements and characteristics; this fundamental meaning–form separation, being a property reflected in the SCORM standard [16], is outlined in Fig. 17.

Learning objects are not bounded to simple elements such as paragraph items, but may range from multimedia content and more comprehensive instructional content to external instructional software and relevant tools. In a wider perspective, learning objects could include, and are not limited to, passive items, active items, collaboration gateways, personal contacts and social events. In a similar manner to the previous meta-data categories, alternative templates for learning objects need to be supplied, which will be appropriately combined with different course templates. Some typical attributes of learning objects, in the context of a *lecture-session* course template, may include:

- Type, author, owner
- Terms of distribution
- Format
- Pedagogical attributes, such as:

Fig. 15 Two different templates for learner information illustrating the genuine need for domain-specific learner meta-structures to accommodate typical domain-variant learner attributes

Excerpt of a "worker-trainee" template	Excerpt of a "student-learner-template
• Public personal information.	• Public personal information.
• Private personal information.	• Private personal information.
• Authorisation levels.	• Authorisation levels.
• Work role history.	• Grades.
• Role assigned.	• Semester.
• Start / end date.	• Lesson.
• Domain keywords.	• Written examinations.
• Level of expertise gained.	• Oral examinations.
• Associates.	• Projects.
• Supervisor.	• Evaluation.
• Training history.	• Lesson.
• Course taken.	• Score.
• Start / end dates.	• Behavioural.
• Domain keywords.	• Cooperation characteristics.
• Evaluation record.	• Social characteristics.
• Co-trainers.	• Creativity
• Expertise	• Interests
• Domain.	• Domain.
• Keywords.	• Keywords.
• Experience.	• Relevant activities.
• Intended engagement.	• Associates.
• Work role.	
• Expertise required.	

Fig. 16 Excerpt from an interactive-book course template

- Subject
- Title
- Summary
- Relevant keywords
- Date introduced
- Revision history
- Purpose
- Prerequisites
- Relevance with other courses
- Content
  - Table of contents
  - Chapters
  - *...list continued on the right-side*
- Title
- Introduction
- Keywords
- Table of contents
- Links
- Sections
  - Title
  - Links
  - Units
    - Unit items.
    - Footnotes.
    - Comments.
    - Illustrations.

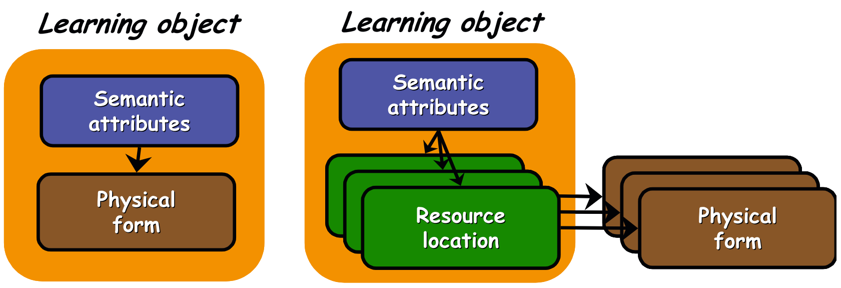


Fig. 17 The separation of meaning and form in learning objects, allowing a single learning object to have alternative forms, all representing the same semantic content

- Teaching/instruction style
- Grade level
- Mastery level
- Prerequisites

The designed learning object templates should provide a set of semantic attributes, which will appropriately link to course templates to enable the implementation of facilities for:

- Search, evaluation, selection and acquisition of learning objects
- Sharing across different courses
- Semantic manipulation in making personalised learner-centred lessons
- Documentation and recognition of the completion of existing, or new, learning/performance objectives associated with learning objects.

As a starting point, the IEEE LTSC Learning Object Metadata (LOM)—see (IEEE LTSC) have been employed, as they were considered as a comprehensive model of learning objects. The original specification aimed to be domain-free, and in this sense, it virtually constitutes a universally applicable template. However, due to such intended generality, it offers a taxonomy of semantic attributes which does not allow domain-specific concepts to be optimally revealed, something which is clearly a serious constraint towards domain-oriented automatic course adaptation.

4.3 System development

The adaptation-oriented decision-making is carried out to accomplish adaptation of the course content by applying software engineering methods for interface adaptation [14] in the context of course content. Content adaptation is applied each time a particular course topic is to be interactively delivered. Adaptation is based on learner information, course information, learning objects and process information. Normally, it is expected that in numerous cases course administrators or tutors will desire to provide their own logic of linking learner information to particular courses or learning objects. In this context, it is imperative to provide an *editable decision making mechanism*, which will allow administrators to easily define their own logic for linking together learner-, course-, learning object- and process attributes. To this purpose, the Decision Making Specification Language (DMSL) is employed, which was originally designed to allow the specification of adaptation-oriented selection logic to choose among alternatives according to decision parameters. The DMSL has been applied for the implementation of interface adaptation logic in unified user interfaces [14], while an interactive rule editor has been developed generating DMSL specifications [12] (see Fig. 18).

Such development techniques have been also applied for the implementation of adapted content delivery for a multimedia information system delivering user-adapted content (see [19]), the PALIO system. In particular, the specific method known as “dynamic query instantiation from polymorphic query patterns” (see Fig. 19), adopted for the implementation of the PALIO system, is also employed in the context of adapted course content delivery. In the PALIO system, the original decision-making process was based on parameters such as nationality, age, location, interests or hobbies, time of day, visit history and group information (i.e. family, friends, couple, colleagues etc.). The information model reflected a typical relational database structure, while content retrieval was carried out using SQL queries in XML. In this context, to enable adapted information delivery, instead of implementing hard-coded SQL queries, query patterns have been designed (such as those shown in Fig. 19), with specific polymorphic placeholders filled in by dynamically decided concrete sub-query patterns.

4.3.1 Dynamic course-content assembly

The fundamental need to support adapted course-content delivery implies the capability to have mutually exclusive alternative content constituents, where the most suitable is decided and activated at runtime. This scenario reveals the need for dynamic content containment where contained course items may normally encapsulate other contained items, i.e. they can be containers themselves. At the implementation level, this introduces the need for a demanding assembly process, since the specific form of every contained object cannot be “a priori” predicted and defined. Moreover, it is impossible to ensure that the delivery format of container components is always interoperable with the delivery format of contained objects. For instance, a textual container document cannot encapsulate directly a text paragraph in binary image format.

To resolve this issue, the *parametric polymorphic containment* technique [14] was applied. Following this technique, each container component aiming to support alternative contained components is implemented to perform internally runtime decision-making, retrieval of contained objects, physical assembly and activation. This way, it is the responsibility of component vendors to explicitly turn container course components into intelligent templates deciding and retrieving contained items on the fly, thus separating the semantic aspects of course components from the physical linking and embedding. Clearly, this implies that the overall course structure is not singular or monomorphic, but its hierarchical design encompasses articulator points where the hierarchical decomposition may have alternative forms, i.e. having a polymorphic hierarchical decomposition. This approach is symmetric to unified user interface design, where the user-task hierarchical decomposition entails alternative decompositions, leading to polymorphic task hierarchies.

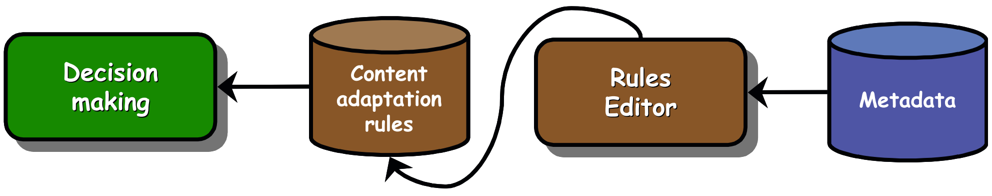


Fig. 18 The runtime adaptation-oriented decision-making process, executing adaptation rules defined through a rule editor

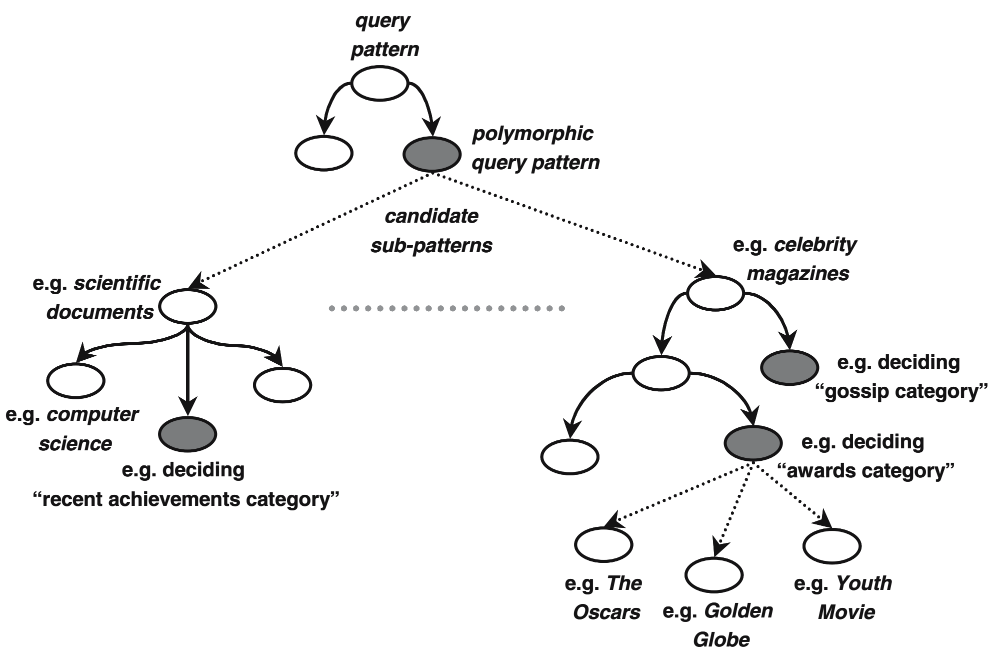

query pattern
polymorphic query pattern
candidate sub-patterns
e.g. scientific documents
e.g. celebrity magazines
e.g. computer science
e.g. deciding "recent achievements category"
e.g. deciding "gossip category"
e.g. deciding "awards category"
e.g. The Oscars
e.g. Golden Globe
e.g. Youth Movie

Fig. 19 Polymorphic query patterns, with mutually exclusive query sub-patterns, to support adapted run-time query specialization

More generic techniques exist as part of the SCORM standard, such as the Content Aggregation Model (CAM). However, such techniques are not yet accompanied with comprehensive software engineering propositions; more specifically, CAM is currently instantiated as a web-based approach relying on a JavaScript Application Program Interface (API) to provide a standardized way of communication and control by a Learning Management System, regardless of what tools are used to develop the content object itself.

4.3.2 *Distributed decision-making logic*

Since every container component is responsible in deciding upon the adaptation-oriented selection of contained course items, the specification of localized decision blocks is encapsulated within container components, and unique naming of every component instance in the context of its container is adopted. In Fig. 20, an example of a decision block for a contained component in a math course is shown to decide the type of contained component to be chosen.

5 A universally accessible chess game

5.1 Design objectives and requirements

The main objective that led to the birth of UA-Chess game, an abbreviation for ''universally accessible chess computer game'', was to provide a fully functional web-based chess game that could be concurrently played by people with different abilities and preferences, including people with disabilities (e.g. low-vision, blind and hand-motor impaired).

This had to be accomplished by supporting alternative input and output modalities and interaction techniques that could co-exist and co-operate at runtime within the delivered user interface, while effectively combined with fully customisable player profiles (see Fig. 21).

Additionally, UA-Chess was required to allow for two-player games over the Internet, as well as games with two opponents sharing the same computer, where the user interface (input and output) is adapted to the particular active player-user profile.

Finally, the inclusive game was required to follow the official Laws of Chess defined by the World Chess

Fig. 20 An excerpt from the decision-making specification block in DMSL for a container course component to adaptively decide upon the most student-appropriate incarnation of a particular contained component

```
if (student."efficiency.math" in {"best", "very good"}) then
    activate "Advanced exercises";
else
if (student."efficiency.math" in {"good", "quite good"}) then
    activate "Normal exercises";
else
if (student."efficiency.math" in {"bad", "very bad"}) then
    activate "Simple exercises";
else
    activate "Student profile error";
```

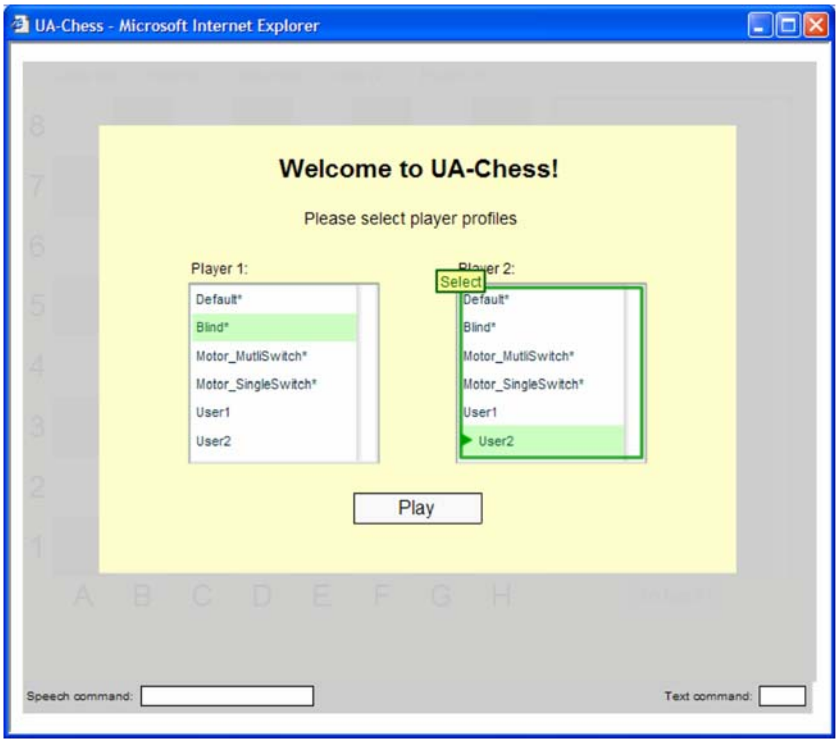


Fig. 22 The introductory "welcome screen" in UA-Chess for end-user profile selection

Federation (FIDE, http://www.fide.com/official/handbook.asp?level = EE101).

This section introduces and discusses the design and implementation of the UA-Chess game that effectively accommodates all the previously mentioned design objectives.

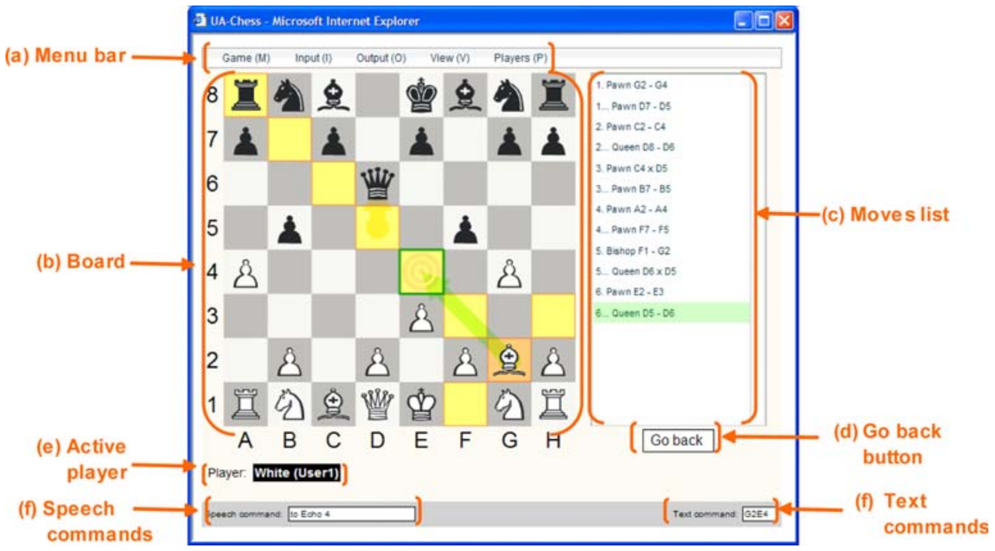
5.2 User interface

Every aspect of the game's functionality is fully accessible through the mouse, the keyboard (or any type of switches emulating keystrokes) and voice input (i.e. speech recognition). UA-Chess has self-voicing capabilities, as it includes a built-in screen reader that offers full auditory access to all interface elements. The game can be dynamically resized according to user preference, supporting zoom in/out at varying levels. Furthermore, several alternative interaction techniques (the parameters of which can be customized) are supported for each device. For example, the keyboard can be used for direct positional input (e.g. to select the square A7 by typing "A7"), for "tabbing" through all the alternative interaction elements, or in combination with different scanning techniques. The interaction capabilities of UA-Chess allow it to provide access to people with combinations of disabilities, such as blind hand-motor impaired persons.

The main component of the game's user interface is a chessboard (Fig. 22, label b) that has the ability to present to the player (visually or orally) game-related information, such as valid moves (Fig. 23a), the last move played (Fig. 23b), if the king is in check etc. Next to the board, there is a moves' history list (Fig. 22, part c) that allows the user to effectively rollback the game to a previous state. The list uses an adapted shorthand chess notation for the description of moves, but when an item is read it is "translated" to plain language. For example, the entry "2... *Queen* G8×G5" will be read as "*move* 2, *black queen moved from* G8 *to* G5 *capturing a pawn*". A menu bar (Fig. 22, part a) is located at the top of the window. All the game functions (e.g. new, load, save etc.), and several user profile customisation parameters (e.g. input/output preferences, player profiles), are available using the menu bar and the related shortcut keys. Feedback is provided about the player who has the move (Fig. 22, part e), and the speech and text commands that were recognized by the system (Fig. 22, parts f, g).

In brief, the following functionality is offered:

- *Game*: New (local or network), load, save and exit.
- *Input*: Activate/deactivate speech recognition, scanning and text commands.
- *Input parameters*: select between manual or automatic scanning, set scanning speed and frame size.
- *Output*: Activate/deactivate speech synthesis. By default, speech synthesis provides audio descriptions only about chessboard events (e.g. a piece is selected or has moved). In addition to this, a screen reader mode is available that provides additional auditory information (e.g. about the current focus, menus, etc.), so that the game can be played by individuals with different levels of visual impairments, or people who have dyslexia. If the screen reader mode is used in



```
1. Pawn G2 - G4
1... Pawn D7 - D5
2. Pawn C2 - C4
2... Queen D8 - D6
3. Pawn C4 x D5
3... Pawn B7 - B5
4. Pawn A2 - A4
4... Pawn F7 - F5
5. Bishop F1 - G2
5... Queen D6 x D5
6. Pawn E2 - E3
6... Queen D5 - D6
```

Fig. 22 A snapshot of the UA-Chess user interface in use

combination with scanning, then the current state of the scanning frame is also described (e.g. "enter menu game", "exit board"), so that the system can be used by people having a combination of visual and motor-impairments.

- *Output parameters*: Show/hide the chessboard, show/hide visual cues about the last move, select board orientation (white at bottom, white at left, white at top, white at right).
- *Oral descriptions*: Listen to information about the board, such as the selected piece, the contents of the chessboard, the contents of the selected piece's rank, the contents of the selected piece's file, the contents of the selected piece's lower diagonals, the contents of the selected piece's upper diagonals, the positions of opponent pieces, all the selected piece's possible moves, which pieces can move, who is playing, the last move made, and all the moves made so far. The user can also ask the system to repeat the last thing said, stop talking or describe the current input focus.
- *Players*: Reset the active player's profile or swap players, i.e. the player who has the white pieces will take the black ones and vice versa.

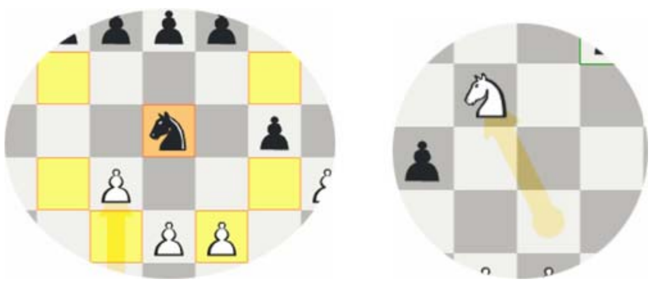


Fig. 23 Representations of: **a** the available moves (marked in *yellow*) and **b** the last move (marked with an *orange arrow*)

5.3 Using alternative modalities

The UA-Chess game supports several alternative input modalities, devices and interaction techniques that can be used exclusively or concurrently at runtime. Some representative examples are provided below.

5.3.1 *Using the mouse*

The mouse is used as with any other "classic" Windows application. When the mouse is over a piece that can move, the square on which the piece resides is highlighted with a yellow colour (Fig. 24, part a). By clicking on the piece, it is selected, and its square is highlighted with orange, while all the squares to which the selected piece can move are highlighted with yellow (Fig. 24, part b). When the mouse is over a square to which the piece can move, an orange target appears on the square and a green arrow connects the selected piece with the target square (Fig. 24, part c). By clicking on the target square, the selected piece moves there.

5.3.2 *Using switch-based scanning*

Scanning, as it has been detailed in the cashier-trainer application, is an interaction technique that is mainly used for providing computer access to people with hand-motor disabilities. The basic idea is that a special "marker" (e.g. a coloured frame) indicates the interaction item (e.g. a button, a menu) that has the input focus. The user has the options to change the current focus or select the current one.

Figure 25 shows the following sample play:

(a) First the focus is on the chessboard.
(b) The user "selects" the currently focussed object and the focus (i.e. green frame) shifts over the first piece that can move, which is also highlighted.

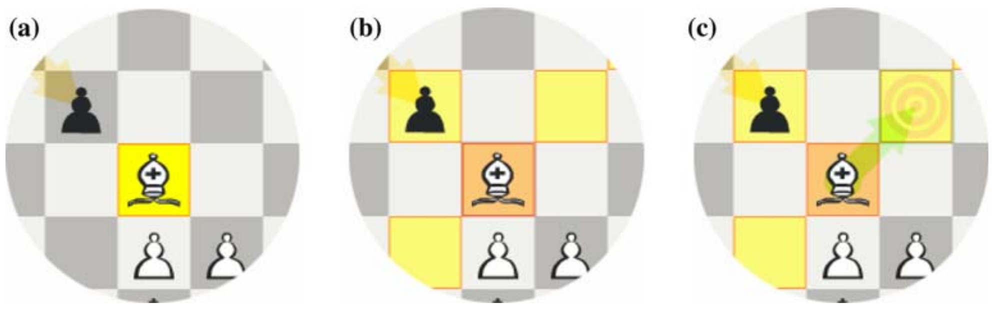


Fig. 24 Selecting and moving a piece directly using the mouse

(c) The user shifts the focus among the pieces that can move and selects one of them. Thus, the focus is transferred to the first possible move.
(d) The user moves the focus over another possible move.
(e) The piece moves to the selected square and input control is handed to the opponent player.

5.3.3 Using voice recognition

Using speech recognition, users can have totally "hands-free" access to all the functionality of UA-Chess. When speech recognition is active, a text box labelled "Speech command:" is visible below the bottom left corner of the board (Fig. 26), in which speech commands "heard" by the system are presented; a sample move using speech recognition is as follows:

(1) The user selects a piece by saying "*from <square name>*". A *<square name>* is composed of the square's file (i.e. a letter from "A" to "H"), plus its rank (i.e. a number from "1" to "8"), e.g. A2, B7 and C4. In addition to using the "traditional" letter names, the NATO phonetic alphabet can be used to improve speech recognition results.
(2) Then the user selects a move by saying "*to <square name>*".
(3) Finally, the user confirms the move by saying "*move*".

5.4 System development

A first prototype of the game was developed using Visual Basic [4] to further study the concept in practice. User tests and expert reviews allowed to improve interaction design issues and fix functional and usability problems. Unfortunately, this development approach suffered from two drawbacks: (a) the game could only

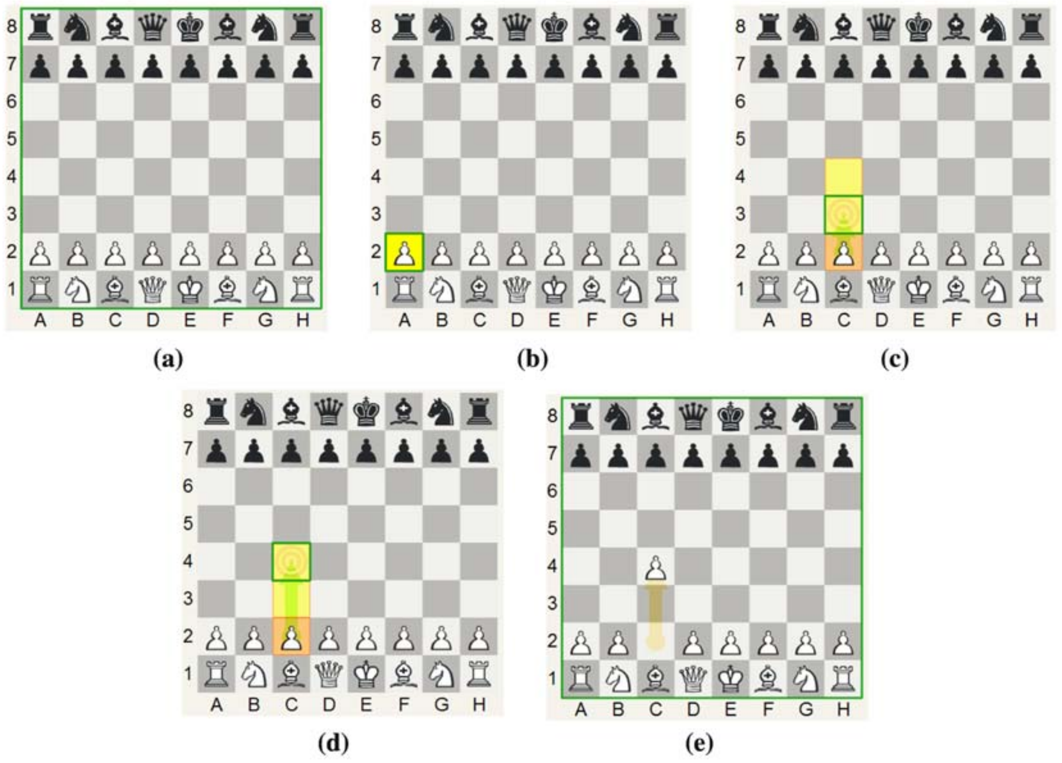
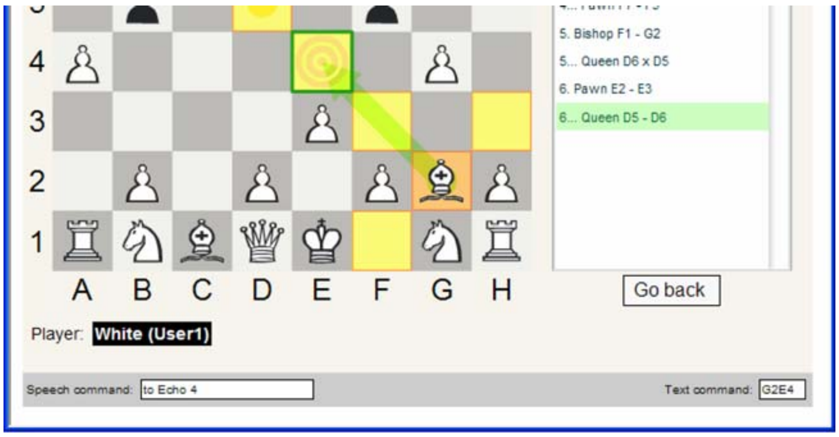


Fig. 25 Selecting and moving a piece using switch-based scanning



Player: White (User1)

Speech command: to Echo 4

Text command: G2E4

Fig. 26 The interface text label where the currently recognized speech command appears

run in computers using the MS Windows operating system, thus excluding a large number of potential users; and (b) it required to be downloaded and installed, thus making it harder to get and use (especially for disabled persons), and also creating a problem with future fixed and updated versions.

Thus, for the development of the final game, Macromedia Flash MX Professional 7 was selected as the development platform. This plug-in is available for a large variety of operating systems (Windows, Mac OS, Linux and Solaris) and web browsers (e.g. MS Internet Explorer, Netscape, Mozilla, Opera). Since Flash does not offer speech input and output capabilities, in order to support speech recognition and synthesis Speech Application Language Tags (SALT) technology (www.saltforum.org) was employed. SALT is an emergent standard for developing voice-enabled applications for the Web that extends existing markup languages, such as HTML and XHTML. Currently the only SALT-compliant browser available is Microsoft Internet Explorer 6 (using a related plug-in), but the OpenSALT project (hap.speech.cs.cmu.edu/salt/) has announced its plans to make freely available a SALT-compliant browser for Linux, based on the Mozilla web browser. In this context, a programming interface was developed to integrate SALT (that runs in the web page) with Flash (that runs as an object in the web page).

To support hierarchical scanning for the motor-impaired users and non-visual interaction techniques for the visually impaired, custom interaction objects were created (e.g. the chess board and the pieces) and existing Flash objects were augmented (e.g. buttons, menu, lists) with related capabilities. Furthermore, a focus/scan manager was developed to orchestrate multi-modal user interaction.

5.5 Innovation

Currently there are two chess programs that are accessible to the blind: WinBoard (ftp://www.ftp.freedom-scientific.com/users/hj/winboard/WinBoard.exe) that is adapted for compatibility with a specific screen reader (JAWS) and KChess (http://www.arkangles.com/kchess) that provides some basic information using speech and is also compatible with screen reader software. For motor-impaired users, or people with combinations of impairments, there is currently no other accessible chess program that the authors are aware of. Surprisingly, it has been impossible to find another application like UA-Chess in the entertainment domain that could be concurrently played by two gamers with different abilities (or disabilities), using a variety of alternative input/output modalities and techniques in any combination, both through the Internet (i.e. remote mode) and also sharing the same computer (i.e. local mode). UA-Chess is freely available online at the URL address www.ics.forth.gr/uachess. Additionally, it is important to note that the UA-Chess game was selected as one of the finalists of the 2004 European "Design for All" awards.

5.6 Usability evaluation

An initial subjective usability evaluation process has been carried out with six participants, one of who was an able-bodied user, two blind users and three motor-impaired users. The evaluation process has been carried out around two basic scenarios, actually constituting two rounds per player in an informal competition. All players had only a little prior knowledge of chess, so it was necessary to spend some introductory time on the game and its basic rules. The player pairs for each round were randomly chosen, just to make the process more challenging for the end-users. The results of the study were quite encouraging, although somehow expected. More specifically, apart from the very good scores given by the blind and motor-impaired users for all metrics (all metrics are below three, which is considered to be a very good score), it was very quickly clear that the disabled users were immediately enthusiastic with the fact they were given the opportunity to actually play a computer game. Moreover, one very positive element in the overall evaluation process was that the participants did not know their opponent until the game was actually completed. This proved to be another unexpected motivating factor that turned the study into a more interesting “event”, something quite typical for computer-game play sessions. The results of the evaluation study are briefed in Fig. 27. The enthusiastic opinion of disabled users for the UA-Chess game is evident from the low scores given for all metrics (lower values correspond to a better evaluation than higher values). On the contrary, the able-bodied user that has participated in the evaluation experiment was far less enthusiastic with the chess game, in comparison to the rest of the participants. In an informal interview with this user, it quickly became evident that the negative score was mainly due to the falsely expected three-dimensional visualizations, graphical effects and animations, typically met in today’s action games. Although it is in our future plans to incorporate within UA-Chess such appealing features, those were not clearly the main objective of the original design of the game.

6 Conclusions and future work

The e-learning field encompasses a wide variety of e-learning systems, with different pedagogical approaches, semantic content and interaction characteristics. In this context, the work reported in this paper has placed primary emphasis on the inclusive characteristics of e-learning systems, moving accessibility forward, while also pursuing automatic learner-oriented content personalization. The reported experience has concerned four e-learning systems:

- Two training applications for people with cognitive disabilities and hand-motor impairments. Those systems have been entirely evaluated and are in use for more than 3 years.
- One e-learning platform for learner-adapted courseware, that is currently under development.
- One entertainment-oriented e-learning application, falling in the recently appearing domain of universally accessible games. This application was just recently completed and has been launched for public use over the web.

From the evaluation and deployment experience of the training applications, the following design-oriented conclusions have been drawn:

- A large number of early evaluation cycles needs to be organized and conducted, always in close co-operation with therapists.
- In many cases, it is necessary to implement varying design features, since there are tasks where the need for optimal design mandates design variations for some users, by enabling individualized deployment through configuration files.
- In numerous dialogue scenarios, the default graphical feedback methods do not suffice, but there is a need for more graphically emphatic ways.
- The usability evaluation process for work training requires intensive testing with multiple real-life scenarios.
- Accessible interaction methods need to support all together: speed of interaction, accuracy of actions and physical stamina, which may mandate the design of an application-specific accessible dialogue (e.g. switch-based hierarchical scanning).
- The graphical interface should be kept very simple, pop-ups are to be avoided, while the overall dialogue design should reflect a “shallow” hierarchy of alternative “screens”.
- Software or hardware speech synthesis quality should be intensively tested with different users, exploiting all the API capabilities to control the speed and voice styles; in many cases the recoding of familiar voices may be the only optimal solution (for people with cognitive disabilities).

From the accumulated experience in the context of learner-adapted e-learning platform development, the following implementation-oriented conclusions are drawn:

- The concrete software engineering interpretation of standards like LTSC or SCORM is still blurred, as these mainly introduce semantic design directives, rather than prescriptive implementation techniques.
- Learner modelling, from meta-data to learner-profile management and administration, is clearly the easiest development part.
- Domain-specific course meta-structures are prominent so as to have better matching of e-learning deployment requirements to the eventual e-learning system features.

Fig. 27 The results of the chess usability evaluation process. (The range is from 1 to 7, where 1 is the highest/best possible score)

Participant	User Category	scenario 1 ASQ (play round 1)	scenario 2 ASQ (play round 2)	OVERALL	SYSUSE	INFOQUAL	INTERQUAL
Player 1	Blind	2.33	2.67	2.37	2.25	2.43	2.67
Player 2	Blind	2.67	2.67	2.53	2.63	2.43	2.67
Player 3	Motor-impaired	2.33	2.00	2.32	2.38	2.29	2.33
Player 4	Motor-impaired	2.00	2.00	2.05	2.13	2.16	2.00
Player 5	Motor-impaired	2.33	2.33	2.42	2.38	2.43	2.67
Player 6	Able-bodied	4.00	3.67	4.11	4.25	4.14	3.67
	Average	**2.61**	**2.56**	**2.63**	**2.67**	**2.65**	**2.67**

- The decision-making logic should be a separate property of learning objects, being editable in a rule-based fashion, while supporting directly executable (not necessarily interpreted, but can be compiled on the fly) specifications.
- Generic universal APIs are very hard to accomplish for learning object instances, although this is set as a primary target of SCORM; the freedom of container-exported APIs for candidate-contained components was preferred in the reported work.
- Interoperability cannot be open, as it presupposes a particular component-ware technology (i.e. not all interoperability technologies interoperate); it was preferred to postpone the lower-level interoperation decision to containers vendors, as it has been difficult to choose a globally optimal approach (e.g. XML, scripting, CORBA, Beans, DCOM etc.).
- Polymorphic containment for components can be very easily accomplished, placing the responsibility for containment conformance to contained components.

Finally, in the context of universally accessible games, the following conclusions are drawn:

- The capability to turn a typical graphical computer game into a universally accessible version is largely dependent on the possibility of transforming the game board, input control and feedback methods to accessible counterparts. In the case of UA-Chess, the non-visual representation introduced the most difficult challenges.
- The management of concurrent I/O with multi-modal feedback is implementation-wise a very demanding feature, which required the internal programming of an abstract task engine.
- The delivery of a universally accessible game over the web has to address various technological barriers, mostly related to special-purpose I/O (e.g. scanning, voice recognition, speech synthesis, force feedback etc).

Each of the four e-learning systems reviewed in this paper had its distinctive design, implementation and deployment characteristics. It is argued that because of the fact that the e-learning field accounts for a very large number of information systems, any consolidated wisdom towards making a particular category of e-learning systems user-inclusive is bound to have limited chances for open deployment and reuse. In this context, the reported experience is not put forward as a recipe for all classes of inclusive e-learning systems, but primarily aims to deposit a specific consolidated know-how, while also raising awareness on the key design and implementation challenges in the pursuit of inclusive e-learning technology.

Acknowledgments The two training applications reported in this paper have been developed under a sub-contract (2001) from the "Centre of Special Training", of the "Therapeutic Clinic of Lifelong Diseases", located at the city of Agios Nikolaos, region of Crete. Part of the work on learner-adapted courseware reported in this paper has been carried out in the context of the European Commission funded "DELOS Network of Excellence on Digital Libraries"—DELOS NoE (IST-2002-23112).

References

1. Bowman RF (1982) A Pac-Man theory of motivation: tactical implications for classroom instruction. Educ Technol 22(9):14–17
2. CEN/ISSS (2005) CEN (European committee for standardization), Learning technologies workshop, http://www.cenorm.be/cenorm/businessdomains/businessdomains/isss/activity/wslt.asp
3. Giguette R (2003) Pre-Games: games designed to introduce CS1 and CS2 programming assignments. ACM SIGCSE 35(1):288–292
4. Grammenos D, Savidis A, Stephanidis C (2005) UA-chess: a universally accessible board game. In: Stephanidis C (ed) Universal access in HCI: exploring new interaction environments. Proceedings of the 11th international conference on human-computer interaction (HCI International 2005), Las Vegas, Nevada, USA, 22–27 July, vol 7. Lawrence Erlbaum Associates, Mahwah, NJ [CD-ROM]
5. Hill JM, Ray CK, Blair JRS, Carver CA (2003) Puzzles and games: addressing different learning styles in teaching operating systems concepts. In: Proceedings of the technical symposium on computer science education (SIGCSE 2003), pp182–186
6. IEEE LTSC (2005) Learning technology standards committee. WG12: learning object metadata. http://ltsc.ieee.org/wg12/index.html
7. IGDA (2004) IGDA accessibility white paper. http://www.igda.org/accessibility/IGDA_Accessibility_WhitePaper.pdf
8. IMS (2005) IMS, Global learning consortium. http://www.imsglobal.org/
9. ISO/IEC (2005) ISO/IEC JTC1 SC36, ITLET, Information technology for learning, education and training, http://jtc1sc36.org/
10. Kirakowski J (1996) The software usability measurement inventory: background and usage. In: Jordan PW, Thomas B, Weerdmeester BA, McClelland IL (eds) Usability evaluation in industry. Taylor & Francis, London, pp 169–178
11. Lewis RJ (1995) IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. Int J Hum Comput Interact 7(1):57–78
12. Savidis A, Antona M, Stephanidis C (2005) A decision-making specification language for verifiable user-interface adaptation logic. Int J Softw Eng Know Eng 15(6):1063–1094
13. Savidis A, Stephanidis C (2001) Development requirements for implementing unified user interfaces. In: Stephanidis C (ed) User interfaces for all—concepts, methods, and tools (ISBN 0-8058-2967-9, 760 pages). Lawrence Erlbaum Associates, Mahwah, NJ, pp 441–468
14. Savidis A, Stephanidis S (2004) Unified user interface development: the software engineering of universally accessible interactions. Int J Universal Access Inf Soc 3:165–193
15. Savidis A, Vernardos G, Stephanidis C (1997) Embedding scanning techniques accessible to motor-impaired users in the WINDOWS object library. In: Salvendy G, Smith M, Koubek R (eds) Design of computing systems: cognitive considerations (21A). Elsevier, Amsterdam, pp 429–432
16. SCORM (2003) Best practices guide for content developers. The guide, 1st edn. 20030228. http://www.lsal.cmu.edu/lsal/expertise/projects/developersguide/#guide
17. Stephanidis C (ed) (2001) User interfaces for all. Lawrence Erlbaum, NJ
18. Stephanidis C, Paramythis A, Sfyrakis M, Savidis A (2001) A case study in unified user interface development: the AVANTI web browser. In: Stephanidis C (ed) User interfaces for all, Lawrence Erlbaum, NJ, pp 525–568
19. Stephanidis C, Paramythis A, Zarikas V, Savidis A (2004) The PALIO framework for adaptive information services.

In: Seffah A, Javahery H (eds) Multiple user interfaces: cross-platform applications and context-aware interfaces. Wiley, Chichester, UK, pp 69–92

20. Verenikina I, Harris P, Lysaght P (2003) Child's play: computer games, theories of play and children's development. In: Wright J, McDougall A, Murnane J, Lowe J (eds) Young children and learning technologies, vol 34. Selected papers from the International Federation for Information Processing Working Group 3.5 Open Conference, Melbourne, Australia, ACS, pp 99–106. http://crpit.com/Vol34.html