Title	A Framework for Personalised Learning-Plan Recommendations in Game-Based Learning
Author(s)	Hulpus, Ioana; Hayes, Conor
Publication Date	2014
Publication Information	Ioana Hulpus, Conor Hayes, Manuel Oliveira Fradinho (2014) 'A Framework for Personalised Learning-Plan Recommendations in Game-Based Learning' In: Nikos Manouselis, Hendrik Drachsler, Katrien Verbert, Olga C. Santos(Eds.). Recommender Systems for Technology Enhanced Learning. New York : Springer.
Publisher	Springer
Link to publisher's version	http://dx.doi.org/10.1007/978-1-4939-0530-0_5
Item record	http://link.springer.com/chapter/10.1007%2F978-1-4939-0530-0_5; http://hdl.handle.net/10379/4524
DOI	http://dx.doi.org/10.1007/978-1-4939-0530-0_5

A Framework for Personalised Learning-Plan Recommendations in Game-Based Learning

Ioana Hulpuş, Conor Hayes and Manuel Oliveira Fradinho

Abstract Personalised recommender systems receive growing attention from researchers of technology enhanced learning. The learning domain has a great need for personalisation as there is a general consensus that instructional material should be adapted to the knowledge, needs and abilities of learners. At the same time, the increase in the interest in game-based learning opens great opportunities for learning material personalisation, due to the complexity of life situations that can be simulated in serious games environments. In this paper, we present a model for competency development using serious games, which is supported by case-based learning-plan recommendation. While case-based reasoning has been used before for recommending learning objects, this work goes beyond current state-of-the-art, by recommending learning plans in a two-step hierarchical case-based planning strategy. First of all, several abstract plans are retrieved, personalised and recommended to the learner. In the second stage, the chosen plan is incrementally instantiated as the learner engages with the learning material. We also suggest how several learning strategies that resonate with a game-based learning environment, can drive the adaptation of learning material.

1 Introduction

Serious games for educational purposes have a number of potential advantages over more traditional learning methods and on-the-job training. Game-based learning is

Ioana Hulpuş
INSIGHT Centre for Data Analytics, NUI Galway, Ireland, e-mail: ioana.hulpus@insight-centre.org

Conor Hayes
College of Engineering and Informatics, NUI Galway, Ireland, e-mail: conor.hayes@nuigalway.ie

Manuel Oliveira Fradinho
SINTEF e-mail: manuel.oliveira@sintef.no

consistent with constructivist learning theories [42], which emphasize that learning is active and knowledge is built on top of own experiences. Serious games include tolerance and encouragement of risk within a safe environment [24], thus promoting and encouraging experimentation instead of passive learning [27, 13]. They can support learning that is active, experiential, situated, problem and inquiry-based, and they provide immediate feedback. They also involve communities of practice which provide collaborative support to learners [8].

Evidence for their efficacy as educational tools is growing with a growing number of research studies finding improved rates of learning and retention for serious games compared with more traditional learning methods [16, 10, 11]. A very recent literature review on empirical evidence of serious games [11] found that students enjoy the game-based approach and found it motivating. The same study shows that games can be used with success for both behavioural change and cognitive acquisition, but their benefits vary.

Like for all learning activities, the learning objects have to be designed and selected according to pedagogical foundations and the learning experience must be personalised in order to avoid the "one-size-fits-all" learning environment. In these paper, we are dealing with these two aspects of game-based-learning, namely: (i) the recommendation of learning paths that support the learner towards achievement of target competences, and (ii) personalisation through constant performance assessment and on-the-fly adaptation of the learning paths. The main contribution of this paper is twofold: (i) we research how case-based planning can be used for recommending personalised learning plans and (ii) we translate TEL recommendations from hypermedia to serious games, and exploit game adaptation strategies in accordance with the variation learning theory.

We propose a case-based approach to the generation of learning plans and game scenarios. While Case-Based Reasoning (CBR) has proven to yield good results for the adoption of on-line tutoring systems, the planning potential of CBR has yet to be exploited in relation to the creation of learning plans. We research how the game-based learning process can be continuously adapted towards the efficient development of required competencies for each individual learner. This entails the use of a planner that collects feedback from the learner interaction with the suggested plans and uses this feedback to learn which parts of the plan are failing and how to recover. As described by Hammond [21], case-based planning (CBP) systems have the capability of learning from the interaction with the human users. They can also anticipate problems and learn how to avoid them. These features make CBP attractive for use in learning environments.

We also research how alternative plans which target the same goals can be represented, and retrieved based on their outcomes for different learners. The retrieved plans are then incrementally instantiated during execution, taking into account the information derived from constant performance monitoring.

The objective of this article is to present a coherent framework for an on-the-fly adaptive planning system for game-based learning. We do not address narrative theory or specific game design. Rather, the focus is on providing a generic model of the atomic learning components for an adaptive personalised planner where compe-

tencies are taught through game-based learning strategies. At all times, we attempt to justify our models with reference to current learning theory and state-of-the-art techniques in case-based planning and personalisation.

This research was carried out within the context of the TARGET[1] European Project. TARGET's goal is to implement a new type of Technology Enhanced Learning(TEL) environment which shortens the time to competence for learners within enterprises. As such, the examples in this paper refer to learning topics such as innovation and project management.

The remaining of this paper is structured as follows. The next section summarises related work in personalised recommender systems for learning and related work that leverages the use of CBR for the purpose of supporting human learning. Section 3 presents an overview of our research direction in the field of CBP and learning with serious games, and in Section 4, we show how some modern learning theories can drive the planning and story personalisation. In Section 5 we provide a detailed description of our model of hierarchical CBP for personalised learning plans. Section 6 presents a discussion of the implications of our system and some remarks on future work, and then we conclude in Section 7.

2 Background

2.1 Personalised Recommendations in Technology Enhanced Learning

There has been a growing interest in applying recommendation techniques from the e-commerce domain to that of technology enhanced learning. However, for efficient learning, non-technical particularities of the e-learning domain must be considered. Previous work like [14, 32, 41] raises awareness of the differences between personalised recommendations in e-commerce and e-learning. The main guidelines for personalisation stress the importance of considering (i)the learning goal, (ii)prior knowledge of the learner, (iii)the learner's model, (iv)groups of similar learners, (v)rated learning activities, (vi)emerging learning paths and (vii)pedagogical strategies for learning. Our work considers all these aspects, and focuses on how the aspects captured by points (i)-(v) can be used to identify and reuse (vi) - emerging learning paths -, by incorporating (vii) - pedagogical strategies - into the adaptive, personalised recommendation process.

TEL Recommender systems belong to three main categories, similar to their e-commerce ancestors: content-based, collaborative filtering, and hybrid [32]. While TEL recommenders extend classical recommenders by considering the pedagogical needs of the learners rather than only the preferences, they are still subject to drawbacks like: cold-start problem (new learner or new item) and data sparsity [2]. Hybrid recommenders usually overcome part of these problems. In our work, we

[1] http://www.reachyourtarget.org

consider a hybrid approach, combining collaborative filtering techniques with case-based reasoning(CBR) recommenders, a type of content-based approaches [47]. However, in this paper, the main focus lies on the case-based recommendation component. Next section analyses in more detail the related work on CBR and human learning.

Another aspect of interest regarding TEL recommender systems, is the type of recommended material. Most of the works focus on recommending individual learning objects, but in [15], the authors acknowledge the need of suggesting learning paths, where the sequence of the recommended material guides the learner towards achieving his goals. In [26], the authors suggest an adaptive learning plan generator for educational hypermedia. Several alternative paths are generated by analysing the domain concepts ontology and the relations between educational resources, for example, prerequisites. The personalised learning plan is created by computing the suitability of each path to the learner's profile. This work is similar to ours as it uses several abstraction layers in order to extract and personalise learning plans. However, it does not use any machine learning mechanism, therefore constant usage of the system will not improve its performance. More over, as opposed to our work, this model does not consider on-the-fly assessment and adaptation. Nevertheless, the work presented in [26] can be considered complimentary to ours and can be used in a hybrid system as a backup to the CBP approach, which 1) generates learning plans in case of cold-start problems like *data sparsity* and *new item*, 2) validates the ordering of learning resources, and 3) avoids the so-called "conceptual holes" [26] from the learning path.

2.2 Case-Based Reasoning and Human Learning

Case-Based Reasoning (CBR) is an artificial intelligence paradigm that involves reasoning from prior experiences: it retains a memory of previous problems and their solutions and solves new problems by reference to that knowledge. This is the main difference from rule-based reasoning systems, that normally rely on general knowledge of a problem domain, and tend to solve problems from scratch or from first principles. Usually, the case-based reasoner is presented with a problem (the current case). In order to solve it, the reasoner searches its memory of past cases (the case base) to find and retrieve cases that most closely match the current case, by using similarity metrics. When a retrieved case is not identical to the current case, an adaptation phase occurs. During this phase, the retrieved case is modified, taking the differences into account [37]. Finally, the cases are retained in the case base for future use. These four steps are defined by Aamodt and Plaza [1] as Retrieve, Reuse, Revise, and Retain.

Developed from CBR, case-based planning (CBP) systems address problems that are represented by goals and have solutions that are plans. Like traditional case-based reasoners, CBP systems build new cases out of old ones. Unlike CBR systems, CBP systems put emphasis on the prediction of problems: when encountering a

new plan, CBP systems anticipate the problems that can arise and find alternative plans to avoid the problems. Plans are indexed by the goals satisfied and problems avoided [21].

CBR for human learning purposes has been a topic of study for a number of years, with significant developments in the fields of intelligent tutoring systems and adaptive hypermedia. The appeal of a CBR approach is partly due to its roots in cognitive science which focuses on modeling human problem-solving behaviour [39]. There are many examples in the literature of day-to-day human reasoning and planning that highlight the important role of previously experienced situations and of analogy in human problem solving [43, 44, 29]. In [44], Schank argues for a goal-based approach to education, in which case acquisition plays a central role. In [29], Kolodner suggests how CBR can enhance problem-based learning by recommending relevant problems to learners. In both goal-based and problem-based learning, learning occurs in the context of attempting to achieve a mission or find a result.

The research done by Jonassen and Hernandez-Serrano [25], also supports the use of CBR for instructional design based on problem solving. The authors argue for a story-based system supported by CBR that would enable learning from other people's experiences. These experiences form a case library of narratives from employees that describe real-life work-related problems and their solutions. Each experience must have a particular lesson to be learned that the user can reference in a similar situation. This idea resonates well with a game-based learning environment where learners' experiences are saved in common accessible repository. However, we are focused on how to create a suitable learning plan, rather than on how to retrieve a similar narrative.

The ILMDA (Intelligent Learning Material Delivery Agent), designed by Soh and Blank [51] focuses on the learning domain of computer science of undergraduates. It combines CBR with system meta-learning that demonstrating that a detailed analysis and adaptation of the learning process can be used to improve students' results. An approach that comes closer to serious games is presented by Gomez-Martin et al. [18, 19]. They present a metaphorical simulation of the Java Virtual Machine to help students learn Java language compilation and reinforce their understanding of object-oriented programming concepts. Unlike these two systems, where the problems have direct mapping to the correct solution and the targeted domains are well defined, we are creating a system for use in two very complex domains: Project Management and Innovation. In these domains, the problems are open-ended and the required competences are complex and difficult to model. Therefore, our approach is to create an open environment capable of reasoning with very complex, poorly structured domain knowledge. Furthermore, we focus on long term learning goals. For this, a single learning episode is not enough; the system must design consistent and coherent *learning plans*. As such, we use a CBP approach rather than classical CBR.

However, none of these approaches use CBR as a recommendation engine. CBR can be used for recommendations as shown in [47]. Technically, the main difference we consider between a case-based reasoning system and a case-based recommender system, is that while the former imposes the top solution, the latter returns top-n

solutions and recommends them. The inner processing done to rank the results is similar in both types of systems. Previous research that considered CBR for recommending learning activities is presented in [17]. The work provides a simplified CBR model, without the revision/adaptation phase. Our model extends this work by exploiting adaptation methodologies inspired by learning theories. More over, our work also looks into recommending learning paths. In [14], the authors also consider case-based recommendations as a possible approach, and identify as a main disadvantage of the methodology the fact that "the user is limited to a pool of items that are similar to the items he already knows". We argue that this problem can be avoided by adapting the case retrieval methodology.

3 Overview

In this section, we introduce the fundamental aspects of case-based reasoning as it pertains to the support of an adaptive learning system. These ideas will be further elaborated by the model presented in Section 5.

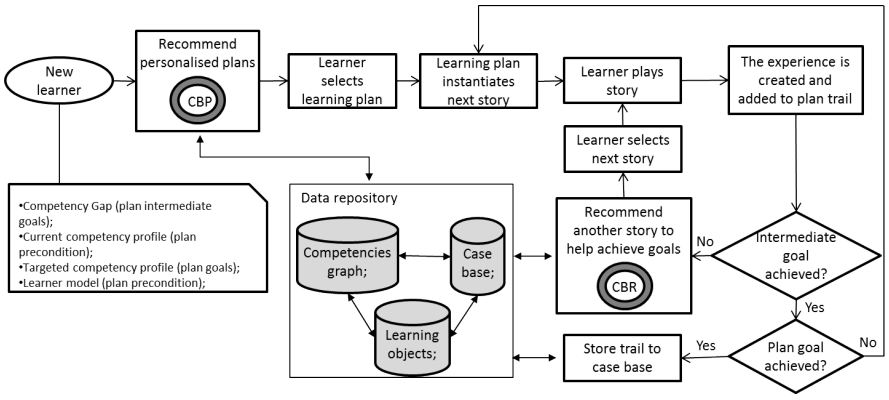

Fig. 1 Proposed CBR-supported learning process

In Figure 1, we illustrate how CBR can be incorporated into the learning process within a game-based learning environment. The core unit of instruction is represented by *stories* which are interactive narratives the learner engages with as he assumes a specific role, with a specific mission. A learning plan is composed of a personalised sequence of stories. The learning plan is created in a two stage process. First, several *abstract plan* are created and recommended, as a sequence of story templates. Then, the actual learning plan is instantiated incrementally, each story being created starting from the corresponding story template, when the plan execution thread reaches it. This process is the central focus of our paper and is described in more detail in the Section 5.

Throughout a story execution, towards the achievement of his mission, the learner is put in various *situations* meant to develop and evaluate his competencies. Each story has at its core a *story template* which can accommodate several situations. In order to create a story starting from a story template, a sequence of these potential situations is selected, based on the learner needs and requirements.

3.1 The Data Repository

In this section, we describe the repositories that contain the competency knowledge, the raw learning objects and the cases.

3.1.1 The Competencies Graph

The term competency carries many definitions in the related literature [22]. The definition that matches the best the way we use and assess competencies is that they are a set of personal characteristics, knowledge, skills and abilities that help successfully perform certain tasks, actions or functions and are relatively stable across different situations [54]. Many companies use the IPMA Competence Baseline[2] which breaks project management in 46 competences, or SHL Universal Competency Framework [3] which defines the "great eight" cross-domain competencies, in order to model and structure the competencies of their employees. The initial TARGET competency framework will use these frameworks. However, each TARGET user community will also be able to contribute to the competency framework. Thus, the set of competencies is very likely to become very versatile as the community of users represent different educational backgrounds, work history and business domains.

The competencies of an individual are characterised by a state of attainment (degree of mastery) which we call level, and which the system estimates by analysing the user's *performance*. Therefore we define a competency profile as a set of competency-level pairs. A learner in our system has assigned both a *current competency profile*, and a *targeted competency profile*.

In this work, we consider the system being deployed and used within enterprises, each TARGET instance having its own competency representation, imposed by the particular enterprise. The set of competencies is very likely to strongly differ among domains and communities. While the framework presented in this work could theoretically fit a Web-based online learning environment, an important research challenge would be to organise domain concepts and competencies so that they can deal with users of very different cultural background. One option would be the use of en-

[2] http://www.ipma.ch/certification/standards/Pages/ICBV3.aspx

[3] http://www.shl.com/OurScience/Documents/SHLUniversalCompetencyFramework.pdf

cyclopedic knowledge bases like DBpedia [4] to extract relations between concepts, and use this graph to create learning paths. Semantic Web technologies can be used to link the DBpedia concepts to the serious games annotated as learning objects metadata, according to [23]. For sake of generality, the only assumption we make about the competencies, is that from their representation, dependency (e.g. prerequisite) relations can be extracted, which would then be used to guide the learning plan creation.

3.1.2 The Learning Objects

The stories represent the personalised learning material that is generated for each learner and are part of the case base described in the next subsection. They are created starting from story templates by the TARGET game engine. These story templates, together with the competency-training situations are stored in the Learning Objects Repository. The TARGET game engine has the responsibility of creating stories from these "raw" learning objects (story templates) by selecting the required competency-training situations, player roles and level, non-player characters and narrative thread. While the TARGET game engine and its story creation mechanism are outside the scope of this paper, we give in section 3.2 a brief example of a story template and its "child" stories.

3.1.3 The Case Base

At the core of any CBR system is the case-base, which in this context brings together all the experiences created by learners using the system. In a CBR system, a case denotes a problem-solution pair. In our system, the problem is represented by the goal, preconditions and learner model, as shown in Figure 1. Still, depending on the solution, we have two kinds of cases: story cases, where the solution is a story, and plan cases where the solution is a plan. A plan is an ordered sequence of stories. Experiences are instances of stories created each time a learner plays a story, whereas trails are instances of plans, therefore sequences of experiences. Experiences and trails are used to evaluate the stories and plans respectively. On the basis of these definitions, we can formalise the case knowledge of the system as containing a set of knowledge assets with a story at the core. Each story holds references to the experiences it has seeded. The stories are interconnected into plans, which are associated with a set of trails that link together experiences. These knowledge assets have associated social data created by the community, such as feedback, ranking, peer assessment, tags, etc. Section 5 contains the details on case representation.

Other knowledge that such an integrated system would use are the social network and the game mechanics, but their description is outside the scope of this paper.

[4] http://dbpedia.org

3.2 Story Generation Toy Example

Figure 2 illustrates an example of a story template with its corresponding possible situations, and the competencies trained and evaluated in each situation. The situations are labeled with letters A-G. The arrows which lead from one situation to another show the possible flows of situations. For example, situation A "*Partner does not produce*", can lead to one or both situations B "*Conflict between partners*" and F "*Tasks not achieved or postponed*". The dashed lines in the figure illustrate the links between situations and the related competencies. For example, situation B trains and evaluates *conflict resolution*.

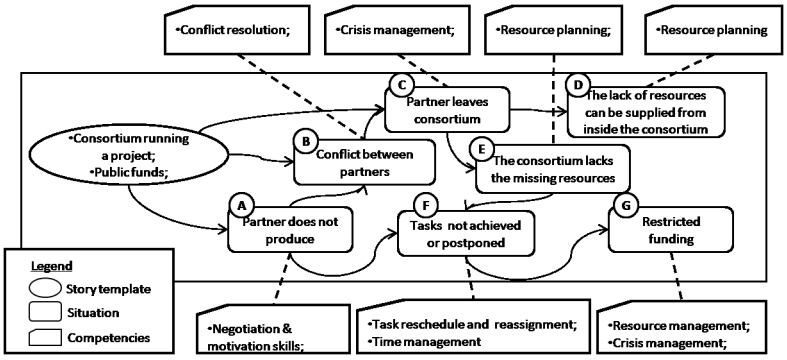

Fig. 2 Example of Story Template and Potential Situations

For each story template, an instantiated story consists of one path through its graph of situations. The game engine will instantiate the story according to the requirements of the learner as stated by the CBP module. The story instantiations consists of: (i) selection of the story situations, (ii) instantiation of story parameters. Given the example in Figure 2, we can consider a user who wants to train in conflict resolution, crisis management and resource planning. Then, a candidate story is created by switching on the situations B, C and D. To train in the required competencies, the learner chooses the role of project coordinator. During his experiences, the user is first evaluated on how he handles the conflict between the partners. Then he is evaluated on how he manages the situation where a partner leaves the consortium where other partners have sufficient resources to overcome the loss. Other candidate stories starting from the same template can be: $B \rightarrow C \rightarrow E$, or even $B \rightarrow C \rightarrow E \rightarrow F \rightarrow G$, which would suit a more experienced learner, or a learner who needs a more complex story.

Regarding the story parameters, for the given example such a parameter would be the number of partners in the consortium, the number of partners involved in the conflict, the personality of the non-player-characters. All these contribute to an easier or more complicated story. Having the set of needed competencies, the case-based planner might choose the template for the abstract plan, but in order to know

how to instantiate the story (i.e, how to choose from the three stories we described above and instantiate the story parameters), it needs to know the performance of the user within the plan. Therefore, each story is instantiated when the plan execution thread reaches it.

3.3 Overview on Plan Generation

At the start of the process, the learner decides to achieve more competencies. A case for the case-based planner is derived from the plan goal (targeted competencies), by the set of possible intermediate goals (competency gap), and the plan preconditions (the learner model and his current competencies).

Drawing on this data and on the competencies knowledge, the system uses case-based planning to generate personalised plans for the learner. From a list of recommended learning plans, the learner chooses the one he prefers. As he or she plays, an experience is generated and added to his or her trail.

Depending on the learner's performance, the system decides if the intermediate competencies have been achieved. If the learner has failed to achieve them, the case-based planner identifies the situation as a failure and tries to recover in two ways: i) the planner anticipated the problem and will have already assigned a recovery plan for a particular story. If this is the case, the planner will choose the recovery plan with highest eligibility value; ii) otherwise the planner will undergo a CBR process to recommend other stories to the learner in order to bring him or her to the required standard in relation to the intermediate competencies. This is similar to the process suggested by variation theory of learning which states that a key feature of learning involves experiencing that phenomenon in a new light [34]. When all the goals of the plan have been achieved, the trail is saved and becomes part of the case base.

The plan generation described above uses a case-based planning approach based on 4 phases.

3.3.1 Plan Retrieve

Starting with the goals and preconditions, the planner searches the case base to find plans with similar descriptions, which yielded good results for the learner. In order to do this, the system must consider different types of knowledge and reasoning methods such as similarity metrics, utility metrics, statistical reasoning and collective filtering. An important focus of research related to this phase concerns the *new-student problem*. In this situation, the system will not yet hold enough information to be able to assign a learner model to the student. In this context, a conversational CBR (CCBR) approach might be used. A CCBR system is used when the problem is not completely known and, therefore, the traditional retriever has no data to match the cases to. The system starts a conversation with the user, asking him questions which discriminate between learner models by traversing a decision tree. As

the learner model is drawn out from this conversation, and the other problem data are known, the system selects the suitable cases. An even more attractive direction would be to adapt CCBR so that, instead of using conversations to figure out the learner model, learners are given stories to play, where the stories are chosen in such a way that the user's actions lead the reasoner along the same discriminative decision tree.

3.3.2 Plan Reuse and Revise

The differences between the goals of retrieved plans and the goals of the current learner are identified and used to adapt the plan. If the goal competencies are not similar, the competencies to be removed are identified and the associated stories are removed from the plan. If the current targeted competencies usually entail the mastery of some new competencies, the plan is adapted so that it targets these competencies. The obtained plan and stories are then analysed using domain knowledge to make sure that they are coherent, and revised if needed.

3.3.3 Plan Retain

The plan and its trail are saved in a temporary storage after it has been played by the learner. Then, periodically these plans and trials are analysed and filtered. For the stories which failed (eg.: the learner did not achieve the related competencies), the planner updates its fail expectation, and saves the recovery plan which worked. The recovery plan is represented by the stories the learner played until they achieved those competencies. At this stage, if the plan is a new one, it is assigned a utility and eligibility value. If the plan is a reused one, these values are updated. When a contingency story has a better eligibility value than the story in the original plan, it replaces the story in the plan. An important challenge here is to filter out the plans and stories which are not considered relevant for future use.

Section 5 discusses these stages in more detail, as well as case representation.

4 Learning Theory Principles

Game-based learning has pedagogical foundations in problem-based learning [42], experiential [28] and inquiry-based learning [3]. More-over, they are also able to support other types of learning strategies because the content of the learning material is highly customisable, simulating real life situations, and capturing the learners' actions when they are faced with various tasks. This section describes some learning principles inspired by modern learning theories, that can be supported by the case-based planner overviewed in the previous section.

4.1 Principles of Linking Stories in a Learning Plan

The learning plan must be created so that the flow of stories the user engages with, lead him to the targeted competencies. For the creation of the learning plan we must consider the fact that the way learning episodes relate to each other is very important in order to keep the learner motivated and on the flow. There are several aspects which we focus on in creating the learning plans. First of all, we have to consider if there exists a domain model where possible competencies are represented and have specific relations between them (e.g. decomposition, prerequisites, constraints). These relations are extracted from the Competencies Graph illustrated in Figure 1 and they guide the ordering of the story templates in the abstract plan.

Secondly, it is important that the learning plan builds new competencies on top of existing ones. Following this principle, the competencies are developed both horizontally and vertically. By horizontally we mean the development and association of new competencies from existing ones, and by vertically we mean reaching higher levels of mastery in a competency. Thus, in the learning plan the story complexity and the difficulty will increase as the user performs.

The third principle is that learning needs practice, and often recursiveness and/or repetition. The *variation theory of learning* [34] and the *cognitive flexibility theory* [53] argue that practice of the same thing in different contexts, not pure repetition, leads to better learning outcomes. Following this principle, a learning plan should train the same competency in typical but varied situations until the learner reaches the desired level and also subject him to at least one atypical situation.

While in the case of case-based planning the plans are usually extracted from the case-base, these principles must be enforced 1). in the case of *data-sparsity* problem when plans must be built from scratch for example in a manner close to [26] and 2). when new learning plans are obtained as adaptations of existing ones, in the "revise" stage of the CBP process.

4.2 Principles for Personalised Story Recommendation

Besides plan generation, we use a case-based reasoner to recommend stories which might help the learner get over the stages where he or she gets stuck in the learning process. When the learner fails to achieve the supposed intermediate goals, the planner detects a fail. This failure might be interpreted by the planner as either an expectation failure or plan failure. A learner might get stuck in a game by not making any relevant progress, which can lead to frustration. The learner is assessed on-the-fly [45], his actions being evidences for competency assessment, as well as for his emotional state [4]. This embedded formative assessment can guide the story personalisation. In this case, the case-based reasoner suggests targeted stories or story episodes, starting from one which poses problems to the learner, but adapted based on the variation patterns from *variation theory of learning*.

The proponents of the variation theory of learning define four main patterns of variation that facilitate learning [33]: (i)*contrast* - experience the world with or without the property of interest; (ii) *generalisation* - experience various worlds containing the object of interest; (iii) *separation* - experience the property of interest by varying it while other aspects stay the same; (iv) *fusion* - experience related or dependent properties of interest simultaneously. Therefore, these dimensions of variation can be employed by the case-based recommender in the adaptation stage of the CBR process. A preliminary study on how variation theory can be implemented in a serious games scenario is presented in [40].

The case-base can be used to also recommend similar experiences of other learners. This enables the environment to integrate *case-based learning (CBL)*[5] [20]. CBL allows the students to view how others act, analyse and compare with their own actions and has already been successfully used in serious games.

In addition, the system should show the learner graphs and statistics on their performance and their learning patterns. In this way, learners have the chance to analyse their overall progress and how it was achieved, and thereby have facilitated *meta-learning* [35], a process of being aware and taking control of one's own learning.

5 Hierarchical CBP for Personalised Learning Plans

5.1 Reasoning with Abstraction to Build Learning Plans

In a game-based learning environment, a learning plan is an ordered list of stories meant to support the learner until he reaches the desired competency profile. The learning plan has to be adapted to the learner data like age, gender, cultural background. As well, it has to dynamically adapt based on the learner performances within the plan. This means that the planner does not have enough knowledge to create the whole plan in the initial stage of the planning. Therefore, at this stage several abstract plans are created, as sequences of story templates, and the learner can choose which one to execute. The story instances are created on-the-fly based on the story templates as the plan execution thread reaches them. At this stage the system has accumulated knowledge from the user's performances so far, and can individualise each story.

This methodology is inspired from the use of abstraction in case-based reasoning. By using abstraction, the less relevant features of a problem description are ignored in a first stage, leading to an abstract solution. Then, as the ignored features of the problem are being considered, the final concrete solution is derived from the abstract one [5]. In our case, the reasoner does not ignore features of the problem, but has

[5] Please note that although case-based learning uses similar wording as case-based reasoning, we use it here to denote a human learning and teaching strategy, and has nothing to do with machine processes.

to reason with an incomplete problem, which becomes complete as the solution is executed.

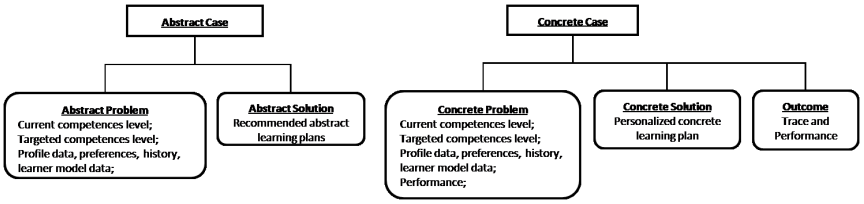

Fig. 3 Single Case Components

Following this hierarchical model, the abstract cases solve the problem requirements related to competency needs and learner profile data, by suggesting several abstract plans. The concrete cases have the problem enriched with the learner's performances, and therefore the solution is an iteratively created concrete learning plan. The two types of cases are represented in Figure 3.

5.2 Hierarchical Case-Based Planning

For planning on several layers of abstraction, many terms have been used in literature, the most common ones being hierarchical case-based reasoning [49] and stratified case-based reasoning [9]. The basic idea is that in the hierarchy of cases, only the "leaf" cases are concrete, and all the other nodes are "abstract cases". The studies presented in [6, 9, 49], to name just a few, prove the advantages of this approach. Compared to classical case-based planning, it shows significant improvements in efficiency of retrieval and adaptation.

There are still differences in these approaches. In some of them, the abstract cases are created by abstraction and generalisation [1] of concrete cases. This is a bottom-up process which consists of merging concrete cases based on similar features. These are then discriminated based on their specific features, obtaining a hierarchical tree structure. In these systems the plans are retrieved entirely, and the new solutions are created by adapting them, e.g., in the PARIS system [6]. Other approaches create the abstract cases starting from task or goal decomposition. The concrete cases are the atomic actions which cannot be decomposed any more. The work described in [31] uses such an approach. In these type of systems, each planning step is retrieved individually and then they are integrated to form the adapted solution. Another system, Déjà-Vu [49], combines the two types of hierarchies.

In our research, each planning step (a story instantiation) is not retrieved individually but is adapted by the user's previous interactions. Hence in our approach plan instantiation and the final steps of plan adaptation occur together. Generated abstract plans are presented to the user and he makes the choice of which one to

follow. Every story generation is directly followed by user execution and system evaluation. The results are used to create new tasks for the subsequent steps.

In our solution, there are two levels of abstraction. In the systems which use abstraction it is common that the depth of the hierarchy is flexible, as the abstract cases are generated dynamically as soon as new cases share common features. The results of [9] show a significant improvement in efficiency when 3-4 levels of abstractions are used. If this proves to be valid in our system too, we will consider the option of using dynamic abstraction within each of the two current layers.

5.3 Abstract Plans

5.3.1 Abstract Case Representation

In order to represent the abstract cases we have to consider that there can exist multiple learning plans achieving the same learning goals. Consequently, all the plans which have similar initial states and goals are grouped under the same root. Then a description node is created for each abstract plan. This description node contains the users who executed the plan in the past, and a summary of their experiences (the plan outcome). This abstract plan outcome includes information like the time the user needed to complete the plan, the average number of story repetitions, and the performances. It is important to note that this summary, although part of the abstract case representation, is extracted from the concrete layer. This way, we compensate for the loss of information which is inherent in reasoning with abstract cases [6]. Including this information in the description node gives us the possibility of combining CBR with collective filtering. In this scenario, collective performance information from similar learners will help in ranking candidate cases. The model also supports the inclusion in the description of learners who were recommended the plan but did not choose to execute it. This information lends itself to providing explanations (e.g. 8 out of 10 learners selected this plan, 6 out of 8 learners completed this plan).

The description nodes have as children the abstract plans they describe. This hierarchy is illustrated in Figure 4. As mentioned in Section 3, an abstract plan is a list of story templates. In Figure 4, the abstract plan 1 is composed of the story templates *ST1* $\rightarrow$ *ST2* $\rightarrow$ *ST3*, and the abstract plan 2 is *ST4* $\rightarrow$ *ST3*.

The figure shows that the learners *Learner 1*, *Learner 2* and *Learner 3* have similar initial and goal competency states. Still, both *Learner 1* and *Learner 2* chose the *Abstract plan 1*, while *Learner 3* chose *Abstract plan 2*. Let us define the initial competency state as *(conflict resolution, beginner), (negotiation, average), (crisis management, beginner), (resource planning, upper average)* and the goal competency state as *(conflict resolution, average), (crisis management, average), (resource planning, expert)*. Then, the story template illustrated in Figure 2 is a good candidate for being part of the two abstract plans. Moreover, since it brings together all the goal competencies, it is a good candidate for being *ST3*.

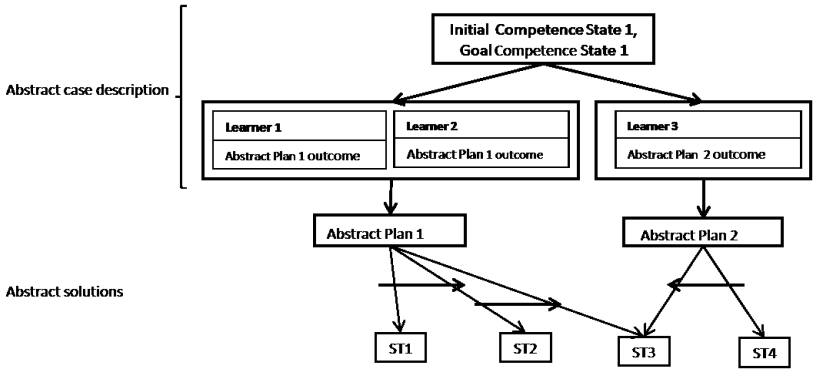

Fig. 4 Abstract Case Representation; ST - story template;

Each story template in the abstract plan, has assigned the competencies it has to train and evaluate within that plan, and an initial set of tasks, based on the available knowledge about the learner. For example, let us consider the story template in Figure 2 is labeled *ST3* in Figure 4. Then, within the two abstract plans, the template is assigned the tasks to select situations which match *conflict resolution, negotiation, crisis management* and *resource planning*, since these are the competencies it was chosen for, even if it can support situations addressing other competencies as well.

This information is kept latent until the instantiation process, when the story is created. It can be seen as an explanation why the template is part of the abstract plan. Still, this data is not enough for a personalised story. To personalise the story, more tasks to fulfill are assigned to the template as described later in section 4.2.

5.3.2 Abstract Plan Retrieval and Reuse

The retrieval of the abstract learning plan is a top-bottom traversal of the tree presented in Figure 4. This consists of two main steps: during the first step the system matches the current problem's initial state and goal to existing cases in the case base. Considering Figure 4, this stage retrieves a set of nodes from the first level.

During the second step the system first retrieves the child nodes of the nodes returned after the first step. Then, for each such child it computes a *suitability* value, rather than a similarity value. The suitability value takes into consideration the learner similarity, the plan outcome for him and as well adaptation complexity [48].

After the most suitable abstract plans are retrieved, they are adapted so that they fulfill all the problem's requests: they fit the learner, his current competency profile as well as his targeted competencies. The adaptation consists of adding/removing/replacing story templates from the original abstract plan. At this stage, the system has to make sure that the order of trained competencies and story templates respects the constraints described in Section 4.

5.4 Concrete Cases

5.4.1 Concrete Case Representation

Concrete case representation inherits from hierarchical representation used by Smyth et al. in Déjà-Vu [49]. The similarity comes from the fact that stories are generated step by step, therefore the final concrete solution is obtained by integrating the individual stories. Still, our suggested planner executes each step before instantiating the next. Both approaches permit multiple case reuse, which means that each planning step can be retrieved and reused from multiple cases.

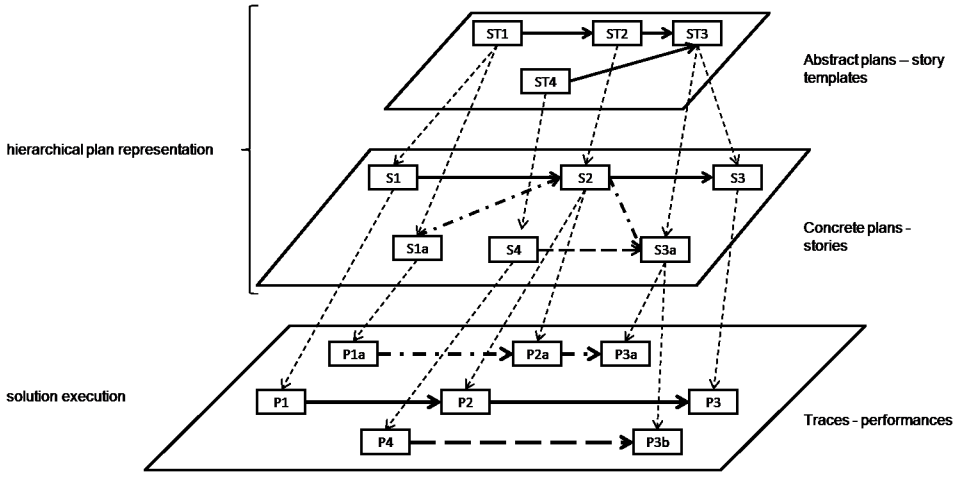

Fig. 5 Learning plans hierarchy; ST - story template; S - story; P - performance

As described in Figure 3, a component of the concrete problem is the set of previous user performances. Therefore, a learning plan that has been even partially executed by the learner is stored along with its performance score as a plan trace. The performances are analysed and depending on the result, the system selects and tailors the next story to play. Figure 5 shows the concrete plans layer, standing between the abstract plans layer and performance layer.

In the example in the figure, there are two abstract plans: $ST1 \rightarrow ST2 \rightarrow ST3$, and $ST4 \rightarrow ST3$. The first abstract plan has two instantiations, i.e., two concrete learning plans: $S1 \rightarrow S2 \rightarrow S3$ and $S1a \rightarrow S2 \rightarrow S3a$. The second abstract plan has only one instantiation in the case base: $S4 \rightarrow S3a$. The arrows from the abstract plan to the concrete plan show how the story templates have been instantiated. For example, *ST1* was instantiated creating *S1* and *S1a*, while *ST2* was instantiated only once, in *S2*, this story being selected for two concrete plans.

The third layer shows how each concrete plan forms a trace as it is being executed. For example, the concrete plan $S1 \rightarrow S2 \rightarrow S3$, was executed once, leading to the trace: $P1 \rightarrow P2 \rightarrow P3$. The vertical arrows show how each story was instantiated

by being played by a user and leading to a performance. For instance, the story *S2* was executed twice, leading to the performances *P2* and *P2a*.

Let us consider the example in Section 3.2 , with two learners, *L1* and *L2*. Because they have similar current and targeted competencies, they are recommended the same abstract plan: *ST1* → *ST2* → *ST3*. Let us further consider that the story template in Figure 2 is labeled *ST3* in Figure 5. Before the instantiation of *ST3*, the learner *L1* has executed two stories, *S1* and *S2*, with performances *P1* and *P2*. At the same stage, the learner *L2* has executed the stories *S1a* and *S2* with performances *P1a* and *P2a*, respectively. As the planner analyses the performances, *L1* seems to make a good progress and successfully execute the tasks is short time. The planner can then decide to instantiate the *ST3* template to a complex story, therefore creating story *S3* as the flow of situations $B \rightarrow C \rightarrow E \rightarrow F \rightarrow G$. To make the story challenging, the planner also chooses to enforce a large consortium, with a spread conflict which determines a key partner to leave and cause a big resource gap. At the same time, if learner *L2* has a slow progress, with blockages and long idle times, the system can decide to instantiate *ST3* into *S3a* as the flow of situations $B \rightarrow C \rightarrow D$. To make the story accessible, it defines a consortium of 4-5 partners with only two conflicting partners. A partner with a low contribution has to leave, and the lost resources can be covered from within the remaining consortium.

5.4.2 Planning on First Principles

The learning plan is created step by step, by instantiating the story templates of the abstract plan, at the moment they are needed or when the learner requests it.

Algorithm 1: NextStory(Learner, AbstractPlan, CurrentPlan, step)

```
   Input: Learner L, AbstractPlan AP, CurrentPlan CP, planning step n
1  S_n = CP[n] ;                          /* S_n is the last executed story in CP */
2  ST_n = AP[n];
3  P_n = Performance(L, S_n);
4  if P_n < ST_n.goal then
5  |   T = GenerateSelfTasks(L,S_n,P_n);
6  |   ST_n.tasks = ST_n.tasks ∪ T;
7  |   S_n^a = CreateStory(ST_n);
8  |   return S_n^a;
9  else
10 |   if n + 1 < AP.length then
11 |   |   T = GenerateTasks(S_n, P_n, L);
12 |   |   DistributeTasks(AP, n + 1, T);
13 |   |   S_{n+1} = CreateStory(ST_{n+1});
14 |   |   return S_{n+1};
```

Algorithm 2: DistributeTasks(AbstractPlan, step, Tasks)

```
   Input: AbstractPlan AP, step n, Tasks T
1  ST_n = AP[n];
2  foreach task t ∈ T do
3  |   if ST_n can satisfy t then
4  |   |   ST_n.tasks = ST_n.tasks ∪ {t};
5  |   |   T = T \ {t};
6  if T ≠ ∅ and n + 1 < AP.length then
7  |   DistributeTasks(AP, n + 1, T);
```

In order to create the next story, the system needs to interpret the previous performances and modify the remainder of the plan accordingly. Algorithm 1 illustrates how this is done. It uses a task creation and distribution mechanism shown in Algorithm 2: after a performance has been analysed a list of tasks is created. If the learner failed to reach the level planned for the current story, the planner can recommend him to replay a variation of the same story with a different difficulty level. Otherwise, the planner sends the package of tasks to the first subsequent story template. The story template keeps for itself the tasks which it can achieve and sends the rest further to the subsequent template in the plan, and so on. In case a story template cannot satisfy any new task, it is considered that it needs no further personalisation, and it is instantiated based on its initial set of tasks, set by the abstract plan.

An example of the concrete-plan creation process based on the task distribution is presented in Figure 6. The dashed arrows in the figure show how each performance triggers the delivery of tasks to the subsequent story template, which keeps for itself the tasks it can achieve and sends the rest forward.

As an example, let us consider that the story template *ST3* represents the story template illustrated in Section 3.2, Figure 2. The template receives the tasks *T1* and *T3* due to the performance *P1*. *T1* states that the complexity of the story should be high, and *T3* requires that the *team management* competency should be approached so that it suits a beginner. *ST3* receives the two tasks but, because *T3* refers to a competency the story template cannot address, it can only keep *T1*. *T3* is sent further to the next story template. Due to previous performance *P2a*, *ST3* also receives tasks *T5* and *T6*. *T5* states that the competency *crisis management* should be approached so that it suits an average level learner. *T6* states that the set of competencies needed to successfully achieve the story mission must include the learner's targeted competencies, but not exclusively.

When *ST3* needs to be instantiated, it has to consider therefore the tasks *T1*, *T5*and *T6*. Because of *T1* and *T6*, the story is created so that it brings a large number of situations. Because of *T5*, the situations have to be chosen and adapted so that *crisis management* is required in many situations, but not very demanding. This requirements lead to the instantiation of story *S3* as the flow of situations $B \rightarrow C \rightarrow E \rightarrow F \rightarrow G$ with parameters instantiated so that situations C and G

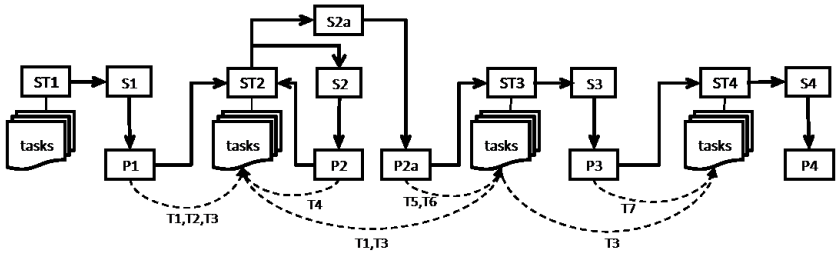

Fig. 6 Plan Instantiation Example.

cannot be handled unless the learner has an average level of proficiency in crisis management (due to *T5*).

5.4.3 Using CBP Adaptation Techniques to Create Concrete Plans

The way the stories are created at this step from the story templates, can be either based on first-principles, or using one of the case-based-planning adaptation techniques like *derivational* or *transformational analogy* [36, 52, 12]. When the stories are generated on first-principles planning, then the system does not need to retrieve concrete plans from the case-base. They are created starting from the abstract plan, using only domain knowledge. All the knowledge about how the tasks are generated from performances is needed. As well, how to instantiate a story starting from a story template and a set of tasks.

The transformational approach relies on the fact that the system saves entire plans and the new solution is created by reusing the old solution. When such an approach is used, then the system does not care to learn about tasks. If the old story's performance was partially similar to the current story's performance, then the system adapts the next story in the old plan to achieve the new story. In this approach domain knowledge is needed to be able to adapt the old story to the new previous performance.

On the other hand, using the derivational analogy, the cases are adapted based on the way the old solution has been built. Here, the system does not save the entire plans, but the decisions taken which lead to the plan generation. In our case, this would mean that the system does not need to know the stories, it is only interested in the tasks which led to their generation. If the old story's performance was partially similar to the current story's performance, then the system adapts the old set of tasks and creates the new tasks. Using these tasks, it generates the story. Here, the system needs domain knowledge on how to integrate the tasks in the story creation.

The transformational approach can be used when the sequence of story templates in the new plan is similar to the old plan, and there is a high chance that the new learner performs similar to the previous learner. Because this situation is unlikely to happen too often, we suggest the derivational analogy as being more appropriate for our problem. Its proven advantage is that it provides more flexibility, because the planner can replay the derivational trace relative to the new problem [36]. In our case, the derivational trace is represented by the tasks, and because the tasks are not dependent on the next stories, they indeed can be applied to the new plan. Another advantage of derivational approach is that it can be used with introspective case-based planning in order to prune fruitless past tasks.

Still, research and evaluation of the possible approaches has to be done before the best fitted solution can be selected.

6 Discussion and Future Work

By now, we have presented how learning plans are created for the learners and adapted to match their needs and performances. Another crucial part of case-based systems is the retain phase, during which the system adds the new cases to the case-base. The case base should avoid redundancy and be kept at a size which does not negatively influence the retrieval and adaptation efficiency. For this, we propose to keep all the traces and experiences in a separate storage, and then periodically carry out maintenance analysis [46] to make sure that only the cases which bring value to the case-base are retained.

Keeping the traces of successful and failed plans allows us to analyse the features and feature weighting that are leading to unsuccessful retrievals. Introspective learning techniques for feature weighting are designed to increase or decrease the weights of selected case features on the basis of problem solving performance [7]. Such techniques have also been used to facilitate easier adaptation of cases [30]. Analysing the repository of plan traces using introspective learning should allow us to improve the retrieval of abstract cases and their adaptation to the learner context.

An important aspect of game-based learning that does not make the focus of this paper is related to user's performance assessment. This process is the focus of another component in the TARGET system, which models the relations between competencies and situations (i.e. the dashed lines in Figure 2). The user's execution during a situation stands as evidence of his level of mastery of the related competencies. The way of automatically interpreting these evidences, assessing the user's competencies and finally student modelling is a challenging research direction on its own. One possible solution would be the use of dynamic belief networks, where the relations between competencies and situations are represented as probabilistic relationships, as suggested by Reye in [38].

Throughout this paper we mention the user data like age, gender and geographical details to be used for finding the suitable plan. Although it has been proven that cultural background, age and gender might influence a person's way of learning, we have to analyse if this data is relevant in our system. Therefore, we will use this data only for analysis during the early stages of the case base. If the analysis of cases proves any relation between learning and these parameters, we will consider them for plan retrieval.

Another aspect we have to consider when plans and stories are recommended is diversity [50]. We need diversity both for the learner and for the system. For the learner, it is important that recommended plans are varied and do not overlap with the user's already executed plans. For the system, it is important that it explores the efficacy of new plans as well, not only relying on old highly evaluated ones.

While the goal of this paper was to present a model of CBP and online learning using serious games, we should discuss our plans for implementation and evaluation. This work is being developed as part of the large European project TARGET, which contains academic and industrial partners and the case-based recommendation engine will be evaluated iteratively in small user trials, as well as fully integrated with the other components of the TARGET system.

7 Conclusion

We have presented a methodological framework for creating personalised learning plans based on serious games - interactive narratives designed to teach particular competencies. We justified our reasons for proposing a novel a case-based planning approach and described in detail our hierarchical case structure and our iterative retrieval and adaptation process. We proposed that the learning process can be continuously adapted for each individual learner. We showed how alternative plans which target the same goals can be represented, and retrieved based on their outcomes for different learners. The retrieved plans are then adapted on-the-fly, based on an evaluation of the learner's performance. We proposed a hierarchical planning methodology which enables the planner to retrieve and personalise the learning plan for each user. We also examined how plan traces from all learners can be exploited to improve the case-base of learning plans. This work is being developed as part of the European project TARGET and will be evaluated iteratively in small user trials.

Acknowledgements

This work was partly supported by the TARGET project under contract number FP7-231717 within the Seventh Framework Program, and by the Lion II project funded by Science Foundation Ireland (SFI) under Grant SFI/08/CE/I1380.

References

1. A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations and system approaches. *AI Communications*, 7(1):39–59, 1994.
2. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.
3. S. Barab, M. Thomas, T. Dodge, R. Carteaux, and H. Tuzun. Making learning fun: Quest atlantis, a game without guns. *ETR and D*, 53(1):86–107, 2005.
4. M. Bedek, P. Seitlinger, S. Kopeinik, and D. Albert. Inferring a learner's cognitive, motivational and emotional state in a digital educational game. *Electronc Journal of e-Learning*, 10(2):172–184, 2012.
5. R. Bergmann and W. Wilke. Building and refining abstract planning cases by change of representation language. *Journal of Artificial Intelligence Research*, 3:53–118, 1995.
6. R. Bergmann and W. Wilke. On the role of abstraction in case-based reasoning. *Advances in Case-Based Reasoning. Lecture Notes in Artificial Intelligence*, pages 28–43, 1996.
7. A. Bonzano, P. Cunningham, B. Smyth, et al. Using introspective learning to improve retrieval in CBR: A case study in air traffic control. *Lecture Notes in Computer Science*, 1266:291–302, 1997.
8. E. Boyle, T. M. Connolly, and T. Hainey. The role of psychology in understanding the impact of computer games. *Entertainment Computing*, 2(2):69–74, Jan. 2011.
9. K. Branting and D. W. Aha. Stratified case-based reasoning: Reusing hierarchical problem solving episodes. In *IJCAI*, pages 384–390, 1995.

10. D. Charles and M. McAlister. Integrating ideas about invisible playgrounds from play theory into online educational digital games. *Entertainment Computing–ICEC 2004*, pages 598–601, 2004.
11. T. M. Connolly, E. A. Boyle, E. MacArthur, T. Hainey, and J. M. Boyle. A systematic literature review of empirical evidence on computer games and serious games. *Computers & Education*, 59(2):661–686, 2012.
12. T. Cox, M., H. Munoz-Avila, and R. Bergmann. Case-based planning. *The Knowledge Engineering Review*, 20(3):283–287, 2006.
13. S. de Freitas. Emerging technologies for learning. Technical report, Becta, 2008.
14. H. Drachsler, H. Hummel, and R. Koper. Recommendations for learners are different: Applying memory-based recommender system techniques to lifelong learning. In *Workshop on Social Information Retrieval for Technology-Enhanced Learning and Exchange*, 2007.
15. H. Drachsler, H. Hummel, and R. Koper. Identifying the goal, user model and conditions of recommender systems for formal and informal learning. *J. Digit. Inf.*, 2008.
16. D. Druckman and R. Bjork. *In the mind's eye: Enhancing human performance*. National Academies Press, 1991.
17. M. Gmez-Albarrn and G. Jimnez-Daz. Recommendation and students authoring in repositories of learning objects: A case-based reasoning approach. *iJET*, 4(S1):35–40, 2009.
18. M. Gómez-Martín, P. Gómez-Martín, and P. González-Calero. Game-driven intelligent tutoring systems. In *Lecture Notes in Computer science 3166*, pages 108–113. Springer, 2004.
19. P. Gómez-Martın, M. Gómez-Martın, B. Dıaz-Agudo, and P. González-Calero. Opportunities for cbr in learning by doing. In *Proceedings of 6th International Conference on Case-Based Reasoning (ICCBR)*. Springer, 2004.
20. J. S. Hammond. Learning by the case method. Harvard Business School Publishing Division, Harvard Business School, 1976.
21. K. Hammond. Case-based planning: A framework for planning from experience. *Cognitive Science*, 14:385–443, 1990.
22. M. Harzallah, G. Berio, and F. Vernadat. Analysis and modeling of individual competencies: Toward better management of human resources. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 36(1):187–207, 2006.
23. M. Hendrix, A. Protopsaltis, I. Dunwell, S. de Freitas, S. Arnab, P. Petridis, C. Rolland, and J. Llanas. Defining a metadata schema for serious games as learning objects. In *The Fourth International Conference on Mobile, Hybrid and On-Line Learning*, 2012.
24. IBM and Seriosity. Virtual worlds, real leaders: Online games put the future of business leadership on display. Technical report, IBM and Seriosity, 2007.
25. D. H. Jonassen and J. Hernandez-Serrano. Case-based reasoning and instructional design: Using stories to support problem solving. *Educational Technology Research and Development*, 50(2):65–77, 2002.
26. P. Karampiperis and D. Sampson. Adaptive learning resources sequencing in educational hypermedia systems. *Educational Technology and Society*, 8:128–147, 2005.
27. M. Kebritchi, A. Hirumi, et al. Examining the pedagogical foundations of modern educational computer games. *Computers & Education*, 51(4):1729–1743, 2008.
28. D. A. Kolb, R. E. Boyatzis, and C. Mainemelis. Experiential learning theory: Previous research and new directions. *Perspectives on Thinking, Learning, and Cognitive Styles*, pages 227–247, 2001.
29. J. L. Kolodner, C. E. Hmelo, and N. H. Narayanan. Problem-based learning meets case-based reasoning. In *Second International Conference of the Learning Sciences*, 1996.
30. D. Leake, A. Kinley, and D. Wilson. Learning to improve case adaptation by introspective reasoning and CBR. *Lecture Notes in Computer Science*, pages 229–229, 1995.
31. C. Lee, C. K.Y.R., and L. A. A case-based planning approach for agent-based service-oriented systems. *Systems, Man and Cybernetics. IEEE International Conference*, pages 625–630, 2008.
32. N. Manouselis, H. Drachsler, R. Vuorikari, H. Hummel, and R. Koper. Recommender systems in technology enhanced learning. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 387–415. Springer US, 2011.

33. F. Marton and M. F. Pang. On some necessary conditions of learning. *Journal of the Learning Sciences*, 15(2):193–220, 2006.
34. F. Marton and K. Trigwell. Variatio est mater studiorum. *Higher Education Research and Development*, 19(3):381–395, 2000.
35. J. H. F. Meyer and M. P. Shanahan. Developing metalearning capacity in students: actionable theory and practical lessons learned in first-year economics. *Innovations in Education and Teaching International*, 41(4):443–458, 2004.
36. H. Munoz-Avila and M. T. Cox. Case-based plan adaptation: An analysis and review. IEEE Intelligent Systems.
37. S. Pal and S. Shiu. *Foundations of soft case-based reasoning*, volume 8. Wiley-interscience, 2004.
38. J. Reye. Student Modelling based on Belief Networks. *International Journal of Artificial Intelligence in Education*, 14:1–33, 2004.
39. M. M. Richter and A. Aamodt. Case-based reasoning foundations. *Knowledge Eng. Review*, 20(3):203–207, 2005.
40. M. Ruskov and W. Seager. What can bits teach us about leadership: A study of the application of variation theory in serious games. In M. Ma, M. Fradinho Oliveira, and J. Madeiras Pereira, editors, *Serious Games Development and Applications*, volume 6944 of *Lecture Notes in Computer Science*, pages 49–60. Springer, 2011.
41. O. C. Santos and J. Boticario. Modeling recommendations for the educational domain. *Procedia CS*, 1(2):2793–2800, 2010.
42. J. R. Savery and T. M. Duffy. Problem based learning: An instructional model and its constructivist framework, 1994.
43. R. Schank and R. Abelson. *Scripts, plans, goals and understanding: An inquiry into human knowledge structures.* Lawrence Erlbaum, 1977.
44. R. C. Schank. Goal based scenario: Case-based reasoning meets learning by doing. *Case-Based Reasoning: Experiences, Lessons and Future Directions*, pages 295–347, 1996.
45. V. J. Shute, M. Ventura, M. Bauer, and D. Zapata-Rivera. Melding the power of serious games and embedded assessment to monitor and foster learning: Flow and grow. In U. Ritterfield, M. J. Cody, and P. Vorderer, editors, *The Social Sciences of Serious Games: Theories and Applications*. Routledge/LEA, 2009.
46. B. Smyth. Case-base maintenance. In *Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, 1998.
47. B. Smyth. Case-based recommendation. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The adaptive web*, pages 342–376. Springer, 2007.
48. B. Smyth and M. T. Keane. Adaptation-guided retrieval: questioning the similarity assumption in reasoning. *Artificial Intelligence*, 102:249–293, 1998.
49. B. Smyth, M. T. Keane, and P. Cunningham. Hierarchical case-based reasoning integrating case-based and decompositional problem-solving techniques for plant-control software design. *IEEE Transactions on Knowledge and Data Engineering*, 13(5), 2001.
50. B. Smyth and P. McClave. Similarity vs. diversity. *Lecture Notes in Computer Science*, pages 347–361, 2001.
51. L.-K. Soh and T. Blank. Integrating case-based reasoning and meta-learning for a self-improving intelligent tutoring system. *International Journal of Artificial Intelligence in Education*, 18:27–58, 2008.
52. L. Spalazzi. A survey on case-based planning. *Artificial Intelligence Review*, 16:3–36, 2001.
53. R. J. Spiro, R. P. Feltovich, M. J. Jacobson, and R. L. Coulson. Cognitive flexibility, constructivism, and hypertext: Random access instruction for advanced knowledge acquisition in ill-structured domains. *Constructivism and the technology of instruction: A conversation*, pages 57–76, 1992.
54. L. Tobias and D. A. Identifying employee competencies in dynamic work domains: Methodological considerations and a case study. *Journal of Universal Computer Science*, 9(12):1500–1518, 2003.