SIViP (2008) 2:371–384
DOI 10.1007/s11760-008-0086-z

ORIGINAL PAPER

Automatic generation of adaptive, educational and multimedia computer games

Mária Bieliková · Marko Divéky · Peter Jurnečka · Rudolf Kajan · L'uboš Omelina

Received: 1 July 2008 / Revised: 21 August 2008 / Accepted: 19 September 2008 / Published online: 21 October 2008

Abstract Education accompanies us throughout our whole life. Many innovations in education have originated from modern technologies. However, the majority of learners—especially children or teenagers—find studying from electronic educational sources and web-based information systems less exciting than playing today's popular computer games that, conversely, lack signs of education. In this paper, we describe an innovative concept of generating three-dimensional interactive multimedia educational games that combine the excitement and looks of popular computer games with the educational potential of e-learning, and the concept's realization by a software system called S.M.I.L.E.: Smart Multipurpose Interactive Learning Environment. One of its key features is the automatic generation of games based on a model created by teachers without needing them to be familiar with programming or game design. Moreover, we consider various learners' abilities and features that enable different users (including handicapped) to learn effectively by playing educational games easily created by teachers. We follow the idea that everyone needs access to quality education and are convinced that by enabling cooperative education not just among learners, but also between handicapped and able-bodied ones, we bring the humane dimension into education.

Keywords Education · Edutainment · Multimedia computer games · Game model · Game generation · Adaptive computer game

M. Bieliková (✉) · M. Divéky · P. Jurnečka · R. Kajan · L'. Omelina
Institution of Informatics and Software Engineering,
Faculty of Informatics and Information Technologies,
Slovak University of Technology,
Ilkovičova 3, 842 16 Bratislava, Slovakia
e-mail: bielik@fiit.stuba.sk
URL: http://www.fiit.stuba.sk/~bielik/

1 Introduction

Today, learners have the possibility to learn interactively and share their ideas and knowledge with each other—even if they are miles apart. Education has significantly evolved in the last decade, thanks to modern technologies. On the one hand, learners currently have access to diverse educational sources, such as multimedia encyclopedias or interactive online tutorials. On the other hand, many learners lack motivation into studying. In fact, we all need to be motivated more into studying.

Educational games (i.e., serious games) are a new genre that evolves rapidly [1]. Computer games offer enjoyment and fun, both of which play a crucial role in effective learning [2]. Moreover, computer games can do a great job in motivating [3]. However, the most popular ones are currently far away from being educational and, on the other hand, educational games often lack the thrill of their popular counterparts. In addition, such games have a fixed plot that cannot be altered by teachers who do not have any programming expertise. Their creation is almost solely restricted to software professionals. Additionally, only a few of them can adapt to player abilities and therefore provide better support for individual players [4].

Another important issue that needs attention is that learners with a handicap are often ignored and forgotten. Such learners cannot explore the advantages that modern e-learning systems bring with them. Handicapped learners have absolutely no options to play today's modern (educational) computer games. No matter how hard they try, it is practically impossible for them to play such games along their able-bodied friends or schoolmates, given the lack of options they have today. That is why we are trying to give them an opportunity to be able to learn alongside others in an entertaining and joyful way.

We propose an innovative concept that forms a unique solution for the above-mentioned problems. It combines the advantages of both interactive educational content (represented by learning objects) and popular computer games by giving teachers (i.e., authors of educational content) the ability to have exciting educational games automatically generated according to their preferences. These games can be played even by handicapped users (e.g., visually impaired or deaf). We also describe the S.M.I.L.E. system (Smart Multipurpose Interactive Learning Environment), which we have developed in order to prove our concept.

On a broader scale, the concept that we propose, including the developed S.M.I.L.E. system, serves as an effective way to create not only educational, but any type of computer games. By focusing on one domain (education), we are able to generate games automatically and thus reduce the workload laid on the game creator (in our case, a teacher) to the minimum.

The paper is structured as follows: In Sect. 2, we describe work related to the subject of this paper. Our model of educational games is described in Sect. 3, along with proposed learning environments and representation of educational learning objects. Section 4 explains the mechanism used to automatically generate educational games, including an example scenario. A software prototype that we have implemented is described in Sect. 5, whereas Sect. 6 explains how we utilized adaptivity and adaptability mechanisms in our work. Finally, Sect. 7 gives our concluding remarks and proposals for future work.

2 Related work

Our goal was to devise a concept that would support the reshaping of education by luring children into educating themselves by playing multimedia computer games created especially for that purpose. There is extensive research confirming the fact that computer games have a great potential in improving the educational process [5,6] and that the use of computer games in classes not only provides learners with knowledge in an enjoyable form, but also helps them develop various cognitive and thinking skills [2].

At present, learners are provided with different kinds of interactive educational games. The drawback of these games is that they are rather limited in the amount of how much knowledge they can give to the learners, since they are all focused on only one narrow area of knowledge. Furthermore, creation of (not only) educational multimedia games is a cumbersome and complex process that requires in-depth knowledge of programming, game design, 3D modeling and much more—an impossible task for an average teacher.

In order to simplify the complex process of creating computer games and make it more effective, game engines have been developed that can be reused across multiple (not only educational) games [7]. There are over 200 game engines available (see http://www.devmaster.net/engines). However, these engines are still too complex to work with, due to their general-purpose nature.

Genre-specific computer game design software (known as toolkits) exist that ease the programming burden (such as Game Editor at http://www.game-editor.com or Game Maker at http://www.gamemaker.nl). However, such software supports the creation of only simple two-dimensional games that are not comparable to modern three-dimensional computer games. Although more robust software, such as Alice (available at http://www.alice.org) or Inscape (available at http://www.inscapers.com) enables to create almost any kind of (educational) computer games, it requires to learn a great amount of scripting and requires game design skills.

However, educational games that are on par with modern three-dimensional computer games exist (e.g., Dimension M available at http://www.dimensionm.com). Such games have been developed by a team of professional game designers and do not let teachers alter their scenario or content in a way that does not require knowledge and experience in programming and game design. In addition, modern educational games are tied only to a specific area of knowledge (such as mathematics in the case of the previously mentioned game entitled Dimension M).

Most games do not include any adaptation (they are interactive, but in the same context, regardless of the user, they all behave the same). Some of them are adaptable (the user can manually set the environment according his preferences). When adaptive, mostly a stereotype user model is used.

To summarize, existing solutions do not allow teachers to create three-dimensional, adaptive, multiplayer and multimedia educational games for their students without requiring programming and game design skills. We therefore focus on creating an extra layer that runs atop of an existing game engine (Torque Game Engine Advanced, available at http://www.garagegames.com/products/torque/tgea) and allows teachers to define educational games without the need to write a single line of code. These games follow an educational games model that is described below.

3 Educational games model

Our model of educational games is based on today's most popular genre of computer games—Role Playing Games (RPGs) [8]. Such games are exciting, easy to control (which is especially important for handicapped users) and have serious educational potential [2].

Games based on the RPG genre take place in a realistic world set in a specific time (e.g., in medieval age, at present time or in the future) or in an imaginary world that is or is

not close to reality. Players are represented by and control an *avatar* (most commonly in a human form) and solve various *quests* throughout the game. These require the player to find certain objects that he needs to correctly use or combine in order to solve a particular problem, and/or require the player to choose a correct answer from a number of given answers to a certain question. Quests are assigned to players by Non-player Characters (NPCs) that cannot be controlled by players and interact with them through dialogs. In our game model each player can collect things into an inventory that contains things necessary for solving particular tasks during gameplay. Moreover, the game objects representing learning objects have assigned location that can be either NPC or another object. Based on this location, an action aimed at collecting the object is invoked during gameplay, e.g., if the game object representing a chemical compound has a NPC set as its location, the player would encounter this NPC for acquiring the object (chemical compound).

3.1 Learning environments

Each educational game represents one of the following learning environments:

- **Teacher-created games** present a practical and enjoyable form of evaluating the learners' level of knowledge and skills from a desired field. These games are specified by teachers either *visually* by defining their concept, or *automatically* generated by the system using specified learning objects and relationships between them and according model parameters set by the teacher.
- A **persistent virtual world** that is made of quests based on all available learning objects, and therefore practically resembles one large game that contains all possible educational quests. It is intended for self-training and raising the players' level of knowledge.

The only difference between the teacher-created games and the virtual world is that the games in teacher-created model can also contain quests marked as *private* by their authors. Practically speaking, the described concept encourages learners to first train and learn themselves by solving quests in the persistent virtual world, and afterwards to test their newly gained knowledge and skills in games that have been created by teachers to test the level of comprehension of a certain knowledge (from various domains such as mathematics, chemistry or biology). Since these games can contain private quests (which are not available to players in the common virtual world), they support testing the level of knowledge of individual learners that is not available in current educational games. Moreover, it enables an adaptation in the process of selecting suitable quests according the level of learner's knowledge.

Fig. 1 A "Did you know. . ." tip with educational content

3.2 Representation and formatting of learning objects

Educational content, represented by learning objects, serves as source of knowledge for learners. Moreover, in case a learner has a problem completing a particular quest during gameplay, he has in our system the possibility to view the learning object related to the problem that is tested by the quest (via the in-game *Knowledge Viewer*, which is used to render learning objects directly in-game—a unique feature not found in existing multimedia game solutions). Moreover, educational content that may be helpful in completing the currently assigned quests is dynamically suggested to learners during gameplay as "Did you know. . ." pop-ups, a standard technique of tips—see Fig. 1.
The purpose of educational content is to help players successfully complete the available educational quests. Learning objects are collaboratively created, edited and managed by teachers and are stored in a format that makes it possible to adapt their formatting for every user's individual preferences.

Educational content contains anything ranging from formatted text and lists to pictures, videos, audio files and other multimedia, such as Macromedia Flash elements. Each learning object is represented in an XML format designed especially for this purpose. As a consequence of the fact that the format separates the learning objects' content from their formatting, we are able to let each user specify how he would like to format all viewed learning objects (e.g., learners with a vision impairment can set the size of all displayed text and have all multimedia elements shown larger, deaf learners can set the option to display available subtitles below every video or audio file—see below for a list all possibilities).

The mechanism of formatting learning objects according to users' preferences is shown in Fig. 2. Each user has a profile containing his preferences according to which all viewed knowledge is formatted—differently for each and every user. Moreover, different settings can be used for the learning objects viewer that corresponds to standard learning environments where a learner can browse and study learning objects (system viewer preferences) and the in-game knowledge

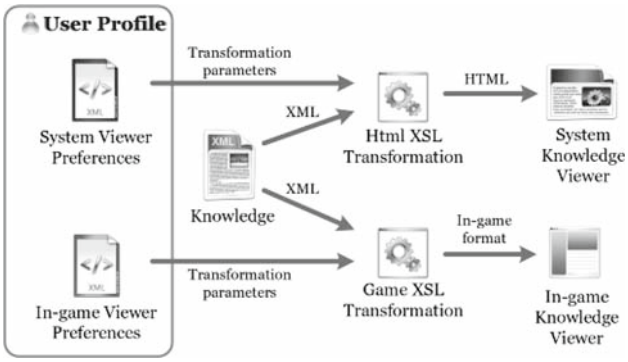

Fig. 2 Formatting of learning objects

viewer that enables displaying learning content within gameplay (in-game viewer preferences).

Preferences contain various formatting parameters (the first three are required and the others are optional):

- Background color of documents
- Font face, size and color of headings
- Font face, size and color of text
- Font face, size and color of captions for multimedia elements
- Font face, size and color of bold, italic, and underlined text
- Font face, size and color of links, numbered and bulleted lists
- Zoom factor of all multimedia elements.

The specified settings serve as parameters for the corresponding XSL transformations—the transformation into HTML and the transformation into the format used by the in-game *Knowledge Viewer*. The XSL transformations take the viewed learning objects as XML input and the user's settings as their parameters, and then transform the knowledge into formats used by the viewers. The resulting documents are formatted according to users' individual preferences.

Users' preferences are independent, i.e., a user can set different formatting parameters for the system *Knowledge Viewer* and different parameters for the in-game *Knowledge Viewer* (or he can have both preferences equally set), resulting in greater flexibility. Users are able to choose from a set of predefined formatting presets based on standardized schemes (e.g., see a preset designed for colorblind learners, right part of Fig. 11 in Sect. 6.1) or may further customize these presets.

Thanks to the fact that both learning objects and the knowledge viewers' formatting settings are stored in XML format, it is possible to replace them and the XSL transformations with a different XML representation, mainly for the integration with other existing bases of learning objects, including e-learning standards [9].

4 Automatic generation of educational games

The key to the automatic generation of educational games is based on the fact that every stored learning object has a defined set of *game objects* associated to it and each game object can have one or more *relationships* defined with other objects (see Fig. 3). For example, chemistry learning objects about water, sulfur trioxide and sulfuric acid (named "Water," "Sulfur Trioxide" and "Sulfuric Acid") may have game objects "H_2O," "SO_3" and "H_2SO_4" associated to them respectively.

When a teacher chooses to generate a new quest or game based on a learning object (e.g., "Sulfuric Acid"), all game objects related to the chosen learning object are found (e.g., "H_2SO_4"), their relationships with other game objects are examined (e.g., relationships with "H_2O" and "SO_3") and a set of *interactions* among them is created—forming a quest that learners accomplish throughout gameplay.

Based on parameters specified by the teacher—such as difficulty, the generated quests may require more or less effort and knowledge depth in order to be successfully completed. For example, if the difficulty was set to easy, the resulting quest will require the players to only find "H_2SO_4" throughout the game. In case the difficulty was set to moderate, students will have to find both "H_2O" and "SO_3" and afterwards mix them together correctly in order to complete the generated quest. A quest of high difficulty would not only require the players to find all objects in any way related to "H_2SO_4," but also to combine them correctly.

4.1 Phases of game generation

Automatic educational game generation consists of two separate phases: (1) *Quest Generation* and (2) *Game Generation*. The *Quest Generation* phase consists of the following steps:

1. *Analysis of game objects* related to the specified learning object, including their relationships.
2. *Formation of a partially ordered set of in-game interactions* from the analyzed relationships.
3. *Creation of generic NPCs* who assign and explain the generated quests to players and reward those who have successfully solved them.

The *Game Generation* phase is made of the following steps:

1. *Content analysis* of all specified quests, together with game objects that they contain, resulting in a list of needed NPCs, game objects and other game assets.
2. *Landscape generation* based on predefined landscape objects and properties of the analyzed game objects.
3. *Grouping of generic NPCs (from quests) into specific in-game NPCs*, since a single in-game NPC can be

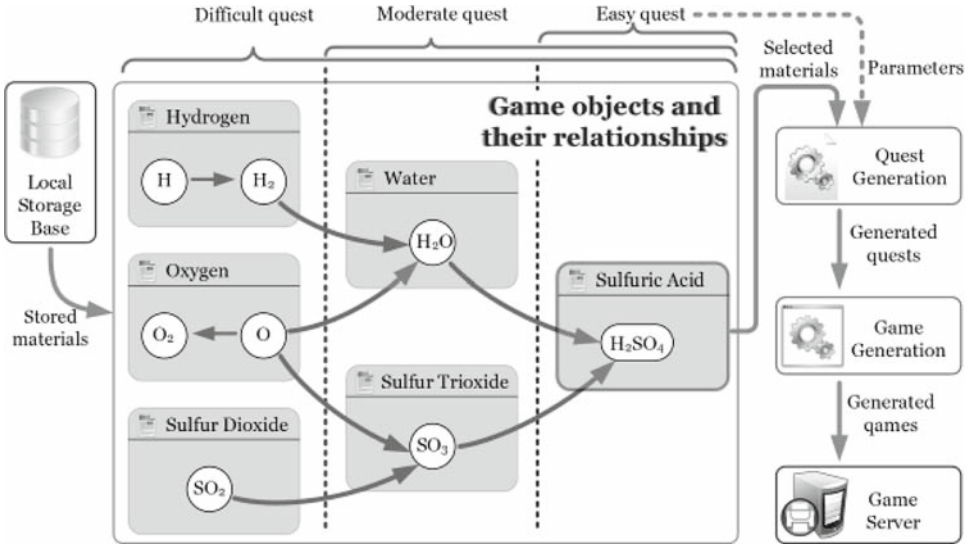

Fig. 3 Automatic generation of educational games from an example set of game objects

responsible for and handle the assignment, explanation and accomplishment of multiple quests.

4. *Generation of textual dialogs* (based on a pre-defined set of sentence fragments, each suited for a different scenario) for the created in-game NPCs.
5. *Placement of all quests and in-game NPCs onto the generated landscape*.
6. *Generation and positioning of waypoints for the movement* of severely sight-handicapped and blind players throughout the landscape (see Sect. 6.1).

Both phases utilize artificial intelligence-based algorithms for solving particular tasks. The *Quest Generation* phase uses a depth-first search strategy with backtracking for quest generation, since a set containing all game objects along with their relationships forms a large tree with one game object as the root node (see Fig. 3 for an example tree with "H_2SO_4" as its root node). The quest generation algorithm first constructs such trees (each containing a game object related to the teacher-specified learning object as its root node), scans these trees for relevant relationships and creates a set of in-game interactions that forms the resulting quest.

The *Game Generation* phase utilizes simulated annealing to find appropriate places for towns during the landscape generation process. First, a random terrain is constructed based on a generated height map. Afterwards, the terrain is scanned for an appropriate area in which a town can be automatically placed. The chosen area must be as flat as possible, and at the same time, it must not look unnatural to players. The algorithm that we have proposed and implemented uses simulated annealing in order to find a satisfactory area that meets the above-mentioned criteria. It is based on optimization of terrain based on calculating a difference of the height of a particular point in the area analyzed and the average height of all points in this terrain. Figure 4 shows an example of an automatically generated town area.

Fig. 4 An automatically generated town (bird's-eye view). The appropriate area was found by utilizing simulated annealing

The proposed concept of automatic game generation is not limited to the domain of education and educational games—despite the educational nature of all examples in this paper. Proposed mechanism enables to create any type of RPG according to the model described in Sect. 3, since the explained model based on relationships between game objects is flexible enough to express almost any action that can be realized by the RPG genre of computer games. Moreover, with the addition of *quiz questions* (see below), the scope of the proposed concept grows even wider.

4.2 An example scenario

Let us suppose that a chemistry teacher wants to teach his students one of the many practical ways that sulfuric acid

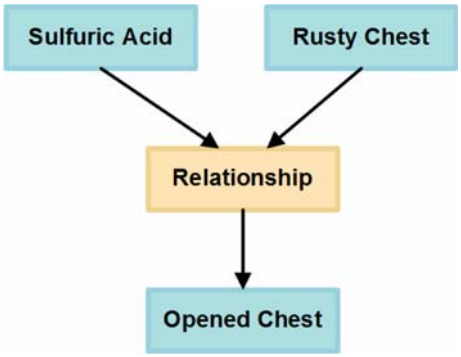

Fig. 5 Relationship describing the fact that pouring sulfuric acid on a rusty chest with an old lock unlocks (and opens) the chest

is being used (e.g., to remove rust from old materials), by creating a game in which players have to find a creative way how to open an old lock on a rusty chest. The process of creating such an educational game using S.M.I.L.E. a system that we have developed in order to validate the proposed concept of automatic educational games generation, consists of the following steps:

1. **Start the *Game Editor***. Upon logging into S.M.I.L.E., the teacher launches the *Game Editor*.
2. **Define game objects**. Next, the teacher creates a new game object representing sulfuric acid by specifying the name of the object, a short description, the object's picture and other (optional) information, such as the object's relative weight (players cannot carry many heavy game objects at the same time) and how long players are able to carry that particular object (i.e., a lit match can be carried for only a moment since it burns out quickly). The teacher may also specify places from where players can obtain the newly created object (such as map locations or specific NPCs).
3. **Define relationships and quiz questions**. After having created game objects representing sulfuric acid and other needed objects (a rusty chest with an old lock—both locked and opened), the teacher creates a relationship among these objects. In this relationship, the teacher specifies "Sulfuric Acid" and the locked "Rusty Chest" as input (by selecting them from a list of available game objects) and an "Opened Chest" as the relationship's output. The described relationship (see Fig. 5) indicates that players can get an old lock on a rusty chest open by using sulfuric acid on the stuck lock. More formally, the described relationship states that combining together game objects representing sulfuric acid and a rusty chest with an old lock results in an unlocked (and opened) chest.
   Additionally, the teacher may come up with an alternative way of describing how other players can open the same old lock on a rusty chest, i.e., different inputs that lead to the same output. For example the chest can be opened by finding a strong NPC that will open the chest by force. Such way can be described by a relationship that takes the locked "Rusty Chest" as input, an "Opened Chest" as output and can be carried out by a character with great strength, for example an NPC called "Tom Strong." Having defined an alternative way to open a rusty lock for his students, the teacher may also create a new *quiz question* (such as "How was sulfuric acid once known?" with a number of possible answers—the correct one being "Oil of Vitriol") and associate it with the second created relationship.
4. **Generate quests**. In the next step, the teacher has the S.M.I.L.E. system generate a new quest involving sulfuric acid and the rusty chest with an old lock by choosing these two game objects from all available game objects and selecting the desired difficulty of the generated quest. The teacher is afterwards shown a visual representation (see quest graph in Fig. 6) of the newly generated quest next to additional information regarding the quest, such as detailed information about every interaction contained in the quest.
5. **Generate a game**. After having saved the generated quest, the teacher chooses to generate a new game based on this quest (and other previously created quests that he has specified). Teachers have the option to specify additional options for the new game, such as the climate and weather based on the selected calendar season.
6. **Start the game**. The newly generated game is automatically added to the *Game Server* and can from now on be played by players.

Let us now suppose that a player, i.e., a student, wants to play an educational game consisting of the quest that his teacher had created above. An example showing how a player completes the teacher's quest is described below. The player will follow these steps—all depicted in Fig. 7:

1. Upon logging into the S.M.I.L.E. system and selecting a game from the list of currently running games on the *Game Server*, the player launches the *Game Client*.
2. After successfully connecting to the selected game, the player can fully explore the three-dimensional interactive game world. Eventually, the player will meet an NPC who gives him a rusty chest with an old lock and asks him to get that chest open in any way—see Fig. 7a. If the player accepts the offer, the above-mentioned quest is started and the rusty chest appears in the player's inventory (as shown in Fig. 7b).
3. The player must somehow get sulfuric acid with which he is able to open the old lock on the rusty chest. One way of getting sulfuric acid is to mix sulfur trioxide with

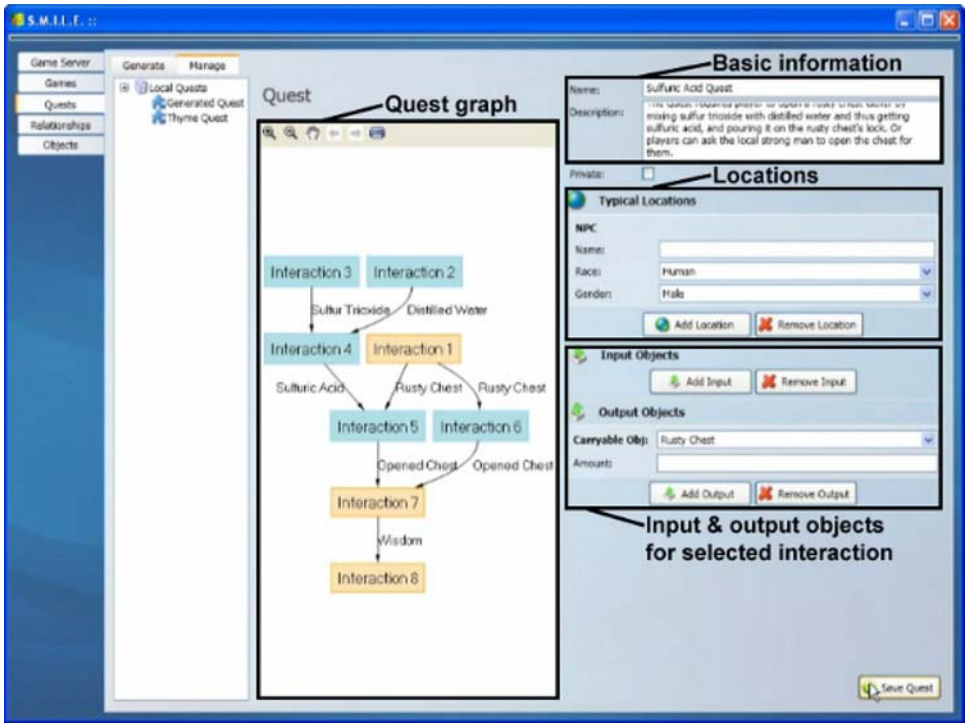

Fig. 6 Visual representation of a sample quest. The interface contains basic information (e.g., a short textual description of the quest) and a graph displaying all interactions that players must successfully carry out in order to complete the quest (required interactions are colored *yellow*, whereas optional ones are *blue*). When the teacher clicks on any interaction, more details appear (e.g., in which in-game location the interaction takes place and which game objects are involved in the interaction)

distilled water (assuming that a teacher had also defined a relationship with sulfur trioxide and distilled water as an input, and sulfuric acid as output). If the game object representing sulfur trioxide had a NPC set as its location, then the player would encounter an NPC, e.g., a chemist shown in Fig. 7c who gives him a dose of sulfur trioxide—see Fig. 7d. Similarly, if the game object representing distilled water had a water distiller set as its location, the player could get a sample from one, as seen in Fig. 7e.

4. Having obtained both distilled water and sulfur trioxide, the player can mix these two items together (see Fig. 7f), respecting a specific ratio (if set by the teacher in the particular relationship)—resulting in successfully obtaining sulfuric acid.
5. After obtaining sulfuric acid, the player can pour it on the rusty chest (see Fig. 7g), thus opening the chest's old lock, receiving an opened chest in the player's inventory (see Fig. 7h), successfully solving the teacher's quest and—most importantly—learning in a enjoyable and entertaining way how to practically use sulfuric acid.

Since the described quest has several possible solutions due to the fact that the teacher could have previously specified an alternative way of getting the rusty chest's old lock open, the player can also find a strong NPC that will first ask him a quiz question (chosen from a number of questions specified by the teacher according to the player's estimated level of knowledge—see Sect. 6.3), and open the rusty chest by force in case the player successfully answers the quiz question.

5 Proposed concept realization

We have experimented with the proposed educational games model by developing the S.M.I.L.E. system. An overview of the system is depicted in Fig. 8.

It is based on client-server architecture and consists of the following modules.

Knowledge Editing. The creation or import of educational sources is done with the assistance of the *Knowledge Editor*, which is a full-featured WYSIWYG editor designed to ease the process of creating educational content for the generated games. One of the most assistive functions of the editor is the feature to automatically search the Internet for multimedia elements (pictures and audio files—see Fig. 9) related to the topic of the currently edited knowledge. Chosen elements can be easily inserted (respecting copyright issues) into the contents of the current learning object by drag 'n' drop. The application utilizes specialized Internet search engines—Picsearch (available at http://www.picsearch.com) for pictures and photos, and FindSounds (available at http://www.findsounds.com) for audio files.

Game Editing. Teachers can collaboratively edit, share and use the stored learning objects for teaching. Most impor-

(a) The player meets an NPC that gives him a rusty chest with an old lock and asks him to get that chest open in any way.

(b) The rusty chest with an old lock appears in the player's inventory.

(c) The player encounters an NPC being a chemist who gives him a dose of sulfur trioxide.

(d) The player's inventory now contains a dose of sulfur trioxide, in addition to the rusty chest.

(e) The player gets a sample of distilled water from a water distiller.

(f) The player mixes distilled water together with sulfur trioxide, respecting a ratio set by the teacher.

(g) The player pours sulfuric acid on the rusty chest, resulting in unlocking the chest's old lock.

(h) The player receives an unlocked (and opened) chest in his inventory.

Fig. 7 The process of playing an example teacher-created educational game by a learner. **a** The player meets an NPC that gives him a rusty chest with an old lock and asks him to get that chest open in any way. **b** The rusty chest with an old lock appears in the player's inventory. **c** The player encounters an NPC being a chemist who gives him a dose of sulfur trioxide. **d** The player's inventory now contains a dose of sulfur trioxide, in addition to the rusty chest. **e** The player gets a sample of distilled water from a water distiller. **f** The player mixes distilled water together with sulfur trioxide, respecting a ratio set by the teacher. **g** The player pours sulfuric acid on the rusty chest, resulting in unlocking the chest's old lock. **h** The player receives an unlocked (and opened) chest in his inventory

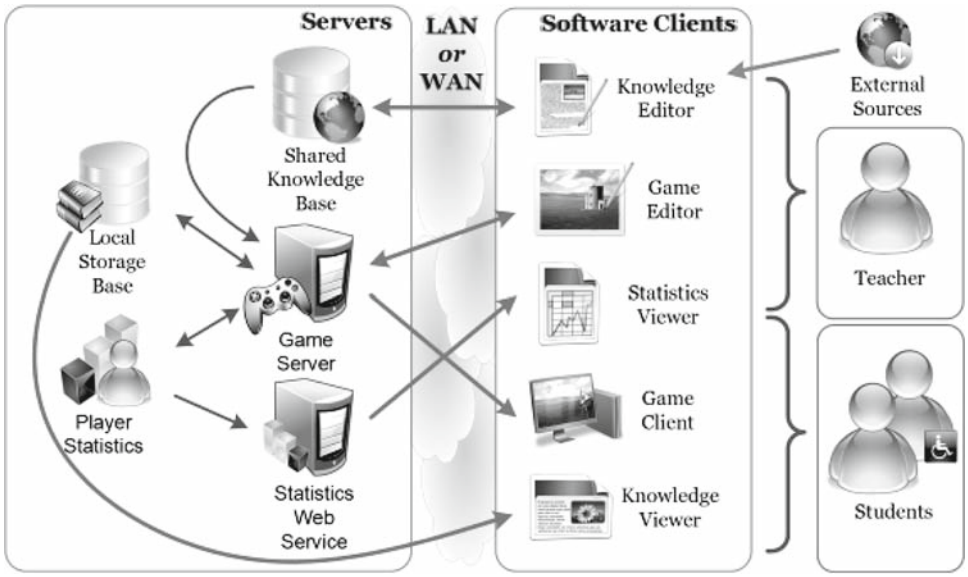

Fig. 8 Overview of the S.M.I.L.E. system

Fig. 9 Automatic context-sensitive Internet search inside the *Knowledge Editor*. The content author (teacher) can start search which gives results from either a picture or sound Internet search engine (based on his intentions). The searched keywords are filled in automatically according to the title of the edited educational content

tantly, teachers are presented with the ability to create their own educational games based on these learning objects via the *Game Editor* (as described in Sect. 4).

Game Playing. All games created by the S.M.I.L.E. system are played on the *Game Server*. Its role is to coordinate and control the actions of players (i.e., synchronization of their movement on the maps—since one game can be played simultaneously by many players, adherence of the game logic according to the scenario as stored in the *Local Storage Base*, gathering of statistical data from gameplay, etc.). The *Game Server* also encapsulates an interface that enables handicapped players to play along with their able-bodied colleagues.

Data Storage. All created games including the necessary maps, textures, models and sounds are stored in the *Game Server*, whereas all learning objects, quests and game objects along with their relationships and additional objects necessary for actual gameplay are stored in the *Local Storage Base*, to where educational content is copied from the *Shared*

Knowledge Base by the teachers' version of the *Knowledge Editor*. This mechanism not only reduces requests to the *Shared Knowledge Base,* which can become a bottleneck of the whole system, but also ensures that the necessary educational content is accessible even when connectivity to the *Shared Knowledge Base* gets interrupted.

Game Types. Learners have the option to learn by playing both games created by their teachers (which are based on quests either directly chosen by teachers, or on quests adaptively selected according to parameters set by the teachers), and also by playing in a persistent virtual world consisting of all quests that are stored in the system and marked *public* by their authors (as described in Sect. 3.1).

Knowledge Viewing. To ease the process of learning, learners can browse and view all learning objects available in the *Local Storage Base* via the *Knowledge Viewer*, which formats the content of displayed learning objects according to every learner's preference (as described in Sect. 3.2). The viewer also allows learners to search for available learning objects (and within knowledge represented by them) via keywords or by browsing through the educational content that is tied to courses they attend. The *Knowledge Viewer* is also accessible in-game, so learners can view the related educational content without interrupting the gameplay.

Statistics. The *Statistics Web Service* gathers information about each learner's in-game performance (e.g., which quests the learner successfully completed and which quests he did not—along with the number of failed attempts and/or the amount of time the completion took). Thanks to the facts that S.M.I.L.E. registers the associations between learning objects and quests that are being solved by players, and that learning objects are classified into domains such as mathematics, chemistry or biology, the system estimates the level of knowledge individually for each and every player (see Sect. 6.3). These statistics can be viewed by the *Statistics Viewer* (or by a web browser) and can be helpful not just for teachers, but also for learners and their parents.

6 Considering different learners

The proposed concept of educational games was designed with both adaptability and adaptivity in mind [10]. Speaking about *adaptivity*, we point out that the system attempts to be different for different learners and groups of learners by taking into account information accumulated in learners' or groups' characteristics [11], such as which educational games (and quests therein) an individual or a group have (or have not) successfully completed, including all recorded mistakes and failed attempts. Based on this data, it is possible to adapt all generated educational games and quests to every learner (i.e., we can navigate players into solving either difficult or easy quests, based on their previous progress of solving the generated games and quests). Previously solved in-game tasks are continuously re-tested by utilizing the Item Response Theory [12].

Speaking about *adaptability*, we point out that the system can be adapted by every learner according to his needs and abilities [11] (in our case mostly related to a possible handicap of users), since users are able to fully customize the appearance of all (including in-game) user interfaces, including the appearance of all viewed educational content (also including content viewed directly in-game). A set of predefined customization presets is already available for users to choose from. Moreover, these presets can be further customized by users.

To provide interfaces for all learners, especially for handicapped, we designed the educational games with the possibility to be fully controlled and interacted with via voice recognition and also with peripherals for the handicapped, such as Braille keyboards. Handicapped learners may also use the keypad of their mobile phone or PDA as an input device. All these interfaces enable handicapped learners to fully control and interact with the educational games.

6.1 Diversity in vision

Learners with a visual handicap control all interfaces by means of voice commands and have the educational content read to them using a computer-synthesized voice. Moreover, such learners have all texts shown as large as they prefer.

Games can be played by means of voice-activated commands using a version of the *Game Client* specialized for blind and vision-impaired users. It reads all available in-game actions to users via computer voice synthesis and lets them speak which actions to be performed by the system. To other players, a blind player looks just like any other player, with the exception that he moves according to waypoints automatically placed when the educational game is created.

Players with minor vision impairment can customize the visual appearance of important in-game elements, such as NPCs—as shown in Fig. 10.

Such players able to fully customize the appearance of not only educational content, but also all user interface components (see Fig. 11).

6.2 Diversity in hearing

Similarly to how the S.M.I.L.E. system provides adapted interfaces for vision-impaired learners, it also supports specialized interfaces for hearing-impaired learners. The standard interface used is purely visual, and so the design does not need to be altered in order to provide support for such learners. However, the only exception to this rule is the playback

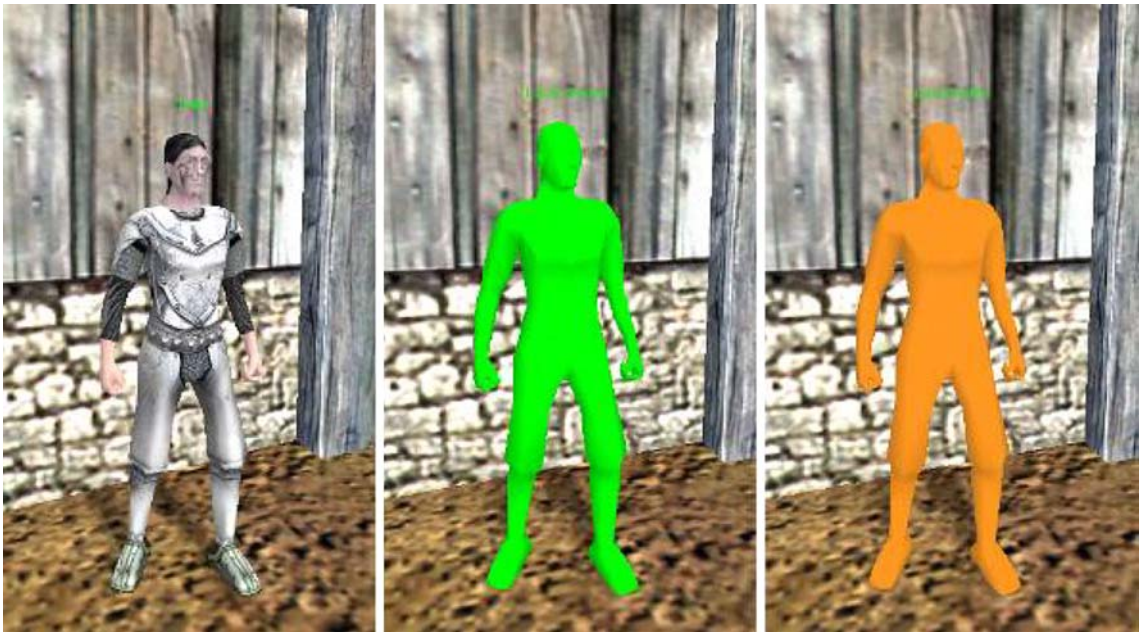

Fig. 10 Adaptation of NPCs for visually impaired players

Fig. 11 Adaptability of user interfaces for various visual disabilities. The *leftmost* pair of screenshots shows the default scheme (*black text on a white background*), whereas the pair in the center shows a scheme for visually impaired learners (*white text on a black background*, plus a 200% zoom factor). The *rightmost* pair of screenshots shows a scheme for colorblind learners (*blue text on a yellow background*)

of videos stored inside learning objects. Therefore, hearing-impaired learners can choose to have subtitles shown below all played videos.

On the other hand, all created educational games utilize environmental and character sounds in their (otherwise visual) interfaces. Deaf players and players with a hearing impairment use interfaces that contain extended graphical effects instead of sounds of any kind. In case a sound should be played, such players are shown a graphical image associated to the particular sound. Moreover, the image is displayed on-screen in the exact spot from where the sound came from, thus enabling deaf and hearing-impaired players to perceive all audible events visually.

6.3 Diversity in level of knowledge

Every quest (whether it was manually designed by a teacher or automatically created according to his preferences by the S.M.I.L.E system) is based on a set of learning objects, since each game object covered in the quest is associated to a particular knowledge to be learned (i.e., a concept). In addition, each learning object is related to at least one specific domain or field (such as mathematics or chemistry). If the player successfully solves (or is unable to solve) a quest related to several domains, we increase (decrease) the estimated level of his skills in all domains accordingly. Having all this in mind, it is possible to classify all quests according to domains that

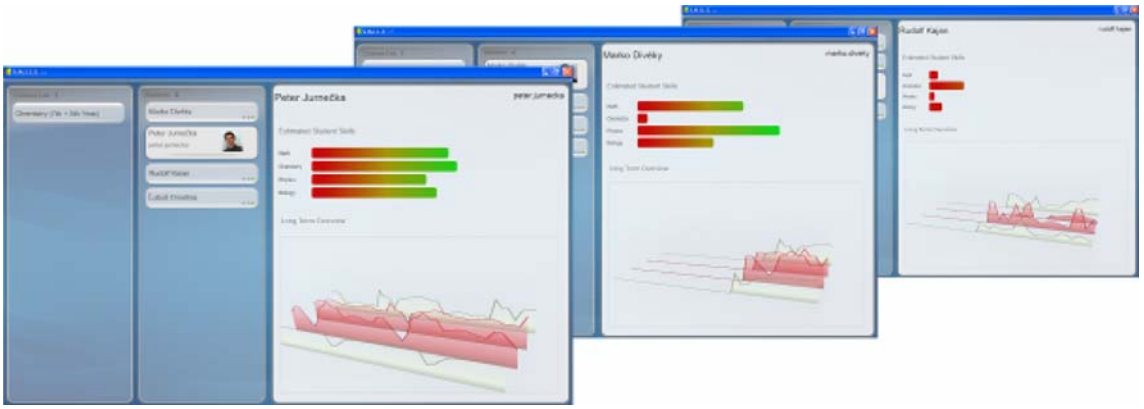

Fig. 12 Students' estimated level of knowledge at domains. Graphs in the screenshots show every student's currently estimated level of knowledge for each defined domain separately (in percentages, i.e., a rating of 100% means that the student has successfully solved all quests related to the particular domain). The timelines show the progression of students' estimated values of knowledge level related to particular topics in time and separately for each domain

they are related to, and estimate how good each player is at each domain based on how he managed to solve particular quests—see Fig. 12.

Thanks to the fact that we estimate the level of knowledge of each domain for every player and the fact that each quest is classified according to domain it is related to, all educational games are adapted individually to every player's (estimated) level of knowledge and skills. Players are dynamically navigated into solving quests (e.g., by pointing players at map locations of the in-game NPCs that assign them) most suitable for their estimated knowledge level. Additionally, while solving quests, options of the appropriate difficulty are chosen for players to solve (e.g., if a player has his estimated knowledge of physics low, he will be asked a simpler quiz question regarding physics in order to solve the quest).

By utilizing the Item Response Theory [12], we are able to predict how players will react to both tasks and questions they are given while solving quests, and thus to measure their level of forgetfulness by testing them repeatedly on quests and quiz questions they have previously successfully completed and answered.

6.4 Diversity in skills

We have also taken the diversity of players' skills into account. Therefore, we let each and every player choose how he interacts with the educational games. For example, players who wish to interact with a standard keyboard have the option to specify their own controls. Naturally, players can also choose from a number of predefined control schemes and can customize these schemes according to their liking.

6.5 Diversity in style

Every player is represented in the game world by an *avatar*—a virtual character according to which other players notice his presence. Thus, the option to personalize every player's avatar plays a crucial role in getting the players immersed into the gameplay and accepting the avatar as the player's own in-game identity.

The first step a player does in order to have his avatar personalized is to choose a specific geometry model. The S.M.I.L.E. system offers a variety of standard models to choose from and, alternatively, players have the option to import their own geometry models from external modeling tools. The chosen model can be further customized, e.g., players can specify its relative height by scaling the desired model.

Naturally, creating and importing a custom geometry model is rather difficult and cumbersome. Therefore, we have enabled players to assign different textures to various parts of their avatar's model, what proved as an effective and amusing method in differentiating players' avatars. Each geometry model, representing the body of an avatar, has distinct zones representing various body parts, e.g., a head, torso, hands and feet. Players have the option to assign a desired texture (clothing) to each zone (see Fig. 13).

A great advantage of this approach is the fact that the number of possible different avatars created grows exponentially according to the number of available geometry models and textures. For example, given the amount of 5 models, 6 custom-clothed zones per model and 10 textures per each zone, we get a total number of 5 million (5×10^6) different avatars. Moreover, giving access to new avatar textures and models every time a player successfully solves all quests in an educational game is a great motivation for players to keep playing educational games created by the S.M.I.L.E. system.

The association of an in-game avatar with the player it represents is determined by a unique ID, such as the player's login name or nickname. This identifier is displayed next to every avatar, which enables players to correctly identify

Fig. 13 Customization of a female human avatar

other players standing next to them in the virtual world of educational games.

Additionally, a unique feature of our S.M.I.L.E. system is the possibility of players to give their avatars their own faces via textures created from the players' photographs (specified in a special format that allows automatic mapping of significant face elements). The ability to see other players' faces on their in-game avatars brings a great amount of reality into games within the S.M.I.L.E. system, and also enables players to identify other players in-game more easily. This feature is especially important and useful for educational purposes (to the opposite it may not be desirable in many cases for standard computer games where players usually do not want their identity revealed).

7 Conclusions and future work

The main contribution of the presented work is in devising a concept that allows teachers themselves—without requiring any knowledge of programming—to define three-dimensional, adaptive, multiplayer and multimedia educational games, which are automatically generated based on teachers' preferences. This process undoubtedly requires some knowledge and skills, but these are easier to master than programming skills and game design techniques.

Moreover, the proposed novel concept can be adapted to different learners and different abilities of a particular learner. User interfaces are designed adaptable such that they support handicapped learners.

In order to prototype the presented concept, we have developed a system called S.M.I.L.E. that encapsulates and combines the advantages of two distinct worlds—interactive educational content and popular computer games. The system also enables handicapped learners to learn together with their non-disabled colleagues. Such a concept has not yet been realized to our knowledge in any of the existing applications.

We have positive feedback from both teachers and students at elementary and secondary schools, where we informally validated our concept of automatic generation of educational games by having teachers successfully generate educational games (what they were not able to do up until now) for their students, who afterwards played these games together.

The content that we have developed is targeted primarily at the most problematic group of learners—students at the age of 10–16 years. Teachers often find it challenging trying to involve such pupils into the educational process. We focus on all learners—even handicapped ones. Specialized interfaces enable handicapped learners to participate in the educational process by playing educational games with their non-disabled friends, and thus socialize with the community.

Our concept of educating by games is applicable for any age group as game landscape and environments, game objects, and quests are generated according to actual data set in advance (e.g., Middle Ages environment for teenagers). Changing the focus to a different age group requires considerable effort (e.g., models of objects in the game environments should be replaced), but all hereby described mechanisms realizing game generation and adaptivity remain the same. The proposed concept is not limited to educational games, since it can be used to generate any type of computer RPGs.

We work on improving adaptivity of the generated educational games by automatic estimation of user characteristics based not only on results of tests, but also on overall behavior [13] together with maintenance of user characteristics in time [14]. We also plan to use the demographical data collected by the *Statistics Web Service* to map and compare the effectiveness of learning with the S.M.I.L.E. system depending on the demographical location of its users. In addition, we plan to validate the effectiveness of various strategies used for personalization, and also to explore the possibilities of adding interactive storytelling [15] techniques in order to achieve a more compelling storyline in the generated educational games.

Acknowledgments This work was partially supported by the Cultural and Educational Grant Agency of the Slovak Republic, grant KEGA 3/5187/07, and a software solution was developed within the Software Design category of the Imagine Cup 2007 competition.

References

1. Zyda, M.: Creating a science of games. Commun. ACM **50**, 27–29 (2007)
2. Dryden, G., Vos, J.: The Learning Revolution. Jalmar Press, Austin (1999)
3. Vorderer, P., Hartmann, T., Klimmt, C.: Explaining the enjoyment of playing video games: the role of competition. In: Proceedings of the Second International Conference on Entertainment Computing, Pittsburgh, 2003, pp. 1–9. ACM, New York (2003)
4. Brusilovsky, P., Peylo, C.: Adaptive and intelligent web-based educational systems. Int. J. Artif. Intell. Educ. **13**, 59–172 (2003)
5. Gee, J.P.: What Video Games Have to Teach Us About Learning and Literacy. Palgrave Macmillan, Basingstoke (2004)
6. Shaffer, D.W., Gee, J.P.: How Computer Games Help Children Learn. Palgrave Macmillan, Basingstoke (2006)
7. Bendis, J.E.: Developing educational virtual worlds with game engines. In: Proceedings of the 34th International Conference and Exhibition on Computer Graphics and Interactive Techniques, SIGGRAPH 2007, San Diego (2007)
8. Tychsen, A.: Role playing games: comparative analysis across two media platforms. In: Proceedings of the 3rd Australasian Conference on Interactive Entertainment, pp. 75–82. ACM, New York (2006)
9. Fallon, C., Brown, S.: E-Learning standards: a guide to purchasing, developing, and deploying standards-conformant e-learning. CRC Press, Boca Raton (2002)
10. Bieliková, M., Divéky, M., Jurnečka, P., Kajan, R., Omelina, L.: Learning with smart multipurpose interactive learning environment. In: Kendall, M., Samways, B. (eds.) Learning to Live in the Knowledge Society, 20th World Computer Congress, International Federation for Information Processing, vol. 281, pp. 101–104. Springer, Berlin (2008)
11. Brusilovsky, P., Maybury, M.T.: From adaptive hypermedia to adaptive Web. In: Brusilovsky, P., Maybury, M.T. (eds.) Commun. ACM, Special Issue on the Adaptive Web **45**(5), 31–33 (2002)
12. Baker, F.: The Basics of Item Response Theory. ERIC Clearinghouse on Assessment and Evaluation, University of Maryland, College Park (2001)
13. Barla, M., Bieliková, M.: Estimation of User Characteristics Using Rule-based Analysis of User Logs. In: UM 2007. Data Mining for User Modeling, 11th International Conference on User Modeling, pp. 5–14, Corfu, Greece (2007)
14. Šimún, M., Andrejko, A, Bieliková, M.: maintenance of learner's characteristics by spreading a change. In: Kendall, M., Samways, B. (eds.) Learning to Live in the Knowledge Society, 20th World Computer Congress, International Federation for Information Processing, vol. 281, pp. 223–226. Springer, Berlin (2008)
15. Charles, F., Mead, S.J., Cavazza, M.: Interactive storytelling: from computer games to interactive stories. Electron. Libr. **20**, 103–112 (2002)