Chapter 3b

Realizing Just-in-Time Personalization – A Technology Overview

Owen Conlan, Cormac Hampson, Kevin Koidl, Stefan Göbel, Florian Mehn

Introduction

Digital Game-based Learning (DGBL) strives to maintain a balance between learning effectiveness and the inherent motivational experience of playing a game. The main challenges lie in ensuring the learning offered is embedded in the gameplay in such a way that it does not interrupt the flow of the gaming experience. This may be achieved by tightly integrating the gaming environment with the educational material, but such tight coupling leads to poor reusability as the gameplay, story narrative and learning material are difficult to separate. In 80Days the responsibilities for gameplay, storytelling and learning are separated into three engines that interoperate to deliver a cohesive learning experience through a compelling gaming setting.

The learning offered in 80Days is tailored to the learner's individual skills and needs. This form of dynamic personalization presents several technological challenges, such as implicitly acquiring information about the learner, cooperating with the story engine to ensure the narrative evolves to accommodate appropriate learning objectives and performing adaptations in a timely and consistent manner.

This chapter focuses on the operation of the Learning Engine, specifically how it models learners and make adaptive recommendations to the Game Engine. The chapter also discusses the interoperation between the Learning Engine and Story Engine to ensure the game meets the needs of the learner.

Challenges in Realizing Personalized Digital Game–based Learning

Producing a game that is both motivational and educationally sound presents a number of significant challenges. The primary of which is

ensuring that the pedagogical premise of the educational elements of the game and the gameplay elements are sufficiently intertwined to present a holistic experience. A common failing of Digital Game-based Learning has been the lack of integration of these elements. This often leads to a fractured experience for the learner with the educational elements appearing as dull interludes in the otherwise exciting game.

Simply combining traditional technology enhanced learning techniques with a game does not automatically produce a motivational learning experience. The flow of experience is often interrupted by the learning material, which appears to be flat and non-interactive when compared with the gaming experience. However, there is a compelling reason for combining games and learning and it stems from the inherent motivation that most games inspire. The learner is driven to complete the game by the feeling of satisfaction that comes with successfully mastering the in-game challenges. Every computer game exhibits some form of learning activity, albeit quite informal. For example, a player must learn the controls of the game, details of the game setting and how to interact successfully with non-player characters (NPCs). As these skills are learned as part of the gameplay and are integral to success in the game, players are willing to invest time in learning them. The vision for successful Digital Game-based Learning is that the educational skills learned through the game should be seen as just as integral to successfully completing the game.

Assuming that an appropriately compelling means can be found to integrate the educational skills and game skills, there exists a fundamental problem – no two learners have exactly the same needs or prior learning experience. ELearning typically suffers from high dropout rates. This is for a variety of reasons, but the material not meeting the learner's needs and their lack of motivation rate highly for not completing a course. If appropriately combining eLearning material with an engaging computer game helps to resolve the motivation issue, then how can the material not meeting the needs of the learner be addressed?

Adaptation and personalization technology offers the potential to tailor an offering to meet the needs of a user. Adaptation techniques, such as those proposed by the Adaptive Hypermedia research domain, often construct an information offering for the user by composing several smaller pieces together. It is imperative that a sound narrative is followed to ensure that the pieces form into a coherent whole. In addition

to a sound theoretical foundation, adaptive education requires quite demanding technological realizations of theories and models. Over the past number of years, a variety of intelligent and adaptive educational systems have been introduced. To date, these systems have focused largely on adaptively ordering and presenting learning tasks.

Intelligent, adaptive, and personalized tutoring systems were developed by different researchers; a detailed review on such technologies was published by Brusilovsky (1999), general reference frameworks are described, for example, by De Bra (1999) or Albert and Mori (2001). Techniques of adaptation and individualisation are primarily adaptive presentation, adaptive navigation support, and adaptive problem solving. In the framework of the ELEKTRA project (Kickmeier-Rust et al, 2006) a new terminology was introduced because game-based approaches to learning are substantially different to traditional eLearning approaches The new concepts, which are tailored to learning environments with large degrees of freedom, are adaptivity on macro and micro levels (Kickmeier-Rust & Albert, 2010; Kickmeier-Rust et al., 2007). Macro-adaptivity refers to traditional techniques of adaptation such as adaptive presentation and adaptive navigation on the level of learning objects (or learning situations in a DGBL). Generally, macro-adaptive interventions are based on a fixed learner model (e.g. traits) or adaptation model (e.g. pedagogical implications) and on typical (knowledge) assessments (via test items). Micro-adaptive interventions, on the other hand, are non-invasive (meaning that an overall narrative is not compromised) and affect the presentation of a specific learning object or learning situation.

Within Digital Game-based Learning a different approach to that seen in intelligent tutoring systems has to be taken since the learning tasks are profoundly embedded the narrative of the game. As the game has its own storyline, any personalization offered must remain cognisant of this narrative structure whilst presenting the educational material in a meaningful manner. Thus, simply reordering learning tasks in a game may make educational sense, but would most probably result in an implausible rearrangement of the game's narrative plot elements. Therefore, existing approaches to intelligent and adaptive educational technology are inappropriate for gaming learning environments. Due to the nature of immersive DGBL the adaptation within such games needs to be continuous and less periodic. This issue can be resolved by integrating micro-adaptivity into the environment where adaptation

occurs within the various learning situations as opposed to around them. Micro-adaptivity creates challenges of its own due to the nature of the experience of game play, and the impact that game world changes can have on a player's experience.

Micro-adaptation allows the details of a learning situation to be tailored to the specific needs of a learner. Macro-adaptation allows the sequence of learning situations to be reordered and more learning situations added to accommodate more learning objectives. This form of adaptation requires close coordination with the storytelling elements of the game to ensure the story remains coherent. Both of these forms of adaptation may be performed dynamically as part of the game. The macro-adaptation tends to happen on a slightly longer time frame as the need for more or fewer learning objectives emerges. The micro-adaptation happens rapidly as the learner interacts with the game. In both forms the information about the learner and their needs must be inferred through observing their behaviour and interaction with the game. Interrupting the game to explicitly query the learner about their needs would break the immersion of the game and disrupt its flow.

There are a number of technical challenges that must be addressed in order to introduce micro- and macro-adaptation into DGBL –

1. The modeling of the learner must be implicit and time efficient. It should be possible to model several facets of the learner, such as their knowledge, gameplay skills and motivation in a timely manner.
2. Micro-adaptation should be able to personalize a learning situation to meet the needs of a learner. These adaptations again should be timely and consistent with the story elements of that situation.
3. Macro-adaptation should able to introduce more learning objectives into the game in order to meet the learner's evolving needs. The introduction of these learning objectives may require the modification of the storyline to ensure a contiguous narrative experience.

The key aspects of these challenges are timing and consistency. The learner should not be aware that the game is adapting to meet their needs, other than a sense that it is appropriate for them. If the gameplay slows down or behaves in an unexpected or inconsistent manner then the adaptations begin to produce a negative experience for the

learner. The danger is that immersion may be broken and the flow of experience disrupted.

Introducing the Story Engine and Learning Engine

In order to ensure that consistency is not compromised the components responsible for the learning and storytelling need to work in tandem. In 80Days the components responsible are the Learning Engine (LE) and Story Engine (SE), respectively. They are separately implemented and maintained logic engines that operate within their respective remits. However, it is important that they cooperate closely to ensure this consistency is maintained. Broadly speaking the Story Engine is responsible for ensuring the sequence of learning situations chosen produces a sensible storyline. The Learning Engine is responsible for the micro- and macro-adaptation decisions that pertain to learning. These may impact the sequencing of learning situations when it is determined that more learning objectives are required. The LE has two main areas in which it performs adaptations – skills and motivation. Skills relate to learning outcomes that may be gained by the learner through a learning situation. Motivation relates to the degree of immersion and involvement experienced by the user whilst learning and playing through the game.

In order for the LE to successfully adapt to the learner's needs it must have a model of that learner. Evidence comes from the Game Engine, via the Story Engine to the Learning Engine and this evidence must be interpreted to build a coherent model of the learner. Figure 3b-1 shows a simplified version of the 80Days architecture with the separation of the three main engines. In order to build a coherent model of the learner the LE has two components – the Skill Assessment Engine (SAE) and the Motivation Assessment Engine (MAE). These sub-engines must, in a timely manner, determine the skills the learner is acquiring and their degree of motivation from the evidence received. This interpretation stage must also gather information about the current game state and adaptations that have already been triggered. This information is important to ensure consistency and appropriateness of future adaptations that may be recommended. This is just one of four stages followed by the LE. These stages are discussed in more detail in the next section.

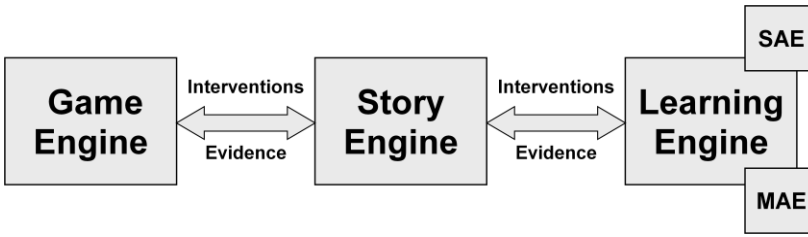

Figure 3b-1. Simplified 80Days Architecture

In 80Days the effectiveness of the different engines were assessed in two demonstrators. The first, known as Lizard, was developed through several iterations and was evaluated in authentic learning settings. Its primary focus was to assess the appropriateness of the micro-adaptations offered through a compelling and appealing game. The second demonstrator, know as Bat Cave, was a technical tool used to assess the potential of macro-adaptation to adapt to the changing needs of a learner by adding additional learning objectives to a game. This tool was assessed by validating that the additional learning situations added to cover these learning objectives remained consistent to the storyline being presented. Lizard involved integration of all three main engines, Game, Story and Learning, whilst Bat Cave focused predominantly on the latter two.

The Four-Stage Approach to Just-in-Time Personalization

The nature of the game evidence sent from the Game Engine (GE) is game specific and consists of player actions, movements, and task successes or failures. This information however is not immediately useful for educational adaptation, requiring a degree of inference by the Learning Engine (LE). Inference within the LE is the first step in the Four-Stage Approach to Just-in-Time Personalization employed to provide effective non-invasive adaptation. The four stages employed are inference, context accumulation, adaptation constraint, and adaptation selection. Further details on the background to the four stage approach are detailed in Peirce 2008.

The design of the LE and the four stage approach allows for the educational adaptation to be performed without regard for the game specifics. The LE effectively infers and abstracts game actions into educational evidence that can be reasoned over in a generic manner, thus

enabling it to be employed for different games with minimal alteration. A key example of this is the abstraction of skills provided through the Skill Assessment Engine (SAE). The SAE effectively maps user actions within the game to skill evidence, and further generates a probabilistic skill model for the learner. More information on the SAE is detailed in the next section.

The second stage of the adaptation process involves accumulating game and learner evidence. In consideration of the large quantity of evidence accumulated, potentially dozens of items per second, the use of XML based models, a traditional approach in many Adaptive Hypermedia Systems such as APeLS (Conlan, 2004), becomes impractical due to manipulation and reasoning speed. Consequently all data is accumulated in a working memory provided by the Drools rule engine. The use of the Drools rule engine provides an efficient means to reason over large data sets using declarative logic.

In order to perform adaptation within the GE the LE must have an a priori abstracted understanding of the adaptations possible. Within the LE these adaptations are negotiated with the Story Engine (SE) and are represented as Adaptive Elements. An Adaptive Element consists of an identifier which represents the corresponding skill and type of intervention such as "B2337_CE" (B2337 being the skill identifier and CE being the identifier for the intervention type). Based on these identifiers the SE can decide which dialog should be rendered in the game engine. An example Adaptive Element in the 80 Days would be the Non Player Character (NPC) Feon giving a cognitive hint such as, "Okay, that's Budapest – the capital of Hungary."

The following are the benefits of using Adaptive Elements:

- Educational adaptation does not need to be concerned with realizing adaptations
- Facilitates the independent authoring of the game engine and the adaptation logic
- Based on SE negotiation story constraints can be considered

The third LE stage of adaptation constraint is concerned with ensuring that only appropriate Adaptive Elements are used. By using constraint rules, only feasible and appropriate Adaptive Elements are made available for selection in the final LE stage. The selection

of adaptation is achieved through adaptation rules that examine the accumulated learner data and the available Adaptive Elements.

The Skill Assessment Engine

Knowledge Space Theory (KST), introduced by Doignon and Falmagne (Doignon 1985), provides a theoretical framework within which the knowledge or skill state of a learner can be determined. It is based on a prerequisite skill structure that describes the relationships between different skills. For example, a learner should typically be able multiply whole numbers before they can multiply decimal numbers. If the learner exhibits evidence of being able to multiply decimal numbers it may be assumed that they can also multiply whole numbers. Such probabilistic reasoning enables a system to infer a learner's skill state based on partial evidence (Conlan 2006).

The fundamental approach taken in KST is to reduce the number of possible pieces of evidence needed about a learner to an optimal set. In this way the Knowledge State of a learner may be assessed through the minimum number of inferences, thus achieving maximum efficiency. This is only possible by examining the domain in which the learning is occurring and identifying the underlying prerequisite relationships that exist between concepts. This is a time consuming and expert task that involves describing a learning domain, such as mathematics, in terms of formal prerequisite relationships.

Specific educational tasks, such as the learner interacting with a virtual experiment, are broken down into specific sub-tasks. Success or failure in these sub-tasks forms evidence that facilitates the probabilistic update of the learner's model. The certainty is dependent on the level of inference required. However, as only partial evidence is needed to assess a skill state it can be done very efficiently. When applied to DEGs KST has the potential to provide the basis of a time sensitive approach to modeling a learner's acquisition of knowledge and skills [].

Interpreting evidence sent by the Game Engine (GE) is central to the first inference step of the four stage approach to Just-in-Time Personalization. The Skill Assessment Engine (SAE), a component of the Learning Engine (LE), is responsible for translating a learner's actions within the game into a list of probabilities that show the likelihood of a skill having been acquired by the learner. This assessment of a learner's

skills must be done in an implicit fashion so as not to negatively impact their flow through the game.

The domain specific skills to be acquired in the 80Days game were organized according to KST into a prerequisite knowledge structure, which was represented as a binary matrix and then parsed by the SAE at design time. The resultant skills file was then loaded into the run-time component of the SAE. Previously the SAE was limited to working with modestly sized knowledge structures due to scalability issues with the initial algorithm implemented. This meant that in the ELEKTRA game (Peirce 2008) the SAE was limited to a knowledge structure of just 25 skills. However in the 80Days game a more efficient simplified updating algorithm was developed and incorporated into the SAE. This meant that knowledge structures of much larger sizes could now be updated efficiently at runtime, with the 80Days game containing a knowledge structure of 156 skills. Furthermore, the current version of the SAE can accommodate knowledge structures much larger than this if required.

During the game, the user faces various learning challenges, with specific educational rules triggered depending on their interactions with learning objects, such as maps and flood prevention simulators. Learning objects are traditionally seen as static pieces of content, usually HTML, with associated metadata. In 80Days a learning object is an interactive experience that is woven into the game narrative. Each learning object has skills associated with it, thus if a rule relating to a learning object is fired through a learner's interaction with the game, the SAE runs its algorithm to determine which skills have increased or decreased in probability.

If a user is performing badly at a specific task then the associated skills will have their probabilities decreased in the SAE. If these probabilities drop below a specified threshold then a related intervention may be triggered in the game (typically through NPC dialogue or in-game display) to help the user overcome this task and acquire the relevant knowledge. Likewise if a user is performing well at a task, the associated skill probabilities are increased within the SAE. If these probabilities exceed a pre-determined value, the user is then deemed to have acquired these skills.

These calculations must be done in less than 200ms (MacKenzie, 1993) so that the delay in the LE selecting an appropriate intervention for the

GE is not noticeable to the user. For the purposes of the work presented here, below 200ms is the definition of *Just-in-Time*. The adjustment in skill probabilities is taken into account in stage two of the four stage approach to Just-in-Time Personalization, where all evidence from the game and user is accumulated. Thus any change in skill probabilities has influence over which adaptive interventions are eventually presented to the user within the game environment.

The Motivation Assessment Engine

As illustrated in the previous section the assessment of the learner's skills during the runtime of the 80 Days game is essential for the Learning Engine (LE) to recommend appropriate cognitive interventions to assist the learner. This selection of interventions is based on the inferred skill state or knowledge of the learner. In addition to the cognitive assessment within the LE, 80Days supports the assessment of motivational aspects also. This functionality required the development of a separate engine to sit alongside the Skill Assessment Engine (SAE) called the Motivation Assessment Engine (MAE). The MAE recommends motivational interventions as it assesses the learner's attention and confidence. Similar to the SAE, both of the motivational aspects, attention and confidence are managed in a probabilistic model.

In order to assess the motivational aspects of the learner the Game Engine (GE) sends messages that relate to their specific actions within the game, such as changing the flight direction of the UFO. The LE can then use the time stamp of these actions to assess the motivational aspects of the learner e.g. it can infer low attention if the learner receives a cognitive recommendation, but does not react to it within a specific time frame. This can result in the LE sending a motivational recommendation to prompt the learner to react, if the MAE's value for attention drops below a predefined threshold. In addition to the timing of specific actions, the skill probability updates are used to indicate motivational aspects. For instance, a continuous stream of actions leading to a decrease of skill probabilities can be used to infer a lack of user confidence and attention, and can prompt a motivational intervention.

Finally the update history and probability reflecting the learner's attention is used to assess if the game's pace should be changed. A game pace change recommendation does not manifest in a dialog, such as the cognitive and motivational recommendations, but allows the game to switch to a faster game pace e.g. from a relaxed pace to a driven pace,

or from a driven pace to a hectic pace. The result of a change in the game pace could be the introduction of time limits, or specifically in 80 Days, the introduction of additional UFO's within the game that apply pressure to achieve the mission goals as fast as possible.

To conclude, the MAE uses three different types of evidence to assess the motivational aspects of the learner. First, the time between different actions within specific learning situations such as changing the flight direction of the UFO after receiving a cognitive hint or remaining idle for a specific time frame. Second, the amount of skill updates leading to a probability decrease of at least one skill. Finally, the update history of the motivational aspects can lead to a change in game pace.

Personalizing the Story

Story Description Language - ICML

In order to describe interactive stories and Storytelling-based application scenarios, much research has been invested into the development of the ICML format (INSCAPE Markup Language). ICML was developed within the INSCAPE project (Balet, 2007), has successfully been used in other projects and was further advanced for Story-based DEG's in 80Days.

The global aim of ICML is to provide not only a standardized and comprehensive description language for a specific storytelling framework, but also a good basis for a standard for any Interactive Storytelling applications in general. Used as the underlying data format for the Storytelling platform/framework, ICML describes the entire structure of the story in a declarative way. In its first version (ICML V1.0), each ICML document was separated into three top-level nodes, which distinguish different levels of narrative structures and their building primitives: Content, story and strategies.

The content node contains a global listing of logical content elements which are part of the story. The mapping to their physical source (e.g. image URL) is done by the framework. Each content element node includes a type description (sound, image, 3d model, etc.), the actions ('play', 'walk to', 'talk', etc.) and properties.

The story node encodes the 'state model' of the story, including all scenes/complex scenes and all possible transitions between scenes. Every scene contains a stage set and an action set node. The stage set

contains references to all content elements which are part of the scene and their size, position, etc. in that specific scene. In contrast, the action set node contains the high level story logic expressed as condition/action rules.

The strategies node contains a list of strategies. A strategy tells the system about corrective actions to be taken in case of a specific performance situation. In other words: it is a rule-base for triggering some methods that may influence the story flow at any point in time during the entire performance of the story

A strategy is comparable to an action set, with the difference that it describes global, story wide rules and doesn't refer to a particular scene. The strategies concept has been introduced within INSCAPEs' Story Pacing research topic whose main purpose was to guarantee a fluent and suspenseful experience in the performance of a highly interactive story by assigning the author certain control over time conditions and duration.

Story Engine

The Story Engine is responsible for executing and performing the interactive stories. The Narration Controller (Story Engine) forms the core of the overall runtime system. It interprets the created stories encoded in ICML described above and executes them by building an executable story graph out of it. During runtime, the graph is traversed by following the transitions from scene to scene – being evaluated at each step of the Narration Controller's main loop. If a transition is executed, the Narration Controller is responsible for informing the Player component (the Game Engine in the 80Days approach) about a change of the current scene. On the other side, the Player has to inform the Narration Controller about user events (game evidence from the GE), which has to be processed by the Narration Controller and/or directly passed to the Learning Engine (LE). Taking into account these user events, the Narration Controller traverses the action set for every scene, evaluates the conditions and informs the Player/GE about the actions to be executed.

How does a story continue at a specific moment? In order to answer that question the central idea is to use the concept of priorities. This means all the aspects discussed in the previous sections are considered, and a set of rules conceptualized. These rules are used by the Narra-

tion Controller to find the 'best' next state and to decide which scene is loaded next.

Hence, the key challenge is to find a fair balance between the initially created story and the 'exceptions' caused by user interactions (unforeseen or at least not intended by the author). As an example in the remainder of this section, a tour through a museum is used. In this case, examples for such exceptions are wrong paths (not following the instructions of a virtual guide), skipped stations (passing artefacts without interacting), too long or short interactions at artefacts (causing problems with external and internal time constraints), etc. The following list provides some of the rules, ordered by priority:

1. External time constraints have the first priority. In the reference example of a story-driven museum tour for young visitors (each pupil explores the museum on his own; no guided tour in a group) the typical period of a museum visit represents a good example of such a fixed external time constraint. A teacher is unlikely to just say 'let's enjoy the interactive tour, walk around and explore the world of dinosaurs by yourself', but rather add constraints such as 'let's meet all together in 1h at the same place').

2. Dramaturgic aspects and story models. Correlating to the characteristics and the rules-of-thumb story models and narrative structures, plot points should be set at specific times, introductionary explanations shouldn't take too much time, the story climax shouldn't be acquired too early, etc.

3. Importance of content and individual story elements. As far as individual story elements (scenes, specific dialogues, content etc.) are concerned, they are attributed by the author with an indicator for importance with the higher weighted elements preferred. For instance, an author might classify the importance of specific dialogue fragments of a chat station as 'very high', since it provides the answer to a leading question, or 'essential knowledge' which the pupils should take out of the museum visit. In contrast, background information about the artist of a painting might be classified as 'interesting' and should only be selected/visualized to the user if there is enough time for it.

Cooperation between the Story Engine and Learning Engine

The communication between the Story Engine (SE) and Learning Engine (LE) is bi-directional. To help support this communication the SE sends a specific ID together with a parameter as a message to the LE. Based on the ID and the parameter, the LE can determine the Learning Situation, Learning Action and in appropriate cases the success or failure of the Learning Action. The received information is then used to perform updates in both the Skill Assessment Engine (SAE) and Motivational Assessment Engine (MAE). If specific skill probabilities or motivational probabilities, such as attention and confidence, reach a predefined threshold, the LE sends a message with an ID and parameter relating to a recommendation. The SE then receives this message and determines the exact dialog match from the recommendation sent by the LE. This avoids the possibility that the same dialogue is displayed twice to the learner. After the SE has sent the selected recommendation dialogue to the GE the LE is informed about which dialogue was selected. This information is used for a further probability update on the related skill (as the user will have received a hint relating to the skill), although this specific update will not prompt the sending of a further recommendation.

It is important to note that this communication has to be very fast in order to provide the appropriate recommendation at the right time. The flying mission within the 80 Days game is one example in which a recommendation which arrives too late could be confusing to the learner. For example if the learner is flying in the wrong direction for a specific time period the LE will issue a recommendation. However if the recommendation arrives too late the learner may have already corrected the flight direction which would mean the recommendation would be incorrect.

To avoid timing issues the interface between the LE and the SE is based on a TCP/IP interface which supports bi-directional sending and receiving of game evidence, and adaptive recommendations in milliseconds. The TCP/IP interface also provides means to request the skill state of the individual learner, or if needed the probability of a specific skill. The interface between the SE and the Game Engine (GE) is built on function calls allowing a seamless integration of the SE with both the GE and the LE.

Meeting the Challenges of Game-based Learning

This chapter has introduced three main technical challenges of Digital Game-based learning, namely implicitly modeling the learner; providing detailed micro-adaptations to adjust individual learning situations; and offering macro-adaptation that can adjust the number of learning situations in order to introduce more learning objectives. Across these challenges timeliness and consistency must be adhered to. In other words, everything presented to the learner must remain coherent, flow naturally and be offered without any slow down in the gaming experience. The Learning Engine has been discussed as the component of 80Days that is responsible for meeting these challenges. It achieves these by cooperating with the Story and Game Engines. Specifically, the cooperation with the Story Engine must ensure that macro-adaptation is performed in a manner consistent with the storyline being told.

Modeling the learner is divided into two sub-components of the Learning Engine – the Skill and Motivation Assessment Engines. The chapter has described how these engines operate in order to determine information about the learner in a timely manner. Moreover, it has discussed the four-stage process that takes this modeled information and recommends appropriate adaptive interventions to be rendered to the user in the Game Engine.

References

Albert, D. & Mori, T. (2001). Contributions of cognitive psychology to the future of e-learning. Bulletin of the Graduate School of Education, Hiroshima University, Part I (Learning and Curriculum Development), 50, 25-34.

Brusilovsky, P. (1999) Adaptive and Intelligent Technologies for Web-based Education. In C. Rollinger and C. Peylo (eds.), Special Issue on Intelligent Systems and Teleteaching, Künstliche Intelligenz, 4, 19-25.

Balet, O. (2007). INSCAPE An Authoring Platform for Interactive Storytelling. Virtual Storytelling. Using Virtual Reality Technologies for Storytelling. Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 176–177. Retrieved from http http://dx.doi.org/10.1007/978-3-540-77039-8_15

Conlan, O. & Wade, V. (2004) "Evaluation of APeLS - an adaptive

eLearning service based on the multi-model, metadata-driven approach," in Lecture Notes in Computer Science in proceedings of the Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Springer, Eindhoven,The Netherlands, pp. 291 - 295. Retrieved August 12, 2009, from http://www.tara.tcd.ie/handle/2262/22608

Conlan, O., Hampson, C., O'Keeffe, I., Heller, J. (2006). Using Knowledge Space Theory to support Learner Modeling and Personalization, World Conference on E- Learning in Corporate, Government, Healthcare, and Higher Education, Hawaii, USA.

De Bra, P. (1999). Design Issues in Adaptive Hypermedia Application Development, Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, pp. 29-39, Toronto and Banff, Canada, 1999. (Editors P. Brusilovsky and P. De Bra, available as CSN 99/07, TUE, or at http://wwwis.win.tue.nl/asum99/.)

Doignon, J.-P., and Falmagne, J.-C. (1985). Spaces for the assessment of knowledge. International Journal of Man-Machine Studies, 23, 175-196.

Kickmeier-Rust, M., Schwarz, D., Albert, D., Verpoorten, D. & Castaigne, J. (2006) "M.: The ELEKTRA project: Towards a new learning experience," in M3 – Interdisciplinary aspects on digital media & education, Vienna: Österreichische Computer Gesellschaft,

Kickmeier-Rust, M., Peirce, N., Conlan, O., Schwarz, D. et al. (2007) "Immersive Digital Games: The Interfaces for Next-Generation E-Learning?," in C Stephanidis (eds), Universal Access in Human-Computer Interaction. Applications and Services, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 647–656. Retrieved from http://dx.doi.org/10.1007/978-3-540-73283-9_71

Kickmeier-Rust, M & Albert, D. (2010) "Micro-adaptivity: protecting immersion in didactically adaptive digital educational games." Journal of Computer Assisted Learning, vol. 26, no. 2, pp. 95-105.

MacKenzie, I.S. and Colin, W. (1993). "Lag as a determinant of human performance in interactive systems," in Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems Amsterdam, The Netherlands: ACM http://portal.acm.org/citation.cfm?doid=169059.169431

Peirce, N., Conlan, O., Wade, V. (2008). "Adaptive Educational Games: Providing Non-invasive Personalised Learning Experiences," Digital Games and Intelligent Toys Based Education, 2008 Second IEEE International Conference on , vol., no., pp.28-35