Gamification of Prosocial Learning
for Increased Youth Inclusion and Academic Achievement

D4.2 - 2nd Intelligent Adaptation and Personalization
This document describes algorithms for online and offline adaptation of games.

Author(s)  Konstantinos Apostolakis (CERTH), Kosmas Dimitropoulos (CERTH), Petros Daras (CERTH)

Contributor(s)  Kiriakos Stefanidis (CERTH), Athanasios Psaltis (CERTH)

Reviewer(s)  Kam Star (PLAYGEN), Eric Robertson (Redikod)

Dissemination level  
- [ ] internal
- [x] public
- [ ] confidential

Document Control Page

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Modified by</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>09/01/2017</td>
<td>Konstantinos Apostolakis</td>
<td>First version draft</td>
</tr>
<tr>
<td>2.0</td>
<td>12/01/2017</td>
<td>Kosmas Dimitropoulos</td>
<td>Second version draft</td>
</tr>
<tr>
<td>3.0</td>
<td>27/01/2017</td>
<td>Konstantinos Apostolakis</td>
<td>Ready for internal review</td>
</tr>
<tr>
<td>4.0</td>
<td>06/02/2017</td>
<td>Konstantinos Apostolakis</td>
<td>Final Version</td>
</tr>
<tr>
<td>Final version</td>
<td>07/02/2017</td>
<td>Pilar Pérez</td>
<td>Format review and final version</td>
</tr>
</tbody>
</table>
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PsL</td>
<td>ProsocialLearn</td>
</tr>
<tr>
<td>PAM</td>
<td>ProsocialLearn Adaptation Manager</td>
</tr>
<tr>
<td>PLO</td>
<td>Prosocial Learning Objective</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>OL</td>
<td>Online Learning</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>AU</td>
<td>Action Unit</td>
</tr>
<tr>
<td>NN</td>
<td>Neural Network</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
</tbody>
</table>
Executive summary

This deliverable is part of work-package four **WP4: Dynamic and Personalized Game Elements for Prosocial Learning**. It is a public document focusing on the results and outcomes comprising the contents of **Deliverable 4.2: 2nd Intelligent adaptation and personalization**. This report will: a) outline the final version of the Prosocial Adaptation Manager (PAM) component which is to be integrated with the ProsocialLearn platform; b) describe a methodology for multimodal recognition of student engagement during gameplay using consumer-grade image and motion-sensing devices; c) elaborate on mechanisms for both online and offline adaptation of games and content towards increasing their effectiveness; and d) demonstrate an example implementation for clarity and assessment of the results. In this respect, this report is both an update, as well as a complementary document to D4.1: 1st Intelligent adaptation and personalization, and readers are encouraged to read through that report first to better grasp the contents described in this deliverable.

This document will be made available on the project website for external parties interested in adaptive and personalized games for optimized learning. It is hoped that the report will assist interested parties in understanding intelligent configuration of prosocial games, and dynamic recommendation of game adaptations for personalized learning.
# Table of Contents

1 Introduction .................................................................................................................. 7  
   1.1 Scope .................................................................................................................... 7  
   1.2 Document structure ............................................................................................... 7  
2 ProsocialLearn Adaptation Manager ............................................................................. 9  
   2.1 PAM Offline & Online Mechanism approach ......................................................... 9  
   2.2 PAM Architecture .................................................................................................. 9  
   2.3 PAM within the ProsocialLearn platform architecture ........................................... 14  
3 Multimodal Engagement Recognition ......................................................................... 17  
   3.1 Introduction .......................................................................................................... 17  
   3.2 Multimodal Affective State Recognition ............................................................... 18  
   3.3 Game-play Features .............................................................................................. 20  
   3.4 Engagement Recognition ..................................................................................... 21  
   3.5 The two versions of the "Path of Trust" game ....................................................... 22  
   3.6 The Engagement Experiment .............................................................................. 24  
4 Offline Adaptation ........................................................................................................ 27  
5 Online Adaptation ........................................................................................................ 30  
   5.1 Introduction .......................................................................................................... 30  
6 Utilizing the ProsocialLearn Adaptation Manager in a sample game .......................... 35  
   6.1 Path of Trust ......................................................................................................... 35  
   6.2 Offline adaptation in Path of Trust ....................................................................... 41  
   6.3 Online Adaptation ............................................................................................... 43  
   6.4 Small Scale Study ............................................................................................... 46  
7 Conclusions .................................................................................................................. 53  
8 References ................................................................................................................... 54  
Appendix A: Prosocial Adaptation Manager (PAM) Manual ........................................... 57  
   A.1 Introduction .......................................................................................................... 57  
   A.2 PAM-Supported indicative adaptation architecture ............................................ 57  
   A.3 Usage Requirements ........................................................................................... 58  
   A.4 Instructions for game developers to use the PAM in their own games .................. 59
A.5 Messages .............................................................................................................................................. 60
A.6 Instructions for executing the PAM on a local machine after installation for testing ........... 60
1 Introduction

This section provides detailed information about the purpose of the Prosocial Adaptation Manager in general, as well as the scope and structure of the document, which set the tone for the intended audience and interested readers.

1.1 Purpose of the document

This document presents the role, system architecture, and constituent adaptation approaches of the ProsocialLearn Adaptation Manager (PAM). As part of the activities carried out in Task 4.1 and leading up to this deliverable, we explain how the Adaptation Manager module is built following the guidelines presented in document D2.6 “Prosocial Game Design Methodology”.

1.2 Scope and Audience of the document

As has been described in deliverable D4.1 “1st Intelligent adaptation and personalization”, the adaptation manager should support adaptation triggered both online, i.e. during gameplay, to increase player engagement and achieve maximum potential for setting up learning outcomes, as well as offline matching of players to specific game scenarios, providing insight to game developers on how to introduce computational factors to generate different sets of circumstances that aim at balancing player ability to game context. In this respect, the adaptation system could be seen as a matching mechanism that adjusts game elements and receives feedback from the user about his/her experience with that element.

In this deliverable, we expand upon the concepts, contents and principles first presented in D4.1 targeting both online and offline adaptation, triggered to personalize and appropriately tailor prosocial content for maximizing individual players-students’ learning of prosocial skills. In this respect we will present the algorithms and processes used for online and offline adaptation, as well as the multimodal engagement recognition algorithm developed within the framework of the ProsocialLearn project. We therefore provide updated descriptions of the different components originally described in D4.1, expanding upon these differences where necessary.

1.3 Document structure

This document contains the following key sections, conveniently detailed in the list below:

- **Section 1: Introduction** – an introductory section, i.e. this present section, which describes the purpose of Task 4.1 that generated this document.
- **Section 2: ProsocialLearn Adaptation Manager** – in this section, the role and purpose of the PAM are presented.
- **Section 3: Multimodal Engagement Recognition** – this section outlines the proposed multimodal engagement recognition algorithm, which is used as input to the online adaptation mechanism.
- **Section 4: Offline Adaptation** – this section provides a description of the offline adaptation module of the PAM.
- **Section 5: Online Adaptation** – this section presents the online adaptation algorithm developed within the ProsocialLearn project.
• **Section 6: Utilizing the ProsocialLearn Adaptation Manager in a sample game** – this section presents an example that demonstrates how the PAM can be utilized with the sample prosocial game Path of Trust.

• **Section 7: Conclusion** – this section presents the conclusion of the work described in this document.

Finally, the Appendix contains the manual of the Prosocial Adaptation Manager (PAM), which provides detailed instructions to game developers on how to integrate the PAM modules in their own games.
2 Prosocial Learn Adaptation Manager

This section outlines the architecture and organization of the Prosocial Adaptation Manager (PAM)'s components. The Section serves as a summary of the main architectural description provided in the previous version of this deliverable (D4.1), updated to the latest changes and configurations made to operationalize the component for small scale studies (Section 6). This Section concludes with positioning of the PAM architecture within the overall architecture of the ProsocialLearn platform.

2.1 PAM Offline & Online Mechanism approach

The PAM is divided into two parts, namely the offline and online adaptation mechanisms. The distinction concerns whether the processing takes place during gameplay or before, in the loading phase of the game. These two mechanisms aim to personalize the prosocial games towards maximizing the players’ engagement during gameplay, or select appropriate settings (i.e. scenarios) for demonstrating prosocial skills acquisition. The reason for having two distinct mechanisms is to provide our platform with better personalization capabilities. Each mechanism processes different information about the player and concerns different types of factors affecting engagement and prosocial behaviour. More specifically, offline adaptation is based on persistent player information that concerns their prosocial skills performance, as defined by the specific constraints of the Prosocial Learning Objectives (PLOs). It aims to match players to the proper game scenario to provide the most appropriate challenge settings (defined by the game developer) for carrying out the appropriate prosocial skills to succeed. On the other hand, online adaptation considers real-time player data concerning player’s engagement estimation through multimodal sensor input data fusion sent at specific time intervals during the game (see Section 3). This mode of adaptation aims at matching players to the most appropriate game elements (again, defined by game developers) to best tailor the game to that player’s preferences In the paragraphs that follow, more detailed information about the two mechanisms is provided.

2.2 PAM Architecture

2.2.1 Components

A UML diagram of the PAM’s components can be seen in Figure 1. A dynamic user profile is employed in order to account for players’ behavioural shifts over time.

The internal structure of the PAM consists of a set of interrelating components that exemplify the adaptation algorithms described in Sections 3 and 4 of this report. Each component depicted in the UML diagram shown in Figure 1 supports a specific functionality, which is briefly described in the bullet list below:
• **Data Representation.** The PAM preserves its own data representation scheme containing all necessary data types for storing information in memory. These data structures include:
  
  o The game information structure, keeping information about the scenarios and elements that a particular game on the ProsocialLearn platform supports.
  
  o The player prosocial skills profile, containing the player ability estimates for all PLOs defined across all games on the ProsocialLearn platform.
  
  o The player engagement profile, containing information about the player’s engagement estimations for the game’s elements accounting for both positive reinforcement as well as corrective feedback on their performance of the required prosocial skills.
  
  The above mentioned data structures and PAM persistence mechanism are all maintained within the \textit{PsLAdaptationBase} internal PAM component. Some basic auxiliary functions usable by all components are also defined within the same component.
  
• **Information Storing.** All data collected for a user are stored into files. The memory data structures are converted to text format and stored in an XML form. More information on this
data storing mechanism will be presented in Section 2.2.3. The actual Save and Loading functions are provided through the \textit{PsLAdaptationXML} internal PAM component.

- **Learning Algorithms.** The PAM’s adaptation components that support both online and offline adaptation utilize online learning (OL) algorithms, as they make personalisation more evident to the user, due to their ability to handle his or hers behavioural shifts and independence from the context of the learning task. As a result, both online and offline adaptation processes supported by the PAM utilize two different OL algorithms.
  
  o An ability ranking system is included to support offline adaptation, in order to match the ability demonstrated by the player towards properly using a certain prosocial skill to the appropriate scenario that is expected to adequately challenge that player’s roleplaying of the same skill within a different context.
  
  o Online adaptation utilizes the engagement monitoring of the player. Through an analysis of the most recent engagement estimates, the PAM is able to select a game element, or group of game elements, that is expected to maximize that player’s engagement, and as a result support the learning process through the proper means of administering either positive reinforcement or corrective feedback during play.

Both OL algorithms are implemented within the \textit{PsLAdaptationCore} internal PAM component.

- **Information Update.** The core algorithms of the PAM described in the previous bullet, construct and exploit dynamic user profiles in order to provide adaptations to the games. Both the prosocial skills as well as the engagement profile of a single user are updated at specific moments within the game. These moments correspond to game states right after a player has performed a specific prosocial skill to overcome a game task designed to test the player’s ability in using that skill, in which an engagement estimate is requested to make the necessary online adaptations (if any). More specifically, the engagement profile is updated during actual gameplay time through communication with the fusion module, while the player’s prosocial skills profile is updated at the end of each game using the scores achieved by the player when using the skill to overcome a series of game tasks throughout the current game session. These processes are implemented within the \textit{PsLAdaptationUpdate} internal PAM component.

- **External Communication.** Communication of the PAM with the other ProsocialLearn platform components must be established in order to exchange information with games. This functionality is provided through a separate, high-level abstraction component within the PAM, \textit{PsLAdaptationModule}, which ensures flexibility on any changes in communication protocols and is responsible for initializing both online and offline adaptation procedures.

Concerning the internal PAM component communication, all of the aforementioned PAM components utilize \textit{PsLAdaptationBase} in order to define and use the data types used in the PAM, as is shown in the UML diagram above. The \textit{PsLAdaptationModule} uses the \textit{PsLAdaptationCore} defined functions to transfer the requests of the game for adaptation. The \textit{PsLAdaptationCore} therefore uses \textit{PsLAdaptationXML} to load the required data at the start of each game session, and executes the learning algorithms using the \textit{PsLAdaptationUpdate} component in order to update player profiles, as
well as other auxiliary data structures. Finally, at the end of each game, the \textit{PsLAdaptationCore} uses the \textit{PsLAdaptationXML} to store the session-specific data into XML format for future use.

2.2.2 Information Exchange

Following a generalized approach in order to be able to support multiple games, information exchange within the PAM component follows the specific pattern described in this Section. At the start of each game session, information is exchanged between the game, the ProsocialLearn platform, and the PAM. Initially, the game should notify the platform about the IDs for that specific game’s scenarios and online adaptation elements (for both positive reinforcement and corrective feedback) that are supported. The platform should then forward this information to the PAM\(^1\). After loading existing data, (or creating it, if need be) the PAM waits for a signal about executing the ability ranking system for selecting the proper game scenario ID through the offline adaptation mechanism, and thus properly initialize the game. This signal should be sent by the game prior to its launch.

During gameplay, the online adaptation mechanism implemented within the PAM receives input generated by the multimodal fusion module (see Section 3) and uses the data to update the dynamic user model maintained for player engagement. During this process, the mechanism might detect the need for making changes to the current set of game elements towards maximizing the player’s engagement estimate. This process, described in detail in Section 5, utilizes past engagement data to replace elements currently associated with the low engagement estimates with others that might trigger a more vigorous response from that particular player. The PAM will thus respond to the game after triggering the online adaptation mechanism with the proper Element ID, which in-game adaptation response processes should decode to make the proposed changes to the game. An example of this process for a sample prosocial game is illustrated in Section 6.

During the game, players will face tasks that challenge their ability to properly use the prosocial skill with respect to a number of factors, as described in D4.3. The game keeps an internal scoring mechanism (as described in Section 3.3), the outcome of which is communicated by the game to the PAM at the end of the session, thus allowing the PAM to update the ability of the player with respect to the corresponding PLO.

The overall information exchange diagram describing the above processes is shown for better clarity in Figure 2.

\footnote{\(^1\) It should be noted here that all communication between the game and the PAM is realized through a web sockets interface. In this report, it is accepted that the game communicates to the PAM via the ProsocialLearn platform, which forwards the game-specific requests through a web socket connecting to the designated port PAM is listening to (defined by the backend platform provider).}
2.2.3 Data Storage

As previously mentioned, the PAM utilizes dynamic player profiles, maintained for both online and offline adaptation mechanisms in order to estimate the preferences of a student, and make personalized adaptations to the games he/she plays. The PAM-specific player profile data include:

- **The prosocial skills profile.** A single XML-based profile description composed of the players’ abilities for each of the PLOs currently supported across all games in the ProsocialLearn platform.

- **The engagement profiles.** A profile composed by the engagement estimates for each of the online adaptation elements supported by a single game. Therefore, a separate engagement profile will be created for a particular user for each game that that user has ever played.

The PAM maintains its own persistence mechanism which stores all profile data into XML files. The XML schema is organized under three categories: Game Data, Prosocial Skills Data, and Engagement Data. An example of that schema is shown in Figure 3.

- **Game Data** contains information about the game such as the scenario IDs and element IDs for both positive reinforcement as well as corrective feedback elements. It also contains the game scenarios’ ratings for prosocial ability (see Section X). A single XML file is created for each game.

- **Prosocial Skills Data** contains information about the player’s prosocial skills profile. A player’s prosocial skills profile consists of that particular player’s ability estimations for every
prosocial learning objective (PLO) being tested through in-game tasks according to the game design.

- **Engagement Data** contains information about the player’s engagement profile. A player’s engagement profile consists of estimations generated by the multimodal fusion module (See section 3). For a specific game, the player’s engagement profile is composed of engagement measurements for every element that the player has had an in-game experience with in that particular game.

Examples of how this data is relevant to each game and how it should be considered by the game developers to design their in-game prosocial skills task challenges will be demonstrated in Section 6.

![Diagram](image)

Figure 3: XML schema for storing player information in Adaptation Manager

### 2.3 PAM within the ProsocialLearn platform architecture
The PAM has been developed in accordance to the 2\textsuperscript{nd} System Requirements and Architecture (D2.4) description of the adaptation architecture components (Figure 4). As is described in that deliverable, one of the teacher’s crucial role within the ProsocialLearn vision is to identify his/her students’ individual preferences in order to setup the ideal conditions that will make the learning process as effective as possible. As will be described in more detail in the later Sections of this document, the role of the Prosocial Adaptation Manager (PAM) is to automatically match the preferences of players/students to the most effective game elements and scenarios in an attempt to maximize the potential benefit of playing prosocial games towards achieving real learning outcomes. In this respect, offline adaptation (Section 4) is based on historical player information with respect to their in-game achievements defined by the game’s modelling of Prosocial Learning Objectives (PLOs), to achieve a proper matching between players and proper game scenarios to setup the learning environment, while online adaptation aims to estimate and maintain players engagement throughout the game session in order to increase the likelihood of players actually learning through prosocial gameplay.

Within the ProsocialLearn architecture, one PAM instance is instantiated for each game instance, thus handling individualized offline and online adaptation for that particular game. The PAM instance is independent of any other PAM instances running in parallel for a different game instance or class. Whenever a game instance is initiated, the corresponding PAM instance is initialised by loading the designated player’s profile data from the set of students in the Learning Group from the Learning Record Store (LRS), along with the related Prosocial Learning Objectives supported by the game and defined by the game developers for the teacher to support for the lesson in prosocial behaviour. Throughout gameplay, the PAM will periodically connect to the LRS component to retrieve player prosocial state (i.e. engagement and affect data, as well as PLO scores at the end of the session, as demonstrated in the UML diagram shown in Figure 2 back in Section 2.2.2). The LRS therefore acts as an information broker between the multimodal engagement fusion components and the PAM, in that it receives data on player valence/arousal and engagement from the fusion algorithms, as well as recording PAM-generated PLO outcomes achieved by the players at the end of the game session. The PAM uses the information to calculate skill rankings for individual students in relation to the group, taking into account all games designed to administer lessons with respect to the particular
PLO, including the specific game situations played. The ranking scores are used to calculate student performance and to make recommendations for offline adaptation across all games including that PLO. In addition, each game offers game elements with points of variability classified according to the Prosocial Learn vocabulary for adaptation. If the PAM determines that the student would benefit from online adaptation, then the PAM recommends a change by asserting an adaptation statement to the game server via the messaging component. As is shown in Figure 4, any online adaptation statements should also be stored within the LRS.
3 Multimodal Engagement Recognition

3.1 Introduction

Engagement is an important determinant for user’s interaction with technology. The measurement of user engagement enables not only the better design of interactive applications, but also the development of intelligent, sophisticated and adaptive environments. This is mainly due to the fact that engagement plays a key role in better understanding general user behavior and overall efficacy of goal or task-oriented behavior within computer-based environments (Wiebea et al., 2014), such as social networks, video-games, web-based applications or educational environments. In the scientific literature and especially concerning the topic of education, student engagement has been closely associated with various academic, behavioural and social factors, as there are numerous studies dating back decades, which show that low student engagement leads to poor academic performance or even high drop-out rates (Larson et al., 1991).

Student engagement is a very complicated and multifaceted phenomenon with different dimensions and therefore, there are various definitions of its term in the literature. Newmann (Newmann, 1992) first discussed the importance of engagement in the educational process defining student engagement as the student's psychological investment in and effort directed toward learning, understanding, or mastering the knowledge, skills, or crafts that academic work is intended to promote. Additional studies investigated the relationship between school engagement and dropping out (Sullivan, 2009)(Fredricks, 2004), while other researchers identified the engagement in terms of autonomy (Parson & Taylor, 2011)(Reeve, 2004). In a more recent study, Zepke et al. (Zepke et al., 2010) also suggested that engaged students usually succeed on their activities when they are intrinsically motivated and feel capable of working in an autonomous way. On the other hand, Parsons and Taylor (Parsons & Taylor, 2011) argue that only performing the activities is not enough to identify the engagement of students. They'd rather also inspect the qualitative aspect of time spent on a task, indicating a time window in which activities should be performed to regard their actors as being engaged throughout performing these activities. Hence, students show their commitment by managing time efficiently while working on deadlines (Akey, 2006)(Saeed & Zyngier, 2010). Additionally, other studies revealed that engagement can also be evaluated from the perspective of collaboration and teamwork (Bulger et al, 2008)(Zepke, 2010), while recent research works have focused mainly on the strong relation between fun, satisfaction and engagement (Prensky, 2002)(Shernoff, 2003)(Chatterjee, 2010).

In order to better study the role of engagement in learning, Finn’s (Finn et al, 1995) introduced the “Participation-Identification” model, which makes a clear distinction between behavioral and affective engagement (Culver, 2015- Interested readers are encouraged to refer to D4.1 Section 5.2.1 for a detailed description of these engagement types). More specifically, the behavioral dimension of the model is related to the degree of students’ participation in the learning process, while identification refers mainly to the students’ emotional attitude towards learning, i.e., the students’ affect within the classroom and the sense of belonging at school. Later, Fredericks et al. (Fredericks et al.,2004) proposed one of the most frequently cited models of student engagement that is based on three distinct components: behavioral, affective and cognitive engagement. The latter includes features such as problem solving, preference for challenging work, investment in learning,
metacognitive strategies and persistence during difficult tasks (Anderson et al., 2004). More recently, other researchers (Appleton et al., 2006) have further proposed that engagement comprises four subtypes, such as academic, behavioral, cognitive, and psychological.

Within the ProsocialLearn project, we developed a novel multimodal student engagement recognition approach for serious games in education inspired by the theoretical grounds of educational psychology. The proposed approach aims to capture the different dimensions of engagement, i.e., behavioral, cognitive and affective, based on the combination of real-time engagement cues from different input modalities, such as facial expressions, body postures and various game events. For the combination of different dimensions of engagement, a machine learning approach, based on artificial neural networks, was adopted, while engagement labelling data were acquired through retrospective self-reports, i.e., Game Engagement Questionnaires - GEQs (see Section 3.4).

### 3.2 Multimodal Affective State Recognition

Facial motion plays a major role in expressing emotions and conveying messages. In our experiments, we used Kinect SDK’s face tracking engine for the extraction of facial features. The engine is able to track facial muscle activities, i.e., Action Units (AUs), which can be seen as a form of mid-level representation of student’s facial expressions. To classify expressions into the six basic emotion categories (anger, disgust, fear, happiness, sadness, and surprise), we concatenated the posteriors of all AUs in a unified vector representation and we trained a neural network using the RGB-D bimodal database described in (Psaltis et al., 2016), as shown on the left part of Figure 6.

Body movements, body postures, or the quantity or quality of movement behavior in general, can also be of help to differentiate between emotions (Wallbott, 1998). To this end, we decided to extract a number of 3D body features, which are deeply inspired by the relevant psychological literature (Kaza et al., 2016)(Piana, et al., 2013). The 3D body movement features are extracted from joint-oriented skeleton tracking using the depth information provided by the Kinect sensor. More specifically, the extracted features are classified into the following broad categories: i) kinematic related features: kinetic energy, velocity and acceleration, ii) spatial extent related features: bounding box, density and index of contraction, iii) smoothness related features: curvature and smoothness index, iv) symmetry related features: wrists, elbows, knees and feet symmetry, v) leaning related features: forward and backward leaning of a torso and head as well as right and left leaning and vi) distance related features: distances between hands, distance between hand and head as well as hand and torso. An example of the kinetic energy measurement during the play of the "Path of Trust" prosocial game, is demonstrated in Figure 5.
The multimodal fusion process is responsible for measuring the affective state of a student using a series of visual cues. More specifically, given a sequence of Kinect’s data streams, we extract feature vectors from users’ facial expressions as well as from body gestures, as is described in (Psaltis et al., 2016). Then, the extracted feature vectors are fed to separate unimodal classifiers. After learning each Neural Network (NN) model, the posteriors of the hidden variables can then be used as a new representation for the data. By adopting unified representations of the data, we can learn high-order correlations across modalities. Deeper networks with fewer hidden variables can provide simpler and more descriptive model. Therefore, we consider training an Artificial Neural Network (ANN) over the pre-trained layers for each modality, as motivated by deep learning methods. We stack ANNs and train them layer-wise by starting at the base layer and moving up. This is a directed model since there is no feedback from higher layers to the lower layers, as shown in the lower part of Figure 6. This layer-wise architecture improves performance, while avoiding overfitting.
The multimodal affective state recognition is a frame-by-frame process. In order to have an indication of the player’s affective engagement, i.e., a total measurement of player’s affective activity during gameplay, we initially map the dominant student emotion (i.e., the emotion with the highest probability in each frame) to the Valance-Arousal (V-A) space and then we estimate the average variation of player’s affective state. For the mapping of the dominant emotion to the V-A space, the 2D position of the dominant emotion is estimated as follows:

\[
\begin{aligned}
V_d &= V_m + \lambda (1 - p) \sigma_d \\
A_d &= A_m + \mu (1 - p) \sigma_d
\end{aligned}
\]

where \(V_m\) and \(A_m\) are the coordinates of the mean value of the dominant emotion in V-A space, \(\sigma_d\) indicates its standard deviation (Piana et al., 2013), \(p\) is the probability of the dominant emotion, while factors \(\lambda\) and \(\mu\) are scalars with values equal to ±1 depending on V-A values of the second most dominant emotion (e.g., \(\lambda=1\) if the valance value of the second most dominant emotion is greater than \(V_m\)). After mapping the emotions to V-A space, the average variation \(D_a\) of student’s affective state can be easily calculated as follows:

\[
D_a = \frac{1}{F} \sum_{i=2}^{F} \|x_{i-1} - x_i\|^2
\]

where \(F\) is the total number of frames and \(x_i\) indicates the current affective state of the student in V-A space.

3.3 Game-play Features

Student’s interactions with the game can provide valuable information for the two other engagement dimensions, i.e., behavioral and cognitive. Towards this end, we extract features based on the analysis of specific game-play events and their corresponding time-stamps in order to achieve a more targeted measurement of these two aspects of engagement. According to the literature, the dimension of behavioral engagement is defined as focused activity on a task, with a typical measurement being time on task (Hookham et al., 2016)(Annetta et al., 2010) while playing the video-game. On the other hand, cognitive engagement is defined as mental activity associated with the presented content and is measured by successfully achieving the desired goal of the game or by pre and post testing of outcomes (Annetta et al., 2010). In the approach described in this deliverable, the behavioral engagement of the student is measured by estimating his/her average time of responsiveness \(R \in [0,1]\) in all challenges \(c_i\) of the game (e.g., in “Path of Trust”, a challenge can be the collection of a diamond), with \(i=1,2,...,n\):
\[ R = \frac{1}{n} \sum_{i=1}^{n} R_{i} = \frac{1}{n} \sum_{i=1}^{n} \frac{t_{ci}^{i}}{t_{total}^{i}} \]

where \( t_{ci}^{i} \) and \( t_{total}^{i} \) indicate the student’s time of responsiveness and the total available time in challenge \( c_{i} \), respectively.

Similarly, for measuring the cognitive engagement of student during gameplay, we estimate whether the desired goal has been achieved in each challenge \( c_{i} \) (e.g., whether student has decided to cooperate with and trust the Guide or follow a different strategy that leads to a non-prosocial behavior, the number of selected diamonds out of the total available diamonds or the number of monsters/traps that the player has avoided out of the total number of monsters/traps in a challenge) and we estimate the total score \( S_{r_{j}} \) in each task \( r_{j} \) of the game, with \( j=1,2…m \), (each task can contain one or more challenges). Finally, we estimate the average score \( S \in [0,1] \) of the student in all tasks of the game:

\[ S = \frac{1}{m} \sum_{j=1}^{m} S_{r_{j}} = \frac{1}{m} \sum_{j=1}^{m} \frac{g_{r_{j}}^{j}}{g_{total}^{j}} \]

where \( g_{r_{j}}^{j} \) and \( g_{total}^{j} \) indicate the number of successfully achieved goals and the total number of goals in a task \( r_{j} \), respectively.

We have to note here that the above metrics/features are normalized and are completely independent of the game, that is, a game developer can easily estimate the values of \( R \) and \( S \in [0,1] \) in any serious game by simply defining the challenges, the tasks, the available time and the goals of his/her game.

### 3.4 Engagement Recognition

According to D4.1,

- **Behavioral engagement** is defined as focused activity on a task, with a typical measurement being time on task.
- **Cognitive engagement** is defined as mental activity associated with presented content, and is measured by successfully achieving the desired goal of the game (e.g. 6 out of 6 diamonds collected), or by pre and post testing of outcomes.
- **Affective engagement** relates to emotional responses of players to game content.

Having defined the parameters, i.e., the three dimensions, of our engagement model, we subsequently need to annotate our data in order to label them for the training of our classifier.
Towards this end, we adopted a retrospective self-reports approach, based on GEQ questionnaire, for both games, as it is described in detail in the “Engagement Experiment” section. Figure 7 illustrates the GEQ measurement approach (Brockmyer et al., 2009) and shows how player’s answers are mapped to the engagement scale. More specifically, symbols N, M, and Y displayed in Figure 7 refer to “No”, “Maybe”, and “Yes”, respectively to each question displayed on the right. Since each answer corresponds to a specific engagement value, we aggregated the values of all answers and we estimated the average engagement value for each gameplay.

![Figure 7: GEQ measurement approach (Brockmyer et al., 2009).](image)

Finally, the estimated engagement values were used as labels of the recorded data, i.e., engagement vectors $E=[D_a, R, S]$, for the training of an ANN. In our experiments we used two classes, “Not Engaged” and “Engaged”, however, one can add more classes, e.g., “Nominally Engaged” or “Very Engaged”.

### 3.5 The two versions of the "Path of Trust" game

To measure the student engagement instead of relying only on self-reports or external annotations, as is commonly done, in our study we used two different versions of the same prosocial game, namely “Path of Trust”, with different degree of challenge (we developed a stripped-down, intentionally “boring” version of the game, to complement the more challenging version of the original) in order to trigger different levels of engagement to the users.

More specifically, some minor modifications were made to the original, single player version of the game (Apostolakis et al, 2016), in order to detect player responsiveness to game events altering the outcomes of players’ actions:

First of all, we placed two items in each corridor dungeon piece (instead of the default one-item-per-corridor rule found in the original "Path of Trust" game, as explained in Section 6.1), as well as implemented a random item switching function. This function randomly determined whether one of the items in the corridor (Treasure Piece) would instantly be replaced with a Mummy hazard, slightly before players touched the item and collected the points to be gained from it. The rationale for
including this game event lies within measuring the time it takes players to notice the swap (i.e. being attentive of the game), and subsequently react to it, as they attempt to avoid the surprise hazard. A logging function was also implemented to keep track of the time, responsiveness, accomplished outcome and player affect during these events.

Additional to the aforementioned changes and in order to isolate the behavioural, cognitive and affective dimensions of student engagement using real-time facial, body and in-game engagement cues, we heavily modified the single player version of the original game in a separate build, stripping down content and producing an intentional, non-challenging version of the game. In it, all colorful 3D graphics, textures, music and sound effects were replaced by simple, geometric shapes. Instead of the two character avatars, the player now controlled a simple cube totem, as seen in Figure8. The game mechanics were slightly altered as well. Since all narrative elements were taken out of the original game, there was no exchange of information taking place between the player and the AI-controlled Guide. Instead all the doors hindering the player from knowing what lies in the next room (thus contributing to spatial immersion in the game world) were omitted from this version, providing players with complete knowledge of the items lying in their path. As a result, this game version provided no challenge whatsoever, as the game’s slower pace and absence of closed doors allowed players plenty of time to move their totem out of harm’s way (red totems) and line it up to the treasure pieces (green and blue spheres). This was intentionally designed to hinder the game from supporting the fundamental psychological need for competence, as described in self-determination theory (SDT) (Deci & Ryan, 2000). According to SDT, personal well-being is believed to be enhanced when one’s actions and interactions satisfy, among other things, a sense of efficacy. Maintaining interest and loyalty of players in video games is directly linked to competence need satisfaction described in the SDT (Weinstein & Ryan, 2010): too much of a challenge can lead to frustration, while too less of a challenge can ignite boredom. We therefore expected players to report no interest in re-visiting the stripped-down version of “Path of Trust”, as opposed to the original, which would allow us to trigger different levels of engagement during gameplay in a way that would make them detectable through the multi-modal, consumer-grade sensing devices used for mandatory game control (i.e. Microsoft Kinect).
3.6 The Engagement Experiment

Participants in our study were a total of $N=72$ children from three primary schools. The students ranged in age from 8 to 10 years old. In total, 38 boys and 34 girls completed the entire session. The study was approved by the Institute of Educational Policy (IEP), a private legal entity under the supervision of the Greek Ministry of Education, Research and Religious Affairs.

The study took place in four different classes assigning around 16-20 children per classroom. Our user study coordination team assigned two persons per classroom to carry out the experiments, with one person hosting the game and the other assisting children with regards to the printed questionnaires queries. One desktop PC, equipped with a Kinect for Xbox One sensor and with pre-installed versions of the single player Path of Trust games described in Section III, was situated within each classroom. Students were asked to play both versions of the game in succession, with a short time for filling out the GEQ in between sessions. A short description of the Path of Trust back-story was briefly delivered by the game host, along with instructions on the game gesture-driven interface with regards to moving the player character and touch to collect / avoid hazards mechanics. In order not to build-up any expectations about the games and thus receive a more genuine response with respect to each version’s appeal and motivational affordances, we allowed children to play the stripped-down version of the game first.

After playing, each child was asked to answer a GEQ (with the help of the assigned experiment coordinator to clarify questions which could confuse or were seen as somewhat difficult to assess by children on their own) to evaluate player’s engagement and overall game experience. The process was subsequently repeated with the original version of the game. Each game session had pure gameplay duration worth two and a half minutes (150 seconds) of time. In total, each child took approximately 10-15 minutes to complete the study.
Aggregate results from GEQ questionnaires for both versions of the Path of Trust game demonstrate a clear preference of children towards the original version. A Statistical analysis can be seen in Figure 10. Judging from the normalized representation of player individual engagement levels superimposed on the graph displayed in Figure 11, we clearly see the majority of participants showed a clear (and in many cases, significant) preference for the original version.

For the evaluation of the proposed methodology, we applied a cross validation approach with four folds and we compared its performance against that of the three distinct dimensions and their combinations in pairs. More specifically, as shown in Figure 12, the proposed methodology, which is based on the combination of behavioral, cognitive and affective engagement, outperforms all other approaches with a classification rate of 85%. As we can also see, the role of sensors (i.e., the affective dimension of engagement) is crucial in the classification process with a detection rate of 78% against 73.3% and 75% for behavioral and cognitive engagement, respectively.
Figure 11: Students’ individual engagement scores, based on GEQ analysis. Red ‘x’ marks indicate the engagement level of players corresponding to the stripped-down version of the game, while blue circles correspond to the PoT.

Figure 12: Comparison of the proposed methodology against behavioral, cognitive, affective engagement and their combinations in pairs.
4 Offline Adaptation

Offline adaptation is realised in the loading phase of the game. Its purpose is to select game conditions that are expected to drive the player towards expressing prosocial behaviour. These conditions are referred to as game scenarios. Every scenario belongs to a certain Prosocial Learning Objective (PLO) that is determined by the teacher before the game starts. These PLOs concern the prosocial ability that the teacher intents to train the student at. Each game is capable of utilizing one or more PLOs and for each of them contains a pool of game scenarios offered for offline adaptation.

Before a game begins, the teacher selects the learning objective for training a specific prosocial skill and a registered PsL game that supports it. At the start of each game, the PsL platform exchanges information about the scenarios IDs for the specific game and communicates with the Adaptation Manager which in turn, checks if stored data concerning the active player and the game exist. If data exist, then the game loads the information and fills-in the data structures needed for both offline and online adaptation mechanisms. In cases where there is no existing data, the data structures are created for the specific game and player. The offline adaptation executes the ability ranking system and the player is matched with a scenario based on his or hers ability level. This scenario is then used to initialize the game.

More specifically, the ability ranking system developed within ProsocialLearn is based on the well-known Elo rating system, which was initially proposed by Arpad Elo (Elo & Sloan, 1978) for measuring the ability of chess players. Inspired by the work in (RAGE Project, 2015), where game scenarios were matched to players according to their competence, in ProsocialLearn we adopted a similar ability matching methodology to model the players’ prosocial ability. Hence, in ProsocialLearn, the Elo
computation is based on pairwise comparisons between players and scenarios. In particular, the new rating $R_i'$ of player $i$ can be easily estimated as:

$$R_i' = R_i + K(S_i - E(S_i)),$$

where $K$ is a weight for rating the uncertainty for $i$, $S_i$ is $i$'s score (there are 3 possible scores $S_i$ for every PLO: 1 for prosocial win, 0 for failure and 0.5 for any other case), and $E(S_j)$ is $i$'s expected probability of winning. The uncertainty $K$, controls the rate of convergence of the system. That is, small values of $K$ make the system learn the ability of a player slowly, while a large $K$ would mean that the system relies its estimation only on recent events and for that it is inaccurate.

The expected score $E(S_j)$ for a player $j$ concerns what the system expects for an outcome of a certain match, considering the ratings for both players. Its definition is given by

$$E(S_j) = \frac{1}{1 + 10^{(S_i - S_j)/N}},$$

where, $N$ is the total games played among players $j$ and $i$. As we see from the above definition, the expected value is a function of the difference of the two components ratings. That is, if the difference is negligible, the expected value will tend towards 0.5, while large differences force the result towards 0 or 1 according to which player preserved the largest rating.

![Figure 14: Prosocial ability and matching to scenarios](image)

Similarly to a prosocial profile for a player, the prosocial profile of a game is the aggregation of the ratings for all the scenarios that it supports. In the same manner as the player profile updates, the scenario ratings are been computed by the experience (outcomes) that players have with the game’s scenarios.
Before each game begins, the offline adaptation mechanism can determine the proper scenario for the player, namely, the one that preserves those conditions that maximize his or hers prosocial ability for the current PLO. As we mentioned in the previous sections, the new rating for a player depends both on his/hers past rating and the score that corresponds to the outcome for the current prosocial task. Similar to the typical use of the Elo system for providing interesting matchmakings between contestants, it is our intention to match a player of a certain prosocial ability to a scenario of similar rating for a PLO. This is intended for the following reason: we want the player to experience a scenario that is not too difficult, nor too easy to accomplish. That way, we aim to achieve true enhancement of prosocial behavior, compelling the player to accomplish a task that lies on the limit of his or hers prosocial ability.

Following (Klinkenberg et al., 2011), we can determine the proper scenario for the player in real-time by selecting the one which rating has the minimum distance to the player’s rating. In mathematical notation, the rule that gives the id for the next scenario reads

\[ id = \arg \min_{s} |P_s - P_{est}|, \]

where \( P_s \) is the rating of a scenario and \( P_{est} \) is a stochastic estimate of the prosocial ability of the player, given by

\[ P_{est} = P_p + \ln\frac{r}{1-r}, \]

where \( P_p \) is the true prosocial ability of the player, and \( r \) is a real number drawn from a normal distribution. The reason that the matching between players and scenarios is done stochastically, is solely to avoid repetitions of the same scenario over a short period of time.
5 Online Adaptation

5.1 Introduction

Online adaptation is realised during the actual gameplay of a PsL game, as shown in figure below. Its purpose is to select the proper game elements that contribute to the enhancement of the player’s engagement in a prosocial objective. These game elements concern the standard pedagogical practice of expressing positive reinforcement and corrective feedback on a student’s performance:

- **Positive reinforcement** is the process of strengthening a person’s behaviour as a consequence of applying a stimulus.
- **Corrective feedback** provides instructions to players to correct their behaviour and supports them in identifying and paying more attention to the outcome of their actions and what they ought to do to be more successful. It provides specific, often textually represented, instructions to players to correct their behaviour and supports them in identifying and paying more attention to the outcome of their actions and what they ought to do to be more successful.

Each game must offer a pool of elements (e.g. text messages, sounds or graphics) realizing positive reinforcement or/and corrective feedback. Online adaptation considers real-time player data concerning player’s engagement estimation during specific time intervals in the game. Hence, the player’s engagement estimation, as described in Section 3, is vital within the context of prosociality, due to the concept’s gamification as in-game tasks for building up a set of predefined abilities.

Figure 15: Online adaptation
For a certain game, the recordings of user engagement for his/hers experience with each positive reinforcement or/and corrective feedback element encountered, are kept into buffers of equal size. In a typical duration of a game, the player may encounter many of these elements. The buffers have a fixed size and are circular, meaning that when the buffer is filled up and a new engagement measurement occurs, then the oldest stored value is discarded and the other values are shifted towards the end in order to create an empty space in the beginning for the new value. The positions in the buffer account for the order in time in which the measurements occurred at. Thus, by shifting positions towards the end of the buffer, we keep only the most recent recordings to represent the user’s engagement state. The collection of the engagement recordings for every element of the game, consist the player’s engagement profile for that game, as shown in the figure below.

**Figure 16: A player's engagement profile.**

When the buffers of the engagement profile are filled up with values, a mathematical measure should process them and derive a score or ranking for each of them. These rankings would imply the elements that mostly contribute to the process of enhancing the user’s motivation. Therefore, it is our intention to incorporate such a measure for analysing a player’s engagement profile and inferring the player's preference for motivation regarding the accomplishment of a particular task. A well acknowledged weighting scheme can be found in the context of Reinforcement Learning, where the computation of the overall reward of a learning process relies on gradually discounting rewards over time. In such a case, one considers immediate rewards to be more important than later rewards and expresses this fact with the notion of a discount factor (van Hasselt, 2010). The value for this factor must be set to be lower than 1. Using such a definition, we ensure that the value of the reward is finite at any step of the learning process. Therefore, this definition bounds our computation and ensures that the resulting value will not become very large (i.e., the variable will not get out of range). Another interesting characteristic for the factor is that it controls the convergence of the learning process. For example, in some cases, a lower discount factor might imply faster learning. In
any case, applications with short term objectives might use lower discount factors than applications with long term objectives.

More specifically, for non-episodic tasks, i.e. tasks that do not have an ending state, the overall reward over time for an agent is given by the following rule

\[ R = \sum_{t=0}^{\infty} y^t r_{t+1}, \quad 0 \leq y \leq 1 \]

where \( y \) is the discount factor and weights the recordings in decreasing order over time.

Using the above measure on player’s engagement estimates, we can therefore create rankings of game elements that the player has experienced over a period of time, through receiving positive reinforcement and corrective feedback. An example of the proposed approach for adjusting the reinforcement game elements is given in the following table. The bold row in the table shows the element that is considered to produce the highest engagement experience to the user.

<table>
<thead>
<tr>
<th>Positive Reinforcement Game Element</th>
<th>Player’s Preference Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typical Well-Done Message</td>
<td>0.932</td>
</tr>
<tr>
<td>Fireworks Exploding</td>
<td><strong>1.426</strong></td>
</tr>
<tr>
<td>Blinking Congratulation Message &amp; Player’s Score Increase</td>
<td>1.064</td>
</tr>
</tbody>
</table>

In order to learn the user’s engagement profile, the real-time adaptation module must provide an online learning mechanism. This mechanism will gradually learn the user’s engagement profile, considering both exploiting a user profile in order to satisfy the need for personalization, but also supplying the user with new elements to experience in order to take into account the case of behavioral shifts over time. This situation is known as the exploration-exploitation strategy. A special case for utilizing such a strategy is the multi-armed bandit problem. Having its roots in probability theory, the multi-armed bandit problem describes the case in which a gambler has to decide among a number of slot machines (i.e., one-armed bandits) which of them to play, how many times to play each, and in which order to play them. After each play, each machine provides a reward draw from a probability distribution that represents the machine’s functionality. The problem’s objective is to maximize the sum of rewards earned through a sequence of lever pulls. Thus, the gambler at each trial must decide about exploitation of the machine that has the highest expected payoff and exploration to get more information about the expected payoffs of the other machines. This trade-off between exploration and exploitation is also faced in reinforcement learning. This similar problem consists of an artificial agent that simultaneously attempts to acquire new knowledge (called “exploration”) and optimize his or her decisions based on existing knowledge (called “exploitation”). The agent attempts to balance these competing processes in order to maximize its total reward over a time period.
Within the framework of the ProsocialLearn project, we adopted the epsilon-Decreasing (ED) algorithm, which is a variation of the well-known epsilon-greedy algorithm (Epsilon-greedy or ε-greedy, is probably the simplest action–selection strategy to solve the multi-armed bandit problem). More specifically, at each iteration, the algorithm selects a random solution from the population with a certain probability $1 - \epsilon$, while with the remaining probability $\epsilon$ selects the solution with the highest ranking. The evaluation process ends after a number of evaluations are completed. The value of $\epsilon$ decreases as the experiment progresses, resulting in highly explorative behavior at the start and highly exploitative behavior at the finish. Generally, the $\epsilon$ in ED method gets decreased according to a monotonously decreasing function (Takahashi, 2011). An example of such a function is given below:

$$e(t) = 0.5/(1.0 + rt)$$

where $r$ is the algorithm's parameter for controlling the rate of decrease.

The $\epsilon$-decreasing strategy fits the requirements of the PsL online adaptation for learning players’ engagement profiles. Initially, there are no engagement profile data for the algorithm to consult for a real-time selection of a positive reinforcement element. During that time period, the $\epsilon$-decreasing algorithm expresses highly explorative behavior in order for the player to experience as many different elements as possible. In this stage, the algorithm tries to maximize the coverage of the search space (Shani, 2011) (i.e., experience of available game elements). When few engagement data occur in the player’s profile, the algorithm continues to keep an explorative behavior in order to increase its confidence for the profile’s appropriateness. When the profile is filled with engagement values (i.e., engagement buffers for each positive reinforcement element are filled), the $\epsilon$ value favors mainly exploitation, but does not terminate its exploration phase. The focus on exploitation in that stage, is of major significance in order to provide reliable results for predicting the actual state of the user. Additionally, the retention of the exploration phase guarantees adaptivity to user behavioral shifts over time. That way, the personalization capabilities of the system are enhanced.

Finally, to address the adaptation problem in multiplayer games, we have extended the algorithms for online and offline adaptation by adopting the Borda Count Aggregation approach, which is a single-winner election method. More specifically, we compute each player’s score for each item (scenario or element) based on its position in player's preference list:

$$\text{score}_p(i) = n - \sigma_p(i) + 1$$

and then we estimate the group score for each scenario or element:

$$\text{score}_g(i) = \sum_{p \epsilon P} \text{score}_p(i)$$
<table>
<thead>
<tr>
<th>Element</th>
<th>Preference $d_p(i)$</th>
<th>Element score $(i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>E2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>E3</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Preference $d_p(i)$</th>
<th>Element score $(i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>E2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>E3</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements</th>
<th>Group score $(i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>3</td>
</tr>
<tr>
<td>E2</td>
<td>6</td>
</tr>
<tr>
<td>E3</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 17: Group score.
6 Utilizing the ProsocialLearn Adaptation Manager in a sample game

This report has so far provided a thorough description of the ProsocialLearn Adaptation Manager (PAM). Two distinct adaptation mechanisms for online and offline adaptation have been presented, aiming to provide personalized game elements and proper challenge setting for children to properly learn how to use prosocial skills using the ProsocialLearn platform. The refined technical overview and analysis of components for both mechanisms has been explained in detail throughout the sections of this document, the theoretical justification has been presented in the previous version of this report. We conclude this overview with an example that demonstrates how the PAM can be utilized with the sample prosocial game *Path of Trust*, in order to support personalized adaptations in an otherwise straightforward endless running platform game. This Section is provided as more of a guide for aspiring ProsoiciLearn game developers, for demonstrating the design processes taking place to support communication with the PAM, as well as the expected outcomes, through a carefully designed small-scale experiment demonstrating the feasibility of the approach. The Section is organized as follows: Section 6.1 will briefly present the *Path of Trust* game used as a base for demonstrating both offline and online adaptation considerations. Section 6.2 will demonstrate the necessary changes made to the game in order to support offline adaptation mechanism provided by the PAM, explaining how different game scenarios were fleshed out. Section 6.3 will similarly outline how support for PAM’s online adaptation mechanism was implemented, presenting some examples of different elements supporting both positive reinforcement as well as corrective feedback. Finally, in Section 6.4 we present a small scale study that demonstrates functional capabilities of the implemented game and some thoughts on future work.

6.1 Path of Trust

*Path of Trust* is an endless running game about two characters having to cooperate in order to navigate a maze and collect treasures while avoiding enemies and other hazards in the process. The game features a single player mode that supports PLOs with respect to the prosocial skill of evaluating trustworthiness of the NPC character, while the two-player mode of the game focuses on teaching the benefits of cooperation. In the remainder of this text we will focus on the multi-player version of the game. In this version of the game, the two players take control of the two characters and set out to collect as many treasure points as they can within the designated time limit. The players’ characters navigate a maze, structured by junctions and corridors, as is shown in Figure 18. The player who is controlling the character moving through the maze is referred to as the *Muscle*, and is deprived of spatial awareness within the maze (i.e. the player can only see the area he/she is currently in), while the other character referred to as the *Guide*, uses a top-down map view to navigate both of them (as the Guide character sits on the shoulders of the Muscle character) safely through the maze without being caught. The dungeon corridors are populated by items, which can either be collected (upon touch) by the characters to score points or, in case the item represents a hazard, need to be avoided. These items are described in Table 1. The dungeon’s Junctions are special areas in which the characters’ path can be altered by following up to three different
Figure 18: Junction and Corridor dungeon tile types making up the *Path of Trust* map maze. The bold green arrow indicates entry point for the characters. The white arrows indicate possible navigation paths.

<table>
<thead>
<tr>
<th>Item Name</th>
<th>Item Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diamond</strong> (D)</td>
<td>Treasure Point, adds up the total tally of points collected by each player in the multiplayer game mode. The Muscle Player gets two points, while the Guide player receives one point.</td>
</tr>
<tr>
<td><strong>Mummy</strong> (M)</td>
<td>Avoidable creatures living in the dungeon. They will stall players’ progress and demand one point as payment for being disturbed. When the Muscle Player is captured by a Mummy, he/she loses one Treasure Point. The Guide player’s Treasure Point remain unaffected.</td>
</tr>
<tr>
<td><strong>Portal</strong> (P)</td>
<td>Unavoidable item that allows the two characters to switch bodies. When the Muscle Player passes through a Portal, he/she immediately becomes the Guide Player and vice versa. No rewards or penalties are applied to the players’ total Treasure Points’ tallies.</td>
</tr>
<tr>
<td><strong>Trap</strong> (T)</td>
<td>Avoidable hazard. A Muscle Player touching the trap leads to an immediate Game Over screen.</td>
</tr>
</tbody>
</table>

Table 1. Items in the *Path of Trust* multiplayer game world.
directions (turning left, right or continuing forward). As a narrative element in the Path of Trust storyline, the Guide is the only character with any knowledge on the items contained within the adjacent corridors. This element is utilized to provide a memory-based mini-game designed to engage the player currently controlling the Guide, as they wait for the other player to navigate to the next Junction. The Guide is supposed to pick which direction the duo should take next and make a proposition to the player controlling the Muscle. The Muscle is then supposed to trust in the Guide’s suggestion and follow their directions, and cooperate by either touching the item in the corridor or avoiding it. A switching roles game mechanic (see D4.3) allows players to actively swap characters, thus experiencing the cooperation from the other player’s perspective. As an ending condition, the game will end when any one of the two players reaches a designated number of points (six, in the user study). Furthermore, the game features a time limit and a set of different endings for the players to reach according to their performance. Considering the cooperation skill, players are expected to work as a team in a way that will allow them to finish the game with six (or more, in some cases) points each, thus demonstrating the skills necessary to appreciate following each other’s directions and understanding when they should sacrifice some of their own resources in order for the group to benefit in the long run (as they will accumulate more points as a team, as opposed to each player trying to gather all the points for themselves).

### 6.1.1 Guide gameplay

As mentioned previously, the Guide is shown a top-down view of the common map in a 2D perspective, like in Figure 19. The Guide always gets to see up to three corridors ahead (left, forward, right, see Figure 19a). The Guide also gets to “peek” into the contents of each corridor for a small period of time (Memory Window, Figure 19b). This way the Guide will know if the corridors to the left, right and forward contain Treasure, Mummies, Traps or Portals. After a limited number of seconds the indicators will disappear and the player will have a small window (the Decision Window) in which to communicate his choice of direction to the Muscle player. Players have to choose between suggesting going ‘left’, ‘right’ or ‘forward’ (Figure 19c). The Guide’s Treasure Points are increased by 1, whenever the Muscle player touches a Diamond item. Of course, it is in the best interest of the group to sometimes lead the Muscle intentionally towards a Mummy or Portal in order to balance out their uneven Treasure Points (minimizing the chance of a premature game over, should the Muscle touch a Diamond that would be their sixth point) or swap bodies to re-route the way points are being collected.

### 6.1.2 Muscle gameplay

The Muscle is shown a 3D view of the dungeon scene in a third-person view as depicted in Figure 20. The player can see the junctions and corridors, the two characters and the items (when inside a corridor). Once the Guide communicates his command, a direction indicator will be shown on the screen for a brief period of time (Figure 20a). During this time window, the Muscle player will be able to input his next move. The game indicates to the player that he/she is expected to make the choice by showing the indicator throughout the entire distance the characters have to travel before reaching the actual turning point (this constitutes the Muscle Decision Window). After the choice of
direction has been made (Figure 20b), the characters will enter a corridor and have a limited room in which they can place their characters in order to touch treasure points or avoid Mummies and Traps. They can do that by moving the character to the left or right, during a limited time period determined by their speed (The Action Window, see Figure 20c). Again, it is sometimes in the best interest of the group, that the Muscle should intentionally choose not to touch a Diamond, if that would be the last Diamond they could carry, resulting in the player having way more points than the Guide player, or intentionally touch a Mummy in order to close that points gap (see Section 6.1.4).

6.1.3 Synchronization

The game achieves synchronization between the two players by having the Guide’s Memory Window take place during the Muscle’s Action Window, a sensible solution, as the junction the Muscle is heading to is static and therefore the next three (two or one) adjacent corridors are already known. As the Muscle strives to collect or avoid the item inside the corridor, the Guide engages in a short memory game, which allows the player to choose towards which direction to go next. After exiting
the corridor, the Muscle already has received the Guide’s instruction, while the Guide’s game world is being refreshed (i.e. oriented towards the direction the duo is now facing). The game is won if one of the characters reaches the appointed goal of treasure points (six in our study case). The game is designed in a way that an equilibrium is possible, i.e. there is potential for both players to co-operate in a way that both players can win at the same time (by collecting the final diamond, the Muscle player who currently has -2 treasure points from the goal and the Guide player who currently has -1 treasure points from the goal would win the game together). Table 2 presents an example of a possible game simulation in which the players win the game by collecting 12 diamonds each.

6.1.4 Task-driven Prosocial Learning Objective for cooperation

Within the Path of Trust multiplayer game mode, the prosocial skill (and corresponding PLO) concerns being able to identify the benefits of cooperation. As previously mentioned in previous Sections in this report, the game contains specifically designed game tasks to test the player’s ability in using that skill. The player have to properly using the skill to overcome a series of game tasks throughout a single game session in order for the player’s prosocial skills profile to be updated at the end of each game using the scores attained at each task. For the remainder of Section 6 and for simplicity, we will focus on the Muscle player’s game tasks designed within Path of Trust.
Table 2. Example Simulation: Player 1 starts as Guide, Player 2 starts as Muscle. Players assumed to lead one another safely towards Treasure Points (TPs) while switching roles during Rounds 4, 8 and 10 to maintain a balance in their TP tally. End result is Victory for both players, as they reach their goal of 12 TPs each at the end of Round 11.

We consider the Muscle player’s cooperation game task to start right after the doors to a Corridor have been opened, thus revealing to the player the item within that Corridor. With respect to the cooperation skill, and within the short time frame that is given between passing through the Corridor doors and reaching the item’s position at touch-range, the player is expected to consider the following strategy:

- Acknowledge the item and its utility. The player has to identify a Diamond, Trap, Portal or Mummy and draw immediate connections with its utility, i.e. how the players themselves as well as the group is affected by touching that specific item.
- Decide whether to touch or avoid the item. The player has to decide whether touching the item will hurt, or benefit the group.
- Proceed with the desired action.

According to this strategy, players are able to evaluate their cooperation with their partner, and identify its benefits through the online adaptation mechanism (see Section 2.3). For example, in our case study, a Muscle player currently has 5 points, while their Guide only has 2 points. After entering a Corridor with a Diamond item, the Muscle player has to acknowledge that touching the Diamond will cause the game to end (as they will have accumulated 5+2=7 points, which is over the 6 point end-game threshold set by the game). Furthermore, that player will have to decide whether to touch the Diamond, given the fact that he/she will finish the game with 7 points while their Guide will only have 2+1=3 points by doing so. The player acknowledges that the group could do better (aiming for a 6-6 end game finish) and decides not to touch the Diamond. As the player acted towards the best interest of the group, their cooperation skill is met with positive reinforcement, by presenting a corresponding element programmed within the game. The same logic can be applied for cases in which the players approach the other item types. The game administers positive reinforcement or corrective feedback elements according to how well players regulate their actions to serve a common goal. While portals and traps are pretty straightforward (one is unavoidable while the other
must always be avoided), touching or avoiding Diamonds and Mummies might trigger different reactions from the game based on the players’ current score, past decisions and potential follow-up item (for example, a seemingly questionable action taken now, could be of benefit to the group if the player bases his strategy based on the follow-up item being a Diamond). Sometimes, waiting for a Portal while avoiding every other item, might turn out to be the best strategy. This is thoroughly demonstrated in the graph shown in Figure 21. Each rectangular box represents a current score between the two players A-B, in which A is the score of the Muscle player and B is the score of the Guide player. A blue tinted box represents redundancy in the graph (i.e. states which have been previously explored). A red-tinted box represents unfavourable end-game, while a green-tinted box similarly represents favourable end-game. A red arrow represents a Muscle player action (touching a Mummy – $M$, avoiding a Mummy - $\textoff{M}$, avoiding a Diamond - $\textoff{D}$, touching a Diamond – $D$) that triggers a Corrective Feedback element, while a green arrow represents an action triggering a Positive Reinforcement element. This logic is implemented as part of the game (not the PAM), and triggers whenever the Muscle player Action window is initiated. When the Action Window stops, and the Muscle Decision Window is about to start the game triggers communication with the PAM for online adaptation (see Section 6.3).

### 6.2 Offline adaptation in Path of Trust

As previously mentioned, offline adaptation within the context of the PAM requires the game developers to define a set of scenarios, i.e. varying levels of challenge as to the way the prosocial skills are being tested. In Section 6.1.4, the game tasks designed to test the players’ abilities to properly demonstrate the necessary cooperation skill were defined. Players formulating a strategy according to Figure 21 will have more chances of having their team emerge victorious from their cooperation. Therefore, in order to challenge players’ skills, more challenging dungeon layouts are required. This is achieved within Path of Trust, by increasing the number of traps and portals, while simultaneously decreasing the chances of encountering Junctions in which two or three similar, score-affecting items (e.g. two Diamonds or three Mummies) can be present. It is expected that increasingly more complex item distribution layouts will take significantly more time and effort on behalf of the players to complete with a favourable team outcome than others. As a result, we created three, increasingly difficult dungeon layouts, and corresponded each one to a different game scenario for the offline adaptation mechanism of the PAM to choose from. To indicate these changes to the players, we created different sets of graphical assets and GUI indicators for each scenario, thus ending up with three different levels to support variety for the game as players progress their skills abilities. The levels currently supported within Path of Trust are the Egyptian Pyramid, the Knossos Labyrinth and Aztec Temple levels. The resulting game scenarios are shown in Figure 22.
Figure 21: Task-driven strategies for the Muscle player in *Path of Trust* and corresponding Positive Reinforcement/Corrective Feedback triggers for all possible game states resulting after the Muscle player’s Action Window with respect to all possible player actions (M, IM, D, ID).
6.3 Online Adaptation

As argued in D2.6, game designers are urged to incorporate a range of positive reinforcement and corrective feedback rewards into their game mechanisms. As previously mentioned, online adaptation within the context of the PAM requires the game developers to define a set of elements. Elements can be anything within the game, from specific sound effects to graphics, GUI elements and even dynamic difficulty adjustment. Within the context of ProsocialLearn, we consider game elements to be in service of the prosocial skills skill-streaming approach described in D2.6, thus we considered elements within Path of Trust to be specifically tied to manners through which the game presents positive reinforcement or corrective feedback to the players, in response to their performance after a game task has been completed (see Section 6.2).

According to D2.6, p24:

“Positive reinforcement consists of presenting a positive reward after a correct behaviour in order to increase such behaviour.”

We defined three different in-game elements for presenting Positive Reinforcement enhancers to the players. These are all graphical representations of the game commending the players for a job well done. The actual elements are depicted in Figure 23, namely a plain Well Done message, a flashy Congratulations! animated pop-up icon and a full-screen Fireworks Display. As players progress through the game, online adaptation mechanisms will analyse player engagement with respect to these elements and trigger one of these elements for administering positive reinforcement. For example, a player demonstrating strong engagement reaction towards seeing the Fireworks Display element (explained in Section 3) will increase his/her chances of being commended with a Fireworks Display during future Positive Reinforcement feedback from the game.
For Corrective Feedback, we followed a different approach, seeing how the elements should contribute towards the player understanding their mistakes and trying again in order to succeed. According to D2.6, p.25:

"Corrective feedback can be used when a child doesn’t act as is expected during the session. The teacher (or the computer) should point to the specific behaviour and give the child an opportunity to do the action again. Teachers or the computer should follow the three steps below to give corrective feedback: (1) point to the specific behaviour that the child performed incorrectly, (2) use a statement to describe how the behaviour impacted others (e.g. I do not like it when you do not share your paper with John), (3) ask the child to perform the specific appropriate behaviour and (4) praise the child for performing the specific appropriate behaviour."

Therefore, any verbal communication of corrective feedback should be specific, and relate to the action of the player has completed, to adequately communicate what the child is doing wrong. Seeing how there’s multiple ways in which players can fail a task in Path of Trust (see Figure 21) according to the best strategy to achieve a favourable outcome (e.g. 6-6 endgame), there are a
variety of messages to convey to the user with respect to their chosen action in need of corrective feedback:

![Guide Character Coach pop-up icon indicator](image)
![Female Coach audio recording session](image)
![Male Coach audio recording session](image)

Figure 24: Path of Trust Corrective Feedback elements supporting PAM online adaptation mechanism: a) Guide Character Coach pop-up icon indicator; b) Female Coach audio recording session; c) Male Coach audio recording session.

- Player touched a Mummy ($M$) without actually having to sacrifice any points towards the best interest of the group (e.g. both players points count is low). In this case, the player risks losing valuable time as the clock runs out to replenish lost points. Corrective Feedback should therefore inform the player accordingly.

- Player touched a Mummy ($M$) without actually having any points to sacrifice. This is a special case to the bullet above, in which, Corrective Feedback reminds the player that their action had no effect whatsoever, but resulted in mutual time loss.

- Player touched a Mummy ($M$), resulting in the Guide player having significantly more points than the player. In this case, the group risks reaching an unfavourable endgame (e.g. if current score is, for example 2-5) and thus, Corrective Feedback elements should explain why the player has to try and avoid Mummies until the group re-balances their point distribution (by entering a Portal, for example and having the new Muscle player touching some Mummies to lighten their load of Diamonds).

- Player touched a Diamond ($D$), or avoided a Mummy ($!M$), resulting in the Guide player having significantly more points than the player. In this case, the group risks reaching an unfavourable endgame (e.g. if current score is, for example 2-5) and thus, Corrective Feedback elements should explain why the player has to try and avoid Diamonds until the group re-balances their point distribution (by entering a Portal, for example and having the new Muscle player touching some Mummies to lighten their load of Diamonds).

- Player avoided Diamond ($!D$) despite having good enough reason to take it (i.e. action won’t result in unfavourable endgame, both players points count is low). In this case, Corrective Feedback tells the player they have to consider the group’s time limit and make an effort to contribute towards the team getting more points.
Player touched a Diamond (D), resulting in the player having significantly more points than the Guide player. In this case, the group risks reaching an unfavourable endgame (e.g. if current score is, for example 5-2) and thus, Corrective Feedback elements should explain why the player has to try and avoid Diamonds or touch Mummies until the group re-balances their point distribution.

We therefore grouped all messages into corrective feedback types, matching each type to an online adaptation element. For this we employed: a) the use of a pop-up image of the Guide character pointing to the behaviour the player performed incorrectly, and asking the player to perform the appropriate action in the future; b) an audio message of the exact same message voiced by a Female Coach (no visual feedback); c) same audio message voiced by a Male Coach. The types are demonstrated in Figure 24. As players progress through the game, online adaptation mechanisms will analyse player engagement with respect to these elements and trigger one of these elements for administering corrective feedback in future erroneous actions. For example, a player demonstrating strongly engaged reaction towards hearing any of the above messages from the Female Coach (as explained in Section 3), will increase his/her chances of being corrected by the Female Coach during future Corrective Feedback administered by the game.

6.4 Small Scale Study

After defining offline and online adaptive features within Path of Trust, we registered the game with the PAM by following the instructions on the PAM Manual (see Appendix A). This process takes approximately 5 minutes to complete by hand (creating the XML files and saving them in the correct location within the PAM installation directory), but it is expected to happen automatically once the ProsocialLearn platform for game registration is finalized (i.e., through the upload process in which developers will indicate the PLOs associated with their game). In order to test the game’s interaction with the PAM, we organized and conducted a small scale study during December 2016. We will outline a short presentation of the study participants, procedure and results observations throughout the remainder of this Section. A more detailed overview of the results will be made available in the corresponding deliverable expected from WP7 (D7.9 “2nd Results of small experimental studies).

6.4.1 Participants

Participants in our study were a total of 20 children, students attending the EA private primary school located in Athens, Greece. The students ranged in age from 8 to 9 years old. In total, 11 groups (referenced as group A to group K, with two children randomly being paired with a second partner due to logistical reasons) completed the entire session. The study was approved by the Institute of Educational Policy (IEP), a private legal entity under the supervision of the Greek Ministry of Education, Research and Religious Affairs. Prior to the study, the students’ parents were handed out and signed informed consent forms allowing their children to participate in the study.

6.4.2 Procedure

Prior to this study, we prepared a login system in order to enable storing data on designated user profiles without compromising sensitive user data. Towards this end, we prepared numerous pre-registered accounts and stored their credentials within the system login infrastructure. Each
username/password combination was then handed out to participating children, who were allowed to randomly pick an illustrated card containing their personal account information. Examples of these cards can be seen in Figure 25. Each child was told to hold on to their card and present it to the experiment coordinators to help them log that player into the system. Both the internal game logging mechanism, as well as the PAM persistence mechanisms used the specified account names to update information on recurring players.

Our study spanned a duration of a total four (4) days. Each day, the study took approximately two hours in order to complete an entire session. In order to be more time-efficient, we conducted the study simultaneously using two mobile workstations co-located in the classroom, dividing the number of children participating on a daily basis by two and assigning recurring participants to the proper station in which their credentials were already stored. Each workstation was comprised of two high-range laptops\textsuperscript{2,3,4}, namely the MSI and Dell Workstations (named after the laptops’ manufacturers). Each of the four laptops was equipped with a Kinect for Xbox One sensor. The two laptops in each workstation were connected using an Ethernet cable. One laptop in each workstation was randomly selected to host the game server and the PAM, while both laptops had pre-installed

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{example_path_of_trust}
\caption{Example Path of Trust illustrated account cards distributed to players with credentials printed on them (Username=Password).}
\end{figure}

\textsuperscript{2} 2 x MSI GL62 6QF-632NL (Intel Core i7 2.60 GHz, 2 GB NVIDIA GeForce GTX 960M)
\textsuperscript{3} 1 x Dell Precision M6700 (Intel QM77 Express, 4GB NVIDIA Quadro K4000M)
\textsuperscript{4} 1 x Dell Precision M4800 (Intel QM87 Express, 2GB NVIDIA Quadro K1100M)
versions of a ProsocialLearn front-end client application, which was used to grant access to the game as well as receive user input from the Kinect. Additionally, the client was configured to simulate the functionality of the ProsocialLearn platform (in lack of outside Internet connection), by incorporating both user login infrastructure and multimodal fusion components to extract user engagement (see Section 3).

A short description of the Path of Trust backstory was briefly delivered by one person within the experiment coordination team, using a narrative PowerPoint presentation demonstrating instructions on the game gesture-driven interface with regards to moving the player character and touch to collect / avoid hazards mechanics. Players were told of the endgame conditions and time limit to perform the game tasks. Each game session had pure gameplay duration of two and a half minutes (150 seconds). In total, each child took approximately 10-15 minutes to complete the study, given that each pair of players were allowed to play the game for four consecutive sessions, in order to monitor the adaptation mechanisms’ fitness to the players’ dynamically changing profiles over time.

6.4.3 Results

Our observations with respect to the game’s adaptation efficiency have been associated to the possible game outcomes at the end of each game session. To clarify the paragraphs below we summarize the game’s possible outcomes before delving deeper into our observations on offline and online adaptation mechanism in Path of Trust:

- **PROSOCIAL Outcome**: indicates game finish in which both players have captured six points or more, through an efficient collaboration that includes at least one role-switching action (i.e. passing through a Portal). This outcome ensures the group collects the maximum amount of points as a group.

- **TIMEOUT Outcome**: indicates game finish in which none of the two players have managed to capture six points or more during the designated time limit given to complete the level. This outcome does not necessarily mean improper collaboration, but rather demonstrates difficulties in maintaining a balanced collaboration with the other player (e.g. improper directions or failure to capture/avoid Diamond/Mummy).

- **NON-PROSOCIAL Outcome**: indicates game finish in which one of the players has managed to capture six points or more, while the other player total tally of points is below the given threshold. This state indicates an improper assessment of the collaborative techniques required to achieve a PROSOCIAL outcome, as it is easily attainable by one player collecting all the points in his path without ever switching roles.

- **OVER Outcome**: indicates game finish by instant game over, e.g. when the Muscle player fails to avoid a Trap object. This outcome indicates the worst case scenario, as it includes both faulty directions from the Guide (e.g. pointing out a direction towards the Trap Object) as well as the Muscle (either not following on directions received to avoid visiting a Trap Corridor, or failing to spot, and subsequently avoid the Trap due to (the possibility of) not being sufficiently engaged and paying attention at the time.)
6.4.3.1 Offline adaptation

Our study has confirmed our approach to increase difficulty of the three dungeon layouts. The end-game results demonstrate an efficient distribution of challenge across the three game scenarios for offline adaptation, as players managed to achieve a PROSOCIAL outcome in 5 individual game sessions in *Egyptian Pyramid*, while 8 sessions ended with a NON-PROSOCIAL” and 4 sessions with a TIMEOUT outcome. Players’ progress in *Knossos Labyrinth* fared significantly worse, as approximately 67% of individual sessions ended with players being felled by a trap (OVER), while players in *Aztec Temple* did not manage to record a single session with 6 points or more earned for an individual partner (i.e., no PROSOCIAL or NON-PROSOCIAL outcomes achieved, despite the latter being quite easily attainable, as described in the previous paragraph). The players’ performance with respect to the game’s PLO in cooperation is reflected upon the offline adaptation’s choice of appropriate scenarios for both players, as the easiest level, *Egyptian Pyramid* received substantially more play time (17 individual sessions across all player groups) in contrast to *Knossos Labyrinth* (6 individual sessions) and *Aztec Temple* (7 individual sessions) levels. The end-game outcome results are graphically depicted in Figure 26.

![Figure 26: Individual game session outcomes per Path of Trust scenario](image)

6.4.3.2 Online adaptation

We approached our analysis of the online adaptation comparing player adherence to the online Corrective Feedback (CF) / Positive Reinforcement (PR) messaging mechanisms towards measuring the final game outcome. In this analysis, we are interested in analysing player behaviour immediately after receiving feedback from the game, thus monitoring their next action through the next feedback received. In this respect, player in-game behaviour can be distinct into four categories, namely:

- Receiving Positive Reinforcement right after a Positive Reinforcement message (PR-PR);
- Receiving Positive Reinforcement right after a Corrective Feedback message (CF-PR);
- Receiving Corrective Feedback right after a Positive Reinforcement message (PR-CF); and
- Receiving Corrective Feedback right after a Corrective Feedback message (CF-CF).

As can be seen in Table 3, players who achieved PROSOCIAL outcomes demonstrated a significant capacity for adherence to the game’s Positive Reinforcement / Corrective Feedback, taking significantly less time (shown by the number of consecutive feedbacks given by the game as a
response to the Muscle player’s actions) to reach the desired outcome and following up on previous game feedback with commendable action (e.g. receiving Positive Reinforcement after their previous action was addressed by either a PR or CF message) during approximately 70% of the times (PR-PR, CF-PR), while following up twice with non-commendable action to a CF message (CF-CF) in just approximately 4% of the time.

Some similarity can be observed for players managing a NON-PROSOCIAL or TIMEOUT outcome, demonstrating a significant overall amount of time taken to achieve these outcomes (i.e. shown by the overall number of consecutive feedbacks given, clearly in contrast to the PROSOCIAL case). In the first case, a very strong indicator of player ignorance to feedback instructions can be seen in players responding to the previous PR or CF message with non-commendable action (PR-CF, CF-CF) in approximately 65% of the time, which indicates a stark contrast to playthroughs ending in a PROSOCIAL outcome. On the other hand, TIMEOUT cases demonstrate an overall evenly distributed pattern of actions, which indicates a slight preference of players towards being positively rewarded (58%) after a certain action was taken in response to the message previously received (PR-PR, CF-PR). This can be explained by closely observing the game logs for these playthroughs, in which TIMEOUT outcomes were reached after a significant number of turns had passed, and with player scores being mostly evenly distributed (e.g, 4-5, 3-3, 4-3 etc.) but time was not enough for players to achieve the “Prosocial” outcome. In other words, players demonstrated significantly more adherence to the prosocial objective of being collaborative than players who ended up with a NON-PROSOCIAL outcome, but yet rather significantly lower capacity for continuously doing so, as in the PROSOCIAL outcome attaining groups. This capacity is also taken into consideration towards determining player PLO rankings for offline adaptation, as there is some difference to the TIMEOUT occurring while players have a score of 5-5 or 1-3, yet both cases could have gone either way if the game had infinite time to reach the designated number of points.

These outcomes, in conjunction to the ones in the PROSOCIAL case in Table 3, demonstrate how players tend to better cooperate when adhering to the game’s PR/CF messages, tailored personally to their engagement preferences. Players actively ignoring these messages tend to experience much more difficulty, as they are more prone to underachieve (e.g. NON-PROSOCIAL outcome) or, risk falling into a trap (OVERoutcome), which usually occurs when players demonstrate a clear preference to follow up on PR/CF messages with non-commendable action (PR-CF, CFR-CF in 62,5% of the times, especially considering the limited game time due to the instant game over a Trap object brings about to the game).

<table>
<thead>
<tr>
<th>PROSOCIAL</th>
<th>PR-PR</th>
<th>PR-CF</th>
<th>CF-PR</th>
<th>CF-CF</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of instances</td>
<td>8</td>
<td>6</td>
<td>8</td>
<td>1</td>
<td>23</td>
</tr>
<tr>
<td>Percentage</td>
<td>~34,8%</td>
<td>~26,1%</td>
<td>~34,8%</td>
<td>~4,3%</td>
<td>100%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NON-PROSOCIAL</th>
<th>PR-PR</th>
<th>PR-CF</th>
<th>CF-PR</th>
<th>CF-CF</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR-PR</td>
<td>PR-CF</td>
<td>CF-PR</td>
<td>CF-CF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>
### Table 3. Distribution of in-game Muscle player behaviours and game outcomes per individual session.

<table>
<thead>
<tr>
<th></th>
<th>PR-PR</th>
<th>PR-CF</th>
<th>CF-PR</th>
<th>CF-CF</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No. of instances</strong></td>
<td>16</td>
<td>14</td>
<td>13</td>
<td>7</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>32%</td>
<td>28%</td>
<td>26%</td>
<td>14%</td>
<td>100%</td>
</tr>
</tbody>
</table>

#### TIMEOUT

<table>
<thead>
<tr>
<th></th>
<th>PR-PR</th>
<th>PR-CF</th>
<th>CF-PR</th>
<th>CF-CF</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No. of instances</strong></td>
<td>12</td>
<td>9</td>
<td>3</td>
<td>19</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>~27.9%</td>
<td>~20.9%</td>
<td>~7%</td>
<td>~44.2%</td>
<td>100%</td>
</tr>
</tbody>
</table>

#### OVER

<table>
<thead>
<tr>
<th></th>
<th>PR-PR</th>
<th>PR-CF</th>
<th>CF-PR</th>
<th>CF-CF</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No. of instances</strong></td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>25%</td>
<td>37.5%</td>
<td>12.5%</td>
<td>25%</td>
<td>100%</td>
</tr>
</tbody>
</table>

In total, by looking over the amount of players following up on CF messages with corrective action (e.g. CF-PR), we see a clear link towards maximizing the likelihood of achieving the desired PLO outcome, and even more so, much faster than players choosing to ignore the CF and act out on their own volition in contrast to the desired outcome (CF-CF). Furthermore it is almost certain that players who continuously ignore CF (e.g. CF-CF) will end up underachieving, (NON-PROSOCIAL, OVER) in contrast to those that demonstrate a higher capacity for taking corrective action (TIMEOUT). These conclusions can be summed up in Figure 27.
These results demonstrate a promising “raw” potential for game adaptation to help players in achieving prosocial outcomes through continuous gameplay of adaptable prosocial games over time. The main issue observed is in achieving player adherence to the in-game instructions, which as is argued in D2.6, can be achieved by encapsulating game sessions within the Prosocial model for Teaching and Learning Social and emotional skills (e.g. properly instructing, demonstrating the necessary skills prior to the game session and generalizing during post-session debriefing with the teacher). We show in this work that adaptable PR/CF elements can contribute towards maximizing the likelihood of attaining PLO outcomes, which in turn can help an offline mechanism match player groups to the proper amount of challenge in order to help players advance their learning. We aim to follow-up on this results analysis with a thorough investigation of player real-time engagement indications, and how these can be linked towards the different outcomes attained by the group. Towards this end, a fully adaptable version of the game (with PR/CF strategies developed for the player controlling the Guide character in a similar fashion as described in this deliverable) will be developed. Our final results will then be published through future deliverables with respect to small scale studies (WP7).
7 Conclusions

This deliverable is an updated version of deliverable D4.1 “1st Intelligent Adaptation and Personalization” and it provides a description of the final version of ProsocialLearn Adaptation Manager (PAM). It presented two distinct mechanisms, namely, offline and online adaptation, aiming to enhance prosocial behaviour in children using the ProsocialLearn platform as well as the multimodal engagement recognition algorithm that is used as input to the online adaptation module. Finally, in this deliverable we present an example that demonstrates how the PAM can be utilized with the sample prosocial game Path of Trust.
References


Culver, J., (2015), Relationship Quality and Student Engagement, Wayne State University.


Newmann, F. M., (1992), Student Engagement and Achievement in American Secondary Schools, Teachers College Press.


Rage (2015), http://rageproject.eu/
Appendix A: Prosocial Adaptation Manager (PAM) Manual

This Appendix is offered as an Instruction Manual for both ProsocialLearn platform and individual game developers using the platform’s adaptation capabilities, in order to: a) integrate the PAM with the rest of the platform; and b) successfully make games compatible to the Prosocial Adaptation Manager (PAM) component and efficiently use it for triggering personalization of game elements tailored to the players’ engagement estimates and PLO achievement performance.

A.1 Introduction

- The Prosocial Adaptation Manager (PAM) is a ProsocialLearn platform component responsible for personalizing prosocial games to meet the students’ needs and preferences.
- PAM is fully heterogeneous since any game provided with adaptive features can be configured to communicate with it.
- Two phases of adaptation are supported: offline adaptation and online adaptation, offering enhanced personalization capabilities to the games connecting to the PAM.
- Multiplayer games are supported through a special group adaptation mode, promoting group-specific social interactions.
- Communication with the games is provided through a minimalistic protocol based on simple web socket messages, requiring only the game developer-designated identifiers for the games’ adaptive features.

A.2 PAM-Supported indicative adaptation architecture

An example architecture for game adaptation supported by the PAM is illustrated in Figure 28. The components presented are briefly described in the bullet list below:

- **Game Server:** A server providing access to the game, i.e. the digital place where the game world “lives” and individual sessions are hosted.
- **Fusion Module:** Fuses multimodal information from an array of supported sensors (e.g. Webcam, Microphone, Keyboard, Microsoft Kinect), in order to extract player engagement.
- **ProsocialLearn (PsL) Client:** Front-end user interface integrating multiple sensors. The client executes all connection to the game server to grant access to the game. It connects to the Fusion Module to receive user engagement estimations from multimodal fusion. In this sample architecture, the PsL Client is responsible for communicating these engagement values to the PAM.
- **Client Synchronization Interface (CSI):** Supports multiple PsL Client connections. It is responsible for merging PsL Client messages that belong to the same game session.
- **Prosocial Adaptation Manager (PAM):** Provides offline, online, and group adaptation capabilities to the game hosted on the Game Server (similarly, different games hosted on multiple game servers). For each game, a dedicated “Core” is created, supporting all three adaptation capabilities.
A.3 Usage Requirements

The following list of requirements is intended for ProsocialLearn platform developers, in order to provide support structures that enable the use of the PAM with the rest of the system infrastructure:

1. **Adaptive Features**: Each game should include two sets of adaptive features, namely the game scenarios and elements. Scenarios are selected in offline (player history-based) adaptation to adjust game conditions before the game starts, and elements are used in online (real-time) adaptation for feedback and other means of keeping the players' engagement high while playing.

2. **XML file listing game PLOs**: Titles for all PLOs (i.e., skills being taught to students by the game) must be included in an XML file and placed into the PAM application folder.

3. **XML file listing adaptive feature IDs**: For every prosocial game registered, the ProsocialLearn platform should automatically generate an XML file for the PAM, containing the IDs of the scenarios and elements offered by the game for offline and online adaptation, respectively. This XML must be placed into Game IDs folder.
4. **Client Synchronization Interface (CSI):** Though any single-player game can connect directly to the PAM via a web socket interface, the ProsocialLearn platform lacks of a service that combines two or more clients into a single game session. If no such service exists, PAM cannot handle multiplayer games, since each player establishes connection asynchronously. CSI should offer such synchronization support, interconnecting PAM with multiple player clients.

A.4 **Instructions for game developers to use the PAM in their own games**

Game Developers should consider the following requirements for making their game compatible with the PAM:

1. **XML for PLOs (should be supported by the platform):** Fill out the PLOs XML file contained into the application folder, in order to include the preferred PLOs (i.e., prosocial skills) into the PAM adaptation routines.

2. **XML for IDs (should be supported by the platform):** Create an XML file that contains the IDs for the adaptive features, i.e. scenarios and elements offered by the game for offline and online adaptation, respectively. This XML must be placed into Game IDs folder inside the PAM installation directory. An example XML file contained in that folder will have the following structure:

   ```xml
   <?xml version="1.0"?>
   <GAME_IDS>
   <PLOs SIZE="1">
     <PLO ID="Following Instructions">
       <SCENARIOS SCENARIO_1_ID="S2" SCENARIO_2_ID="S3" SCENARIO_3_ID="S4"/>
     </PLO>
   </PLOs>
   <ELEMENTS ELEMENT_1_ID="E8" ELEMENT_2_ID="E9" ELEMENT_3_ID="E10"
            ELEMENT_4_ID="E11" ELEMENT_5_ID="E12"/>
   </GAME_IDS>
   
   Figure 29: Example game Game IDs XML file listing adaptive features of the game.
   
3. **Web Socket:** Create a web socket in your game to be able to connect to the IP and port that the PAM has been configured to listen to (i.e., the values contained inside the config.txt file within the PAM installation directory, and should be exposed to developers by the platform infrastructure).

4. **Listener:** Create a listener for PAM messages (see Section below). Communication with PAM involves only 3 types of messages for sending data to the PAM and 2 types for receiving input.
from the PAM. Reserved words (identifiers) are separated from update information or values with an underscore (“_”) character and are formatted in capital letters.

A.5 Messages

- Communication of the PsL Client with the PAM is achieved through web socket messages. For multiplayer games, the CSI should open one web socket connection with the PAM for every game session.
- Only 3 types of messages are supported for initialization, engagement estimation, and prosocial outcome. These are demonstrated in the Table below:

<table>
<thead>
<tr>
<th>Time</th>
<th>Client sends to PAM</th>
<th>Client receives from PAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start of the game</td>
<td>GAME_Path of Trust_Following Instructions_Bob</td>
<td>PAM_OFFLINE_scenario5_element4</td>
</tr>
<tr>
<td>During the game (multiple times)</td>
<td>ENGAGEMENT_0.743</td>
<td>PAM_ONLINE_element2</td>
</tr>
<tr>
<td>End of game</td>
<td>PROSOCIALITY_ProsocialOutcome</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4. Example messages for a communication between “Path of Trust” and PAM. The variable ProsocialOutcome can take the following values: PROSOCIAL, FAILURE or UNKNOWN, according to the outcome of the game.

A.6 Instructions for executing the PAM on a local machine after installation for testing

1. Navigate to the PAM installation directory (by default, it should be installed at C:/Program Files (x86)/CERTH Visual Computing Lab/ Prosocial Adaptation Manager). Open config.txt with any preferred text editor and enter your machine’s IP address and a preferred port.

2. Execute PAM.exe (on Windows 8 and newer operating systems, right-click on the desktop icon and select “Run as administrator” in order to allow the PAM to store data on your local hard drive) to start the web socket server. The service is terminated through the Task Manager (Ctrl + Alt + Del).

Running TestGame sample to ensure PAM is properly running on your machine

1. Execute TestGame.exe to connect an example blank game application to the PAM.
2. You should observe a folder titled PsL Adaptation and some sub-folders be added into the PAM’s installation directory.
3. Approximately 30 seconds after `TestGame.exe` was executed, XML files will appear inside the PsL Adaptation folder.

4. Terminate the game through the Task Manager (Ctrl + Alt + Del, find the `TestGame` process and select *End Process*).

5. Repeat steps 3 to 5 and observe changes to the updated profiles within the `PsL Adaptation` directory.

6. Execute `TestGame.exe` and `TestGame2.exe` in parallel to check simultaneous game support (i.e. demonstrates how the PAM can be used to handle multiple games simultaneously). Remember to terminate all services through the Task Manager (Ctrl + Alt + Del).

**Path of Trust sample JavaScript code**

```javascript
if (message_type == 'PAM') {
    if (message_body == 'OFFLINE') {
        starting_scenario = message_parts[3];
        starting_element  = message_parts[4];
    }

    else if (message_body == 'ONLINE') {
        var new_element = message_parts[3];

        if (client && client.game && client.game.wv_client && client.game.wv_host) {
            var my_client = (client.game.player_host == client) ?
                client.game.wv_host : client.game.wv_client;

            if (my_client.readyState === 1)
                my_client.send("S_PAM_ONLINE_" + new_element);
            }
        }
    }
}
```

*Figure 30: Game server listener code example.*
if (game_state == 'TIMEOUT') {
    // ... code ...
    if (client_game.player_host.readyState === 1)
        client_game.player_host.send('S_PROSOCIALITY_UNKNOWN');
    if (client_game.player_player_client.readyState === 1)
        client_game.player_player_client.send('S_PROSOCIALITY_UNKNOWN');

    if (client_game.player_host.readyState === 1)
        client_game.player_host.send('S_GAME_END');
    if (client_game.player_player_client.readyState === 1)
        client_game.player_player_client.send('S_GAME_END');
}
else if (game_state == 'SINGLEPLAYERWIN') {
    // ... code ...
    if (client_game.player_host.readyState === 1)
        client_game.player_host.send('S_PROSOCIALITY_FAILURE');
    if (client_game.player_player_client.readyState === 1)
        client_game.player_player_client.send('S_PROSOCIALITY_FAILURE');

    if (client_game.player_host.readyState === 1)
        client_game.player_host.send('S_GAME_END');
    if (client_game.player_player_client.readyState === 1)
        client_game.player_player_client.send('S_GAME_END');
}
else if (game_state == 'PROSOCIALWIN') {
    // ... code ...
    if (client_game.player_host.readyState === 1)
        client_game.player_host.send('S_PROSOCIALITY_PROSOCIAL');
    if (client_game.player_player_client.readyState === 1)
        client_game.player_player_client.send('S_PROSOCIALITY_PROSOCIAL');

    if (client_game.player_host.readyState === 1)
        client_game.player_host.send('S_GAME_END');
    if (client_game.player_player_client.readyState === 1)
        client_game.player_player_client.send('S_GAME_END');
}

Figure 31: Example game server code handling different game outcomes.
else if (game_state == 'ENGAGEMENT') {
    var my_client = (client.game.wv_host == client) ?
        client.game.player_host : client.game.player_client;

    if (client.game.player_host.readyState === 1)
        client.game.player_host.send (
            "S_ENGAGEMENT_" +
            responsiveness_player1 + '_' + accomplishment_player1);

    if (client.game.player_client.readyState === 1)
        client.game.player_client.send (
            "S_ENGAGEMENT_" +
            responsiveness_player2 + '_' + accomplishment_player2);
}

Figure 32: Game server code for triggering engagement computation.