Creative Emotional Reasoning Computational Tools Fostering Co-Creativity in Learning Processes

www.c2learn.eu

USER PROFILING AND BEHAVIOUR DETECTION

C²LEARN PROJECT DELIVERABLE NO. D3.4.1

Authors: National Centre for Scientific Research "Demokritos" (NCSR-D)

Dissemination level: *Public*

The C²Learn project has been supported by the European Commission through the Seventh Framework Programme (FP7), under grant agreement no 318480 (November 2012 – October 2015). The contents of this document do not represent the views of the European Commission and the Commission cannot be held responsible for any use which may be made of the information contained therein. Responsibility for the information and views set out in this document lies entirely with the authors.

DOCUMENT IDENTITY

Project category	Details
Deliverable code	D3.4.1
Full title	User Profiling and Behaviour Detection
Work package	WP3
Task	T3.4 User Profiling and Behaviour Detection
Consortium partners leading	NCSR-D
Consortium partners contributing	NCSR-D

DOCUMENT HISTORY

Version	Date	Handling partner	Description
v.0.1	10/09/2013	NCSR-D	Document Setup, ToC
v.0.2	20/09/2013	NCSR-D	Introduction, Theoretical background
v.02	20/10/2013	NCSR-D	Initial draft
v.03	28/10/2013	NCSR-D	Revised draft
v.1.0 - Final Version	31/10/2013	NCSR-D	Final

EXECUTIVE SUMMARY

C²Learn at a glance

C²Learn (www.c2learn.eu) is a three-year research project supported by the European Commission through the Seventh Framework Programme (FP7), in the theme of Information and Communications Technologies (ICT) and particularly in the area of Technology-Enhanced Learning (TEL) (FP7 grant agreement no 318480). The project started on 1st November 2012 with the aim to shed new light on, and propose and test concrete ways in which our current understanding of creativity in education and creative thinking, on the one hand, and technology-enhanced learning tools and digital games, on the other hand, can be fruitfully combined to provide young learners and their teachers with innovative opportunities for creative learning. The project designs an innovative digital gaming and social networking environment incorporating diverse computational tools, the use of which can foster co-creativity in learning processes in the context of both formal and informal educational settings. The C²Learn environment is envisioned as an open-world 'sandbox' (non-linear) virtual space enabling learners to freely explore ideas, concepts, and the shared knowledge available on the semantic web and the communities that they are part of. This innovation is co-designed, implemented and tested in systematic interaction and exchange with stakeholders following participatory design and participative evaluation principles. This happens in and around school communities covering a learner age spectrum from 10 to 18+ years.

About this document

Deliverable D3.4.1 comprises the first version of the User Profiling and Behaviour Detection technology and in particular: (a) the theoretical background behind the Personalization technology that will be exploited for the T3.4 and (b) a user manual of the PServer, the User profiling software that integrates all the relevant modules and offers the personalization functionality.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND TERMS

A) Abbreviated names of the project consortium partners

Abbreviation	Explanation
EA	Ellinogermaniki Agogi, Greece (coordinator)
UEDIN	The University Of Edinburgh, UK
OU	The Open University, UK
NCSR-D	National Center For Scientific Research "Demokritos", Greece
UoM	Universita ta Malta, Malta
SGI	Serious Games Interactive, Denmark
BMUKK	Bundesministerium Für Unterricht, Kunst Und Kultur, Austria

B) Other abbreviations

Abbreviation	Explanation
SOAP	Simple Object Access Protocol
WSDL	Web Services Description Language

INTRODUCTION

One of the main targets for game design through the C2Learn project is to deliver challenges tailored not only to particular player's general level of problem-solving skill, but also to the player's adeptness at particular problem-solving strategies. The game will be able to trace a player's sequence of actions, detect player experience patterns and adjust key parameters of the game, such as the game environment, the reward schemes and the modes of interaction to maximize playfulness and thereby empower creativity.

1.1 DELIVERABLE SUMMARY

In the context of the C2Learn project, personalization and problem solving techniques, such as Creative Emotional Reasoning, are associated using the user profiling technology. In particular, a certain construct of user profiles named *user stereotypes,* are exploited to deliver the required personalization functionality.

In this deliverable, we present a short introduction to the user profiling technology that will be exploited to build the user profiling service, a description of the knowledge structures and the methodology that is used to create the stereotypes and finally the system that integrates all the relevant modules and offers the personalization functionality.

1.2 THEORETICAL BACKGROUND

User profiling or user modelling technology aims to make information systems user-friendly, by adapting the behaviour of the system to the needs of the individual. A *personal user model* primarily contains information that characterise the interaction of the user with the system and other users, if interaction between users is supported. As an example, a user model for a service delivering books contains the reading preferences of that particular user. Additionally, a user model may contain personal information about the user, such as age, body size, etc.

Further to the models for individual users, generic user models that apply to groups of users are also widely used. This is the case where there is a lack of personal activity regarding the user interaction with the system (e.g., cold start problem). These group models are often called *user stereotypes*. A stereotype is one of the earliest types of a generic model that appeared in the literature (Rich, 1979) and is nowadays often encountered in digital libraries and museums. Stereotypes are typically based on external knowledge about the user, such as the user's level of expertise or other personal information. For example, a stereotype might link users of a certain age with specific reading preferences, such as History or Comics.

Early techniques exploit personal information, provided by the users, to construct the stereotype manually. In particular, in order for a system to be able to use stereotypes effectively, it must have two kinds of information. It must know the stereotypes themselves - the collections of characteristics or *attributes,* and a set of *triggers*, i.e., the events that activate the stereotype. A user can be characterized by a set of *attributes*, each of which contains a value (or values). The particular set of *attributes* used will, of course, be determined by the domain and purpose of the system as a whole, but could include such things as level of expertise with the system, specific concepts dealt with by the system that are of particular interest, or specific system tasks that are of particular concern. Stereotypes are simply collections of *attributes* -value combinations that describe groups of system users.

A system that is going to use stereotypes must also know about a set of *triggers* - those events whose occurrence signals the appropriateness of particular stereotypes. When a stereotype is activated, the predictions it makes about various characteristics should be incorporated into the model of the user.

The manual construction of stereotypes usually involves the classification of users by an expert and/or the analysis of data relating to the interests of individual users. Acquiring the stereotypes in this way is a difficult task. Thus, there is the requirement for *learning user stereotypes*. In particular, machine learning techniques such as classification (Lock and Kudenko, 2006), (Paliouras et al, 1999) can be applied to construct stereotypes. The stereotypes correspond to the description of the user classes associated with system-dependent information, i.e., information that the user is interested in, such as products, books, etc. As a concrete example, a decision rule could be constructed representing a stereotype for a particular product category. For example a stereotype rule could be *if education="high school" AND class taken NOT "history", THEN the user is interested in "mathematics"*.

Personal information about the users of a system is not always available and therefore the construction of user stereotypes may not be possible. In that case, the organization of users into groups with common interests can still be useful. This can be realized with another type of generic user models, named *user communities* (Paliouras, 2002), which do not contain personal information about the users but they can extract and represent the preferences and interests of users from their behaviour, by employing machine learning techniques.

1.3 USER PROFILING SERVICE

The C2Learn framework offers a User Profiling Service which is designed and implemented to support User profiling functionalities. The User Profiling Service identifies game player preferences and behavioral patterns, and detects the creative skills of the user in particular game scenarios.

The connection of User Profiling Service and Behaviour Detection Module to the rest of the C2Learn's modules is depicted in Figure 1. The service communicates with the game engine to detect user behaviour and takes into account the principles defined in the Assessment Methodology in order to construct behavioural models for the players. The players' models are fed to the Procedural Content Generation module and to the Adaptive Dialogue System in order to alter the game content and game dialogues respectively.

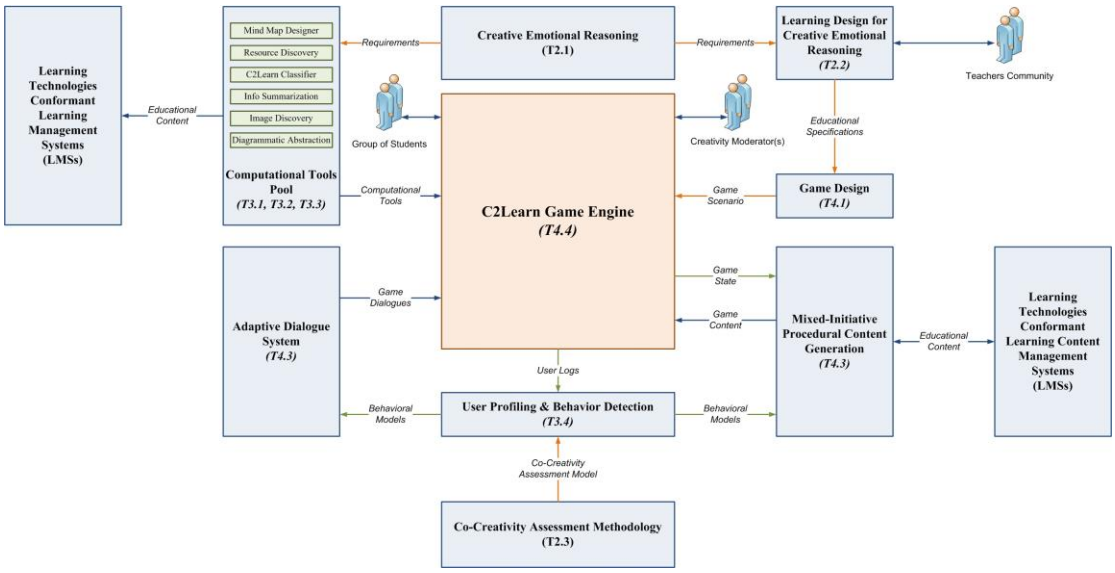

Figure 1. C2Learn *core process and components*

The User Profiling service exploits personal, behavioural and contextual data in order to construct a player model or models for group players. In particular, player profiling algorithms will be used to construct models of player experience such as emotions[1], motivations and goal setting strategies. The game will be able to trace a player's sequence of actions, detect player experience patterns and adjust key parameters of the game, such as the game environment, the reward schemes and the modes of interaction to maximize playfulness and thereby empower creativity. This would allow the system to deliver challenges tailored not only to a particular player's general level of problem-solving skill, but also to the player's adeptness at particular problem-solving strategies. Moreover, when problem solving tasks are designed any kind of social inter-connectedness displayed by each player can also be utilized.

[1] T3.3 Emotive Reasoning and Emotion Detection

2. USER PROFILES FOR C2LEARN

User Profiles for the C2Learn platform are associated to the processes of the Co-Creativity Assessment, the Mixed-Initiative Procedural Content Generation and the Adaptive Dialogues, as shown in Figure 1. Following this association we can identify two main classes of User Profiles, which are manifested as Creativity-Based User Stereotypes and Game-Based User Stereotypes. These classes are discussed at the following sections.

2.1 CREATIVITY-BASED USER STEREOTYPES

Creativity-based user stereotypes are directly related to the process of Co-Creativity Assessment. In particular, users can be assigned to these stereotypes based on the analysis and evaluation of their creativity. The main assumption regarding these stereotypes is that they are *orthogonal*, i.e. a user/ learner can be associated with only one of the given stereotypes at a specific time instance.

In the following paragraphs, we provide an example of how such stereotypes could be defined using the Tentative Categorization Scheme for evaluation described in Deliverable D2.3.1, Co-creativity Assessment Methodology.

Creativity Category	Characteristics
Attending to ***ethics*** and ***impact of ideas***	1. Creates new associations between ideas 2. Actively explores the consequences of the newly created associations between ideas 3. Exhibits awareness of and concern / interest for the impact of new ideas on the group's values 4. Actively promotes ideas that are deemed valuable by the group
Engaging in ***dialogue***	1. Engages in debate over ideas 2. Promotes dialogue within group (poses questions, respects different viewpoints and/or encourages members of the group to voice their ideas) 3. Actively negotiates conflict and/or seeks alternate paths
Being in ***control***	1. Takes a leading role during different phases of the creative process 2. Exhibits a firm grasp of the rules in the system underlying the challenges facing the groups 3. Takes decisions and instigates action
Engaged action	1. Immerses him/herself in the experience of the creative process 2. Facilitates immersion in the experience of the creative process for the rest of the group 3. Willing to take risks and/or leaving his/her 'comfort zone'
Intervention and ***reframing***	1. Creates new analogies as building blocks of the creative process 2. Actively experiments with re-combining elements of the creative challenge 3. Actively facilitates a shift of perspective: a. Uncovers hidden aspects of the creative challenge b. Goes beyond the material provided by the description (elements) of the challenge, recasting the challenge in a new light (as a whole or through re-formulating elements of it)

Table 1. Creativity Categories defined in D2.3.1

Table 1 (taken from D2.3.1), presents a non-orthogonal categorization scheme that will be used in the C²Learn evaluation. This scheme entails five (5) classes of creativity capability that can be directly assessed via a measurement of each capability's characteristics. Furthermore, there exist three (3) performance levels in each of these categories: Low, Medium, and High.

Let that each category characteristic presented in Table 1 can be measured on an integer scale, ranging from zero (0) to five (5) inclusively. Let also that the characteristics of each category are not equally important to the overall assessment of the corresponding category, but rather they carry weights. Then, the overall score of a given learner in a category is the weighted sum of his/ her scores in the different characteristics belonging to the category. More formally, let L be a learner, and let C be a creativity category with a set of characteristics $K = \{k_1, \dots, k_n\}$. Each characteristic k_i carries a weight w_i and it holds that:

$$\sum_{i=1}^{|K|} w_i = 1$$

We denote the score of learner L in a specific characteristic k_i of the category C as: $Score(L,k_i)$. The overall score of the learner in the category C, is thus given by the following equation:

$$CScore(L,C) = \sum_{i=1}^{|K|} w_i \, Score(L,k_i)$$

It is evident that since $Score(L,k_i)$ ranges from zero (0) to five (5) for all characteristics of the category C, the overall score of the category C, $CScore(L,C)$, also ranges in the [0, 5] range.

We proceed to give an example for the calculation of the creativity category score. The quantitative information on the presented example is exemplary at this stage.

Let that for the first category depicted in Table 1, i.e. "Attending to ethics and impact of ideas", the respective weights of its characteristics are the ones depicted in the following table.

Creativity Category	Characteristic	Weight
Attending to ethics and impact of ideas	Creates new associations between ideas	0.2
	Actively explores the consequences of the newly created associations between ideas	0.3
	Exhibits awareness of and concern / interest for the impact of new ideas on the group's values	0.4
	Actively promotes ideas that are deemed valuable by the group	0.1

Let also that the performance levels of this category (Low, Medium, High) are defined as ranges over the observed category score of the learners as follows:

Creativity Category	Performance Level	Overall Score	
		Min	Max
Attending to ethics and impact of ideas	Low	0	2
	Medium	2	4
	High	4	5

Let also that a Learner L has achieved the following scores in each characteristic of this category, following the appropriate creativity assessment process defined in D2.3.1:

Creativity Category	Characteristic	Score(L,k_i)
Attending to ethics and impact of ideas	k_1: Creates new associations between ideas	3
	K2: Actively explores the consequences of the newly created associations between ideas	4
	K3: Exhibits awareness of and concern / interest for the impact of new ideas on the group's values	5
	K4: Actively promotes ideas that are deemed valuable by the group	3

The overall Category Score for the learner L for this category is, therefore:

$$CScore(L,C) = \frac{1}{4}\sum_{i=1}^{4} w_i \, Score(L,k_i) = 4.1$$

Taking into account the definition for the performance levels in the category, it holds that the specific learner has achieved a *High* performance for the specific creativity category.

The creativity-based stereotype scheme that will be used for profiling the C2Learn users is orthogonal, i.e. a learner can be classified to exactly one stereotype. For this purpose, each profiling stereotype uses as features the creativity evaluation categories that are depicted in Table 1.

What's more, each stereotype is defined by the range of values in each creativity category that must hold for a user in order to be associated with the specific stereotype.

Let that we have a profiling stereotype scheme consisting of five (5) stereotypes and that the ranges on the performance levels of creativity categories that a learner must have achieved in order to belong to each stereotype are as depicted in the following table.

Profiling Stereotype	Performance Levels of Creativity Evaluation Categories				
	Attending to ethics and impact of ideas	Engaging in dialogue	Being in control	Engaged action	Intervention and reframing
A	Low, Medium	Medium, High	Medium, High	Low, Medium	High
B	Low, Medium	Low, Medium	Medium, High	High	Low
C	Low	High	Low	Low	Medium, High
D	Medium, High	Low, Medium	Low, Medium	Medium, High	Low, Medium
E	High	High	Low, Medium	Low, Medium	Medium, High

The ranges of the performance values for each category are selected in such a way that the profiling stereotypes are orthogonal, i.e. a learner can be associated with only one stereotype. State-of-the-art machine learning techniques will be used for extracting the profiling stereotypes that pertain to the stated requirements (orthogonality of stereotypes, completeness on the learner's profile space).

Let us also assume that the examined learner L has achieved the following performance levels in the defined creativity categories:

Learner	Performance Levels of Creativity Evaluation Categories				
	Attending to ethics and impact of ideas	Engaging in dialogue	Being in control	Engaged action	Intervention and reframing
L	***High***	Medium	Low	Medium	Low

We compare the performance levels of the learner with the established ranges of each profiling stereotype in order to associate him / her with one of the stereotypes. In the specific example, learner L is associated with the profiling stereotype D.

2.2 GAME-BASED USER STEREOTYPES

Game-based user stereotypes, are constructed using information from players utilizing the C2Learn Game Engine. Since, they are game dependent stereotypes, they can be associated to a lower layer of the user profiling service. Each game has its own stereotypes whose attributes are designated by game designers, whilst attribute values are filled from players' interaction with the game. Students can be initially assigned to a certain stereotype on a particular game and during the same game it might change stereotype based on their activity.

Game-based stereotypes are exploited by the Mixed-Initiative Procedural Content Generation and the Adaptive Dialogue System modules to offer personalization functionality. In particular, they are part of the Creative Suite, which provides the students the relevant tools in order to create digital artifacts. The stereotypes will be used to modify the game experience according to student's observed characteristics and preferences.

As a concrete example we consider the *Sketch Game*. The following stereotypes can be implemented: (a) *Beginner Sketcher*, (b) *Intermediate Sketcher* and (c) *Advanced Sketcher.* For these stereotypes a set of characteristic attributes could be: (a) Color, (b) Shapes 2D, (c) Shapes 3D, (d) Connectors and (e) Brush. Other characteristics can also be defined and used. The Beginner Sketchers stereotype could be the following:

Color	Shapes 2D	Shapes 3D	Connectors	Brush
No	Yes	No	No	Yes

whereas the Advanced Sketcher's Stereotype could be:

Color	Shapes 2D	Shapes 3D	Connectors	Brush
Yes	No	Yes	Yes	No

2.3 CREATIVITY-BASED AND GAME-BASED STEREOTYPE ASSOCIATION

The utilization of the C2Learn Game Platform would have a potential impact on students' creativity. This impact will be recorded in the profile of the student since he/she will modify his/her preferences and eventually, his/her behavioural patterns while playing a certain game. In this manner, a student which has initially been assigned to a certain game-based stereotype, for example the Beginner Sketcher stereotype in a previous section, might be transferred to the Intermediate or even the Advanced Sketcher.

These changes in the game-based stereotypes might also change the values of attributes for the creativity-based stereotypes. Using the evaluation tools described in the Co-Creativity Assessment Methodology (D.2.3.1) we can reveal that a Learner L, who has initially been assigned to a Creativity-based stereotype C might be transferred to the Stereotype B, after using the C2Learn's Game Engine for a certain amount of time.

Following the above rationale, it will be safely assumed that an association exists between the two types of stereotype, from the lower to the higher layers. This association, depicted in Figure 2, is in a latent form and can be discovered by searching for correlations between the attributes of both types of stereotypes. For example, we can observe, that the continuous usage of the Shapes3D attribute on the Advanced Sketcher's stereotype mentioned above, might have an impact on intervention and reframing attribute of a Creativity-based stereotype. Machine learning techniques can be employed to discover such correlations.

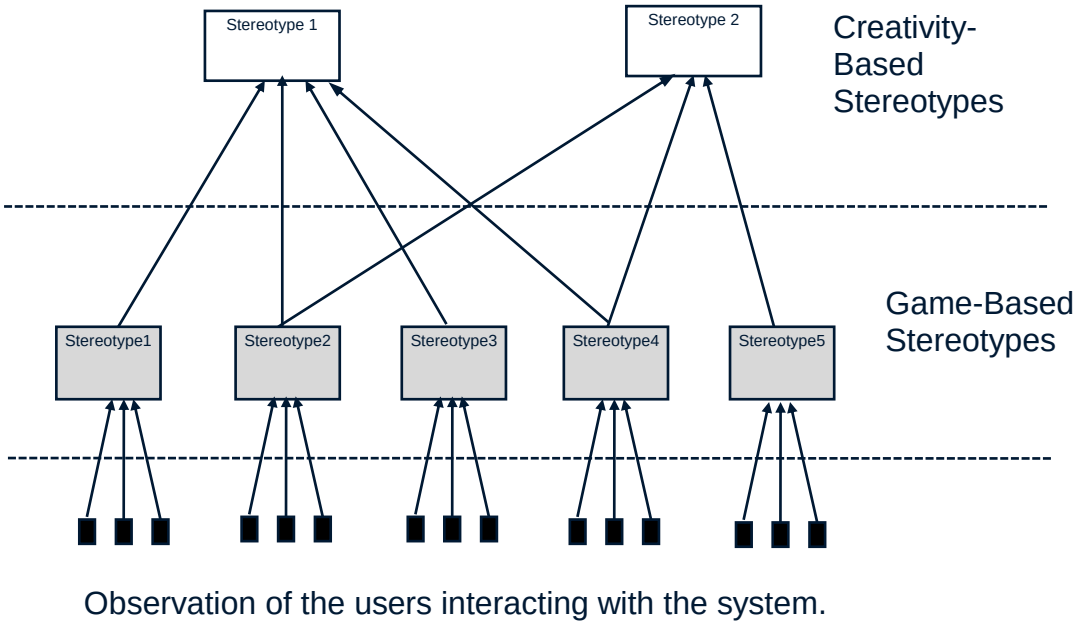

Figure 2. Creativity and Game-Based Stereotype Associations

3. C2LEARN USER PROFILING REQUIREMENTS

The design and implementation of the User Profiling Service should meet a number of functional and non-functional requirements. These requirements are described in the following sections.

3.1 FUNCTIONAL REQUIREMENTS

The following features should be present in the final application.

3.1.1 ACCESS TO ADAPTIVE APPLICATIONS

The User Profiling Service should be available for different application systems, i.e. C2Learn games. To realize this, an interface specification must be supported which provides a versatile communication with other systems.

3.1.2 USER INFORMATION STORAGE

The application systems are responsible for delivering user information to user profiling service. The user profiling system though, on itself does not collect user information directly but must store, organize and process the received data. User information must be organized in a component-based manner, where raw data is recorded in a user profile or user group profiles. Based on these profiles, user models exploit this information, interpret it and infer additional information about the user.

3.1.3 SUPPORT OF USER GROUP PROFILES

The application system should offer the creation and storage not only for personal user profiles, but for group profiles corresponding to more than one users of the application. In particular, the application should user stereotypes and user communities.

3.1.4 INITIALIZE, UPDATE AND DISPATCH USER PROFILES

The implemented user profiles must be initialized with the stored user information. However, if the user information changes, the affected user profiles must be updated.

The user profiling system should be able to deliver user profiles and therefore the application system must be able to query the user profiling system for a particular user profile.

3.1.5 MODULAR BASED ARCHITECTURE

The user profiling system must ensure its flexibility, reusability and extensibility. Therefore its architecture should be modular based, with modules able to perform specific tasks.

3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements of the user profiling service should be adhered to in the resulting system and are not related to functional aspects.

3.2.1 PERFORMANCE

The service should operate at a reasonable speed and with a reasonable response time to commands.

3.2.2 OPERATIONAL PLATFORM

The service should be independent of the underlying hardware and operating system. The software should not require any specialized hardware to run. Any other software resources required for operation of the program should be distributed with the main program.

3.2.3 DATA STORAGE

The user data, as well as any other kind of application data should be stored on a Relational Database Management System (RDBMS).

3.2.4 PRIVACY AND SECURITY

The privacy and security of user information must be ensured by the user profiling service. In particular, a clear set of ethical principles should be underpinned as described in Section 2.5 of D2.3.1. These will in part be derived from Data Protection Regulations and will comply with Directive 95/46/EC to ensure correct handling of data and privacy

3.2.5 SOFTWARE FLEXIBILITY FOR FURTHER DEVELOPMENT

The service should facilitate future additions and modifications to the software.

3.2.6 OPEN SOURCE LICENCING

The application should be distributed under an Open Source Licence and no additional costs should be applied.

4. USER PROFILING APPLICATIONS

4.1 STATE-OF-THE ART OVERVIEW

There is a variety of applications that offer User Profiling Services. The most important of them are briefly described in the following sections.

4.1.1 WEB SPHERE PORTAL

Web Sphere Portal (www-03.ibm.com/software/products/us/en/websportfami) is a commercial application offered by IBM. It supports a series of Web portal design tools, including administration, security, logging, personalization and recommendation functions. The user behaviour, such as products purchased, items added/removed form a shopping basket, user's navigation history, products that match the products the user has selected, and product ratings, is recorded by the application. This information is used to build the user's profile. In addition there be can be defined through rules, some collective profiles, such as: customers of higher and/or lower income, or customers from a certain geographic area etc. For instance, customers of higher income might be presented with premium products. The type of recommendation is based on relevant data availability, such as enough interaction with the site.

4.1.2 ORACLE WEBCENTER SITES & ORACLE REAL-TIME DECISIONS

Oracle WebCenter Sites (www.oracle.com/us/products/middleware/webcenter/sites/overview/index.html) and Oracle Real-time Decisions (http://www.oracle.com/us/solutions/business-analytics/business-

intelligence/real-time-decisions/overview/index.html), is a user profiling platform offered by Oracle with a free licence for no-commercial usage. The product allows a marketer to build rules based on, the so named, implicit or explicit user criteria. Implicit criteria include: location, time of day and day of week, click stream behaviour, and search engine referrer keywords. Explicit criteria include known visitor profile information such as age, gender and specified interests. Rules can be stated based on either the implicit, the explicit criteria or on both; should they be available to promote or suggest products. In addition the Real-Time Decisions component of the system, can build data driven profiles and to generate subsequent recommendations.

4.1.3 SOCIAL MERCHANT

Social Merchant (http://www.thesocialmerchant.com) is a commercial product which offers personalized functionalities for e-commerce sites. It allows online retailers to inject relevance into their ecommerce site, transforming it into a dynamic, personalised sales environment. Retailers can increase conversion rates and revenue by adapting product recommendations and other content based on valuable visitor behavioural data such as the past and current behaviour of individual visitors, the behaviour of others ("crowd wisdom") and the visitor preferences.

Social Merchant supports a deep integration with ecommerce platforms and enable real-time product catalogue awareness, including stock availability. It offers a flexible framework and a simple API which allows to inject relevant product recommendations and personalize across multiple touchpoints, including promotional and transactional emails, social networks, mobile sites/apps, instore kiosks and more. It also supports a variety of personalization strategies through a set of algorithms.

4.1.4 OPENTEXT WEB SOLUTIONS

Open Text Web Solutions (http://www.opentext.com/wsm/index.htm) target specific audiences and deliver Web content in context, making it more relevant and actionable through every online channel. Open Text Web Solutions customizes Information to target users. It allows the creation of user profiles from data gathered implicitly and explicitly. Subsequently it tailors the offered information to meet the specific demands of your target audience, notice any changes quickly, and respond accordingly.

4.1.5 PREDICTIONIO

PredictionIO (http://prediction.io) is an Open Source Machine Learning Server. It empowers programmers and data engineers to build smart applications. PredictionIO is built on the top of solid Open Source technology, and supports the prediction of user behaviour, offering of personalized applications, such as video, news, products etc.

A number of built-in algorithms are available in PredictionIO. An algorithm, and the setting of its parameters, determines how predictive model is constructed. In another word, prediction accuracy or performance can be improved by tuning a suitable algorithm. Moreover, PredictionIO comes with tools and metrics for algorithm evaluation.

4.2 USER PROFILING APPLICATIONS FOR C2LEARN

The user profiling applications mentioned above, meet the following requirements (Table 1) for the implementation and design of a User Profiling service.

		User Profiling Systems				
		WebSphere	**Oracle WebCenter**	**Social Merchant**	**OpenText**	**PredictionIO**
Functional Requirements	Adaptive Application	✓	✓	✓	✓	✓
	User Information Storage	✓	✓	✓	✓	✓
	User Groups					
	Initialize Update Dispatch User Profiles	✓	✓	✓	✓	✓
	Modular Based Architecture					✓
Non-Functional Requirements	Performance	✓	✓	✓	✓	✓
	Operational Platform	✓	✓	✓	✓	✓
	Data Storage	✓	✓	✓	✓	✓
	Privacy and Security	✓	✓	✓	✓	✓
	Software Flexibility	✓	✓	✓	✓	✓
	Open Source				✓	✓

Table 2. User Profiling Systems Comparison

4.3 PSERVER AS THE USER PROFILING SYSTEM FOR C2LEARN

From the above Table, we conclude that the examined user profiling applications meet a number of functional and non-functional requirements for the design and the implementation of the C2Learn User Profiling Service.

However, the systems discussed above failed to meet one important requirement which is the support of the construction of profiles for group of people, such as stereotypes or user communities. Although user profiles could be individual and predict characteristics based on behaviour for a particular user, it will be useful to create models based on data from numerous users.

In the context of the C2Learn project the User Profiling Service will be able to search for like-minded users with similar emotions in user profiles and assign them to the same community.

Moreover, the need for group user models arises from the difficulty of labelling large quantities of data for supervised learning leads to the adoption of unsupervised methods for learning such communities, especially clustering. An important aspect on the choice of the clustering method is that it should allow cluster overlap. This is important for emotion detection since a user might share emotions with more than one community.

The Personalisation Server designed and implemented by the Institute of Informatics and Telecommunications of the NCSR Demokritos meets all the above requirements. PServer is a general-purpose server that offers personalization functionality that can be used by different kinds of applications offering personalized services. The PServer's framework has been designed to be platform and application independent. It can run on different operation systems, and can provide personalization services regardless the application domain. In the following sections we discuss in details, the PServer's architecture and functionality, as well as how this functionality is applied in the C2Learn's context.

5. PERSONALIZATION SERVER (PSERVER)

PServer is an Open Source (Apache License 2.0) application developed by Institute of Informatics and Telecommunications of NCSR "Demokritos", which uses state-of-the-art artificial intelligence to learn the user preferences and interests, to extract user behavioural models (personal or group) and subsequently offers the possibility of personalization of other applications.

5.1 PSERVER ARCHITECTURE

PServer separates user modelling from the rest of the application at both logical and physical levels. At the logical level, PServer features a flexible, domain-independent data model that is based on four entities: users, which are represented by some identifier; stereotypes, which are predefined user groups with specific attributes; features, which are application dependent characteristics that may or may not attract user preference; and properties (or attributes) that depend on persistent user dependent characteristics.

An overview of PServer along with a recommendation engine is depicted in Figure 2. The modules of that appear are explained in the next paragraphs, in particular the manager has the overall control of the system. It should be stressed that the recommendation engine is not part of the PServer, and it will be developed for the C2Learn project.

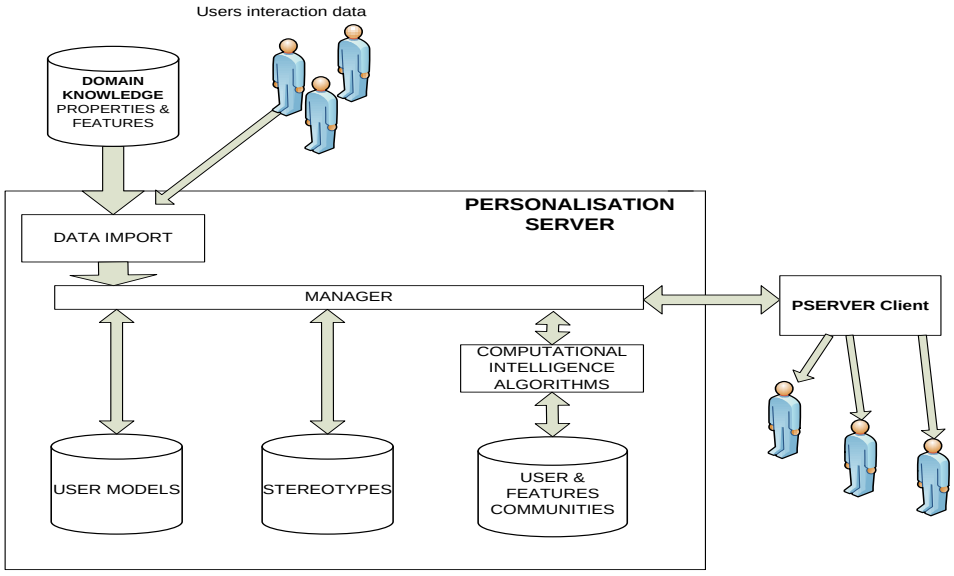

Figure 3. Overview of the Personalisation server.

5.2 LOGICAL LEVEL

PSerser offers three types of user model: *personal user models*, *models of user stereotypes* and *models of user communities:*

- ***Personal user models****:* These models are user specific and are made of *features* and *properties (or attributes)*. Attributes are persistent and external to the system user characteristics (e.g. age, knowledge level) and features are application specific and they

assume values depending on user interest as the user interacts with a system (e.g reading preferences, such as History). Features and attributes are part of the domain knowledge.

- ***Stereotypes**:* These are *predefined* user groups with certain common attributes. (e.g. age, sex, lifestyle etc.). Stereotypes have features and attributes like user models, but it is not necessary for a stereotype to have the same number of features and attributes as another. Based on their attributes, users can be assigned to one or more stereotypes.

- ***User Communities**:* These can be created with machine learning algorithms based on user interaction data, thus they are data driven:

  - Such algorithms detect *similarities* among *users* (based on their feature values) to create user groups. Thus a community is a feature based group (whereas a stereotype is a property based group).

  - Or they detect *similarities* among *features* to create feature groups. Feature groups can be useful in many cases; for instance if two features belong into the same group and a user has expressed interest in the first, then we could reasonably infer that he would be interested in the second.

5.3 PHYSICAL LEVEL

PServer may reside at a different machine from the application with which it communicates, since PServer is implemented as a Web server that listens to a dedicated port, and all requests have the form of HTTP messages. Thus:

- PServer clients are Web applications. In the above figure, the client is a Recommendation Engine.

- Responses are encoded in XML syntax, and especially made XSL stylesheets allow them to be displayed on web browsers, if needed.

- To facilitate applications, a client-side library of classes is available that can be incorporated into the application to handle all low-level communication details.

5.4 PSERVER APPLICATIONS

PServer has already been used successfully in a number of applications and European Projects including the following:

NewSumOnTheWeb (http://www.newsumontheweb.org): PServer has been embedded on NewSumOnTheWeb, which is the web edition of NewSum news portal.

SERVIVE EU Project (http://www.servive.eu): The SERVIVE (SERVice Oriented Intelligent Value Adding nEtwork for Clothing-SMEs embarking in Mass-Customisation) proposed the enlargement of the assortment of customizable clothing items currently on offer, the enhancement of all co-design aspects (functionality and fun) and the development and testing of a new production model based on decentralized networked SME cells.

INDIGO EU Project (http://www.ics.forth.gr/indigo/index.html): INDIGO (Interaction with Personality and Dialogue enabled Robots) developed human-robot communication technology for intelligent mobile robots that operate and serve tasks in populated environments

*CROSSMARC EU Project (*https://www.iit.demokritos.gr/project/crossmarc/): CROSSMARC (Cross-lingual Multi-agent Retail Comparison) developed a commercial-strength technology for e-retail product comparison, which lead to systems able to process pages written in several languages (initially, Greek, English, Italian, and French), and adaptable to new product types via semi-automatic tool.

*MITOS GSRT/Ministry of Development Project (*https://www.iit.demokritos.gr/project/mitos*/)*: MITOS (Information Retrieval and Extraction from Financial News Items in Greek) offers a prototype for the retrieval and extraction of information, related to the interests of particular users, from financial news items written in Greek.

M-PIRO EU Project (https://www.iit.demokritos.gr/project/m-piro): The M-PIRO (Multilingual Personalised Information Object) project developed state of the art techniques and tools in natural language generation, speech synthesis, user modelling, and their interaction with adaptive hypermedia and virtual reality systems.

6. EXPLOITING PSERVER FOR C2LEARN

6.1 PSERVER FUNCTIONALITY

PServer meets all the functional and non-function requirements for the design and implementation of the C2Learn User Profiling Service. It is designed to be platform and application independent and thus it can run on different operating systems, and can provide generic personalisation services to any applications regardless of their content. The application can also be served concurrently.

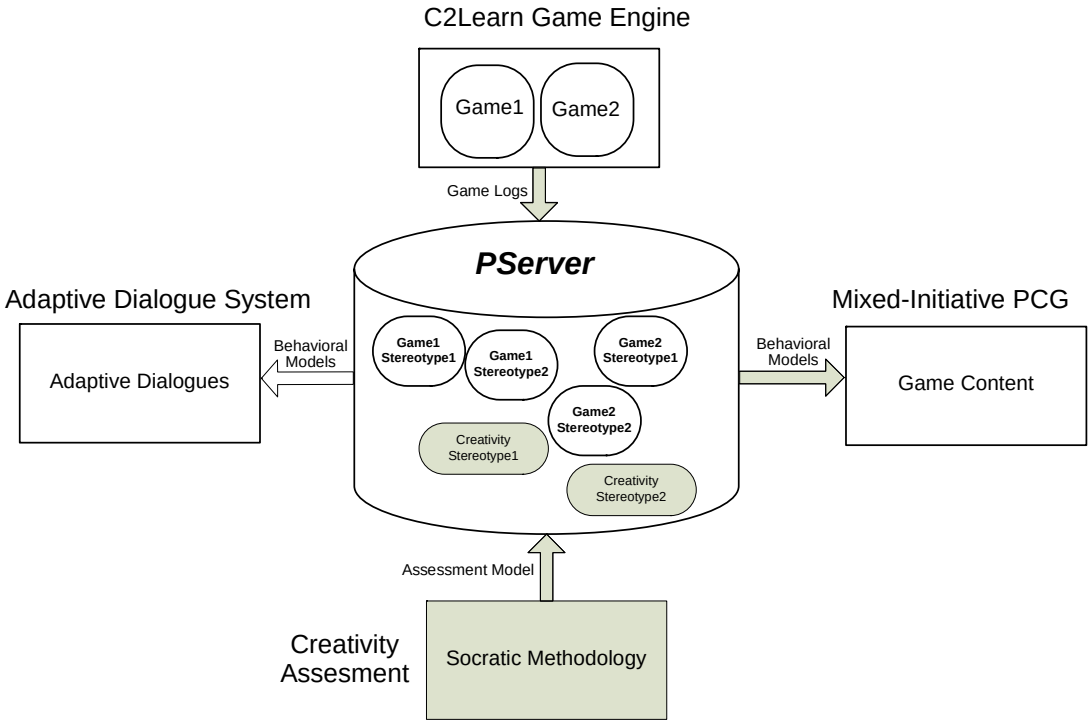

Figure 4 PServer for C2Learn

Figure 4 portrays the specific functionality offered by PServer for C2Learn. The PServer has a dual role. First it operates as the repository for the two types of stereotype, i.e., game-based and creativity-based stereotypes. Game-based stereotypes are created by domain-game experts and their attributes are updated by exploiting the game logs. The creativity-based stereotypes are also built by the creativity domain experts, but their attributes are updated using the Socratic Methodology during the Creativity Assessment Task. Game-based stereotypes are fed to the Mixed-Initiative Procedural Content Generation and the Adaptive Dialogue Systems, as behavioural models in order to offer personalization functionality, i.e., adapt the Game Content and the Dialogues to preferences and user behavioural patterns.

The second role of the PServer, is the monitor the stereotypes activity. In particular, PServer keeps statistical information about the stereotypes, such as time stamps or frequency of usage. The exploitation of this information can help towards the identification of the correlations between the attributes of the two-types of the stereotype and ultimately, the stereotype associations. This will be realized by the implementation as PServer modules, of state-of-the-art Machine Learning algorithms.

6.2 PSERVER IMPLEMENTATION

PServer's implementation has been done with the Java programming language and without any usage of operating system API, and is usable under any known operating system. PServer user information is centrally installed and maintained in a RDBMS. User models are stored in the same RDBMS either in the form of personal user models, or in user stereotypes, user communities, or feature groups.

Apart from its platform independence, PServer is designed to communicate over HTTP protocol. Applications can make simple HTTP requests that contain the parameters of the request, and gather results through XML documents. This is pretty much the same way that RESTful web services work, but it is even "lighter".

In the future, PServer will be extended so as to offer more private and secure information exchange, between the various applications. Moreover, it will be tested with new algorithms that would be able to learn user stereotypes instead of statically defining them.

7. PSERVER API FOR C2LEARN

For the connection to the Game Platforms, such as Unity, and the exchange of information, a set of SOAP Web services will be implemented and provided as an API.

7.1 WEB SERVICE DESCRIPTION

Below follows the XML WSDL model of the Web services.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.2.6-1b01 svn-revision#13094. -->

<definitions targetNamespace="http://c2learn/" name="C2LearnWSService" xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:wsp="http://www.w3.org/ns/ws-policy" xmlns:tns="http://c2learn/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">

 <types>

  <xsd:schema>

   <xsd:import namespace="http://c2learn/" schemaLocation="C2LearnWSService_schema1.xsd"/>

  </xsd:schema>

 </types>

 <message name="setAttributes">

  <part name="parameters" element="tns:setAttributes"/>

 </message>

 <message name="setAttributesResponse">

  <part name="parameters" element="tns:setAttributesResponse"/>

 </message>

 <message name="addUserToStereotype">

  <part name="parameters" element="tns:addUserToStereotype"/>

 </message>

 <message name="addUserToStereotypeResponse">

  <part name="parameters" element="tns:addUserToStereotypeResponse"/>
```

```
 </message>
 <message name="getStereotype">
  <part name="parameters" element="tns:getStereotype"/>
 </message>
 <message name="getStereotypeResponse">
  <part name="parameters" element="tns:getStereotypeResponse"/>
 </message>
 <message name="createStereotype">
  <part name="parameters" element="tns:createStereotype"/>
 </message>
 <message name="createStereotypeResponse">
  <part name="parameters" element="tns:createStereotypeResponse"/>
 </message>
 <portType name="C2LearnWS">
  <operation name="setAttributes">
   <input wsam:Action="http://c2learn/C2LearnWS/setAttributesRequest"
message="tns:setAttributes"/>
   <output wsam:Action="http://c2learn/C2LearnWS/setAttributesResponse"
message="tns:setAttributesResponse"/>
  </operation>
  <operation name="addUserToStereotype">
   <input wsam:Action="http://c2learn/C2LearnWS/addUserToStereotypeRequest"
message="tns:addUserToStereotype"/>
   <output wsam:Action="http://c2learn/C2LearnWS/addUserToStereotypeResponse"
message="tns:addUserToStereotypeResponse"/>
  </operation>
  <operation name="getStereotype">
   <input wsam:Action="http://c2learn/C2LearnWS/getStereotypeRequest"
message="tns:getStereotype"/>
   <output wsam:Action="http://c2learn/C2LearnWS/getStereotypeResponse"
message="tns:getStereotypeResponse"/>
```

```
  </operation>
  <operation name="createStereotype">
   <input wsam:Action="http://c2learn/C2LearnWS/createStereotypeRequest"
message="tns:createStereotype"/>
   <output wsam:Action="http://c2learn/C2LearnWS/createStereotypeResponse"
message="tns:createStereotypeResponse"/>
  </operation>
 </portType>
 <binding name="C2LearnWSPortBinding" type="tns:C2LearnWS">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
  <operation name="setAttributes">
   <soap:operation soapAction=""/>
   <input>
    <soap:body use="literal"/>
   </input>
   <output>
    <soap:body use="literal"/>
   </output>
  </operation>
  <operation name="addUserToStereotype">
   <soap:operation soapAction=""/>
   <input>
    <soap:body use="literal"/>
   </input>
   <output>
    <soap:body use="literal"/>
   </output>
  </operation>
  <operation name="getStereotype">
   <soap:operation soapAction=""/>
```

```
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
  <operation name="createStereotype">
    <soap:operation soapAction=""/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
 </binding>
 <service name="C2LearnWSService">
  <port name="C2LearnWSPort" binding="tns:C2LearnWSPortBinding">
    <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
  </port>
 </service>
</definitions>
```

7.2 METHODS DEFINITION

7.2.1 ADDATTRIBUTES

The method is used to add stereotypes attributes in the PServer.

Arguments

An XML file with attribute names and default values.

Return Value

An XML file with attribute names for confirmation.

7.2.2 CREATESTEREOTYPE

The method is used to add a stereotype in the PServer.

Arguments

Stereotype name as String and an XML file with attribute names and values.

Return Value

The stereotype name.

7.2.3 GETSTEREOTYPE

The method is used to add a stereotype in the PServer.

Arguments

An XML file with attribute names and values.

Return Value

The stereotype's features list.

7.2.4 ADDUSERTOSTEREOTYPE

The method is used to add a stereotype in the PServer.

Arguments

User ID as String and an XML file with attribute names and values.

Return Value

The stereotype's name.

REFERENCES

- Z. Lock, D. Kudenko, “Interaction Between Stereotypes”, In Proc. of International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH2006), 2006.
- G. Paliouras, V. Karkaletsis, C. Papatheodorou and C.D. Spyropoulos, "Exploiting Learning Techniques for the Acquisition of User Stereotypes and Communities," Proceedings of the International Conference on User Modeling (UM), CISM Courses and Lectures, n. 407, pp. 169-178, Springer-Verlag, 1999.
- G. Paliouras, C. Papatheodorou, V. Karkaletsis and C.D. Spyropoulos, "Discovering User Communities on the Internet Using Unsupervised Machine Learning Techniques". Interacting with Computers, v. 14, n. 6, pp. 761-791, 2002
- E. Rich, User modeling via stereotypes. Cognitive Science, 3: 329-354; 1979.

APPENDIX

PSERVER USAGE

PServer's allows the connection from more than one applications or sites. In order to distinguish these client applications, a unique client could be created with username and pass, so as to direct properly the information to PServer. To create a new client, you should go to "PServer clients" page, type in your client username and password, and click on the insert user button.

Figure 5. Creating a client Application to PServer

SYSTEM SPECIFICATION

After the creation of client and in order to integrate PServer into his site – application, the administrator needs to follow the steps below:

Define the User Model

In order to define the User Model, an admin must decide which PServer personalisation features he wants to integrate into his site or application, according to his special needs and requirements. For example, the admin must decide if he wants data about Personal User Models, Stereotypes, User Communities, or all of the above.

Define the data to be stored

At this step, the Admin must specifically determine the data that needs to be stored from the PServer, in order to fulfil the User Model requirements which are described in the above step. For example, if the admin wants to use only the Personal user model, he is not obliged to store their specific User Attributes and characteristics.

Data Feeding to PServer

There are two proposed ways of adding data to PServer. Real Time Data feeding, or periodical data feeding. Here, an admin should decide whether he would like to feed real time data to PServer, or whether he would like to feed data periodically any time he wants. For example, if a given

site has too much traffic, the recommended way is to save the data to a log file and periodically feed PServer with this file, in low traffic periods.

Client Engine

At this last step, an engine should be developed in order to accept the PServer responses, and after proper processing - according to the admins requirements- to present the Personalised results to the end-user.

Setting up PServer on a pre-existing system

In order to acquire PServer's full functionality in a predefined system, the admin should follow all of the four above steps, and additionally, he should develop a wrapper, which would acquire data from the existing DB, according to the User Model requirements (Steps 1 and 2). The wrapper takes the data, and feeds PServer accordingly.

PSERVER REQUESTS

PServer is an HTTP application server that runs plug-ins (pservlet pbeans), serves the requests, and returns responses. Based on this logic, the requests that you make to PServer have specific structure.

http://url/<pbean_name>?clnt=name|pass&com=some_com&param1=val1&param2=val2...

First, you define the URL that links to PServer, then the pservlet that you want to use, and then the client name/pass that makes the request and the pservlet parameters. Every pservlet defines its own parameters. The pservlets that we have written have a common structure; they need to get a com parameter, which defines the pserver command that needs to be called, and then defines the parameters of the specific command that has been specified by the com parameter. The pservlets can return any type of document (HTML, XML, JSON etc), but the ones that we have implemented and we use return XML. The admin pservlet is an exception; it returns HTML documents, and this is needed to access it through a web browser. The XML that is returned is always structured like a 2d array.

PSERVER DATA

There are two ways to enter information into the PServer:

a) the web browser

b) through a program with http requests.

Data are not entered directly into the database, but rather through a client application to the PServer.

http://localhost:1111/pers?clnt=testclient|testpass&com=addattr&job=1

whereas:

'pers'=personal mode (i.e. not stereotypes, or communities. In other words individual users).

'testclient'=the client

'testpass'=password to the client

'\&' separator

'com=addattr' = command add attribute

'job=1', the hypotherical attribute job assumes the value 1

ATTRIBUTES AND FEATURES

INSERTING ATTRIBUTES

For adding attributes:

pers?clnt=testclient|testpass&com=addattr&age=null&sex=null&occupation=null

This adds age, sex, occupation and their default values (null, null, null) as attributes in 'attributes' table.

Add attribute values along with the user:

pers?clnt=testclient|testpass&com=setusr&usr=user1&attr_age=12&attr_sex=male&attr_occupation=journalist

This adds the user1 to table 'user' and the attributes in 'user attributes' table.

Remove attributes:

pers?clnt=testclient|testpass&com=remattr&attr=sex&attr=age
pers?clnt=testclient|testpass&com=remattr&attr=* (deletes all attributes and user attributes)

INSERTING FEATURES

Add features:

pers?clnt=testclient|testpass&com=addftr&movieID=0

This adds movieId and default value 0 as feature in 'up_features table'.

Add feature values to a user:

pers?clnt=testclient|testpass&com=setusr&usr=user1&ftr_movieID=12

This adds to user1, the feature movieID with value 0 in table 'user_attributes'.

Remove features:

pers?clnt=testclient|testpass&com=remftr&ftr=&ftr=movieID

pers?clnt=testclient|testpass&com=remftr&ftr=* (deletes all up_features and user features)

Get default values for features and attributes:

pers?clnt=testclient|testpass&com=getattrdef&attr=*

Increase the value of a feature for a user:

pers?clnt=testclient|testpass&com=incval&usr=user1&movieID=1

STEREOTYPES

Creating Stereotypes:

ster?clnt=testclient|testpass&com=addstr&str=str18_25GR10male

create a stereotype with attributes: age:18-25, country: Greece. Occupation: 10, sex: male

Adding users to a stereotype:

ster?clnt=testclient|testpass&com=addusr&usr=user1&str18_25GR10male=1

add the user1 to stereotype str18_25GR10male with degree 1

Increase the value of a feature for a stereotype:

ster?clnt=testclient|testpass&com=incval&str=str18_25GR10male=1

Remove a stereotype:

ster?clnt=testclient|testpass&com=remstr&str=str18_25GR10male

List all stereotypes:

ster?clnt=testclient|testpass&com=lststr&str=*

ster?clnt=testclient|testpass&com=lststr&str=str18_25GR10male

COMMUNITIES

In order to create user communities, first you have to create the user distances using one of the two metrics provided below.

Using metric cos (Cosine Vector Metric)

commu?clnt=testclient|testpass&com=calcudist&smetric=cos

Using metric ps (Pearson Correlation Metric)

commu?clnt=testclient|testpass&com=calcudist&smetric=ps

The following will create communities with bk (BronKerbosch) algorithm and a given threshold (>0.5). Threshold values can be set from 0 to 1.

commu?clnt=testclient|testpass&com=mkcom&algorithm=bk&th= >0.5

FEATURE GROUPS

In order to create feature groups, previously you must have created the feature distances using one of the two metrics.

Using metric cos (Cosine Vector Metric)

commu?clnt=testclient|testpass&com= calcftrdist&smetric=cos

Using metric ps (Pearson Correlation Metric)

commu?clnt=testclient|testpass&com= calcftrdist&smetric=ps

The following will create communities with bk (BronKerbosch) algorithm and a given threshold (>0.3). Threshold values can be set from 0 to 1.

commu?clnt=testclient|testpass&com=mkftrgrp&algorithm=bk&th=>0.3

DELETING A PSERVER CLIENT

To delete a client such as the testclient along with the relevant data, you must

perform the following:

1. Run Pserver

2. Login in http://localhost:1111

3. You should see the PServer Administration Panel

4. Go to PServer clients.

5. Click on Delete button next to the client you are interested in.

PSERVER INSTALLATION AND CONFIGURATION

PServer is implemented in Java and uses MySQL as its RDBMS. That means that if you want to install Pserver, first you have to install MySQL version 5+; it is also advisable to install the MySQL Admin Tools that provide a GUI, and JVM version 1.5+. You can download these applications from the sites of MySQL and Sun for free. After that, you should download the latest version of PServer files from http://pserver-project.org/downloads.

1. Start MySQL, and import the database schema. Then, you have to create a user (pserver), and assign full privileges to this user. To do this, start a command prompt, and type the following:

   for windows:

   cd <path to mysql/bin folder>

   e.g.: C:\xampp\mysql\bin

   mysql –u root –p

   for linux :

   sudo mysql –u root –p

2. *create database pserver*
3. *use pserver*
4. import now the database schema from the pserver.sql file that comes with PServer files, type the following:

\. <replace this with the correct path of db file>pserver.sql

e.g.: for windows: \. C:\pserver\pserver.sql

e.g.: for linux: \. /home/name/pserver/pserver.sql

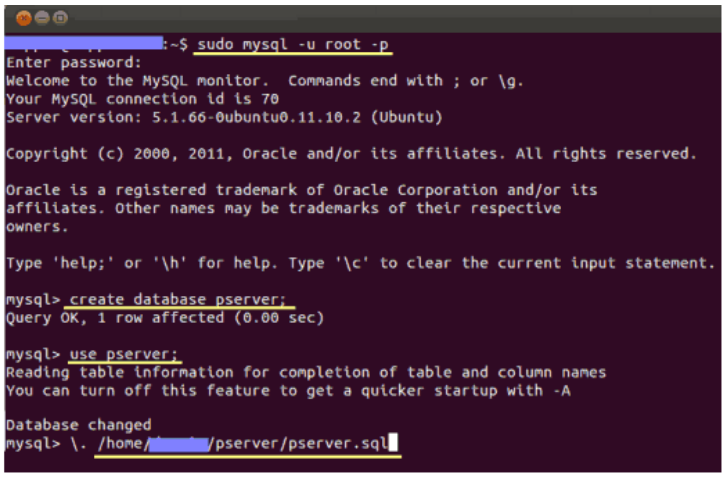

Figure 6. Screenshot from running on a linux machine

5. To add a new user for MySQL with full privileges to the database you created previously, type:

   GRANT ALL PRIVILEGES ON <DB_NAME>.* TO 'my_user'@'localhost'

   IDENTIFIED BY 'my_password' WITH GRANT OPTION;

   e.g.: GRANT ALL PRIVILEGES ON pserver.* TO 'pserver'@'localhost'

   IDENTIFIED BY 'ff0123' WITH GRANT OPTION;

6. Basic PServer Configuration

   We set up variables in the server.ini and pbeans.ini files that exist in the pserver folder, as follows.

   database_user=pserver (no quotes, the user that you created)

   database_url= mysql\://127.0.0.1/pserver (or the host of your mysql)

   database_pass=ff0123 (no quotes, with the password that you entered previously)

RUN PSERVER

To run Pserver, open the terminal, and go inside the pserver folder, and run the PersServer.jar.

For example, in linux you must type these two commands shown below:

1) cd <the path of pserver folder>pserver

2) java -jar PersServer.jar

Now, PServer runs, and is ready for your requests.

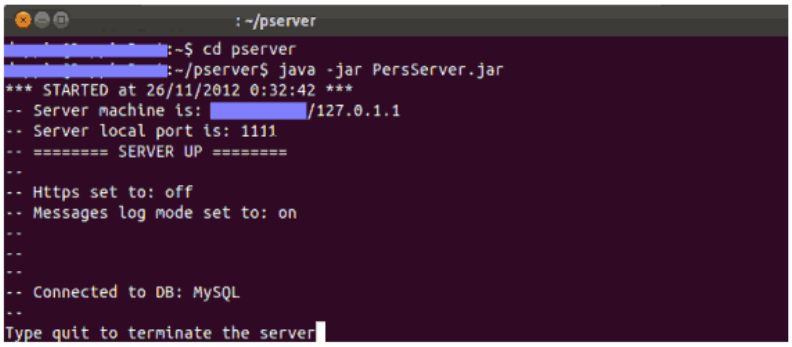

Figure 7. PServer running terminal

When the PServer is up and running, the administration panel helps on creating or deleting PServer clients, change many other settings of Pserver, and use the online help for the requests that we see below. To log in into the administration panel, open a browser, type http:// localhost :1111 and then log in with your username and password you set in the pbeans.ini file.