University of Macedonia
Interdepartmental Program of Postgraduate Studies in
Information Systems

Behavioral User Modeling for Web and Cloud End-User Development Environments

AIKATERINI TZAFILKOU

A thesis submitted for the degree of Doctor of Philosophy

Supervising Professor Nicolaos Protogeros

Preliminary Examiners Anastasios A. Economides, Georgios Evangelidis

Thessaloniki, Greece

November 2018

Statement of Original Authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Ευχαριστίες

Θα ήθελα πρωτίστως να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Πρωτόγερο Νικόλαο. Η υποστήριξη και η καθοδήγησή του καθ' όλη τη διάρκεια της διδακτορικής μου φοίτησης με βοήθησαν να βελτιώσω σημαντικά τις ακαδημαϊκές, ερευνητικές και τεχνικές μου ικανότητες.

Ένα μεγάλο 'ευχαριστώ' οφείλω και στους γονείς μου καθώς και στο σύζυγό μου Γιώργο για την υποστήριξη και την υπομονή τους όλα αυτά τα χρόνια.

Περίληψη

Η παρούσα διδακτορική διατριβή καλείται να αντιμετωπίσει τις σύγχρονες προκλήσεις μοντελοποίησης τελικών χρηστών σε περιβάλλοντα ανάπτυξης εφαρμογών, βασισμένα στον ιστό και στην υπολογιστική νέφους. Η κατεύθυνση που ακολουθεί είναι η καταγραφή της συμπεριφοράς του χρήστη κατά την αλληλεπίδρασή του με τα εν λόγω συστήματα και η συγκέντρωση ανθρώπινων χαρακτηριστικών.

Αρχικά, σχεδιάζεται μια καινοτόμα προσέγγιση ανάπτυξης εφαρμογών βάσεων δεδομένων με στόχο την υποβοήθηση των τελικών χρηστών στην κατασκευή εφαρμογών επικεντρωμένων στο σχεσιακό μοντέλο βάσεων δεδομένων. Η προτεινόμενη τεχνική ονομάζεται Simple-Talking και αποκρύπτει κάθε τεχνική ορολογία και την αντικαθιστά με έννοιες πιο οικείες προς στον τελικό χρήστη. Η τεχνική Simple-Talking ενσωματώνεται σε ένα πρωτότυπο εργαλείο βασισμένο σε αρχιτεκτονική υπολογιστικής νέφους και αξιολογείται μέσω πειραματικής διαδικασίας. Στα αποτελέσματα σημειώνεται υψηλή απόδοση χρηστών χωρίς αυτή να επηρεάζεται από το βαθμό εμπειρίας τους στη σχεδίαση βάσεων δεδομένων. Επιρροή στην απόδοση φαίνεται να ασκεί το φύλο, καθώς και η αντίληψη ευκολίας χρήσης και η αντίληψη χρησιμότητας.

Στη συνέχεια δομείται το μοντέλο RULES, ένα νέο μοντέλο χρήστη αποτελούμενο από βασικές ιδιότητες φύλου που τείνουν να επηρεάζουν τη συμπεριφορά και την απόδοση των τελικών χρηστών. Το RULES περιλαμβάνει τα πέντε πιο σημαντικά χαρακτηριστικά των χρηστών όπως: αντίληψη κινδύνου, αυτοπεποίθηση, θέληση για μάθηση κλπ. τα οποία προέκυψαν μετά από εκτεταμένη βιβλιογραφική ανασκόπηση δημοφιλών θεωριών της Αλληλεπίδρασης Ανθρώπου-Υπολογιστή.

Για τη μοντελοποίηση του χρήστη με βάση το μοντέλο RULES αναπτύσσεται μία ακόμη ερευνητική κατεύθυνση, η ‘σιωπηρή’ παρακολούθηση και καταγραφή της συμπεριφοράς των χρηστών. Για να επιτευχθεί, εφαρμόζονται μέθοδοι καταγραφής των κινήσεων του ποντικιού, του πληκτρολογίου και του ματιού, κατά την αλληλεπίδραση των χρηστών με συστήματα ανάπτυξης εφαρμογών. Η καταγραφή των κινήσεων του ποντικού και του πληκτρολογίου περιλαμβάνει την ταξινόμηση ενός

πλήθους μετρήσιμων δεικτών γεγονότων ποντικιού και πληκτρολογίου, που μπορούν να ανιχνευτούν μέσω ειδικού λογισμικού, στα προγράμματα περιήγησης, σε πραγματικό χρόνο και την αποθήκευση σε μια βάση δεδομένων. Η μεθοδολογία αξιολογείται πειραματικά, και από τη στατιστική ανάλυση των δεδομένων προκύπτει ένα πλήθος σημαντικών συσχετίσεων ανάμεσα στις μετρικές ποντικιού/ πληκτρολογίου και στις ιδιότητες του μοντέλου RULES.

Παρόμοια, η μεθοδολογία καταγραφής των κινήσεων του ματιού περιλαμβάνει την ταξινόμηση δεικτών οφθαλμικής συμπεριφοράς όπως η εστίαση και η διάμετρος της κόρης. Για την πειραματική αξιολόγηση χρησιμοποιείται το εργαλείο οφθαλμικής καταγραφής Tobii Eye Tracker σε ένα δείγμα εθελοντών. Κατά την ανάλυση δεδομένων ανιχνεύονται σημαντικές συσχετίσεις ανάμεσα στους δείκτες οφθαλμικής συμπεριφοράς και στις ιδιότητες του μοντέλου RULES. Τα ευρήματα των ανωτέρω μελετών για το φύλο και τα συμπεριφορικά δεδομένα, φαίνεται ότι αποτελούν χρήσιμη πληροφορία για τη μοντελοποίηση του χρήστη στο πλαίσιο των συστημάτων ανάπτυξης εφαρμογών από τελικούς χρήστες.

Τελικά, το προτεινόμενο μοντέλο χρήστη αναπαρίσταται σε μορφή οντολογίας μέσω της χρήση τεχνολογιών Σημασιολογικού Ιστού. Με τη χρήση σημασιολογικών κανόνων παρουσιάζεται μια μεθοδολογία αξιοποίησης των καταγεγραμμένων δεδομένων με στόχο την κατάταξη του χρήστη σε ένα αντιπροσωπευτικό μοντέλο, αποδίδοντας τιμές στο επίπεδο γνώσης του και τις αντιληπτικές του ιδιότητες. Η προτεινόμενη οντολογική προσέγγιση εξασφαλίζει τη μοντελοποίηση των χρηστών όταν η αρχική γνώση είναι μικρή και νέες συμπεριφορές μπορούν να παρουσιαστούν με την πάροδο του χρόνου.

Abstract

This thesis addresses the challenges of user modeling in cloud and web based End-User Development (EUD) environments by monitoring users' behavior and taking into account human characteristics.

Initially, an innovative development approach is designed to assist end-users in building database-centric applications following the relational scheme. The proposed approach is called *Simple-Talking* since all technical terminology is replaced by simple words and concepts more familiar to the end-user. The *Simple-Talking* approach is integrated into a prototype cloud-based EUD tool and is evaluated via an experimental study. The results reveal high scores of user performance without being affected by the user's expertise level. Impact on performance seems to be exercised by gender, as well as perceived ease of use and perceived usefulness.

Secondly, the RULES model is structured. RULES is a new user model consisting of basic gender properties that tend to affect the end-users' behavior and performance. RULES model includes the five most important end-user features such as: risk perception, self-efficacy, willingness to learn, etc., that resulted from an extensive literature review of popular Human Computer Interaction theories.

To model the user based on the RULES properties, a research is developed to implicitly monitor the end-users behavior. For this, methods to monitor mouse, keyboard and eye movements are applied during the user-system interaction. Mouse and keyboard tracking methodology includes the taxonomy of measurable items like mouse events and keystroke dynamics that can be detected in real time on the client-side and stored in a database. The methodology is experimentally evaluated, and the results of the statistical analysis reveal a number of significant correlations between the mouse/ keyboard metrics and the properties of the RULES model.

Similarly, the methodology for recording eye movements includes the classification of eye-metrics such as fixation and pupil diameter. The eye tracking approach is experimentally evaluated on a sample of volunteers using the Tobii Eye Tracker

software and hardware tool. Significant correlations between eye behavioral data and properties of the RULES model are detected during the data analysis.

According to the thesis findings, gender and behavioral metrics (as derived from mouse/keyboard and eye monitoring research) are key EUD attributes to define end-user developer models.

The proposed user model is semantically represented via an ontology-based framework. The model is further exploited to define an inference mechanism by which the collected data can be used in diagnosing users' domain knowledge level, perceptional and acceptance characteristics. This approach ensures that user modeling is possible when initial knowledge is small and new behaviors are encountered over time.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Ubiquitous systems and cloud computing have paved the way for developing smart adaptive systems that can improve the quality of services and the user experience, drawing on information and knowledge of user behavior within his environment. This process can be achieved by incorporating user modeling mechanisms. User modeling involves an inference process from observable data such as user actions, into unobservable information such as user preference, intentions, cognitive and psychological states, etc. Within end-user development environments (EUD) users are end-user programmers and the main unobservable implied information relates to domain knowledge and behavioral attributes such as perception and acceptance. Also, human factors such as gender appear to be important variables in the behavior and performance of end-user developers.

This thesis proposes a user modeling framework that takes into account human factors and end-user behavior using devices such as mouse, keyboard, and eye movement tracker to create a behavioral user profile. Users can also explicitly provide their domain knowledge level in specific areas related to the EUD and gender information so as to build a comprehensive end-user developer model. Based on this model, the framework could provide better quality services, adapted to the users' distinct needs. The benefits of such a framework are the potential improvement of user experience during their interactions with web-based and cloud-based EUD systems and the improvement of user performance, allowing them to develop or customize efficient and reusable software.

The rest of this chapter is structured as follows. The next section presents the motivation and vision of the current study. Section 1.2 lists the research objectives of this thesis and presents the overall concept of the suggested framework while Section 1.3 outlines the structure of the thesis. Finally, section 1.4 lists the thesis related publications in international peer-review journals and conferences.

1.1 Motivation and Vision

'Get to know the user first'

Web personalization and adaptation services can provide the end-users with a highly optimized user experience and assist them in enhancing their task performance. Human factors like gender, domain knowledge and real time user behavior like mouse and eye movements play a significant role in defining the active user's profile. Various user modeling approaches have been designed to construct users' appropriate profiles and personalize their feedback both in content and in interface design. Although user models have been broadly implemented in intelligent tutoring environments and in recommender systems, there are not any modeling approaches in the field of End-User Development.

EUD is a Human-Computer-Interaction (HCI) field targeted at the population of the end-user developers. End-user developers are non-professional and not experienced programmers who attempt to customize or construct their own applications via EUD tools. Today, the end-user developers' population is getting larger, significantly outnumbering the population of professional developers.

Despite the recent evolution in EUD tools from desktop to web and mobile environments and the adaptation of software engineering principles, end-user developers still face multiple difficulties to produce reusable software, comparable to professional one. One of the main reasons is that end-users' performance is affected by various factors that most software tools tend to neglect. Contrary to professional developers, end-user developers are more 'vulnerable' to human factors. For instance, there are strong gender differences among end-user developers; these differences tend to significantly shrink among professional developers. Also, the increased population of end-user developers renders the mental differences among them even larger than the differences between professional developers.

Although its huge applicability in business, education and every day activities, EUD research has not designed any user modeling approaches to optimize end-users' experience and enhance their performance. To this end, we design and propose a user

modeling framework for the end-user developers' population aiming to dynamically construct user models that take into account gender-oriented and EUD behavioral characteristics, as well as domain knowledge level on EUD related concepts, such as database and web development. Furthermore, the lack of real-time monitoring mechanisms for user modeling both in EUD and other HCI areas, triggered our interest to analyze the end-users' mouse, keyboard and eye behavior and include it as behavioral feedback in the EUD model. To this end, gender, domain knowledge, EUD/HCI behavioral attributes and real time (mouse/ keyboard/ eye) behavior compose the main parts of the proposed user modeling framework.

1.2 Research Objectives and Overall Concept

This thesis is focused on user modeling in web and cloud EUD environments. The aim is to design a behavioral user modeling framework for end-user developers during their interaction with EUD systems. The proposed model takes into consideration gender, domain knowledge, and real time user behavior using devices such as mouse, keyboard and eye tracker. These monitoring methodologies are used and evaluated as implicit feedback gathering mechanisms in EUD environments. Finally, semantic web technologies are used to represent the final ontology-based user model. The information stored in the user ontology could be used for various purposes, such as personalized feedback and system's self-adaptation.

To link fragments to a framework, a set of research objectives need to be considered. Each research objective refers to a separate research part combining EUD and another technological or HCI research field as summarized in the following list:

- Cloud Computing;
- Gender in HCI;
- Eye Tracking;
- Mouse Tracking;
- User Modeling and Semantic Web.

We then describe the research objectives for each one of the research parts mentioned above. The outcomes of each objective will be combined to design and construct the final user model.

Cloud Computing: There is no significant progress towards Cloud-based EUD environments, although EUD has considerably evolved from desktop software to web, mobile and the Internet of Things (IoT) applications. In this thesis we define Cloud EUD as a EUD subfield encompassing aspects of traditional and web EUD, programming psychology principles and Cloud Computing technologies.

In addition, EUD research literature has not shown significant contributions to 'end-user-friendly' approaches for the developing of database-centric web and mobile applications. Database-centric development environments tend to require end-users to have at least some basic knowledge of SQL syntax and database logic. On the other hand, empirical research in EUD shows that end-users are unfamiliar with database logic since it seems not to be close to their natural way of thinking. As a result, end-user developers have difficulty building and using their own database-centric applications. To this end, our first objectives are formed as following:

> ***Objective 1.1:*** *To design a natural language EUD approach to assist end-user developers in constructing database centric applications.*
>
> ***Objective 1.2:*** *To design and develop a prototype Cloud EUD tool to integrate the above approach (of Objective 1.1) and also be used in the Thesis experiments.*

These two objectives are addressed in chapter 2 where we present the main components of Cloud EUD and a natural language EUD approach to assist end-users in developing database-centric web/mobile applications. We integrate the proposed approach in a cloud-based EUD tool and we evaluate its usefulness and ease of use through an experimental analysis. The developed tool is also used as a prototype tool for the experimental tests in chapters 3 and 4.

Gender in HCI: Gender is an important human factor in HCI research. Multiple HCI theories step on gender differences to analyze end-user differences in perception, decision making, debugging strategies, learning preferences, etc. Among these

theories are the popular theories of Self-Efficacy, Technology Acceptance, Attention Investment, etc. The most recent emerging field of Gender HCI addresses gender issues in EUD environments and attempts to reduce the gender and performance gap among end-user developers.

Due to the important role that gender plays in EUD behavior and performance and the lack of gender-oriented EUD user model representations, we set the following objective:

> ***Objective 2:*** *To suggest a user-model structure composed of gender-oriented EUD behavioral attributes.*

This objective is addressed in chapter 3 where the "RULES" attributes are proposed as a component of EUD behavioral user models. RULES attributes include risk-perception, self-efficacy, perceived usefulness, ease of use and willingness to learn. The proposed model is evaluated through the experimental tests presented in the same chapter. RULES attributes will be used in the final model composition as the part of the users' 'behavioral traits' component.

Eye Tracking: Eye tracking data is a reliable source of implicit user feedback during user-system interaction since it infers aspects of users' cognitive and mental states. Despite its important HCI contribution, eye tracking has not been widely integrated in the behavioral research area of EUD. For this reason, our next objective is the following:

> ***Objective 3:*** *To examine the role of eye behavior in EUD tasks and its association with EUD behavioral attributes.*

This objective is addressed in chapter 4 where we conduct an eye tracking experiment using the prototype EUD tool presented in chapter 2. The experimental results reveal the usefulness of including eye metrics like fixations and pupil size in the collected user feedback in EUD activities. These eye metrics are used in the behavioral implicit input components in the proposed modeling framework.

Mouse Tracking: Mouse tracking methodologies are ideal to implicitly gather behavioral user feedback in real time during user-system interaction. Mouse data can reflect specific user behavior patterns regardless of the task and the environment, and can be used to model users' behavior. In addition keystroke dynamics have also been used in HCI research to detect user cognitive and affective states, like mood and emotions. Although techniques for studying mouse movements and keystroke dynamics have been considered to be an effective means of HCI, they have not yet integrated into common daily cloud and/or web-based activities such as End User Development. Hence, our following objective is:

> ***Objective 4:*** *To examine the role of mouse and keystroke behavior in EUD tasks and their association with EUD behavioral attributes.*

To meet the requirements of the experimental study, we need the appropriate prototype tools. These tools are a real-time mouse and keystroke monitoring tool and a prototype EUD tool for conducting the experiment. The reason we do not use the tool of chapter 2 is because of its wizard-like design which is not ideal for gathering sufficient amount of mouse actions. Therefore, we need to design a web or cloud EUD tool using a different interface style (e.g. single-page design) to provide users with multiple clickable options (e.g. buttons, links) and typing form fields. Hence, our sub-objectives for the mouse and keystroke monitoring experiment are as follows:

> ***Objective 5.1*** *Develop a prototype mouse and keystroke monitoring tool to be used in the experiment.*

> ***Objective 5.2*** *Develop a prototype EUD tool (suitable for tracking mouse behavior) to be used in the experiment.*

These objectives are all discussed in chapter 5 where we conduct a mouse and keystroke experiment using a prototype monitoring mechanism and a web EUD tool for constructing database entities through forms construction. The experimental results reveal significant correlations between mouse behavioral metrics and end-user developers' behavioral attributes. Hence, these metrics are also included, along with eye metrics, in the implicit input components in the proposed modeling framework.

User Modeling and Semantic Web: Ontology-based user modeling approaches can be used for various purposes such as personalization and adaptation services. They have been mainly used to model context knowledge and one of their main benefits is that they support reasoning to infer new knowledge on user characteristics. Semantic modeling approaches like OWL-based ontological models, address the problems of heterogeneity and aggregation, rendering user models reusable in the Semantic Web. Based on these, our final objective is the following:

> ***Objective 6:*** *Semantically build and represent an ontological user modeling framework, based on the findings and conclusions of gender, eye and mouse movements.*

This objective is addressed in chapter 6 where we design and build an OWL-based ontological user modeling framework for the end-user developers. A semantically expressed rule-based reasoning is integrated in the ontology to infer knowledge on users' behavioral traits and domain knowledge level. The proposed ontology is constructed based on the research findings of the previous chapters. The efficiency of the representation is evaluated through a use case scenario which runs on Protégé OWL environment.

Figure 1.1 summarizes the overall concept of the suggested framework.

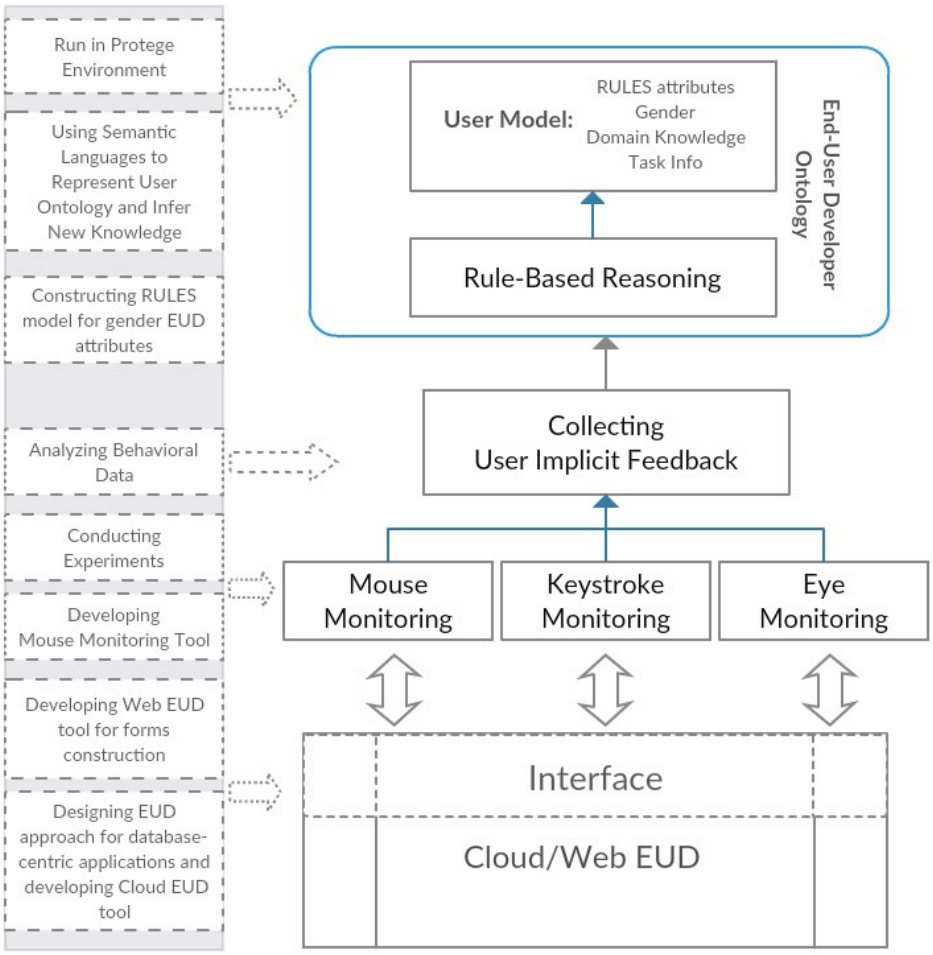

Figure 1.1 Thesis Overall Concept and Research Components

As depicted in Figure 1.1 in the left rectangle, the thesis research elements include the main parts of the research objectives described previously.

At the top of the right section we see the main thesis outcome which is the End-User Developer Ontology representation, including a rule-based reasoning mechanism to infer knowledge on the user model dimensions. The final user model is composed of the gender-oriented behavioral RULES attributes, the users' gender, their domain knowledge level and information of their EUD task. The user-system interaction takes place in Cloud and web EUD environments where mouse, keyboard and eye movements are tracked in real time to implicitly collect behavioral feedback. This feedback along with explicitly provided feedback (regarding gender and domain knowledge) and EUD task related information is stored and used by the ontology to define the final user model.

1.3 Structure of Thesis

The thesis contains literature reviews and research that could be categorized as parts of four main strands of work: the design and development of a cloud-based EUD approach for database-centric applications (Chapter 2), the analysis of gender in EUD activities and the proposal of a gender-oriented user model structure (Chapter 3), the experimental examination of eye and mouse tracking methodologies for implicit feedback gathering in web and cloud EUD environments (Chapter 4 and 5), and the semantic representation of an ontological end-user developer model based on explicit and implicit user feedback (Chapter 6).

Chapter 2 presents the literature review on existing EUD approaches paving the way towards Cloud EUD systems. It also proposes a natural-language EUD approach, named 'simple-talking', for database-centric applications and integrates it in a Cloud EUD approach. The innovative contribution of the proposed approach is that it complies with the end-user developers' natural way of thinking to assist them in designing relational database schemes for their applications. The aim of this chapter is to outline the human side of EUD by describing the psychological aspects of end-user developers and their mental differences with professional developers. This chapter also aims to inform on the benefits that cloud computing technologies, like Software as a Service and cloud manufacturing systems, can bring to the end-user developers and the EUD community in general. The human-side aspect of EUD outlined in this chapter triggers the interest on the analysis of gender that is conducted in chapter 3.

Chapter 3 presents the main HCI gender-related theories outlining the role of gender in EUD. The aim of this chapter is to propose a set of gender-oriented behavioral attributes, named RULES, for EUD environments. RULES include a set of perception and acceptance items that will be used as the dependent variables in the experimental studies of eye and mouse tracking in chapters 4 and 5. RULES will also compose the Behavioral Traits component of the proposed EUD user model presented in chapter 6.

Chapters 4 and 5 present the experimental evaluation of eye and mouse tracking approaches in web and cloud EUD environments. These chapters aim to show the contribution of eye and mouse (including keystroke) monitoring methodologies in diagnosing end-user developers' behavioral states. Eye, mouse and keystroke actions will compose the implicit feedback component that is used in the modeling framework to infer knowledge on users' RULES attributes' values, presented in chapter 6.

Chapter 6 describes the overall EUD modeling framework taking into account the findings of the previous chapters. This chapter aims to define a new framework to ontologically represent end-user developer models and infer knowledge on behavioral traits and domain knowledge properties by implementing semantic rule based reasoning approaches. The constructed user ontology can be used to store implicit and explicit user input and use this input to dynamically assign users in a behavioral model. The proposed model's dimensions include user characteristics as gender, EUD task information, user domain knowledge and behavioral traits. The generated model presented in this chapter, can be used by adaptation agents to provide personalized and adapted services.

Finally, Chapter 7 concludes this thesis with a summary of research and findings, and an outline of the thesis contribution. It also identifies directions for future work and ways in which they could be addressed.

1.4 Related Publications

International Journal Publications

1 Katerina Tzafilkou, Nicolaos Protogeros: *Mouse Behavioral Patterns and Keystroke Dynamics in End-User Development: what can they tell us about users' behavioral attributes?.* Computers in Human Behavior 02/2018;, DOI:10.1016/j.chb.2018.02.012

2 Katerina Tzafilkou, Nicolaos Protogeros: *Examining gender issues in perception and acceptance in web-based end-user development activities*. Education and Information Technologies 10/2017;, DOI:10.1007/s10639-017-9650-x

3 Katerina Tzafilkou, Nicolaos Protogeros: *Diagnosing User Perception and Acceptance using Eye Tracking in Web-based End-User Development*. Computers in Human Behavior 07/2017; 72., DOI:10.1016/j.chb.2017.02.035

4 Katerina Tzafilkou, Nicolaos Protogeros, Charalampos Karagiannidis, Adamantios Koumpis: *Gender-based behavioral analysis for end-user development and the 'RULES' attributes*. Education and Information Technologies 07/2016;, DOI:10.1007/s10639-016-9519-4

5 Protogeros Nicolaos, Tzafilkou Katerina: *Simple-talking database development: Let the end-user design a relational schema by using simple words.* Computers in Human Behavior 07/2015; 48., DOI:10.1016/j.chb.2015.02.002

6 Katerina Tzafilkou, Nicolaos Protogeros, Adamantios Koumpis: *User-centred cloud service adaptation: an adaptation framework for cloud services to enhance user experience*. International Journal of Computer Integrated Manufacturing 04/2015;, DOI:10.1080/0951192X.2015.1030697

7 Katerina Tzafilkou, Nicolaos Protogeros, Charalampos Yakinthos: *End-User Development of CRM Systems: Towards a Behavioral End- User Profiling based on Gender and Expertise*. International Journal of Electronic Business 01/2015; 12(3)., DOI:10.1504/IJEB.2015.071390

8 Katerina Tzafilkou, Nicolaos Protogeros, Charalampos Yakinthos: *Human Factors in End-user Development of Marketing-IS: A Behavioral User Profiling Approach*. Procedia - Social and Behavioral Sciences 08/2014; 148:245–253., DOI:10.1016/j.sbspro.2014.07.040

9 Katerina Tzafilkou, Nicolaos Protogeros, Charalampos Yakinthos: *Mouse Tracking for Web Marketing: Enhancing User Experience in Web Application Software by Measuring Self-Efficacy and Hesitation Levels*. DOI:10.15556/IJSIM.01.04.005

International Conference Proceedings

10 Katerina Tzafilkou, Nicolaos Protogeros, Adamantios Koumpis: *Eye movements and end-user performance in web development tools*. Sixth International Conference on Virtual and Networked Organizations Emergent Technologies and Tools, ViNOrg'17, Póvoa de Varzim, Portugal; 11/2017

11 Katerina Tzafilkou, Chouliara Adamantia, Protogeros Nicolaos, Charalampos Karagiannidis, Koumpis Adamantios: *Engaging end-users in creating data-intensive mobile applications: A creative 'e-learning-by-doing' approach*. International Conference on Interactive Mobile Communication Technologies and Learning (IMCL). IEEE, Thessaloniki, Greece; 11/2015, DOI:10.1109/IMCTL.2015.7359602

12 Katerina Tzafilkou, Protogeros Nicolaos, Ginoglou Dimitrios: *Accounting in the Cloud: Cloud-based Accounting Information Systems for Small and Medium sized Enterprises*. International Conference On Accounting And Finance, Syros Island,Greece; 09/2014

13 Katerina Tzafilkou, Protogeros Nicolaos, Yakinthos Charalampos: *End-user development of CRM systems: towards a behavioral end-user profiling.* International Conference on Strategic Innovative Marketing (ICSIM). Elsevier, Inderscience, Prague, Czech Republic; 09/2013

CHAPTER 2

TOWARDS A CLOUD END-USER DEVELOPMENT FRAMEWORK FOR DATABASE-CENTRIC APPLICATIONS

The focus of this chapter is on the End-User Development (EUD) principles, the main end-user programming psychology aspects and the EUD evolution from desktop to web, mobile and the Internet of Things approaches. Because of the limited implementations of Cloud based EUD approaches we also present the benefits that Cloud brings to end-users and we develop a prototype cloud-based EUD framework for database centric applications. We name the developing approach 'Simple-Talking' since it uses natural language to assist end-user developers in unconsciously designing a relational scheme for their application. The Cloud prototype tool is evaluated and used in behavioral experiments presented in next chapters. The main contributions of this chapter have been published in Protogeros and Tzafilkou (2015), Tzafilkou et al. (2015a) and in Tzafilkou et al. (2015b).

2.1 Introduction

End-user-development (EUD) research has emerged to support end-user development of web automations, mobile devices, personal information management systems, business processes, programming home appliance devices, etc. EUD encompasses many aspects of the software development lifecycle, supporting not only creating new programs and testing/debugging them, but also designing, specifying, and reusing them (Ko et al., 2011). Recently it has also been recognized that EUD involves much more than providing an end-user with the right tools. EUD has its place in a socio-technical environment where issues such as learning motivation and organization are important determinants of its success (De Suza et al., 2011; Tetteroo et al., 2014). Also, end-users' psychological aspects seem to be of great value and interest since the increased number of non-developers (who are more than professional developers) render the mental differences among end-user

developers ‘even larger than the difference between (ordinary) developers and end-user developers’ (Blackwell, 2017).

Since its early beginning EUD was mainly focused on desktop applications and spreadsheet like environments. Today web based EUD for web applications has emerged to provide end-users with tools and platforms to efficiently construct their own, personal and business related web or mobile applications. Rhode (2006) had defined this kind of web-based EUD as EUDweb arguing for its compositing of three HCI sub domains: Psychology of Programming, End-User Development and Web Engineering.

Recent trends like maturation of cloud computing, mass customization and the Internet of Things (IoT) resulted in even higher demand for flexible, feature-rich and extensible platforms for end-user development. As a fact, the recently emerged field of EUD for the IoT has paved new approaches for the end-users to build and customize smart and interactive solutions.

Despite though this emerging need for interconnectivity and ubiquitous access, Cloud EUD solutions have not flourished yet. Although plenty platforms for Cloud software development and Cloud programming environments are now available in the market, we have not yet seen Cloud-based EUD tools. Cloud EUD could facilitate end-user developers in the construction of comprehensive and of high quality software applications. EUD can benefit from cloud technology in terms of ease of use, big data storage, ubiquitous access, on demand usage, and other user-oriented services.

Following Rhode’s (2006) EUDweb composition, we define Cloud EUD as composed of the pillars of Cloud Computing, End-User Development (including all its subfields such as EUDweb, mobile EUD, EUD for the Internet of Things) and Psychology of Programming (see Figure 2.1).

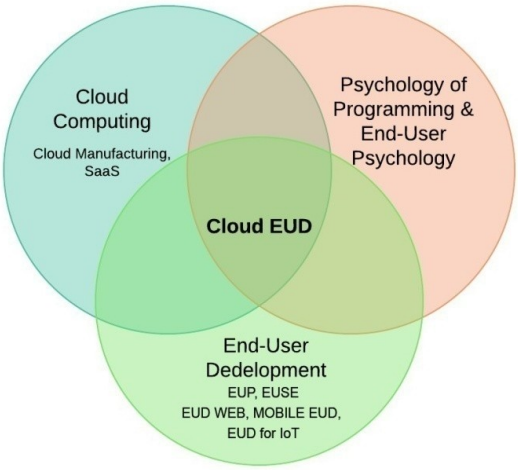

Figure 2. 1 Cloud EUD considered as the cross-section of EUD, cloud computing, psychology of programming, and end-user psychology.

2.2 Literature Background

Composing the stepping stone for the EUD evolution, psychological aspects of programming have been deeply studied since 1970's. In this section we present the psychological aspects of programming ('Psychology of Programming') as have been analyzed in the literature and its integration in EUD research.

2.2.1 Psychology of Programming

The origins of Psychology of Programming (PoP) date back to 1970s, at least to the first version of Weinberg's book 'The Psychology of Computer Programming' (1971). In the mid 1980's, it was observed that programming language research and funding emphasized technical aspects of the domain and neglected psychological ones: "Millions for compilers, but hardly a penny for understanding human programming language use. Of course there is lots of programming language design, but it comes from computer scientists. And though technical papers on languages contain much appeal to ease of use and learning, they patently contain almost no psychological evidence nor any appeal to psychological science. According to Newel and Card (1985), "the human and computer parts of programming languages have been developed in radical asymmetry".

During these past thirty years, researchers have realized that programming tools and technologies should not be evaluated based only on their computational power, but also on their usability from the human point of view, regarding their cognitive effects as well. Such an approach was hopefully expected to render programmers make fewer errors, produce better software, and work more efficiently (Sajaniemi, 2008).

As Sajaniemi (2008) declares, the PoP research community consists of both cognitive psychologists and computer scientists. The main motivation for computer scientists is the improvement of current tools and the development of new ones, as well as the discovery of general principles concerning humans in the context of programming tasks. Psychologists are interested in new theories of human cognition applicable in other domains too. It is obvious that the field demands profound knowledge of cognitive psychology, social sciences, and software engineering.

Dealing with psychological aspects of programming, PoP entertains a broad spectrum of research approaches, from theoretical perspectives drawing on psychological theory to empirical perspectives grounded in real-world software engineering experience, such as improving the usability of programming languages and environments by better anticipating human needs. In this respect, Blackwell (2006) underlines that PoP can be considered as a specialized field within HCI.

It is widely admissible that psychologically, computer programming is a human activity which involves cognitions, such as reading and writing computer language, learning, problem solving, and reasoning. In addition, it is commonly accepted that personality traits affect programming performance even more that intelligence (Weinberg, 1998).

End-user programming and end-user development have always been a topic of great interest in Psychology of Programming. According to Blackwell and Hague (2001), end-user developers offer the prospect to study programming behavior in its ‘natural’ state. Based on the argument that some programming paradigms are more ‘natural’ than others, and hence more appropriate for inexperienced programmers, Blackwell (1996) concludes that end-user developers are useful object of study as ‘natural’ programmers.

Rhode (2005) declares that in its essence end-user development (EUD) or end-user programming is equivalent to “the psychology of programming for nonprogrammers”.

Recently, Blackwell (2017) supports that since there are so many more non developers in the world than professional developers; it seems likely that the differences among end-user developers may be even larger than the difference between (ordinary) developers and end-user developers.

2.2.2 End-User Development

According to Rhode (2006), “*In its essence end-user development or end-user programming is equivalent to the psychology of programming for nonprogrammers*”. Based on the main psychological theories of end-user programmers, End-User Development (EUD) has recently evolved from traditional desktop programming tools (like Spreadsheets) to modern user-centered web-based environments rendering the end-user developers capable of building usable and efficient web and/or mobile applications. In this section we state the definitions and origins of EUD presenting also its main components and research directions it has followed till today.

An ‘end-user developer’ has not received professional training in, is not paid for doing development, and would not describe development as being their profession Blackwell (2017). What characterizes end-user developers is that they express a need to modify on their own the computer systems they use and to gain more control over their computer applications (Lieberman et al., 2006; Repenning and Ioannidou, 2006). The reasons they choose to do this include the following: ‘because they like it, because it is immediately useful, or because they believe they will be good at it’ (Blackwell, 2017).

In other words, people who are not professional developers can use EUD environments to create or modify software artifacts and complex data objects without significant knowledge of a programming language. Because these artifacts help end-users to automate certain activities, end-user programs are extremely valuable to the people who create them, despite the fact that these programs are often smaller and less complex than those created by professional programmers (Scaffidi, 2009).

Accoding to Spahn et al. (2008), the term of End User Development has evolved over time. Inspired by the Pygmalion system of David Smith (Smith, 1975) which introduced the computer icons and Programming by Example approaches, the research field of End User Programming (EUP) emerged in the end of the 1970s (Cypher, 1993). During the 1980s, the term "end user computing" (EUC) became popular (Brancheau and Brown, 1993) and later on the term "end user tailoring" (EUT) emerged. EUT refers to the change of stable aspects of an artifact (Henderson and Kyng, 1991).

> End-User Development is a set of methods, techniques and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artifact (Lieberman et al., 2006).

EUD research has recently made significant progress. In the beginning it was mainly focusing on supporting end users' software development using spreadsheets and event-based computing paradigms. Today, EUD research has emerged to support end-user development of web automations, mobile devices, personal information management systems, business processes, programming home appliance devices, and even internet-of-things programming (Burnett et al., 2017). Further, EUD research now encompasses more aspects of the software development lifecycle, supporting not only creating new programs and testing/debugging them, but also designing, specifying, and reusing them (Ko et al., 2011).

Also, EUD's continuing development as a social phenomenon has important implications for the relationship between end-user developers and professional software developers (Fischer and Giaccardi 2006; Costabile et al., 2006). The rise of EUD to date enables end-users to respond to the reality that professional software developers are not likely to understand and plan for every user requirement when developing software (Burnett and Scaffidi, 2011).

Burnett and Scaffidi (2011) support that end users know their own context and needs better than anybody else, and they often have real-time awareness of shifts in their respective domains. Through EUD end users can tune software to fit their requirements more closely than would be possible without EUD. And this is because

end users significantly outnumber professional developers and hence enables a much larger pool of people to participate in software development.

The “programming or development environments” used by end users include spreadsheet systems, web authoring tools, and graphical languages for creating educational simulations (e.g. Burnett et al., 2001; Lieberaman, 2001; Little et al., 2007; Pane et al., 2006; Repenning, and Ioannidou, 2006). Using these systems, end users create programs in forms such as spreadsheets, dynamic web applications, educational simulations, etc. Some ways in which end users create these programs include writing and editing formulas, dragging and dropping objects onto a logical workspace, connecting objects in a diagram, or demonstrating intended logic to the system (Burnett, 2009).

In EUD the software itself is not the goal but rather a tool to solve a situational problem (Nardi, 1993). According to Cao et al. (2010) end users prefer to build software in the try-and-error fashion instead of following a strict plan. While end users put value on correctness and reliability of created solutions, they are unwilling to invest time into systematic testing and debugging. Additionally, end users seem to be overconfident regarding the correctness of their solutions (Panko, 2008). Hence, the responsibility of error detection and consistency checks is therefore put on development environments and assistance mechanisms.

The central EUD problem is finding good tradeoff between ease-of-use and power (Rode, 2008). Hence, the EUD’s main goal is to decrease the conflict between application complexity and learning effort (Myers et al., 2000) and to provide environments that allow for a gentle learning curve. Although complex programming languages can address a wide range of problems they tend to increase the learning burden on the user. On the other hand, easy-to-learn languages are often domain specific and limit the development possibilities (Paternò, 2013). Designers want to build EUD systems that are as self-evident and as easy to use as possible, so that users with little or no programming experience will be able to use them. But on the other hand, they want the systems to be powerful, supporting traditional constructs of programming languages such as abstraction, modularity, and reuse (Rode, 2008).

Figure 2.2 below depicts the main subsets and technological approaches in EUD till today. As we see EUD evolution encompasses End-User Software Engineering (EUSE), End-User Programming (EUP), web-based EUD and mobile EUD. Its area is based on a set of EUD principles and is oriented towards specific directions.

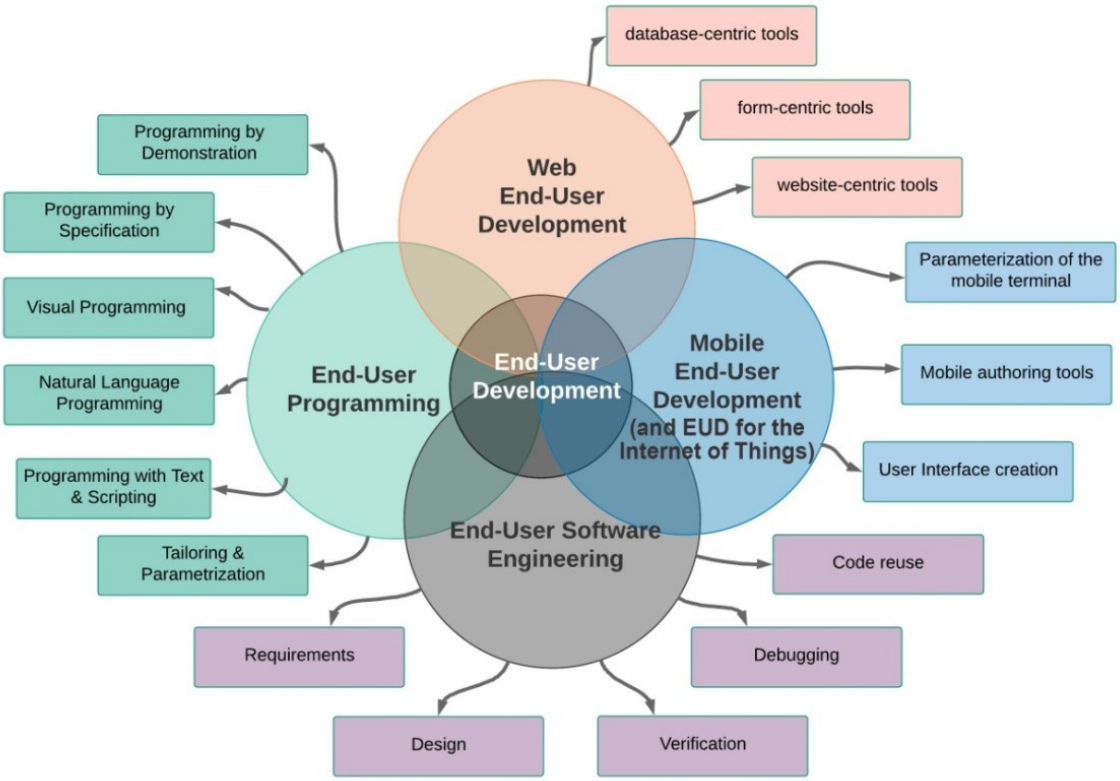

Figure 2. 2 EUD subsets and research directions so far

In the following sections we make a basic review of the depicted EUD subsets and their research progress over time.

2.2.3 End-User Programming and End-User Software Engineering

EUD also includes two main research subsets: End-user programming (EUP) and end-user software engineering (EUSE). End-user programming is the most mature from a research and practice perspective (Burnett and Scaffidi, 2011).

The term of 'end-user programming' was popularized by Nardi (1993) in her investigations into spreadsheet use in office workplaces. Ko et al. (2011) define 'end-user programming' as "programming to achieve the result of a program primarily for

personal, rather public use." In general, EUP enables end users to create their own programs (Ko et al., 2011).

The important distinction here is that the program itself is not primarily intended for use by a large number of users with varying needs but from end-users who need the program for their own domain and to accomplish their goals (Carmien and Fischer, 2008).

EUD notation often overlaps with EUP since they share common principles, technological approaches and paradigms. Also, EUD seems to be the next step after EUP which was mainly limited in spreadsheet like desktop environments. An actual difference between EUP and EUD is that EUD methods, techniques, and tools span the entire software development lifecycle, including modifying and extending software, not just the "create" phase.

There are several EUP approached designed so far to assist end-user developers in modifying or constructing their artifacts. In Table 2.1 we summarize the main EUD technological approaches have been suggested so far in the EUD literature.

Table 2. 1 EUD technological approaches

EUD approach	Short description
Programming by Demonstration	Providing examples of the desired outcome (Blackwell, 2006).
Visual Programming	Using graphical representations to communicate technical concepts (Burnett and Scaffidi, 2011).
Programming by Specification	Describing the desired outcome to be generated (Burnett and Scaffidi, 2011).
Natural Language Programming	Transporting human familiar concepts to computer terms (Spahn et al., 2008).
Scripting	Using scripting languages to automate manual actions (Blackwell, 2006).
Programming with text	Using text to program the desired outcome (Leshed et al., 2008).

The second related concept overlapping with EUD is end-user software engineering. EUSE is a relatively new subset of EUD which focuses on the end-users' developed software quality. This area has arisen because of the ample evidence that the

programs end users create are filled with expensive errors (Panko, 1998; Burnett, 2010; Ko et al., 2011).

EUSE is a particular type of EUD research that focuses on the quality of end-user developed software (Ko et al., 2011). EUSE is a relatively new subset of EUD that began about a decade ago (Burnett and Scaffidi, 2011).

EUSE aims to improve the reliability of software produced by end-user programmers. It is an interdisciplinary research field, drawing from computer science, software engineering, human-computer interaction, education, psychology and other disciplines.

Many technological approaches have emerged to integrate software engineering principles within the end-user development field. For example the tool of Freedom (Nili et al. 2009) leverages the two complementary domains of web engineering and end-user programming. Other research works such as DENIM (Newman, 2003) and Whyline (Ko and Myers, 2004) are also considering the web engineering and end-user programming. DENIM (Newman, 2003) allows end-users to sketch the web site, its organization, and navigation and can be considered a beginner's design tool. Whyline (Ko and Myers, 2004) assists end-users in asking the right questions during the debug phase.

Examples of requirements (goals) in EUSE include personalizing the way that an application or computer behaves, automating time-consuming tasks, performing computations that are hard to do accurately by hand, or communicating information (Ko et al., 2011; Blackwell, 2004; Blackwell, 2006; Rosson et al., 2002).

Concerning verification and validation activities, the most developed end-user testing approach is 'What you See Is What You Test'(WYSIWYT), which employs a 'Surprise-Explain-Reward' psychological strategy, explained in Burnett et al. (2011), to guide users through the process of systematically testing spreadsheets (Burnett and Scaffidi, 2011).

Burnett (2009) illustrates EUSE in the following examples of the EUSES consortium projects:

- What You See Is What you Test (WYSIWYT) methodology and Surprise-Explain-Reward strategy.
- Debugging Machine-Learned Programs.
- Gender in End-User Software Engineering.

As Burnett and Scaffidi (2011) note, with continuing advances in EUSE, end users will not only be able to create a variety of software on their own, but they will also to be able to assess and improve that software's quality on their own. For example, the Idea Garden (Cao et al., 2012), a theory-based EUSE approach to help end users generate new ideas and problem-solve in a self-directed way (Burnett et al., 2017) emerged.

Various other theory-based approaches have been developed to support EUSE activities most can be found in Burnett et al. (2017)

Burnett and Scaffidi (2011) conclude, with continuing advances in EUSE, end users will not only be able to create a variety of software on their own, but they will also to be able to assess and improve that software's quality on their own.

2.2.4 End-User Development for Web Applications

While the first EUD tools were mainly focused on desktop graphical applications, in recent years a considerable amount of work has been carried out to apply the EUD approach to web environments (Paternò, 2013).

According to Rode et al. (2005), web EUD tools can be categorized to three main categories: database-centric tools that are primarily intended to help end-users put databases online for viewing and editing purposes, form-centric tools that are intended to help end-users create forms for collecting data and website-centric tools whose primary purpose is assisting the user with the creation of static or dynamic websites.

Efforts taken by the research community so far in the context of end-user programming and the web, include FAR (Burnett et al., 2001), WebSheets (Wolber, 2002), FlashLight (Rode and Rosson, 2003), CLICK (Rode et al., 2005) and Freedom (Nili et al., 2009). Freedom is a model-driven, web-based development platform that addresses the business users' development needs by supporting the concept of Goal

Driven Development and providing effective, web-based, development tools that assist business users developing DIY web applications. The WebSheets (Wolber et al., 2002) and Click (Rode et al., 2005) environments both assist end users in developing web applications at a level of abstraction, providing a generation of database-driven web applications. There are several other systems that impose similar constrains on designs, such as the Yahoo Pipes and the Apple's Automator (Ko et al., 2011).

Efforts such as WebML (Ceri, 2000) research the model-based web development. WebML offers the developer a full scale modeling language that can be used to model a web application end-to-end (content, page flow, database interaction, etc.). Once the model is defined, an application can be generated. It is reported that the development and maintenance of WebML applications led to 30% increased productivity with 46 distinct applications maintained by 5 part-time, junior developers (Fraternali et al., 2006).

Similarly, there are efforts that have examined template-based web development tools (Turau, 2002; Zdun, 2002). Turau (2002) describes a formal definition of a very lightweight model that validates and manages the data that the user inserts to the databases. In this category we can include some open source frameworks like Zend, Ruby on Rails and other that are also lightweight approaches to using models and templates to construct applications. In these approaches the end user developer designs a set of templates that are used to create the content of the pages when rendered by the user's browsers.

The programming by example approach has also been implemented in Web environments through different mechanisms. Nichols and Lau (2008) describe a system that allows users to create a mobile version of a Website through a combination of navigating through the desired portion of the site and explicitly selecting content.

Other research approaches include Chickenfoot (2010) is an extension of Mozilla Firefox, which allows users to modify Web pages without knowing HTML and using a tool (the browser) familiar to most.

Marmite (2007) tool allows users to select some operators, place them in a data flow representation, and view the current state of the data corresponding to a particular

operator in a table, which shows what the data looks like after it has passed through the operator.

2.2.5 End-User Development for Mobile Applications

Recent advances in smart phones in terms of connectivity, processing power, and interaction resources have enabled the creation of mobile EUD environments. Also, software development using mobile devices is becoming increasingly popular. However, limited work has been dedicated so far to EUD for mobile applications.

Fortunately, recent advances in smart phones have enabled the creation of mobile EUD environments. Mobile EUD approaches mainly address the following aspects: parameterization of the mobile terminal (Tuomela et al., 2003) frameworks to support mobile authoring and execution (Danado et al., 2010), mobile authoring tools (Cuccurullo et al., 2011; Danado et al., 2010) creation of UIs through sketching or by adding interactive techniques in the touch screen (Seifert et al., 2011). Examples of desktop EUD environments targeting mobile applications have been tourism, museum guides, and home applications (Danado and Paterno, 2014; Namoun et al., 2016).

According to Namoun et al. (2016), a review of the mobile EUD reveals three types of EUD activities performed by mobile users:

- Creation of mobile apps;
- Creation of mashups;
- Creation/modification/extension of games.

The first EUD environments to create applications for mobile devices have mainly targeted desktop environments: they assume that people use the desktop for developing the application, which is then deployed in the mobile device.

For instance, Ghiani et al. (2009) have developed an environment that allows customization of mobile solutions for museum guides, performed mainly on desktop systems. Akesson et al. (2003) suggest a user-oriented framework to ease the reconfiguration of ubiquitous domestic environments.

iCAP (Dey et al., 2006) allows end-users to visually design application prototypes by defining elements and rules.

Floch (2011) suggests the design of a city guide that can be tailored by end users in order to include information from different service providers according to the visitor's position and visiting purpose.

Collapse-to-zoom (Baudisch et al., 2004) is a technique for allowing the end user to view web pages on mobiles' screen by interactively removing irrelevant content.

All the above examples regard desktop EUD environments targeted at the development or configuration of mobile applications. However, desktop EUD environments lack the advantages of enabling end users to create applications opportunistically, on the move. Some limited Cloud EUD approaches for mobile apps have recently been developed. For instance, MIT has leveraged App Inventor EUD tool for mobile applications. The tool's EUD approach follows a blocks-based programming paradigm to visually design Android apps (Shiller et al., 2014). The constructed application is a simple client JavaScript app but App Inventor2 new features offer the ability to use cloud data storage functionalities in Google Cloud and in Google Fusion Tables. The tool renders end-users, including kids able to create their own applications for personal use and social impact.

Lao (2017) used App Inventor to make cloud technology available to end users and to assist them in creating mobile applications. Lao developed a CloudDB for the MIT App Inventor that allows users to store, retrieve, and share various types of data in tag-value pairs on a Redis server for their mobile applications.

2.2.6 End-User Development for the Internet of Things

The new era of Internet of Things (IoT) has changed the way people use the web, mobile and sensor-based devices. IoT helps in improving quality of life and in offering an even richer and satisfying experience of use of everyday objects (Barricelli and Valtolina, 2015).

As Danado and Paterno (2014) explain, 'IoT offers unprecedented opportunities to achieve deeper, more meaningful and faster insights by putting the user at the center of informative systems, ambient and personal sensors, communicative tools, and mobile and ubiquitous computing devices'. Since IoT applications need to address

deeply contextualized user needs, end-users need to manage on their own physical devices, interactive systems, and quantified-self data (Barricelli and Valtolina, 2015). Consequently, IoT brings about a transformation to the role of the end-user, who assumes an increasing number of responsibilities traditionally intended for professional developers. To this end, end-users are getting transformed from traditional 'users of an application program' (Cypher, 1993) to end-user developers (Lieberman et al., 2006). The technology and practices that can support this transformation is the field of End-User Development (Tetteroo and Markopoulos, 2015) and hence the new research field of 'EUD for the IoT' has recently emerged.

EUD research (e.g. Beckwith et al., 2005, 2006a, 2006b; Burnett et al., 2008, 2010, 2011; Saadé et al., 2012; Subrahmaniyan et al., 2008) though has shown that end-users tend to be 'prone' to the influence of human factors while interacting with computer environments. Hence, not only technical but also user-side factors should be well analyzed before end-user developers get deeply engaged in EUD activities for the IoT.

Economides (2017) argues that there are various important factors that affect users' attitude toward adopting IoT solutions. According to the author, these factors can be classified into the following categories: i) User-System Interaction (including perceived ease of use, perceived accessibility, perceived control, etc.), ii) IoT System Operation (including perceived efficiency, perceived performance, perceived reliability, etc.) and iii) IoT System Results (including perceived usefulness, perceived satisfaction, perceived anxiety, etc.).

Many of the above mentioned factors have been shown to influence end-user developers' behavior and engagement in EUD activities (Beckwith et al., 2005, 2006a, 2006b; Burnett et al., 2008, 2010, 2011; Saadé et al., 2012; Subrahmaniyan et al., 2008). Also, literature (e.g. Chagas et al., 2015; Tetteroo et al., 2015) highlights the need for more comprehensive and fundamental understanding of the current practice of EUD for IoT. As Tetteroo et al. (2015) observe 'we need new ideas on tools, services and architectural infrastructures able to support EUD in the context of the IoT'.

Some limited yet EUD design approaches have been recently designed to allow end-user developers build and tailor IoT applications to their specific needs and

preferences. For instance, the Puzzle framework (Danado and Paterno, 2014) enables end users without programming experience to create smart applications in mobile touch-based devices. Puzzle allows users to visualize the current status of intelligent objects, such as home appliances and generally, to prototype new Internet of Things applications through a user-centered design approach.

Another popular EUD approach for web, mobile and also IoT applications is the the 'If This Then That' (IFTTT) platform (Ur et al., 2014; Fogli et al., 2016). IFTTT is a free cloud service available via web or mobile app for the creation and use of simple programs that execute actions according to triggers. IFTTT enables end-users to define custom rules for automation following some well-known EUD design principles (like mashups). Triggers and actions include online third-party services as well as connected IoT devices.

Focusing on the 'human-side' aspects, Burnett and Scaffidi (2015) suggest some EUD debugging approaches to "enable ordinary users to customize, control, and fix Internet of Things applications that are trying to help them", and also to allow IoT systems to be inclusive to both male and female end users.

Chagas et al. (2017) uses a scenario-based approach to suggest the use of semiotic engineering, a Human Centric Computing (HCC) perspective, for analyzing the interaction between end-user developers and IoT systems. Chagas (2017) uses the IFTTT paradigm (Ur et al., 2014; Fogli et al., 2016) to interact with the Phillips Hue light ecosystem and analyze the complicate parts of this interaction. The author concludes that by driving our attention to the quality of the communication between people (developers and users) through distributed interfaces we can 'solve' complex EUD for the IoT scenarios.

2.2.7 End-User Development for Database-Centric Applications

Interacting with databases has become commonplace in most end-user activities. This gives rise to the need for database design by the end-users. However, traditional database design is still heavy-weight, requiring technical expertise that end-users do

not possess. Only a few articles address EUD issues in the context of database interaction.

Multiple approaches have been developed to support the fast creation of custom database-centric web applications by removing the need to use a programming language. Examples of such approaches are model-driven (Brdjanin and Maric, 2013; Fraternali et al., 2010) and goal-driven approaches (Adi et al., 2008). However, the level of abstraction offered to the users by EUD model-driven or goal-driven prototype applications is not sufficient. As a matter of fact the above-mentioned approaches refer to End-User Programming rather than to End-User Development. This implies that systems using these approaches are still not usable by end-user developers since they require basic (if not advanced) knowledge of the underlying modeling language and maybe some database design logic and relational schema terminology. The end-users have to be familiar with the design of Entity-Relational diagrams, tables' construction, relationships assignment and data validation in order to use them more efficiently.

For instance, Rode et al. (2005; 2006) suggested and developed the PhpClick prototype application that provides the end-user with a comprehensive EUD environment to build their own applications. PhpClick fully integrates the process of modeling the look and feel, component behavior, database connections, publishing and hosting, while working on a high level of abstraction appropriate for non-programmers. It also supports the automated design of the database schema at the run-time, while the user builds the application. Despite its user-centric design, PhpClick still demands from the user some basic database knowledge in order to create the tables and their fields. As its creators claim, PhpClick is for medium-complex custom web applications, meaning that end-users need to be familiar with some basic concepts to efficiently develop a web application.

The emerging Do-It-Yourself approach was primarily targeted non-programmers by enabling them to build forms, which automatically lead to corresponding tables that are reported on the same page, without the need to create a database schema in the abstract. In this context, the app2you/FORWARD application (Ong, 2010) offers a page-driven design in a WYSIWYG fashion to facilitate non-programmers to create web-

based business process. Its architecture hides database schemas, queries, constraints and other low-level details. The app2you/FORWARD provides the users with wizards that abstract the relational database terminology and explain all the concepts at a high level. However, creating a full scope application using app2you/FORWARD does require some basic SQL queries knowledge (e.g. WHERE clauses and EXISTS conditions).

Following the programming by example, the Visque prototype application (Borges and Macías, 2010) was developed to support easy web interaction with relational databases for the end-users to obtain valuable information from them. Visque was also based on the paradigm of visual prototype applications in database systems in order to manipulate models and data for unskilled end-users. The solution provided is a combination of End-User Development techniques, web-based user interface design and data models in a user centric way. Although Visque assumes no skills in SQL statements, it is for the end-users to build queries and not the database framework.

Following the component-based approach, XIDE (Litvinova, 2010) was designed to support the development of web applications based on component-based database architecture. The system provides several features that help in the development process and at the same time abstracts from the user difficulties that are not able to come due to the lack of programming knowledge. XIDE offers a EUD environment for web application through customizing the exiting components, but in order for the user to create new database components programming skills are required.

Another approach was suggested by Deng et al. (2011) where end-user developers can not only build database driven applications but they can also reuse and extend these applications. In particular, the authors describe a framework that allows a basic data model with several co-existing variations to satisfy the requirements of different group in a common domain and provide tools to help users to extend the data model if new requirements emerge.

In the context of GUI database design, CRIUS (Qian et al., 2010) has attempted to address the recently emerged challenge of enabling non-expert user to “give birth” to a database schema. CRIUS follows the spreadsheet-like paradigm to support schema

creation and modification using an intuitive drag-and-drop interface. CRIUS allows end users to create and modify the schemas associated with their data by using a simple drag and drop interface. However, the spreadsheet-like nature of CRIUS and its overall design framework does not render it a suitable prototype application for building database driven applications for the web.

Several graphical query languages for mobile systems have also been developed. For example, MoSQL is an icon-based mobile information system (Chang, 2003) and Query by Zoom (QbZ) is a query language based on Semantic Zoom for small screen devices (Silveira, 2012). Based on the Query- by-Object (QBO) approach which enables end users to obtain information in a database without using SQL, Akiyama and Watanobe (2012) have developed a visual user interface on mobile devices so that the users can intuitively manipulate the objects for information retrieval. Their system assists end-users who have no skills in query languages to search required items in a step-by-step approach. The tables in the corresponding database are mapped to objects and algorithm of SQL generation builds SQL queries.

As mentioned, most of the above cited works are targeted at the end-users' facilitation to build queries in order to manipulate, search and retrieve data from existed databases and/or demand at least some basic SQL knowledge or familiarity with database terminology.

The end-user developers' mental model (i.e. perception) regarding database concepts was examined by Rode et al. (2006). This work's experimental findings showed that although the end-users perceive well enough the table structure paradigm, an existence of more than one tables with linked data records would cause them total confusion. Moreover, end-users proved to be 'unable' to design a normalized database representation. Instead, they tended to follow the spreadsheet like paradigm to store data in a redundant way totally omitting the need of foreign-keys and uniqueness of data. Based on these research results the authors assume that the end-users lack even an abstract perception to recognize the need of foreign keys (referring to the concept and not the terminology) assignment. They also are not well aware of the necessity to organize their data in a way that avoids data redundancy.

The above described situation implies that end-users who attempt to develop their own database-driven applications meet many difficulties and possibly fail in the design of their initial database schema.

2.2.8 Cloud Computing and Benefits for the End-Users

Cloud computing has emerged as a dominant computing model in IT infrastructures, enabling flexible, ubiquitous, on-demand and cost-effective access to a wide pool of shared resources (Barroso et al., 2013).

The term 'Cloud Computing' describes a variety of tools that provide completely outsourced Internet-based solutions to the need to access applications and information anywhere anytime (Kepcyk, 2011).

Today cloud computing promises a new approach to IT economics, but also presents new technological and social challenges. Cloud computing is not just about data center technology, it's also about streamlining business processes to make organizations and people more strategic, more responsive to change and more oriented to service delivery. Change is an essential characteristic of software development, as software systems must respond to evolving requirements, platforms, and other environmental pressures. Software systems, like economic theories and Darwin's Galapagos finches, are embedded in an environment that is also continually changing (Dawkins, 1976; Jacobs, 2000). Cloud software evolution depends highly on user needs and technological trends. One important trend is the multiple device ownership. Users now have many devices for accessing the same services. Moreover, with Cloud computing, the relationship between the user and machine has turned into a many-to-many relationship, which results in the situation where we have different services to different people.

According to Li et al. (2010), Cloud Manufacturing systems should be able to collect real-time data from manufacturing resources (e.g., machines, robots, and assembly lines), store these data in the cloud, remotely monitor and control these manufacturing resources. As Li et al. (2010) stress, cloud-based data storage provides users with ubiquitous access to a broad range of data stored in the networked servers.

Hence these data will be reachable and continuously available both to the users and cloud service providers.

Finally, the cloud emerged Software as a Service (SaaS) model which is typically used to make a reusable cloud service widely available to a range of cloud consumers (Erl et al., 2013) can bring several benefits to end-users. Following, we present some main SaaS characteristics that according to Reese (2010) are the main reasons for the end-users' SaaS adoption:

- Availability via web browser;
- On-demand availability;
- Payment terms based on usage;
- Minimal IT demands;
- Multitenancy.

In general, SaaS offers ease of use, ease customization, ubiquitous access and various other advantages to the end-users. For instance, SaaS end-users do not need to care where the software is hosted, what kind of operating system it uses, or whether it is written in PHP, Java, or .NET. Also, SaaS end-users do not need to install any software on their devices.

2.3 Literature Concluding Remarks and Challenge

Cloud EUD has not flourished yet since most of the EUD tools are desktop or (more recently) web oriented. Mobile EUD has made its first steps but little efficient approaches have been implemented. As literature showed, Cloud offers multiple benefits to the end-users, and hence end-user developers should enjoy the advantages of cloud-based EUD solutions such as scalability and availability.

Regarding web-based EUD tools for database-centric applications, we have not detected in the reviewed literature any EUD web approaches to build comprehensive applications including their database scheme. Almost every suggested approach for the end-users to develop a relational database structure demands at least some basic SQL knowledge or familiarity with database terminology.

The empirical research in EUD indicates that end-users are unfamiliar with the database logic since it seems not to be close to their natural way of thinking (mental model). Hence, new approaches should be traced to assist the end-users design effective database schemas without the need to be trained on database concepts and terminology. EUD prototype applications should follow the end-users' viewpoint instead of imposing them to follow the database logic as it is defined by professional developers (Scaffidi, 2006; Scaffidi et al., 2008).

However, creating EUD prototypes that provide a decent user performance requires a deep understanding of end-user capabilities, limitations and needs. Hence, our challenge is to design new a human-centric database-driven application to address the research question "Can non experienced and untrained end-users efficiently design relational schemes to develop database driven applications?" Based on this question, our essential concern is to render non experienced end-users capable of building a database-centric application and efficiently design its relational schema without the need to be trained in database logic and terminology.

Our challenge includes the Cloud EUD adaptation aiming to implement our suggested approach in a cloud-based EUD tool. By developing a prototype EUD tool end-users can create their database applications, 'enjoying' the cloud benefits underlined in the reviewed literature.

2.5 The proposed approach: Simple Talking - A Cloud EUD framework for database-centric applications

Based on the afore-described challenge we propose a prototype EUD approach named '**simple-talking'**. The simple-talking approach combines multiple EUD and non-programmer friendly techniques such as natural language, simplicity, error protection, wizard based design, information visualization, and programming by example.

The 'simple-talking' approach assists the end-users in creating tables, fields, correlations, integrity constrains, primary and foreign keys and defining data types in a natural way by abstracting them from these terms' semantics. It gives the impression to the user that is 'discussing' with the system in simple words while describing the ER

model of their personal database application. Finally, the system automatically generates the relational schema according to the user description.

2.5.1 Technical Overview

We developed a prototype cloud-based EUD application to integrate the simple-talking approach. The application is easily accessible via a remote IP and it was used in several laboratory tests under the frame of the thesis objectives.

The application was developed in PHP using the Zend framework. The databases we used to store our data were MySQL and PostgreSQL for abstraction purposes, i.e. to be able to connect to cloud database.

The overall cloud architecture of the prototype EUD tool is depicted in Figure 2.3.

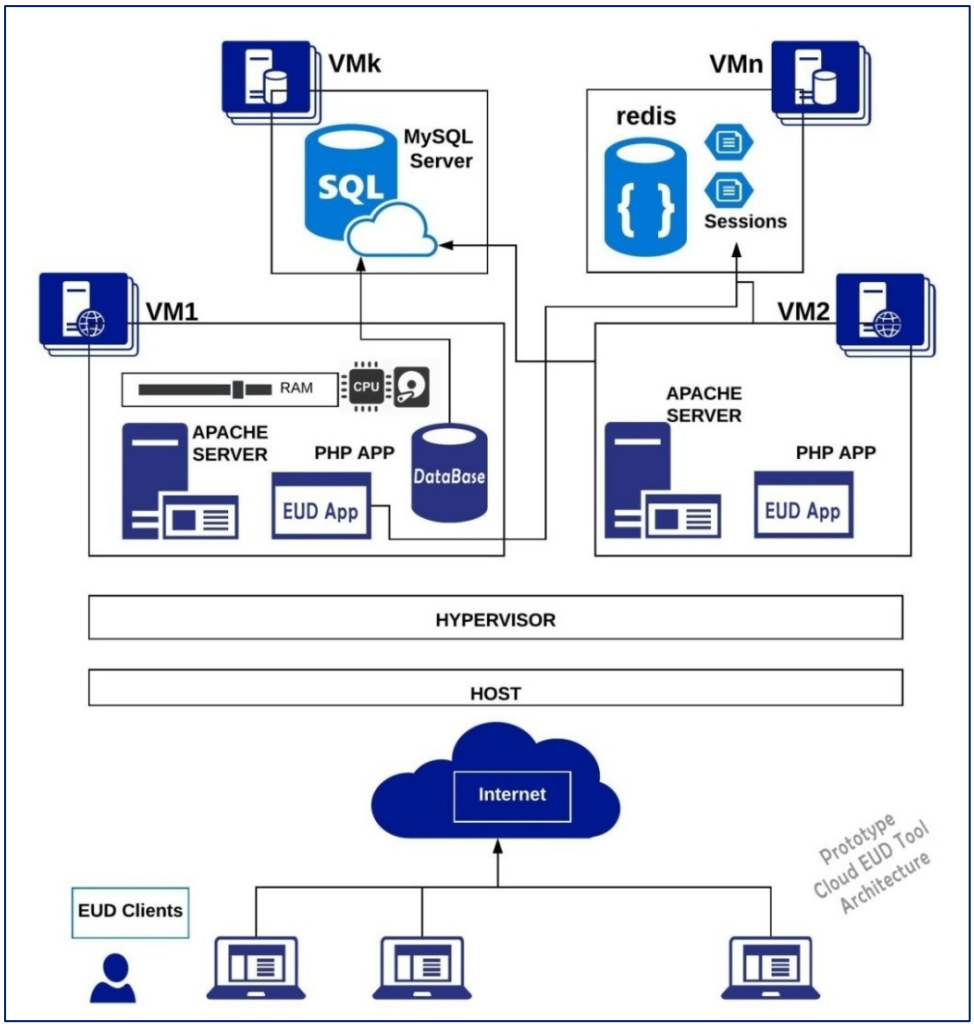

Figure 2. 3 Prototype Cloud EUD tool architecture

The tool was designed following the cloud principles in terms of database and session scalability. As depicted in Figure 2.3 every time an end-user connects to the application a new session and a new database connection is created. The more connections the more RAM, CPU and disk load and hence the load balancer 'decides' for the constructions of new instances of virtual machines (e.g. VM2).

User sessions are stored at a Redis server. This way all sessions are scaled-up. Similarly, MySQL database is not in the same virtual machine with Apache server and PHP application, but in one or more remote virtual machines.

2.5.2 Presentation

Overall Concept

As explained, it is hard for the end-user developer to efficiently produce a database model without training. These models are full of difficult to understand concepts, complex logic and abstruse terminology.

In the 'simple-talking' approach we investigate the possibility of replacing the training process with a simple question flow and a 'real-world word' to a database terminology word mapping. In the concept of this research we have come-up with a two-step process: the 'Word–Mapping' and the 'Question-Forming'.

The first step in the approach consists in mapping (when needed) each database term to an every-day simple word. To this objective we have created a table that maps each term of the database terminology to real world words. In the second step we ask the user to provide a simple but meaningful action verb both in the active and passive voices (e.g., has, buys, places, takes, belongs to) to name the relationship. Then we use the action verbs along with the mapped terms to create appropriate questions for each and every concept we need to extract from the user's mind.

The Word–Mapping Step

Table 2.2 contains the database terms and the real world word mapping taking place in the first step of the approach.

Table 2. 2 Mapping Database Term to Real-World Word

Database Term	Word mapping
Entity/Relation	Object
Strong entity	Basic object
Weak entity	Dependent object
Relationship	Relationship
Field/Column	Attribute

As depicted in Table 2.2 the abstruse terms have been replaced. For instance, the term 'object' is mapped to the term 'entity' since it is more common and the users are more familiar to. The term 'field' is mapped to the term 'attribute' since it is often used in everyday life to refer to the characteristics and properties of various objects. The term 'relationship' is self-explanatory and used as is. Terms such as 'foreign key' and 'primary key', are omitted from the mapping table since they do not appear in the questions but are implicitly extracted. For example the primary keys are surrogate keys and automatically generated for every object (entity) the user defines.

The Question-Forming Step

The 'simple-talking' approach follows a Question-Forming logic in order to extract from the user's mind the concepts of the database items (relationships, data types etc.). Based on this logic our approach can decide on the details of the database items, such as the relationship type and direction, the data type of an attribute, the validity of an attribute (whether it can really exist as an attribute in a relational schema), etc. Following, we present the Question-Forming logic in the cases of relationships, attributes and data type's construction and configuration.

The Relationships case

In this step the user is asked to provide a name for the relationship that is a simple but meaningful action verb. The user is guided to assign relationship names that are significant to the application and that are commonly understood in everyday language. Since all relationships are bi-directional in one direction, the active voice of the verb applies and in the opposite direction the passive voice applies.

Table 2.3 presents the questions provided to the user so as he/she can describe the relationship between two objects.

Table 2. 3 Relationship Titling

Requested Term	Question
Relationship title	Provide with an expression that defines the relationship between first and second object.
Reverse relationship title	Provide with an expression (possibly using the verb of the previous expression in the passive voice) that defines the relationship between second and first object

For instance, if a user has created an object 'CUSTOMER' and an object 'PRODUCT', the verb he/she will probably choose to describe the relationship between 'CUSTOMER' and 'PRODUCT' is the verb '*buys*'. In the opposite direction of the relationship (between 'PRODUCT' and CUSTOMER') the user will probably choose the passive voice of the verb, i.e. '*is bought by'*.

Regarding the type of relationships (one-to-many, etc.) and the type of entities (weak or strong), just the terms' translation is not enough and for this reason our approach adopts the Verbs-and-Questions logic, in order to extract from the user all the information needed to define each item's type in order to effectively produce the ER to relational mapping. The 'simple-talking' approach allows the end-users to use their own words (verbs) in order to describe the type of relationships among the entities.

The user has to describe the relationship between two objects, giving an expression by using a verb that best describes the relationship, by answering on the following statements:

- 'Provide with a Relationship name'
- 'Provide with a Relationship description'
- 'Choose the first member (list of objects given) to participate to the relationship'
- 'Choose the second member (list of objects given) to participate to the relationship'

- ‘What is the first member’s role? (fill only if first and second member are the same)’
- ‘What is the second member’s role? (fill only if first and second member are the same)’
- ‘Give the expression (using a verb-could be passive) that defines the relationship between second and first object.’
- Then the approach ‘decides’ on the relationship’s type and the participation constrains based on a number of questions such as:
- ‘Object-one’ ‘verb given by the user’ many ‘object-two’? (e.g. ‘Customer’ ‘buy’ many ‘product’?)
- ‘Object-two’ passive verb given by the user (e.g. belongs to) many ‘object-one’? (e.g. ‘Product ‘belongs to’ customer?)
- ‘Object-one’ verb given by the user (e.g. has) at least one ‘object-two’? (e.g. ‘Customer’ ‘has’ at least one ‘product’?)
- ‘Object-two’ passive verb given by the user (e.g. belongs to) at least one ‘object-one’? (e.g. ‘Product ‘belongs to’ at least one ‘customer?)

Once the relationship type is defined by the end of the question cycle the foreign keys can be generated and the appropriate constraints applied to the generated objects.

In the case of a ‘many-to-many’ relationship, the approach automatically generates a third/ intermediate table and creates the primary (*id-of-middle-table*) and foreign (*id-of-first-table, id-of-second-table*) keys.

Regarding the participation constrains the user can define them simply by answering to a question and then the SQL generating code will insert the statement “NOT NULL” next to the corresponding attributes’ names.

The Attributes case

For every basic object the user has created, he/she can choose to add or edit attributes (fields). When the user adds an attribute for a basic object he/she has to

answer YES or NO in some specific questions in order for the attribute's 'appropriateness' to be checked. By 'appropriateness' we refer to the fact the user could by mistake describe an object or a relationship as an attribute, so the approach has to check for this before constructing it. The user's answers will define whether the attribute is 'appropriate' to be created or its creation would lead to a false relational schema. In case of an 'inappropriate' attribute the user will be automatically directed to another wizard avoiding the creation of wrong fields. If for example, based on the user's answers the system 'decides' that the user does not mean to create a field but a relationship between this object and another one that may or may not exist, he/she will be directed to the wizard creating a relationship and maybe to the wizard creating a new object.

The Data Types case

In order for the user to appropriately define the data type of the attribute (field) he/she created , the approach 'asks' him/her to complete a form regarding the data type, the maximum limit of characters number, the default value, the NULL and the UNIQUE determination. In this phase, the 'simple-talking' approach is inspired by the Topes project (Scaffidi et al., 2008) which uses examples to conduct a user friendly data validation for the non-programmers users' population.

Hence, since the end user is unfamiliar with most of these concepts, the 'simple-talking' approach replaces them with real examples and simple statements. The user will not see any data type terminology such as string, integer, float, Boolean, date etc. instead he/she will be provided with real examples of such data types to choose one. If a user wants to define a data type as integer he/she has to choose the example "only digits – e.g. 7 , 25 , 146" and if he/she wants a float number he/she will choose "only digits – e.g. 7.234 , 25.45 , 146.2" and so on. In case the field's value can be NULL the user has to check the statement: "Attribute value can be empty, unknown or inexistent".

2.5.3 Evaluation

The Simple-Talking approach has been evaluated in Protogeros and Tzafilkou (2015) where the authors conducted a field test on two different expertise groups of end user

developers to asses both their performance and their perceived acceptance. The study results demonstrated a high value of user performance for both expertise groups revealing the usefulness and validity of the Simple-Talking approach when integrated in EUD environments. Also, the high results of participants' perceived ease of use and usefulness reinforced our arguments and indicated the validity of the prototype cloud-based EUD tool.

2.6 Summary and Contribution of the Chapter

In this chapter we presented the main EUD approaches and its related sub-fields. Also, we outlined the EUD evolution steps from desktop systems to web, mobile and EUD the Internet of Things. Despite this evolution we have noticed no significant progress on cloud-based EUD environments. Hence, by describing the benefits that cloud-based systems and SaaS adaptation bring to the end-users we proposed Cloud EUD as a contributing step in today's EUD evolution.

In addition, through our research in EUD tools and approaches for database-centric applications we found no approach that can support the complete design of a relational schema by end-users unfamiliar with the relational logic. To this end, we designed a prototype EUD approach, named 'simple-talking' and a cloud-based EUD environment to integrate it. The added value of this work is that it attempts to integrate a prototype user-centered approach in EUD tools in order to assist end-user developers in effectively and efficiently building their database-centric applications and unconsciously design a database relational schema. No other approach so far has attempted to replace the database logic and terminology to every-day words. The 'simple-talking' approach attempts to combine behavioral research and software engineering to provide the HCI community with next generation and more 'natural' EUD tool cloud and/or web environments.

CHAPTER 3

CONSIDERING GENDER ATTRIBUTES IN END-USER DEVELOPER MODELS

In chapter 2 we presented the main EUD approaches as well as the end-user developers' psychological aspects. Also, we proposed a natural language approach to assist end-user developers in building relational database-centric mobile applications using a cloud-based EUD environment. In this chapter we will present the main HCI behavioral theories that outline gender differences between end-user developers. Then we will propose a user model structure composed of a set of the reviewed behavioral attributes. The model's usefulness and validity will be evaluated through an example application and a laboratory test conducted using the prototype EUD tool presented in chapter 2. The main contributions of this chapter have been published in Tzafilkou and Protogeros (2017b), Tzafilkou et al. (2016), Tzafilkou et al. (2015) and Tzafilkou et al. (2014).

3.1 Introduction

In the past decades numerous studies reported on marked gender differences interacting with computers, such as different conception of computers, different motives for using computers, different preferences, different styles (Saadé et al., 2012) and even different cognitive styles (Hubona and Shirah, 2004).

In the field of Human Computer Interaction (HCI) and its subfield of End-User Development (EUD) many research works (e.g. Beckwith et al., 2005, 2006a, 2006b; Beckwith and Burnett, 2004; Burnett, 2009, 2010; Burnett et al., 2008, 2010, 2011; Saadé et al., 2012; Subrahmaniyan et al., 2008) have found that end-users tend to be 'prone' to the influence of human factors such as gender, while interacting with computer environments.

Behavioral studies also accept and explain the existing grounded differences in the way male and female end users process information and generally behave during their interaction with computer systems (Beckwith et al., 2006; Saadé et al., 2012;

Subrahmaniyan et al., 2008). Gender has also been singled out as an important variable in the design of user interfaces and visualization techniques. It is also considered as an important user diversity issue for achieving "universal usability" of web-based and other computer services (Hubona and Shirah, 2004).

Researchers have been long reporting theory and empirical data pointing to gender differences in the use of end-user programming environments. Evidence of these differences has accumulated, indicating gender differences in programming environment appeal, playful tinkering with features, usage and attitudes toward end-user programming features, as well as end-user debugging strategies (Beckwith and Burnett, 2004; 2007; Beckwith 2003; Beckwith et al., 2005; 2006; 2007; Burnett 2009; Burnett et al., 2008; 2010; 2011; Kissinger et al., 2006; Grigoeranu et al., 2006). In these studies, the two genders have been shown to both use different features and to use same features in a different way. One of the most important conclusions of these studies is that the features most conducive to females' success are different from the features most conducive to males' success (Beckwith and Burnett, 2004).

Ignoring the gender issue, while designing end-user programming environments, would miss the opportunity of enhancing the effectiveness of end-user programmers. Such a solution could be achieved by incorporating appropriate mechanisms to support gender associated differences in decision making, learning, and problem solving (Beckwith and Burnett, 2004).

Given all these evidence, it can be said that gender differences are potentially critical to our understanding of how end users make their decisions about adopting and using new computer software to develop their own artifacts.

3.2 Literature Background

This section outlines some of the most predominant end user behavioral theories that shed light on gender factors in HCI and EUD behavioral research, namely: (i) *Attention Investment Theory*, (ii) *Technology Acceptance Theory*, (iii) *Self-Efficacy Theory*, (iv)*Information Gap Theory, and* (v) *Personality Traits Theory*. Finally, we include in the

considered theories the recently emerged *Gender HCI* research field which focuses on current behavioral issues in EUD area.

The above theories were selected mainly due to their impact on EUD, as this is demonstrated in the related literature. The proposed model's constructs have been derived by these theories. These constructs need to be few in number because the range of the adaptation responses that can be provided to the user according to their model is not very large (EUD adaptation approaches and interface design approaches, such as elements of intelligent interfaces, are limited).

3.2.1 The Attention Investment Theory

The Attention Investment theory (Blackwell, 2002) proposes a model of how end-users make decisions (to use for example, particular environment features) when engaged in problem solving. As described in Blackwell and Hague (2001), the Attention Investment model is a "decision-theoretic account of programming behavior".

Simply explained by Burnett et al. (2011), Attention Investment is an analytical model of user problem-solving behavior that allows a designer to account for the costs, benefits and risks that users need to consider in deciding how to complete a task. That is, end-users, in deciding to take any action, first weigh the perceived costs, pay-offs, risks and benefits of taking that action. Perception of risk thus plays an important role in a user's decision making about whether to use end-user programming features. Risk Perception is actually a subpart of the more generic Problem Solving attribute. As explained in Blackwell's Attention Investment Model theory (Blackwell, 2002; Blackwell and Green, 1999) Risk Perception strongly influences the end users' behavior through their cost/benefit evaluation. According to Blackwell (2002) terminology, Risk is the Probability that no pay-off will result, or even that additional future costs will be incurred from the way the user has chosen to spend attention. Blackwell's Attention Investment model provides a cognitive model of these insights, describing individuals' allocations of attention as cognitive "investments". According to the model, a user weighs four factors (not necessarily explicitly) before taking an action: (1) perceived benefits, (2) expected pay-off, (3) perceived cost, and (4) perceived risks. If they decide that costs and/or risks are too high in relation to benefits they may choose not

to follow the action. Hence, Risk Perception plays an important role in a user's decision making as to whether to use specific application features (Beckwith and Burnett 2004). The Attention Investment model is related to other descriptions of end-user strategy such as Carroll and Rosson's Paradox of the Active User (1987) which describes the way that users are reluctant to suspend productive use of already-learned (but perhaps inefficient) methods, and tend not to engage in learning further skills, even though this might bring longer-term benefits.

According to the definition of Bauer (1960), Risk Perception is "the combination of uncertainty plus seriousness of outcome". Researchers have discovered that if an individual feels uncertain, uncomfortable and/or anxious toward a new service, then the greatest influence on the adoption decision is the individual's risk perception (Featherman and Fuller, 2003). Similarly, Beckwith et al. (2010) suggest that such behavior can also apply to end-users deciding to use new features in problem-solving and other computing environments.

Gender comes into play because it influences the perception of cost, benefit, and risk. There is evidence that women perceive higher risk in everyday choices and behaviors than men do (Finucane et al., 2000). Just as the Attention Investment Model predicts, higher perception of risk can lead to differences in actual behavior, and such differences have been many times tied to gender. For example, due to the self-efficacy differences a female's perception of the cost of learning a new feature may be higher than a male's perceived cost to learn the same feature. Moreover, detected differences in motivations to use technology also suggest gender differences in perceived benefits (Burnett et al., 2011).

3.2.2 The Technology Acceptance Theory

The original Technology Acceptance (TAM) theories (i.e. those developed by Davis 1989) do not necessarily focus on end-users as their primary audience, and the technologies studied are general software technologies. Nevertheless, there are strong ties to the more specific research of end-user problem solvers (Beckwith and Burnett, 2007).

In particular, the recently emerged End-User Computer Acceptance (EUC) theory introduces the most relevant human factors affecting the end-users' overall behavior and performance as follows (Chen and Corkindale, 2008; Cyr et al., 2007; Sun and Zhang, 2008).

- Computer Self-Efficacy: A person's perception of their ability to use computers to complete a task.
- Computer Enjoyment: A person's reception of joy from using software.
- Perceived Ease of Use: The degree to which a person believes that using specific software will be easy and effortless.
- Perceived Usefulness: The degree to which a person believes that using specific software is useful for his/her job performance.
- Subjective Norm: The degree to which a person believes that other people that believe that that he/she is capable of accomplishing a particular task.
- Internal Computing Support and Training: Training and technical support provided inside the company.
- Task-Technology Fit: The degree to which an organization's application meets the information needs of the task.
- External Computing Support and Training: Management support or external training provided from outside the company.

According to TAM, user acceptance, and ultimately technology use, is determined by two key factors (among the above mentioned): perceived usefulness and perceived ease of use (Venkatesh and Morris, 2000). The relative importance of each differs by gender (Venkatesh and Morris, 2000). Gender related empirical findings in TAM show that female and male users differ in beliefs, intention and usage. Venkatesh and Morris (2000) found that women are more influenced by perceived ease of use in adapting new technology whereas men are more strongly influenced by perceived usefulness. The same is concluded in Terzis and Economides (2011) where the authors, based on TAM constructs, examined gender differences in users' behavioral intention to use a computer based assessment (CBA). Their findings also indicate gender differences in perceived ease of use, behavioral intention and social influence.

According to Teo, (2015), the limited empirical findings in this area suggest that gender differences, or lack thereof, related to perceived usefulness of technology may depend on the context in which the relevant technology will be used.

The Unified Theory of Adoption and Use of Technology (UTAUT) (Venkatesh et al., 2003) is the latest model which has been conceived to understand the nature of technology usage. Gender has been attributed as a significant variable in explaining the technology acceptance behavior of users. In particular, UTAUT has extended TAM by including a set of direct and indirect factors in predicting intention to use a technology. The indirect factors include gender, age, experience with the technology, and voluntarily use of technology. The direct factors include performance expectancy, effort expectancy, social influence and facilitating conditions.

Many different research works (e.g. Chang et al., 2007; Cheng et al., 2008; Hung et al., 2007; Liu et al., 2008; Sumak et al., 2010; Teo, 2011; Terzis and Economides, 2011; Venkatesh et al., 2008; Wang et al., 2009; 2010; Wang and Shih, 2009; Wu et al., 2007; 2008; Zhang et al., 2010; Zhou, 2008; etc.), have conducted UTAUT studies by applying a variety of research methodologies in different environments.

According to Lee et al. (2003) reviewing the performance of UTAUT or assessing the findings and limitations is crucial to recognize possible future research topics and guide future research endeavours. In Williams et al. (2015), the authors present a comprehensive literature review on the most prominent UTAUT research works.

According to Goswami and Dutta (2016), gender plays a significant role in determining the intention of accepting new technology and there are cases where gender differences cannot be discerned.

Gender differences have been found in UTAUT (e.g. Venkatesh et al., 2003) although age has influenced some relationships. For example, performance expectancy has been more influential for males' and young workers' intention to use a technology, while effort expectancy has been more important for females' and older workers' intention to use a technology (Venkatesh et al., 2003).

UTAUT extended research has also been conducted to examine gender differences (e.g. Wang et al, 2010) and related gender differences have been detected in the acceptance of mobile activities and mobile learning, where they were found to moderate the effects of social influence and self-management of learning (Wang et al., 2009; Wang and Shih, 2009).

Many other UTUAT research works that examine the influence of gender on technology adoption can be found in the recent literature review of Goswami and Dutta (2016).

Technology acceptance gender issues are crucial in the interaction between users and EUD systems, since Perceived-Ease of Use and Perceived-Usefulness are strong variables that can determine the end-users' perceived experience and their developing task performance (Beckwith, et al., 2006a; 2007; Beckwith et al., 2005; Burnett, 2009, Burnett et al., 2008; 2010; 2011; Lee, 2008). Hence, technology acceptance should be integrated in the behavioral analyses of EUD researches.

3.2.3 The Self-Efficacy Theory

Bandura's (1977; 1986) self-efficacy theory defines self-efficacy as a person's belief in his/her ability to do a specific task. Presuming ability to complete a task, self-efficacy distinguishes how individuals will approach and perform the task.

Self-efficacy actually predicts reaction and behavior in challenging situations (Bandura, 1986). According to Bandura's (1977) social-cognitive theory, people with low self-efficacy for a task tend to expend less effort on the task, use simpler cognitive strategies, show less persistence when encountering difficulties, and reveal lower performance rate than people with high self-efficacy. Not only does self-efficacy predict end-user task behavior, but it ultimately affects performance outcomes, influencing whether or not an individual succeeds at the task. Individuals with high self-efficacy for a specific task have several characteristics that aid their success in these tasks, characteristics that 'self-doubters' lack (Bandura, 1986). Hence, research has linked self-efficacy much closely to performance accomplishments, level of effort, and the persistence a person is willing to expend on a task.

As Blackwell et al. (2009) point out, being a challenging task, software development renders a person with low self-efficacy less likely to persist when a task becomes challenging or may calculate attention investment trade-offs differently.

In software applications, studies have found gender differences in self-efficacy. Females generally have lower computer self-efficacy than males, and this has been tied to feature usage (Hartzel, 2003) and many times to debugging features.

Being a specialized subset of self-efficacy, 'computer self-efficacy' is defined as a person's judgment of his/her capabilities to use computers in a variety of situations (Compeau and Higgens, 1995).

Researchers have reported gender differences in computer self-efficacy across nationalities and across levels of computer expertise (e.g. Beckwith et al., 2007; Beyer et al., 2003; Colley and Comber, 2003; Margolis and Fisher, 2003). Low computer self-efficacy among females is a prevalent research result in the literature. Females' low self-efficacy is further complicated by their tendency to attribute failure in a task to their own lack of capability, whereas the males attribute this to the difficulty of the task (Stipek and Gralinski, 1991).

3.2.4 The Information Gap Theory

Loewenstein's (1994) Information-Gap theory draws several earlier theories on curiosity, in to one theory. According to Loewenstein, "curiosity arises when one's informational reference points in a particular domain become elevated above one's current level of knowledge", where the "informational reference point" is what one wants to know.

The Information-Gap theory was the backbone in the development of the Surprise-Explain-Reward Strategy (Robertson et al., 2004; Ruthruff et al., 2004; Wilson et al., 2003), an approach aimed at changing end-user programmers' perceptions of risk and reward. The development and success of the Surprise-Explain-Reward strategy relies on raising a user's curiosity to an ideal level, such that he/she becomes aware of missing knowledge, but perceives it as attainable. For this reason, understanding the

underlying theory of the Information-Gap theory is important for the design of problem-solving environments that depend on it.

Research into curiosity indicates that surprising a user to arise his/her curiosity can render him/her search for an explanation. Hence, the ‘‘surprise’’ component is intended to distract users from bias toward a habitual strategy (arising their curiosity), the ‘‘explain’’ provides enough information for the user to reassess the attention investment factors, and the ‘‘reward’’ makes clear to the user what the payback has been for that investment choice, in a way that will influence future choices of strategy. Scaffidi et al. (2010) implement a Surprise-Explain-Reward Model to ‘manipulate’ end-users testing and debugging behavior in spreadsheet environments. In particular, they use colored borders to surprise the user and attract user’s attention to areas that need testing, also providing him/her with a potential reward.

As regarding to gender, it can significantly impact the effectiveness of using a “surprise” component to arise the user’s curiosity. In particular, the differences in end-users’ self-efficacy can determine the level a user’s curiosity could reach at, and hence predict the ‘surprise’ effectiveness to the particular user.

3.2.5 The Personality Traits Theory

Traits are lasting aspects of individuality that differentiate one person from another. All people share the same basic set of traits, but people are different in how much the trait applies to them (McCrae and Costa, 1999). Research works (Costa and McCrae 1992; Costa et al., 2001) suggest that individual personality traits can be classified into five basic dispositional traits which are typically called the Big Five or OCEAN: Openness, Conscientiousness, Extraversion, Agreeableness, and Emotional Stability (sometimes denoted as the opposite, Neuroticism).

As Shneiderman (1980) has stated in his famous book “Software Psychology”, “Personality variables play a critical role in determining interaction among programmers and in the work style of individual programmers”. Many researchers have noted the personal characteristics of individuals who appear intrinsically motivated to notice and explore new technology options (e.g. Rosson et al., 2004; Zang

and Rosson, 2010). Researchers generally agree that personality is more important than intelligence in programming tasks. Being such a crucial factor for programmers one could easily deduce that personality plays a key role for end-user programmers as well. And indeed many recent studies suggest that the user's personality in HCI plays an important role for the overall success of the interaction (Bantrica et al., 2012; Bickmore et al., 2005; Nass and Brave, 2005).

Many researchers have also associated personality traits to technology acceptance and computer use in general. For instance, the results in Terzis et al. (2012) indicate that the five personality factors affect Computer Based Assessment (CBA) acceptance. Devaraj at al. (2008) and Özbek et al. (2014) showed that user personality has effects on perceived usefulness and subjective norms toward the acceptance and use of technology. Papamitsou and Economides (2014) showed that personality factors are significant predictors in the temporal estimation of students' performance in computer based testing.

Moreover, many scientists from different research areas as Psychology, Neurology, Philosophy and Affective Computing agree that human reasoning and decision-making use human psychological aspects (Thagard, 2006; Trappl et al., 2003). Therefore, when humans try to personalize services to other humans, psychological aspects like Personality Traits, should be taken into account.

Research evidence suggests that the two genders differ with respect to personality traits (Costa and McCrae, 2001; Mastor, 2003; Srivastava et al., 2003). The literature also suggests a potential link between personality traits, gender, and computer self-efficacy. As already mentioned, computer self-efficacy (that is strongly gender defined) is important in user acceptance of information technology and end-user performance. Recent research results (Saleem et al., 2011) indicate that four of the five stable personality traits, as measured by the Big-five factors of personality, contribute to explain computer self-efficacy. Taking gender into account, results show that the traits of neuroticism, extraversion, and agreeableness are significantly related to computer self-efficacy for women but not for men. In this light, we strongly believe personality

traits to be coexisting determinant factors for the perceived gender differences in EUD, as presented in the next sections.

3.2.6 The Gender HCI research field

Gender HCI examines ways in which software features can interact with gender differences. While HCI concerns itself with the design and evaluation of interactive systems for human beings, such as user interface design, Gender HCI focuses on the differences in how males and females relate to these interactive systems and their respective designs. In simple words gender HCI focuses on human-computer interaction properties that take gender differences into account in the design of software.

Gender HCI (Beckwith and Burnett, 2004) explores the different way male and female end-users behave when interacting with computer systems, especially with desktop-based end-user programming environments. The authors' research was a continuation of investigating visual programming for managing spreadsheets. Gender HCI has proved that men and women tend to have different perceptions and preferences with respect to the use and satisfaction with different features of these computer systems. The main aspects of Gender HCI include self-efficacy and confidence issues as related to problem solving tasks on a given interface design, willingness to try out new and different features, performance of tasks, tinkering/exploratory behavior, motivations for system usage, general attitudes towards interface designs, etc.

In particular, Gender HCI research focuses on the following aspects:

- Confidence as related to problem solving tasks on a given interface design. (In this point we have to note that when referring to Gender HCI, confidence reflects to self-efficacy, i.e. a person's belief in his ability to perform a specific task (Bandura, 1977) and it should not be confused with self-confidence, which has a more general meaning including the value one is giving to himself).
- General attitudes towards interface designs, web apps, and how and why to use them.
- Willingness to try out new and different features in a familiar interface design.

- Performance of tasks on large vs. small user interface displays.
- Graphic design reactions.
- Testing and debugging strategies.
- Tinkering/ Exploratory behavior.
- Motivations for systems' usage.

Examples of Gender HCI research works regarding EUD issues are the ones of Burnett (2009), Burnett et al. (2008; 2010; 2011), Beckwith and Burnett (2004; 2007) and Beckwith et al. (2005; 2006; 2007) in which the researchers study the users' gender based differences while developing debugging strategies and/or using features while working on spreadsheet environments. They have designed and implemented features for spreadsheet prototypes that take the gender differences into account (Burnett et al., 2011).

Ignoring the gender issue, while designing end-user programming environments, would miss the opportunity of enhancing the effectiveness of end-user programmers. Such a solution could be achieved by incorporating appropriate mechanisms to support gender associated differences in decision making, learning, and problem solving (Beckwith and Burnett, 2004).

To this end, the recently emerged 'GenderMag' (Gender Inclusiveness Magnifier) project (Burnett et al., 2016) aims to 'solve' HCI gender issues not only within software by addressing the factors that affect female and male problem solving attitudes. The main research question it attempts to solve is the following: "would females and males be better supported at problem-solving if the problem-solving software they used took into account individual differences that often cluster by gender?'

The GenderMag method includes a set of faceted personas that bring five facets of gender difference research to life. It is actually a software inspection method that aims to help software creators identify gender-inclusiveness issues in their technologies. The project's research results show that a variety of practitioners who design software were able to use the GenderMag method to find gender-inclusiveness issues in problem-solving software (Burnett et al., 2016; Hill et al., 2017).

3.2.7 Domain knowledge, performance and gender issues

Gender is not the only determinant factor in the end-user developers' developing behavior. McLeod and MacDonell (2011) state that developers possess a range of characteristics that can influence how they approach and practice development. McLeod and MacDonell explain that these characteristics include technical skills, capabilities, domain knowledge, and experience, interpersonal, communication and social skills, application domain knowledge, commitment, motivation, trustworthiness, norms, values and beliefs. They also concern the attributes of personality type, experience with systems and organizational status.

It is well accepted that user expertise and skill affects the way users interact with software. As the end users' perception is mainly based on previous experience, different end users have different perceptions. Thus, it is common for users to be separated into novices and experts. Many differences in the working and thinking mode have been detected to exist between end-user groups of different expertise level and various interface styles have been adapted to serve better their distinct needs and preferences (Burnett et al., 2011). Basic behavioral differentiation between novice and more experienced end-users are comprehensively presented in Popovic (2000) and in Jason et al. (2010).

The fact that we refer to end-users should not exclude the existence of different expertise levels among them. End-users can never be experts (that's why they are called 'end-users') but as Costabile et al. (2008) explain they can belong to different classes according to their task-activity nature (development or customization) and their computer familiarity and/or experience degree which can be reflected to the way the users choose to complete their tasks (e.g. macros, domain-specific languages, web development etc.).

Research has also pointed out computer gender differences as regards to performance and perceived items (e.g. Beyer and Bowden, 1997; Hoxmeier et al., 2000; Tzafilkou et al., 2017b; Vermeer et al, 2000). According to Stipek and Gralinski (1991), women are more likely to attribute their task failure to their lack of capability. Also, according to

Brosnan (1998), women tend to underestimate their performance and ability in computer related tasks.

There are some personalization technologies that have been adopted in EUD approaches to enable end-users to receive optimized advice based upon their domain knoledge level (Burnett et al., 2011).

Liu et al. (2010) study shows the importance of user profiles in the interface design process and emphasizes novice and expert users' difference in various interaction situations showing also some of the implications for interface design. Basic behavioral differentiations between novice and experts between end users are comprehensively presented in Popovic (2010) and in Jason et al (2010).

Regarding the relationship between end users' domain knowledge level and performance, it is considered that an adaptive user interface (AUI) which dynamically changes from a novice user interface (UI) to an expert user interface could potentially improve users' performance. That is, user performance can be increased when the computer UI characteristics match the user skill level (Jason et al., 2010).

Based on Bandura's (1986) research, performance accomplishments are typically the strongest determining factor in an individual's assessment of self-efficacy. As Johnson et al. (2016) explain, end-users' self-efficacy in case of unsuccessful performance on one task can affect future efficacy estimations for the same or other related task. Hence, since women tend to perceive lower self-efficacy than men, they are more prone to future lower performance (Beckwith et al., 2007; Beyer et al., 2003; Colley and Comber, 2003; Cao et al., 2010; Durndell and Hagg, 2002; Durndell et al., 2000; Grigoreanu et al., 2008; Margolis and Fisher, 2003; Vekiri and Chronaki, 2008).

Burnett et al. (2010) use in their work the term 'programming populations' to refer to the different users' experience levels and examine the existence of gender-based behavioral differentiation among users within these populations. What comes to a conclusion is the fact that gender differences tend to cross all the experience levels. However, their degree of fluctuation seems to shrink and some differences even disappear between genders when users have more expertise.

Also, using gender, age, experience and job title as the main factors for the detection of behavioral differences in the examined populations, Burnett et al. (2010) have concluded that gender plays the most important role while age almost none. Moreover, they have proved that women tend to stably possess fewer experience and less technical job regardless the population they belong to. Not surprisingly, experience and job title seem to be closely related to gender attributes such as confidence, tinkering and feature usage.

The observable presence of the gender gap in all these 'user-categories' makes us willing and curious to take both gender and expertise level under equal consideration while allocating end-users into more concise categories (or 'populations').

There have also been some EUD personalization technologies that enable end-users to receive optimized advice based on their experience level (Burnett et al., 2011). A good example is the Wire system, developed by De Angeli et al. (2011), which is used as a EUD recommendation tool. However, little research has indeed tried to base such personalization implementations on the detection of gender associated behaviors of novice and 'expert' end users.

3.3 Concentrating the Gender EUD Attributes

In order to organize the relevant literature into a coherent form, we have theoretically devised a list-taxonomy of this work. Each element of the list is a particular issue that has emerged as a recurring theme from the literature we survey here. These are recognized as affecting the end-users' behavior as well as their computer performance, especially in EUD activities.

Many of the listed issues have interdependencies among them but each issue was selected because of the received attention as a gender-sensitive attribute for the end-users in multiple literature works.

Attempting to focus strictly on EUD research, our list does not include HCI broad gender-differentiated attributes that are being indicated in some HCI literature works, such as Problem Solving, Learning Style and Information Processing, because they are general and this would increase complexity in our user model 'construction'.

These attributes' values (e.g. linear/nonlinear Problem Solving, holistic/elaborative Information Processing, abstract/concrete Learning Style) can significantly increase the end-users' overall behavior and performance (e.g. nonlinear Problem solving could reveal correlation among unrelated concepts, lack of objective, attention deficit, loss of control, higher risk of failure, etc.; elaborative Information Processing could reveal higher cognitive effort, information overload for simple tasks, and abstract Learning Style could reveal Difficulty in concentrating on one thing at a time).

Although excluded (due to their general nature), many of the reviewed theoretical foundations (e.g. Information Gap Theory – Loewenstein, 1994; Attention Investment Theory –Blackwell, 2002; etc.) have been developed to reveal all the aforementioned specific behavioral attributes.

Hence, instead of using these generic attributes, we include in our model some basic sub-parts of them. For instance, Risk-Perception can be considered as a sub-part of the generic problem-solving attitude (Blackwell, 2002), Curiosity is reflected in the generic information processing attitude (Loewenstein, 1994) and Tinkering can be reflected in the Learning style; for example tinkering behavior can reveal an experiential related learning style (e.g. Davies et al., 2012; Marin, 2014).

Moreover, related literature indicates some additional gender-determined factors as influencing the end-users' behavior, such as Facilitating Conditions and Social Influence (Terzis and Economides, 2011), but we do not consider them as end-user developers' behavioral attributes, since they depend on 'external' sources.

We do not aim to provide analytic statistical results showing that end-user developers differentiate their behavior depending on such attributes, since their examination can be found in the corresponding research works. Our distinct contribution is to concentrate and clearly classify all the gender-based behavioral attributes that main EUD-related HCI researches found to influence the end-users' developing performance. The purpose of this section is to concentrate on the attributes rather than on the confirmation of the gender differences reflected on them. All these gender differences have been already confirmed to exist by the underlying literature works

and as we have already mentioned what is missing is a comprehensive behavioral model to analyze the end-user developers' behavior. Based on the literature review these attributes tend to differently affect male and female end-users developing performance.

In Figure 3.1 we attempt to aggregate the main behavioral gender-driven EUD attributes as they derive from the reviewed literature.

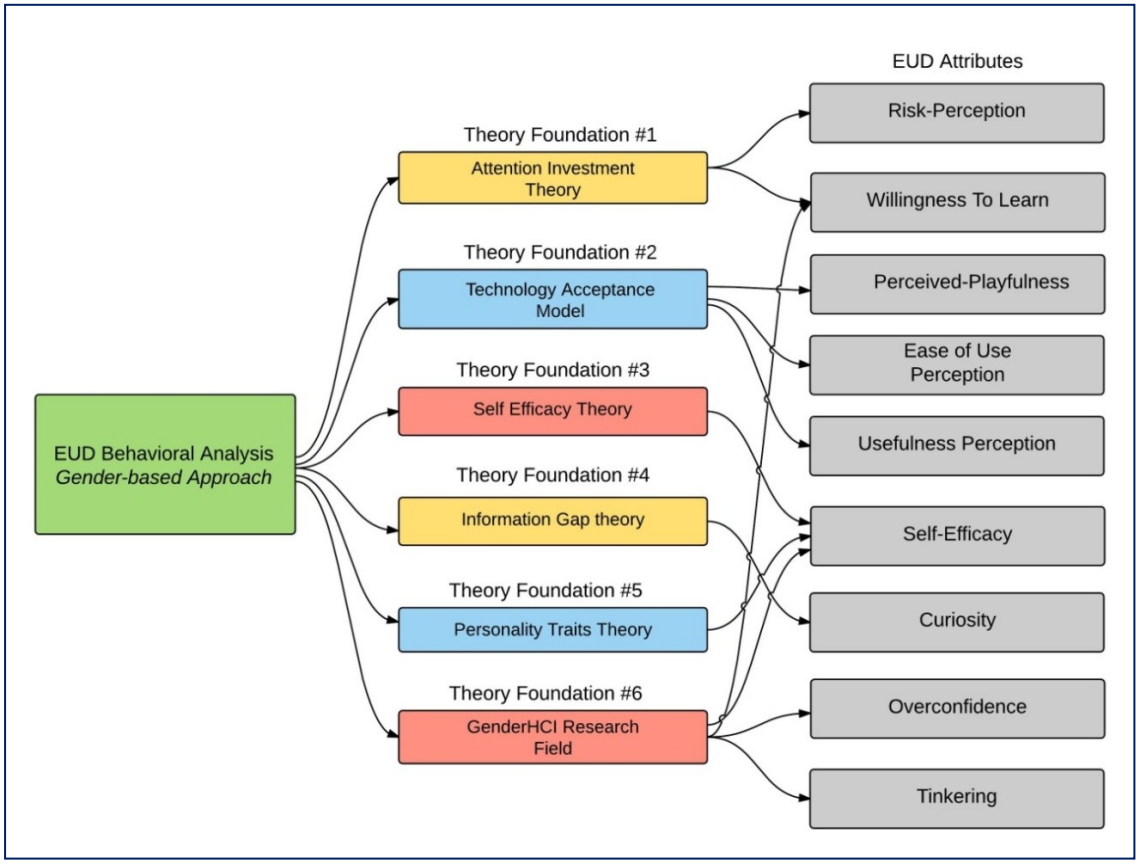

Figure 3. 1 EUD gender-behavioral attributes and relevant theories

Table 3.1 presents the source literature as well.

Table 3. 1 List of Gender-based EUD behavioral attributes

Behavioral Attribute	***Theory Foundation***	***Support Evidence***
Self-Efficacy (or Computer Self-Efficacy)	Self-Efficacy Theory; Technology Acceptance Theory; Personality Traits Theory; Gender HCI	Beckwith et al., 2005; 2006;2007; Hartzel 2003; Burnett et al., 2010;2011; Beckwith and Burnett, 2004;Grigorenau et al., 2008; McIlroy et al., 2001; Shea and Bidjerano, 2010; Terzis and Economides, 2011

Overconfidence	Gender HCI	Burnett et al., 2003; Beckwith and Burnett, 2004
Curiosity	Information Gap Theory	Hartzel, 2003; Grigorenau et al, 2008
Tinkering (or Exploring-Behavior)	Gender HCI; Technology Acceptance Theory	Beckwith et al., 2005; 2006; 2007; Burnett et al., 2010;2011; Grigoeranu et al., 2008; Martinson, 2005; Rode, 2008
Willingness To Learn (new tech, or to use new features), or else Learning Willingness	Attention Investment Theory; Gender HCI	Beckwith et al., 2005; Broos, 2005; Burnett et al., 2010; Grigorenau et al., 2008
Risk-Perception	Attention Investment Theory; Gender HCI	Beckwith and Burnett, 2004; Finucane et al., 2000; Byrnes et al., 1999
Usefulness-Perception	Technology Acceptance Theory	Chen and Corkindale, 2008; Cyr et al., 2007; McLeod and MacDonell, 2011; Featherman and Fuller, 2003; Saadé et al., 2012; Sun and Zhang, 2008
Ease-of-Use-Perception	Technology Acceptance Theory	Beckwith and Burnett, 2004; Featherman and Fuller, 2003; Saadé et al., 2012
Perceived-Playfulness (or Playful-Behavior)	Technology Acceptance Theory	Grigorenau et al., 2008; Terzis and Economides, 2011

Drawing from the aforementioned literature findings we can conclude that behavioral variables tend to meet different values cross different gender/expertise user groups. Thus, gender and expertise level indirectly or directly influence the end-users' preferences and behavior, which in turn influence their task performance.

3.4 Literature Review Concluding Remarks and Challenge

Many studies have emphasized the existence of different mental models between programmers and non-programmers, as well as of different priorities and motivations:

they follow different approaches and reasoning strategies to modeling, performing and documenting the tasks to be carried out in a given application domain (Costabile et al., 2008; Blackwell and Morison, 2010). In this context, research in Human-Computer Interaction (HCI) has put considerable effort over the past decades to build theories and models which attempt to explain end-users' behavior while using computer software to customize, program and/or develop artefacts, since "*we can build better End-User Development tools if we know how end-users think*" (Rode et al., 2005).

Most of the resulting theories assume that end-user behavior is influenced and many times determined by a set of attributes/user characteristics, such as age, location, level of education, interests, habits, goals, mood, personality traits, learning style, experience / expertise, etc. (McLeod and MacDonell, 2011; Osvalder and Ulfvengren, 2009; Rode, 2005; Rode and Rosson, 2003). It should also be clarified that the end-user population does not form one single mental category since various human factors tend to shape different end user 'behavioral groups'.

However, the EUD research field, as a subfield of HCI has not comprehensively 'exploited' these end-user theories to build appropriate user models and develop adaptive EUD environments (in contrast to other HCI fields such as Learning Management Systems) in order to assist end-user developers build 'better' software. This 'lack' motivated our interest to construct a behavioral user modeling approach that can be used as base line for the future implementation of self-adaptive EUD systems. Our study is also a first attempt to gather and classify the behavioral attributes that tend to influence the end-user developers' performance when working in EUD environments.

Aggregating though all the behavioral attributes referred in the distinct HCI literature works would be 'exhausting' and probably inappropriate and misleading for the particular population of end-user developers, since we consider them be a special subgroup of the generic end-user population. Such a generalized approach would also lead to a user model structure composed of too many parameters, impractical to be implemented in the EUD modeling mechanisms. Hence, we had to restrict our research to the behavioral factors (and the attributes deduced by them) that are important to

the end-user developers and can be 'measured' and identified in their behavior while interacting with EUD environments.

In a combined research study of the most dominant HCI behavioral theories and the relatively new study field of Gender HCI (which is mainly focused on the end-user programmers/developers behavior) we could not but notice that Gender is one EUD behavioral factor of particular importance: recent research has shown that gender is a very strong and determinant factor to the end-users' overall development performance (Beckwith et al., 2005; 2006; Beckwith and Burnett, 2004; 2007). Moreover, many times it determines the end-users choices, debugging strategies, motivations and even the users effort and generic perception while their computer interaction (Burnett et al., 2011; Kulenza et al., 2009). Our combined literature research shows that gender also influences many (if not most) of the behavioral attributes (e.g. self-efficacy, ease-of-use perception, etc.) that are contributing to the end-users' final performance.

Hence our challenge is to design and propose an end-user model based on the reviewed gender-oriented attributes.

3.5 The proposed approach: RULES - A Gender based Behavioral User Model

Social, biological or other reasons that make men and women think or work differently, we do not seek to analyse them, but we do step on these differences and consider gender as a de-facto strong factor influencing the end-users' "computer-personality" and hence their task-developing behavior. In this context, this section uses (as its main parameters) the previously aggregated gender-influenced behavioral attributes existing in the end-user developers' population to build the structure of EUD behavioral user models.

Our suggested model (RULES) can be regarded as the very first step to create end-user developers' behavioral profiles, so as EUD self-adaptive tools can be based on them and assist the end-users in their developing activities.

3.5.1 Presentation

Based on the literature analysis of the previous sections, we propose RULES as an attempt to form a behavioral model for the EUD users. Our suggested approach uses a set of the attributes presented in Table 3.1 in an attempt to define the structure of behavioral end-user profiles, suitable to be used during the user modeling implementations in EUD environments.

Hence, we propose a model that provides a user profile formation approach consisted of five behavioral attributes. As it is explained below, these five attributes can collectively reveal the users' "developing task personality", shedding light on their strong and weak points concerning their overall EUD task performance. Based on the users' gender and behavioral profile (i.e. the five attributes-based user model), the system can implement a decision making mechanism for appropriate adaptation.

Instead of using all of the afore-presented attributes (in Table 3.1) for our model formation, we attempt to exclude some of them, to create a clear and simple model structure, composed of particular attributes (functioning as the model's parameters) that can be easily defined and detected by the EUD system behavior-observation (or monitoring) mechanisms.

The first attribute we exclude is Curiosity. According to Loewenstein, "curiosity arises when one's informational reference points in a particular domain become elevated above one's current level of knowledge", where the "informational reference point" is what one wants to know. Curiosity as a behavioral attribute was the backbone in the development of the surprise-explain-reward strategy (Robertson et al., 2004; Ruthruff et al., 2004; Wilson et al., 2003) an approach, aimed at changing end-user developers' perceptions of risk and reward. The development and success of the surprise-explain-reward strategy relies on raising a user's curiosity to an ideal level, such that he/she becomes aware of missing knowledge, but perceives it as attainable. Research into curiosity indicates that surprising a user to arise his/her curiosity can render him/her search for an explanation, but curiosity needs self-efficacy in order to be expressed by the user.

Moreover, according to Loewenstein's (1994) information gap theory, a user needs to have a certain level of self-efficacy in order to reach a useful level of curiosity. This curiosity will leverage the levels of their exploratory behavior (tinkering) and willingness to learn, enhancing at last their performance (Burnett et al., 2011). Hence, we decided to exclude Curiosity mainly because it is tightly associated to Tinkering and Self-Efficacy (Beckwith et al., 2006; Grigorenau et al., 2008) and Tinkering is much easier to be detected in a user's behavior, a user for instance that chooses new features, searches through the menu items and moves backwards after completing new actions, suggest Curiosity be totally substituted by Tinkering. Self-Efficacy is also easy to be measured through the user mouse movements (Lee and Chen, 2007; Ferreira et al., 2010).

Moreover, curiosity could be hidden behind Learning-Willingness actions and Learning-Willingness can be easily detected in the user's behavior (e.g. through instructions reading, video- tutorials viewing, searching on the web, using new features, etc.). Learning-Willingness is an important EUD behavioral attribute that affects the end-users' overall performance since it reveals the user's motivation power, determines the amount of effort the user makes and the perspectives of his/her future performance enhancement. Willingness-To-Learn is important because it can 'predict' the end-user's willingness to try, persist, tinker and even study to learn how to use new features and EUD technologies (Burnett et al., 2010; Grigorenau et al., 2008).

Perceived-Playfulness could be associated to Tinkering (exploratory behavior), hence we also exclude it. In fact, what 'Perceived-Playfulness' or 'Enjoyness' means in other theories such as the Technology Acceptance Model, can be totally replaced by the term of Tinkering in the area of EUD.

We also exclude the attributes of Overconfidence and Tinkering since they can both be determined by Risk-Perception. For instance, high Risk-Perception can lead to low Tinkering and vice versa (Kim, 2010; Saadé et al., 2012; Terzis and Economides, 2011). Moreover, as already explained, tinkering is based on curiosity (Burnett et al., 2011; Scaffidi et al., 2010) which needs high levels of self-efficacy to be triggered (Loewenstein, 1994). Hence Tinkering cannot exist without self-efficacy.

As regards to Overconfidence, based on the studied literature, Overconfidence leads to errors and low performance but high Risk-Perception can lead to control mood and double checking of the user actions, avoiding speed and other confidence related errors. Since high levels of Risk-Perception could possibly eliminate the errors made by Overconfidence, we decide to keep in our model only one of these two attributes. Additionally, while implementing tracking methodologies to monitor user behavior (e.g. Atterer and Lorenzi, 2008; ClickTale, 2010; Ferreira et al., 2010), overconfidence can many times be confused with high self-efficacy. Although self-efficacy regards the user's confidence on their own skills not revealing 'superficiality', overconfidence related attitude can many times be reflected in extremely high levels of self-efficacy. For all these reasons we decided to exclude Overconfidence and Tinkering and to 'keep' Risk-Perception and Self-Efficacy.

In Table 3.2 we summarize the inclusion and exclusion of the above discussed variables (i.e. the behavioral attributes) in our model. Additionally, for the excluded attributes we present their linkage to other variables, as well as a brief justification for their exclusion based on the reviewed literature foundations.

Table 3. 2 Behavioral attributes' inclusion and exclusion

Behavioral Attribute	**Included**	**Closely Related to**	**Brief Justification**
Self-Efficacy (or Computer Self-Efficacy	YES	-	-
Overconfidence	NO	Risk-Perception	Overconfidence can be associated to perceived risk levels, i.e. too low Risk Perception could lead to high Overconfidence. Risk Perception values can be estimated (in one dimension) based on Overconfidence levels (among others). Overconfidence can many times be confused to high Self-Efficacy.
Curiosity	NO	Tinkering, Self-Efficacy	Curiosity can many times be expressed through Tinkering behavior, and Curiosity needs high levels of Self-Efficacy to be

			expressed.
Tinkering (or Exploring Behavior)	NO	Risk-Perception	Tinkering behavior is rationally associated to perceived risk level, i.e. high Tinkering reveals low Risk Perception (contrary to Overconfidence). Levels of Risk Perception can be detected in the user's Tinkering actions (among others). Tinkering needs Curiosity to be expressed and curiosity needs Self-Efficacy. Hence Tinkering is highly dependent to self-efficacy.
Learning Willingness	YES	-	-
Risk-Perception	YES	-	-
Usefulness-Perception	YES	-	-
Ease of Use-Perception	YES	-	-
Perceived-Playfulness (or Playful-Behavior)	NO	Tinkering	Perceived-Playfulness can be regarded as Playful-Behavior that can be reflected in Tinkering behavior. Productive Tinkering can express enjoyness and interest that can positively affect the users' performance.

Hence, Learning Willingness, Self-Efficacy and Risk Perception are determinant attributes for the end-users' performance and are prominent in the very specific field of EUD. The attributes of Ease-of-Use and Usefulness Perception, which are included in the RULES model, are prominent in the whole HCI area and every usability and user experience related study. Ease-of-Use and Usefulness Perception are very important for the EUD user modeling since ease of use and usefulness are key elements in user-centered design and they can provide a range of adaptation elements that could be 'offered' to the users in order to assist them enhance their developing performance.

The remaining attributes are being excluded due to strong association to the basic ones as explained in Table 3.2, since most of them tend to function as extraneous or mediating variables. Contrary, the basic attributes that form our model, tend to directly influence the end-user performance.

Our suggested User Model (UM) can be depicted as following:

$$UM = \begin{bmatrix} \textbf{R}\text{isk Perception} \\ \textbf{U}\text{sefulness Perception} \\ \textbf{L}\text{earning Willingness} \\ \textbf{E}\text{ase of Use Perception} \\ \textbf{S}\text{elf-Efficacy} \end{bmatrix}$$

Due to the fact that our final model is composed of five behavioral attributes we named it 'The RULES Model', and RULES is the acronym of the five most significant attributes of the previous vector (**R**isk Perception, **U**sefulness Perception, **L**earning Willingness, **E**ase of Use Perception, and **S**elf-Efficacy).

The remaining attributes (in Table 3.1) should not necessarily be omitted but that could play a secondary role in the formation of basic behavioral user profiles in EUD environments.

3.5.2 Evaluation

RULES Model has been evaluated via an experimental field test described in Tzafilkou et al. (2016). As the results showed, RULES attributes are important for EUD environments since they tend to affect performance differently for male and female end-user developers. Also, RULES attributes' bi-correlations reveal gender differences in terms of strength and direction.

3.5.3 Example Application

Following we present a sample user scenario in order to better explain the usefulness of the suggested behavioral user modeling approach.

Let's assume that a EUD system aims to provide personalized services for each user. To this end, the system needs to create a user model (UM) for each individual user so as to adapt accordingly. If the system is based on the proposed model, then the user model will be a vector including the five RULES attributes

The system needs to assign specific values for each of these attributes in the user model of each individual user. This can be done through a number of ways, either automatically (i.e. the system is monitoring usage and extracts this information) or manually (e.g. through a questionnaire that is presented to the user the first that that she/he uses the system). We do not analyses further this issue since it is beyond the scope of this paper, and since adaptive systems literature includes numerous solutions for this issue since it is central to every system that aims to support adaptations.

For instance, let's assume that the system has observed the behavior of a specific user and has created the following user model:

$$UM = \begin{bmatrix} R: LOW \\ U: HIGH \\ L: LOW \\ E: HIGH \\ S: HIGH \end{bmatrix}$$

These values could be used by the systems decision making components to decide on the adequate adaptation responses. In particular, the system can provide different adaptation states based on the user's gender and the values assigned to their model's behavioral attributes. That means that female and male end-users may have the same

attributes values but the system adaptation responses will not always be provided in a universal manner: many times different adaptation approaches are suitable for male and female users even if they reveal same behavior (i.e. same attribute value).

For instance:

Low Risk-Perception means that the user 'dares' to try new features and tinker a lot, but could also mean low task consciousness leading to an error prone behavior (usual to male users). Knowing that user's Risk-Perception is low, the system by combining this information to the user's high Tinkering (which is a lot influenced by Risk-Perception) it should find ways to elevate the user's Risk-Perception if his/her Tinkering is negative (i.e. repeatingI) and to keep it stable if Tinkering is positive (i.e. exploratory- see Table 3.1). Elevation of Risk-Perception could be achieved by the system by providing 'what if' tools that could highlight the negative effects that some choices could have, rendering the user more careful and conscious. On the other hand, high Risk-Perception renders users less likely to make use of unfamiliar features. Risk-based adaptations have been proposed by Beckwith et al. (2005) such as the 'advice' component that by balancing the quality and quantity of provided explanations, it tends to decrease the users perceptions of risk.

High Usefulness-Perception reveals that the user believes that specific software is useful for his/her job performance. This leverages the user's interest and makes him/her try harder and leverage his/her performance. Knowing that Usefulness-Perception assists user to perform well, the system should find ways to keep its 'usefulness' in the same level throughout the whole lifecycle of the user's task. It could for example provide the user with system use cases and example implementations so that the user could read them and associate his/her work to the one of the given examples. It could also provide the user a sandbox version where the user could test his/her artifacts under real conditions to better perceive the system's usefulness. Moreover, since usefulness-perception tends to affect more male users (Ong and Lai 2006; Venkatesh et al., 2000), the system could provide with 'stronger' adaptations (such as many control elements so that user can view their 'progress' and realize the system's usefulness to achieve their goals) in cases of male and-users.

Low Learning Willingness reveals the user's unwillingness to use any new features and spend time on learning how to use them. Hence, the system is well aware of the fact that the user will neither read any tutorial nor any long instruction that the system offers for his/her assistance. Thus, the system should be adapted in a way that the user will not be 'faced' with many new features at once, but he/she could learn to use them progressively (gentle slope) if for example be provided with very short but comprehensive explanations. Moreover, the Surprise-Explain-Reward strategy could be implemented also in the case of low Willingness-to-Learn in order to increase the user's curiosity on specific features and motivate him/her to use them. This could be accomplished with the usage of animation or different colors that highlight the 'new' features.

High Ease-Of-Use-Perception could imply that the user is familiar with this environment style, has some significant experience or that the system is 'perfectly' designed in user-centered way. Knowing that Ease-Of-Use-Perception assists user to perform well, the system should find ways to keep its 'friendliness' in the same level throughout the whole lifecycle of the user's task. That is, the system should try not to change the user interface while the adaptation since the user seems to find it convenient. In the opposite case, the system should adopt a more novice-friendly 'design', to help the user perceive the whole environment as 'easier'. For example more wizard-like entities could be used, since novice users (or user behave as novices) and especially female users tend to reveal strong wizard preference (Beckwith et al., 2005; Burnett et al., 2010). Indeed, the minimalist learning theory (Carroll, 1998) suggests that new system features be introduced by engaging users in activity and providing scaffolding to help them gradually increase their skills.

Other adaptations for novice and expert users can be found in many recent works (such as in Jason et al., 2010; Eachus and Cassidi, 2006). Moreover, based on previous research works (e.g. Kim, 2010; Ong and Lai, 2006; Venkatesh et al., 2003) perceived ease of use is more important for female users since men are more familiar than women towards computer use. So, the system's adaptation responses should be more 'rigorous' and 'strong' for the female end-users.

High Self-Efficacy is also the desirable user behavior concerning the particular attribute. However, in case of low Self-Efficacy the system should find ways to assist the user in his/her task performance, since low Self-Efficacy is strongly tied to bad performance. Many research works (e.g. Beckwith et al 2005; 2006; 2007; Blackwell and Morrison 2010; Ko et al., 2011) have proposed ways to Self-Efficacy related adaptations, such as use of WYSIWYT (What you Use Is What You Test) editors and even supporting videos that tend to have positive self-efficacy results for the female end-users. For male users the system could provide with debugging and explanatory elements, e.g. similar to the 'Whyline' (Ko and Myers, 2004) approach, so that users could always realize their mistakes and their outcomes.

As we can see our approach could predict some basic parts of user's behavior based on the behavioral model he/she belongs to and hence to offer him/her some basic adaptation services. While the actual user-system interaction it is expected that the user's behavior may change (e.g. because of gaining system familiarity, or working on different task, etc.). Thus, we suggest that within the initial personalized environment the appropriate system mechanism should dynamically observe the user's behavior and readapt its behavior in a continuous loop in order to completely adjust to the user's current behavior. However the presentation of this mechanism is out of the scope of this paper.

3.6 Summary and Contribution of the Chapter

This chapter presents a first approach taken for a behavioral model construction, based on gender influenced behavioral attributes existing among the end-users who attempt to develop their own applications, i.e. end-users working in EUD environments (also refereed as end-user developers). Such an approach aims to contribute in the development of self-adaptive EUD tools that measuring these behavioral attributes will be able to implement relative user modeling techniques assisting the users in the developing task.

Our study contribution could be regarded as twofold, meaning that it is both review and proposal work, since it steps on the combined HCI and EUD review research to

aggregate all the parameters needed to propose a EUD oriented behavioral user model structure.

According to our approach:

- Past established behavioral HCI theories have contributed to the recently emerged behavioral research in the EUD area, shedding light on the factor of gender and stressing its importance.
- Gender behavioral attributes can be used as a stepping stone for the analysis of end-user behavior and the suggestion of particular end-user behavioral models.
- A behavioral model can be proposed based on a set of five behavioral attributes, since they tend to influence end-user developers overall performance and in a different mode for male and female end-users.
- The suggested model constitutes an initial approach in the design of specific EUD-centred user modeling techniques and their latter implementation in self-adaptive EUD system environments.

Our study is actually a first attempt to gather and classify the behavioral attributes (been studied in the HCI and EUD behavioral literature) influencing the end-user developers' performance when working in EUD environments. Then, stepping on these attributes, we propose the main structure of an end-user developer's behavioral user model.

Our work also provides experimental evaluation and assessment of the proposed argument. Except analysing the linkage between the HCI theories foundations and our EUD oriented modeling approach we also conducted a real-world behavioral EUD experiment on a sample of end-users.

Hopefully, our approach is intended to shed light on the necessity of such EUD behavioral modeling adaptations and to encourage relative future work.

Unfortunately behavioral EUD analysis is rare in the HCI research community works, and our initial review research is limited on current resources (mainly the Gender HCI field and the presented HCI Theories) which may not be sufficient for a complete and totally objective end-user developers' behavioral 'image'. Thus, we shall not forget

that for the optimal evaluation of our work further behavioral research need to be conducted regarding the EUD area. We are currently studying more factors, complementary to gender, that affect end-user performance and differentiate their behavior, such as expertise or familiarity level of the end-user with EUD environments and programming (or database related) theoretical concepts. We plan to include these factors in our research and conduct more behavioral EUD experiments on larger group samples to possible expand our RULES modeling approach. Our study strongly encourages such scientific efforts.

CHAPTER 4

EYE TRACKING IN END-USER DEVELOPMENT

In chapter 3 we demonstrated the role of gender in HCI and we explained the reasons for being taken under consideration in the design of EUD tools. Also, we introduced the RULES model, a set of behavioral attributes that should be included in the structure of end-user developer models. In this chapter we will investigate the role of eye behavior as implicit feedback in EUD environments. This feedback, along with mouse and keyboard feedback presented in chapter 5 will be used for the construction of our final user modeling framework. The main contributions of this chapter have been published in in Tzafilkou and Protogeros (2017a) and in Tzafilkou et al. (2017).

4.1 Introduction

End-user behavioral attributes such as perception and acceptance have been outlined by many studies (e.g. Beckwith, et al, 2006; 2007; Beckwith et al., 2005; Burnett, 2009, Burnett et al., 2008; 2010; 2011; Beckwith and Burnett, 2007; Chen and Corkindale, 2008; Cyr et al., 2007; Lee, 2008; Sun and Zhang, 2008) as important HCI human factors and have been shown to affect user performance in EUD tasks. Hence, the analysis of end-users' perception and acceptance is crucial to design tools and methodologies that assist end-users to enhance their developing performance. Towards this direction, it is important for the EUD community to design and implement new approaches in order to capture and analyze the end-users' behavioral attributes, including perception and acceptance items, while interacting with today's EUD environments.

Mouse and eye tracking methodologies are ideal to implicitly monitor the end-user behavior in real time. Especially eye tracking, can provide with cognitive and psychological user data, helpful to deeply analyze the users' behavior and mental states during the entire cycle of user-system interaction (e.g. Ball et al., 2003; Just and Carpenter, 1976; Yoon and Narayanan, 2004).

In HCI research, analysis of eye movement data has been mainly studied to evaluate usability issues since eye tracking represents an objective technique that can offer useful advantages for the in-depth analysis of interface usability (Poole and Ball, 2005). Unfortunately, despite its important HCI contribution eye tracking has not been widely integrated in the behavioral research area of EUD.

Triggered by the above described, in the current research part we will examine whether end-users' eye behavior is associated with most of RULES behavioral attributes (presented in chapter 3) including the perception and acceptance items of perceived ease of use, perceived usefulness, self-efficacy and risk perception.

The contribution of this chapter is to support the use of eye tracking methodologies in cloud and/or web-based EUD environments, to implicitly gather end-user feedback and use it for user modeling implementations.

4.2 Literature Background

4.2.1 Eye Tracking in Human Computer Interaction

Eye tracking has generated a great amount of interest in HCI research since the beginning of the twenty-first century when eye tracking technology became more widely accessible. Today, eye tracking is frequently employed to help evaluate and improve designs at various stages of the development cycle (Bojko, 2013).

According to Poole and Ball (2005), eye-movement tracking represents an important, objective technique that can afford useful advantages for the in-depth analysis of interface usability. Also, eye movements can provide a window onto many aspects of user cognition and human factors especially on problem solving, reasoning, mental imagery, and search strategies (e.g. Ball et al., 2003; Just and Carpenter, 1976; Yoon and Narayanan, 2004).

According to Just and Carpenter (1976), what a person is looking at is assumed to indicate the thought "on top of the stack" of cognitive processes. This "eye-mind" hypothesis means that eye-movement recordings can provide a dynamic trace of where a person's attention is being directed in relation to a visual display. Measuring other aspects of eye movements, such as fixations (moments when the eyes are

relatively stationary, taking in or “encoding” information) can also reveal the amount of processing being applied to objects at the point-of-regard. In fact, common eye metrics such as fixation duration and pupil size (diameter) have been correlated to perceived web page relevance (Gwizdka and Zheng, 2015).

In practice the process of inferring useful information from eye-movement recordings, involves the HCI researcher defining Areas of Interest (AOIs) over certain parts of a display or interface under evaluation, and analyzing the eye movements which fall within such areas. In this way, the visibility, meaningfulness and placement of specific interface elements can be objectively evaluated and the resulting findings can be used to improve the design of the interface (Goldberg and Kotval, 1999).

Although traditional HCI eye tracking research was focused on measuring usability and interface design issues, recent HCI eye tracking research also focuses on measuring the overall user-experience (UX). User experience expands usability in a sense that it studies not only the interface efficiency but also the end-user from a human side. That is, eye tracking can also measure the end-user’s perceived items, such as perceived ease of use, hesitation, perceived playfulness, cognitive and mental load, etc. and thus it can provide with a basic knowledge of the users’ internal states while interacting with a computer system.

The main measurements used in eye tracking research are fixations and saccades. Saccades are quick eye movements occurring between fixations. Although there are a few more types of eye movements, saccadic eye movements, consisting of saccades and fixations, are most common to user experience research. Fixation is a metric of great interest in HCI and UX because even though eye tracking only captures foveal vision, it provides useful information about visual attention because, in most cases fixation coincides attention (Bojko, 2013). There are also a multitude of derived metrics that stem from these basic measures, including “gaze” and “scanpath” measurements.

Pupil size and blink rate are also studied. For instance, the metric of pupil dilation has been associated with mental effort and attention (Onorati et al., 2013). Suggesting eye movement as a cognitive load in HCI, Chen and Epps (2014b) state that end-users’

perceptual load should be considered in cognitive load measurement using pupil diameter and blink measure. Pupil diameter and blink rate have been associated to cognitive load in many HCI research studies. For example they were also studied for task analysis to improve HCI (Chen et al., 2014a; 2013; Haapalainen et al., 2010; Iqbal et al., 2005) and for constructing effective and user-personalized training (Chen et al., 2011). Oliveira et al. (2009) and Gwizdka (2014) found that in text documents and images pupil dilated for more relevant stimuli. Furthermore, pupil diameter and/or blink measures have been found useful to the system usability (Nakayama and Katsukura, 2007; Kozsa, 2011). Chen and Epps (2014a) encourage eye tracking research in HCI to infer cognitive load, since as they suggest "knowing the type and level of load that is generating user mental effort will benefit the diagnosis and remedy formulation processes in human-centered design".

4.2.2 Eye Movements and End-User Performance

HCI research has shown that studying human behaviors can provide insight into human performance. One class of behavior that is of interest within HCI research is eye movements (Cuddihi et al., 2005). Therefore, techniques for studying eye movement behavior have been considered as effective means of HCI (Harper, 2015).

However, while eye tracking in HCI has proven useful for addressing the question of how users perform tasks, there have been limited research attempts to address the question of how well users perform tasks (Harper, 2015).

The eye tracking and performance research argument is grounded in prior research that examines the relationship between cognitive load and performance. Previous studies have shown the importance and relevance of eye fixation data as a useful measure of cognitive load (e.g. Rayner, 1998). Cognitive load theory (CLT) describes the limitations induced by working memory (Wang et al., 2014). According to Sweller (1988) people's cognitive capacities are so limited that they can process only limited information chunks concurrently. If the information to be processed exceeds a limit, people are overwhelmed.

The relationship between cognitive load and performance has been studied in many different studies (e.g. Luce, 1986; Payne et al., 1988; Schroder et al., 1967) as it shows to affect flexibility in adapting to a decision environment, response time, information processing capability, etc. These studies show that cognitive load can have a significant impact on human performance. Literature also infers that cognitive load can have an influence on users' visual attention and behavior. A known way to assess the impact of cognitive load is by using subjective measures (Rubio et al., 2004).

Confirming the above arguments, eye tracking studies suggest that cognitive load can impact eye fixation. In particular, Ikehara and Crosby (2005) have shown that cognitive load can affect eye movement. Also, Djamasbi et al. (2011) suggest that fixations can serve as a suitable measure of cognitive load. Their research findings showed that the number of fixations on the pages that were more cognitively demanding was significantly larger than the number of fixations on the pages that were less cognitively demanding. In the eye tracking study of Wang et al. (2014) the research results showed that users' attention is easily distracted as website complexity increases, which leads to more fixation count (i.e. larger number of fixations).

However, the existing studies show that cognitive load can affect performance and fixation, they have not examined whether there is a direct association between fixation and performance. Eye movements is a significant measure to evaluate performance since they are assumed to have a predictable relationship with the covert cognitive processes associated with visual attention which are not thoroughly understood (Harper, 2015). According to Harper (2015), as a result of this relationship, eye-tracking measures, such as total number of fixations, gaze durations, and scan paths can provide detailed information about how users perform tasks. And this type of information would be difficult (if not impossible) to collect using other HCI methods.

In Djamasbi et al. (2012) the authors explored the various possible methods to interpret the relationship between cognitive load, performance and fixation. Their results showed a significant relationship between fixation count and performance. The higher fixation count was correlated with higher number of moves or worse performance.

The Eye Tracking-Performance Connection (EPC) perspective is a novel use of eye tracking in HCI. In contrast to the HCI/Eye Tracking (HET) approach, the goal of an eye-tracking experiment using an EPC design is to use the eye-tracking record to predict the likely success of a user performing a task well (Harper, 2015). In his research, Harper (2015) collects a set of performance and eye-tracking data to examine correlations between eye movement metrics and subject's performance scores. In this way, a researcher can verify that an eye-tracking metric can indeed be found which relates to a user's success at a particular task. Additionally, Harper applies machine-learning techniques to predict user performance based solely upon eye movement features.

Although EUD activities are considered cognitively demanding since they 'require' from non-expert end-user developers to develop their own artifacts without getting trained, eye tracking has not been widely adopted in end-user cognitive and behavioral EUD research. In fact, end-user performance evaluation/prediction has not been examined through eye tracking methodologies in recent web-based EUD environments.

4.2.3 Eye Tracking Research (Gap) in EUD, Perception and Acceptance

Although its UX and usability-centered HCI implementations, eye tracking has not been widely adopted in end-user behavioral or in other EUD research areas. In particular, end-user behavior, perception and acceptance have not been examined through eye tracking methodologies in current web-based EUD environments. The research described in this paper differs from the efforts above because it attempts to integrate the eye tracking methodology in EUD research and to provide with existing and/or new eye metrics to measure or deeper understand the end-users' behavior while performing developing tasks.

Eye tracking has not been widely used to detect and analyze the end-users' perception and acceptance attributes, such as Self-Efficacy, Risk-Perception, Perceived Ease of Use and Perceived Usefulness, which are important to the end-users' developing performance according to the EUD literature.

Regarding perceived self-efficacy, limited research works have been found to use eye tracking as a self-efficacy measure. In particular, in Eachus et al. (2008) the authors have correlated eye tracking data with Internet self-efficacy

Regarding risk-perception, no research works have been found to use eye tracking as a risk-perception or hesitation measure. No possible correlations are known between eye movements and hesitation or risk levels while end-users interact with EUD or other software environments.

Regarding perceived usefulness and ease of use, most eye-tracking HCI research focuses on measuring usability and user experience. Yet no eye tracking research has been conducted to measure perceived usefulness and ease of use on web-based EUD environments.

However, there are a lot of user experience eye metrics in HCI that can also be used to examine user interaction with EUD environments, since according to Bojko (2013), the same measures can represent different cognitive phenomena in the context of different stimuli and goals. Thus, the lack of EUD-targeted eye metrics prompted us to analyzing existing eye-metrics (from user-experience or other research area) in the EUD scope. Based on Bojko's (2013) methodological work on how to conduct eye tracking research for measuring the user experience we have produced an aggregated list of eye tracking (fixation and pupil) metric categories and metrics as shown in Figure 4.1, many of which could be useful for our web-based EUD-oriented research as well.

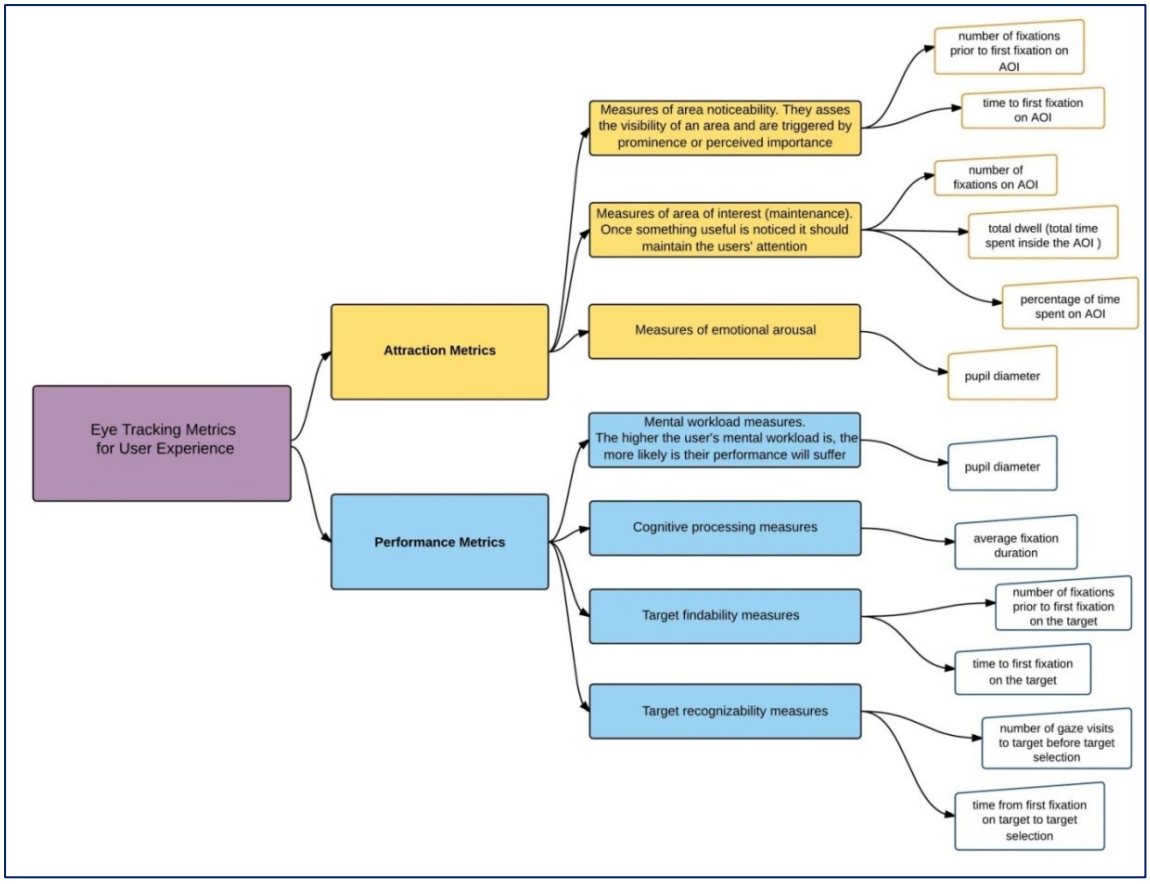

Figure 4. 1 List of eye user experience metrics according to the taxonomy of Bojko (2013)

4.3 Literature Review Concluding Remarks and Challenge

Eye tracking methodologies are ideal to implicitly monitor the end-user behavior in real time. Eye tracking can provide with cognitive and psychological user data, helpful to deeply analyze the users' behavior and internal states during the entire cycle of user-system interaction. In HCI research, analysis of eye movement data has been mainly studied to evaluate usability issues since eye-movement tracking represents an objective technique that can offer useful advantages for the in-depth analysis of interface usability. Unfortunately, despite its important HCI contribution eye tracking has not been widely integrated in the behavioral research area of EUD.

The lack of such useful research works and the fact that eye tracking can be easily implemented as an automatic method requiring limited user's input or self-reporting, triggered our interest to use eye tracking to examine the potential correlation between eye movements and end-user perception and acceptance in nowadays' web-based EUD environments. To this end our main objective is to study whether end-users'

behavioral (in particular perception and acceptance) attributes can be reflected on their eye behavior while interacting with web-based EUD environments.

The main research questions of the current chapter are the following:

Are there any significant correlations between eye movements and perception and acceptance items for end-users of similar expertise level, when interacting with a web-based, EUD system?

Are there any significant correlations between eye movements and user performance for end-users of similar expertise level, in a web-based EUD system?

In a more general sense, we aim to examine whether eye tracking can be considered a useful means towards diagnosing user behavioral characteristics and performance while interacting with web-based EUD environments.

4.4 The proposed approach: Eye Behavior as Implicit Feedback in End-User Development

Our aim is to investigate the usefulness of eye-tracking as implicit feedback in web and/or cloud based EUD environments, in order to include eye metrics in our final user modeling implementation. Hence, we conducted an experiment to examine potential correlations between eye movements and the five EUD behavioral attributes suggested by the RULES model in chapter 3. Also, triggered by the eye-performance literature review, we examined the correlation between eye behavior and user performance as well.

Because of the limited relative previous eye tracking research, we are 'inspired' from user experience metrics and mouse tracking patterns to establish suitable eye tracking measures in our EUD-oriented behavioral research.

To conduct the study we connected a commercial eye tracking software to the wizard-based prototype web EUD tool presented in chapter 2. In this section we first present the eye tracking technology we used to conduct our EUD laboratory test and then we present the laboratory test methodology and results. The study results are encouraging for our final user modeling implementation (presented in chapter 6) since

several eye metrics seem useful to be included in our behavioral user input components.

4.4.1 Eye Tracking Technology

The eye tracking system used in our experiment was the Tobii eye tracker, a commercial platform for the recording and analysis of eye gaze data. Tobii Studio offers a comprehensive platform for the recording and analysis of eye gaze data, facilitating the interpretation of human behavior. Tobii supports a large variety of stimuli and allows both multiple and different stimuli types to be combined into a single recording (Tobii, 2015; 2011).
The selected hardware was composed of a Tobii monitor with an integrated eye tracker device in the front side.

Tobii Studio organizes and stores information on three hierarchical levels: Projects, Tests, and Recordings. Project level contains participant data and one or more tests. Each test created within a project contains one or more media elements (the stimuli), which are organized into a single linear timeline and recordings (Tobii, 2015).

During a recording, the Eye Tracker collects raw eye movement data points every 3.3 to 33 ms (depending on the sampling data rate of the eye tracker). Each data point is identified with a timestamp and “X, Y” coordinates. In order to visualize the data, these coordinates are further processed into fixations and overlaid on a video recording of the stimuli used in the test or used to calculate eye tracking metrics.

In the following we explain how fixations are calculated:

The Tobii Fixation Filter is an implementation of a classification algorithm proposed by Olsson (2007). The algorithm detects quick changes in the gaze point using a sliding window averaging method. Assuming that the eyes move between two different fixations, the following distinctions are made:

- If a segment of the signal is of constant or slowly changing mean value due to drift, we classify it as one fixation.

- If there is an abrupt change in the signal, the eyes have moved to another fixation location.

Another algorithm produces notifications whenever it detects a change in the mean, hence data can be separated into different fixations.

Tobii can be adjusted to calculate and export the AOI dependent eye metrics. This adjustment facilitates the analysis procedure since the system provides with 'ready for analysis' metrics for every participant. The main calculated eye metrics are:

- Time to First Fixation: how long in seconds it takes before a test participant fixates on an active AOI or AOI group for the first time;
- Fixations Before: the number of times the participant fixates on the media before fixating on an AOI or AOI group for the first time;
- First Fixation Duration: the duration in seconds of the first fixation on an AOI or an AOI group;
- Fixation Duration: the duration of each individual fixation within an AOI;
- Total Fixation Duration: the sum of the duration for all fixations within an AOI;
- Fixation Count: the number of times the participant fixates on an AOI or an AOI group;
- Mouse Click Count: the number of times the participant left-clicks with the mouse on an AOI or an AOI group;
- Time to First Mouse Click: how long in seconds it takes before a test participant left-clicks with the mouse on an AOI or AOI group for the first time;
- Time from First Fixation to Next Mouse Click: the time from the first fixation within an active AOI or AOI group until the participant left-clicks within the same active AOI or AOI group.

However, in our experiment we only exported row data since we do not need any AOIs definition. The gaze data received from the eye tracker consists of gaze point coordinates, distance to the eye tracker, pupil size, etc., for the left and right eye separately. The main unprocessed eye gaze data are listed in the following table.

Table 4. 1 Exported eye tracked data types -Source (Tobii, 2011)

Eye/gaze Data	Description	Format
EyePosLeftX (ADCSmm)	Horizontal coordinate of the 3D position of the left eye.	Millimeters
EyePosLeftY (ADCSmm)	Vertical coordinate of the 3D position of the left eye.	Millimeters
EyePosLeftZ (ADCSmm)	Distance/depth coordinates of the 3D position of the left eye.	Millimeters
EyePosRightX (ADCSmm)	Horizontal coordinate of the 3D position of the right eye.	Millimeters
EyePosRightY (ADCSmm)	Vertical coordinate of the 3D position of the right eye.	Millimeters
EyePosRightZ (ADCSmm)	Distance/depth coordinates of the 3D position of the right eye.	Millimeters
DistanceLeft	Distance between the left eye and the eye tracker. The distance is calculated based on the length of the vector from the center of the eye to the User Coordinate System (UCS) origin point on the eye tracker.	Millimeters
DistanceRight	Distance between the right eye and the eye tracker. The distance is calculated based on the length of the vector from the center of the eye to the UCS origin point on the eye tracker.	Millimeters
PupilLeft	Estimated size of the left eye pupil. The algorithms take into account the magnification effect given by the spherical cornea as well as the distance to the eye.	Millimeters
PupilRight	Estimated size of the right eye pupil. The algorithms take into account the magnification effect given by the spherical cornea as well as the distance to the eye.	Millimeters
ValidityLeft	Indicates the confidence level that the left eye has been correctly identified. The values range from 0 (high confidence) to 4 (eye not found).	0;1;2;3;4
ValidityRight	Indicates the confidence level that the right eye has been correctly identified. The values range from 0 (high confidence) to 4 (eye not found).	0;1;2;3;4
GazePointY (ADCSpx)	Vertical coordinate of the averaged left and right eye gaze point on the screen.	Pixels
GazePointX (MCSpx)	Horizontal coordinate of the averaged left and right eye gaze point on the media element.	Pixels
GazePointY (MCSpx)	Vertical coordinate of the averaged left and right eye gaze point on the media element.	Pixels
GazePointLeftX	Horizontal coordinate of the unprocessed gaze point for the left eye on the screen.	Millimeters
GazePointLeftY	Vertical coordinate of the unprocessed gaze point for the left eye on the screen.	Millimeters

GazePointRightX (ADCSmm)	Horizontal coordinate of the unprocessed gaze point for the right eye on the screen.	Millimeters
GazePointRightY (ADCSmm)	Vertical coordinate of the unprocessed gaze point for the right eye on the screen.	Millimeters
StrictAverageGazePointX (ADCSmm)	Horizontal coordinate of the averaged gaze point for both eyes on the screen.	Millimeters
StrictAverageGazePointY (ADCSmm)	Vertical coordinate of the averaged gaze point for both eyes on the screen.	Millimeters

Tobii also exports row gaze event data and activity information. This group of data types (most of them depicted in Table 4.2) provides data describing gaze events classified by the applied fixation filter.

Table 4. 2 Exported gaze event data and activity information -Source (Tobii, 2011)

Gaze event data and AOI activity information	**Description**	**Format**
FixationIndex	Represents the order in which a fixation event was recorded. The index is an auto-increment number starting with 1 (first gaze event detected).	Count
SaccadeIndex	Represents the order in which a saccade event was recorded.	Count
GazeEventType	Type of eye movement event classified by the fixation filter settings applied during the gaze data export.	Fixation; Saccade; Unclassified.
GazeEventDuration	Duration of an eye movement event	Milliseconds
FixationPointX (MCSpx)	Horizontal coordinate of the fixation point on the media.	Pixels
FixationPointY (MCSpx)	Vertical coordinate of the fixation point on the media.	Pixels
AOI [Name of AOI] Hit	Report whether the AOI is active and whether the fixation is located inside of the AOI	Empty;-1;0;1

SaccadicAmplitude	Distance in visual degrees between the previous fixation location and the current fixation location	Degrees

The software also exports key press and mouse events such as mouse click type, horizontal and vertical coordinates of the mouse event location on the screen, information of the key pressed on the keyboard, etc. Timestamp data (timestamp counted from the start of the recording, local time and the time obtained from the internal clock in the eye tracking system) and URL data (the URL of the webpage shown) are also included in the exported data.

Finally, in our experiment we also exported some Gaze Plots to visually analyze the end-users' behavior. The Gaze Plot visualization shows the sequence and position of fixations (dots) on a static media, (e.g. an image or a scene) or a dynamic media (e.g. a movie or a dynamic website). The size of the dots indicates the fixation duration and the numbers in the dots represent the order of the fixations.

4.4.2 Eye Tracking Experiment in End-User Development

4.4.2.1 Research Hypotheses

In this section we construct a set of research hypotheses as a basis to examine the potential correlations between user behavioral attributes (regarding acceptance and perception items) and eye behavior in EUD tasks.

However, since eye tracking in EUD is a new research area and there is no previous research methodology before formatting the hypotheses we should consider the following:

- A EUD system does not consist of static pages (like in most eye tracking studied environments); instead it is composed of many dynamic pages and in some cases of Single Page Applications (SPA). Thus we plan to measure the average eye-behavior during the time needed by a user to complete a development task. This time is different for every user and is defined as 'EUD task duration'. We consider that users can work on different tasks (pages, URLs) at the same time

(timestamp). So we could not compare users' behavior based on timestamps but based on common tasks (URLs).

- Since we will measure the user-side situation (perception and acceptance) and not the system interface design and usability, we will not integrate in our methodology any areas of interest (AOIs). For this reason we have excluded from our research most of the AOI-oriented eye metrics listed in Figure 4.1 and we will use only the most generic and user-oriented ones, i.e. the ones that are appropriate for testing the user 'internal situation' while interacting with a system and not the 'noticebility', 'findability' and attraction of specific AOIs in the user interface.
- Since there is a considerable correlation between eye and mouse movements (Chen et al., 2001), when eye related literature background is missing we suggest and use as EUD eye metrics some related mouse patterns that could also be reflected on eye behavior as we later explain.

In the current research the examined eye metrics have been mainly based on fixations. Psychology researchers have found that most of the information acquisition and cognitive processing occur during eye fixations. Moreover, they have showed that only a small set of fixations is required for participants to acquire and process a complex visual input (Privitera, 2000). Human eyes use fixations to focus on new targets or to extract finer details from a particular area of the scene (Harper, 2015).

The most commonly used fixation metrics are fixation duration and fixation count (or else number of fixations). Longer fixation duration indicates either difficulty in extracting information, or that the object is more engaging in some way. The number of fixations shows the total number of fixations on a given object. It is measured by counting the number of fixations on specific areas of interest or the whole stimulus (Sharafi et al., 2015). Higher number of fixations for the whole stimulus indicates less efficient search for finding relevant information (Goldberg, et al., 2003).

We also use pupil diameter since it tends to reveal mental effort (Bojko, 2013) and mouse clicks in combination to fixations to test our hypotheses.

Finally, task completion time is also taken into account since it is a measure of users' task performance. Less time usually indicates more efficient decision-making and better designed interface (Wang et al., 2014).

Eye measuring Self-Efficacy

Since eye tracking literature is not rich and there is no research on eye tracking self-efficacy in web-based multi-page EUD environments, we have considered the mouse-eye correlation showed in many research works (e.g. Chen et al., 2001; Guo and Agichtein, 2010; Huang et al., 2012;Navalpakkam et al., 2013; Rodden et al., 2008) and decided to use mouse behavioral patterns in order to define and test self-efficacy eye metrics for EUD environments.

Many researchers have concluded that the 'direct mouse movements' can determine self-efficacy levels. Defined as "straight pattern" (Lee and Chen, 2007) the users' 'confident movements' are characterized by a pause before a direct movement towards a target, since "once traced the desired feature (or link) users move the mouse straight to it". On presence of this pattern, researchers infer that there is an earlier decision and the user feels certain about his/her mouse movement. This use of mouse defines direct movements that occur once the user has decided which action to take (Rodden et al., 2008), and this undoubtedly reveals task-oriented self-efficacy. In the context of interaction with web applications, direct movement is characterized by "a direct movement toward a target whit no big pauses" (Ferreira et al., 2010).

In terms of eye tracking self-efficacy, the mouse pattern can be interpreted as following: the user needs first to detect the target with his eyes before moving the mouse straight towards it. The usefulness of the fixated target is determined by the user's decision to click or not to click on it. This logically means that a fixation must occur on every desired target before the mouse click. Hence, any fixation that turns into click could reveal the users' confidence that the fixation area is the desired one. For this reason our first hypothesis is:

H1.1: *Self-Efficacy is significantly related to the number of fixations that turned into clicks during the EUD task.*

According to Bojko (2013), once a desired target is noticed it should maintain the users' attention and this is expressed by the total number of fixations on these areas. Based on this, we assume that the more the fixations the higher the users' certainty that he/she has detected the desired/useful item. Hence, our second hypothesis is:
H1.2: *Self-Efficacy is significantly related to the total number of fixations during the EUD task.*

Eye measuring Risk-Perception

The most commonly used eye tracking measure of mental workload is pupil diameter. According to Bojko (2013) pupil gets larger with high processing demands. The author suggests that the higher the user's mental workload is, the more likely is their performance will suffer. According to Tsang and Wilson (1997), mental workload for a given task depends on the amount of effort a person dedicates to the task and their spare mental capacity. According to Tevel and Burns (2000) perceived-risk can be one important factor that contributes to mental workload. The authors showed that there is a relationship between subjective risk assessment and mental workload in HCI.

For this reason the next hypothesis is:

H2.1: *Risk-Perception is significantly related to the average increment of the pupil's diameter during the EUD task.*

Eye measuring Perceived Ease of Use

One of the measures of cognitive processing difficulty is the average fixation duration. According to Bojko (2013), longer fixations mean more effort to extract information and more difficulty in general. Since PEOU is defined by Davis (1989) as the degree to which a person believes that using the system would be free of effort, the next hypothesis is:
H3.1: *Perceived ease of use is significantly related to the number of fixations that turned into clicks during the EUD task.*

Exploring pupil diameter to index cognitive load has been an active research line in several application domains (Chen and Epps, 2014a). The difference with mental workload, mentioned in H2.1 is that cognitive load can be seen as the effort that users

use, whereas mental effort is the amount of cognitive resources that users actually devote in response to the load (Jong, 2009). Chen and Epps (2014a) showed in their cognitive load experiment that pupil diameter was larger during a high perceptual load task. The authors concluded that participants exerted more mental effort to deal with the high perceptual load task. For this reason we assume that in EUD environments where the users have to 'face' a developing task, high perceptual load can be reflected to low perceived ease of use. Hence, our next hypothesis is:

H3.2: *Perceived ease of use is significantly related to the average increment of the pupil's diameter during the EUD task.*

Eye measuring Perceived Usefulness

As already mentioned, longer fixations mean more effort to extract information. Hence, longer fixations reveal ambiguity and hesitation to take a specific action (i.e. to click on the fixated point). This also may reveal a difficulty to perceive a system as useful, since there are doubts on the predicted outcome/performance if the specific action (click) is completed. That could imply that the more the fixation duration over a specific point, the more the ambiguity on the performance outcome. Hence our last hypothesis is:

H4.1: *Perceived usefulness is significantly related to the average fixation duration during the EUD task.*

Eye measuring Performance

As explained in Djamasbi et al. (2011) fixations can serve as a suitable measure of cognitive load because in their experiment the number of fixations on the pages that were more cognitively demanding was significantly larger than the number of fixations on the pages that were less cognitively demanding. Also, in Djamasbi et al. (2012) the authors showed that there is a significant relationship between fixation number and performance. For this reason our first hypothesis is:

H5.1*: Number of fixations is significantly related to user performance in web EUD tasks.*

According to Bojko (2013), once a desired target is noticed it should maintain the users' attention and this is expressed by the total number of fixations on these areas.

Based on this, we assume that the more fixations that turned into clicks (i.e. the target was 'chosen') the higher the users' certainty that he/she has detected the desired/useful item and hence he/she performs well. For this reason our second hypothesis is:

H5.2: *Fixations that turned into clicks are significantly related to user performance in web EUD tasks.*

One of the measures of cognitive processing difficulty is the average fixation duration. According to Bojko (2013), longer fixations mean more effort to extract information and more cognitive load in general. Also, Djamasbi et al. (2012) results indicated that under high cognitive load, fixation duration was a good predictor of the users' self-reported cognitive load. However, in Wang et al. (2014), research results showed that there is no significant difference in fixation duration among websites with different levels of complexity.

The eye–mind assumption states that a participant fixates her attention on a stimulus until she understands it (Just and Carpenter, 1980) and this increases the average fixation duration time. Longer fixations show that participants spend more time analyzing and interpreting the content of the Areas of Interest AOIs or separate stimuli while working on their tasks. Therefore, they are spending more mental effort to solve their tasks (Sharafi et al., 2015).

As we see there are no clear research results concerning the correlation between fixation duration and performance, we will examine this correlation in our EUD field-test. Our third hypothesis is:

H5.3*: Fixation duration is significantly related to user performance in web EUD tasks.*

In our exploratory study we also plan to examine the relationship between EUD task completion time (task duration) and performance, since it is a measure of users' task performance (Wang et al., 2014) and greater completion time usually reveals higher cognitive load. Hence our last hypothesis is:

H5.4: *Task duration is significantly related to user performance in web EUD tasks.*

4.4.2.2 Evaluation Methodology

Participants and Procedure

There were 10 volunteers (3 male and 7 female) postgraduate students in the department of Information Systems in a Greek university. Their ages were between 25 and 30 years and their bachelor degrees were varied including Social and Political science, Economics, Business Administration, Education and Language. Eight (2 male and 6 female) participants (based on calibration validation testing) of the initial population successfully completed the tasks in three different days. Participants were tested individually and each one spent around 40 minutes to complete the EUD task.

All the participants had been informed that their eye movements would be recorded and they were asked to remove any eye makeup such as mascara and use lens in case they wear glasses. They were also asked to inform us on any serious eye-health issues they might have had.

The experiment design, including calibration testing, stimuli properties, head position and the participants' and environments' factors, was based on Tobii's test methods requirements (Tobii Technology, 2011). Additionally, since pupil diameter was decided to be measured and analyzed, lab moderators made sure that the lighting did not interfere with eye tracking.

To confirm the sample representation, i.e. the participants' similar level of computer and developing experience, an expertise pre-test was conducted. The experience level was measured in a scale from 1 to 5. In addition, some personal information (gender, age, educational background, eye-related health issues) was collected.

The measured mean value of the participants' database familiarity was 1,62, revealing that they could be 'safely' considered as non-professional/non-expert end-users in database-development tasks. Additionally, their programming experience was 1,87, their familiarity with web was 3,37 and their general familiarity with computer use was 3,12 (see Table 4.3). These mean values satisfy our target group (end-users) requirements, i.e. users that are non-experienced programmers, with no or limited knowledge on database concepts but with efficient familiarity with web interaction

and computer use in general. This information could allow us to generalize the results from a small but representative sample and make broad claims about web-based EUD behavior of similar EUD populations.

Table 4. 3 Participants' experience level

Measured Item	mean (0-5)	St. Deviation	St. Error
Database Experience	1,62	0,91	0,41
Programming Experience	1,87	1,18	0,54
Web Experience	3,37	0,92	0,32
Computer Use Experience	3,12	0,83	0,29

The participants were calibrated using a standard 9-point calibration (according to Tobii Technology, 2011) where the point was set to move between the positions at a slightly higher speed than normal. The calibration points were red dots with a central black dot shown on a neutral gray background. An automated calibration procedure was used, which was initiated by the moderator.

Before the beginning of the development procedure, the participants were provided with a document with instructions on the user task. The task was the same described in chapter 2. That is, participants had to build a DVD store mobile app by using the prototype cloud EUD tool, presented in chapter 2.

After completing the developing task, each participant had to answer a questionnaire-based survey consisted of 19 items measuring a set of perception and acceptance variables. The questionnaire was provided to the users as an online survey form, embedded in the last page of the EUD application.

The participants had no previous training on how to use the EUD tool. The use of the prototype tool (the web-EUD system) was simple, following a wizard-based logic and the interface text was translated in Greek.

Data Extraction

Although the eye tracking software provides with ready for analysis metrics as depicted in Table 4.2, we used raw excel data which were exported by Tobii and

imported in a MySQL database. Via SQL queries we retrieved the needed data in the desired mode. The reason we used raw excel data was that we did not need to define any Ares of Interest (AOIs) and hence AOI based predetermined metrics would not be useful.

Due to the dynamic nature of the EUD environment, the eye tracking software needed to be adjusted to recognize the stimuli as 'Web stimuli' in order to treat each URL as a static image.

The software's eye tracking filter was set to fixation and the software generated only fixation related data (no saccades or blinks were captured). The raw data provided all fixations, mouse clicks and time information as well as the coordinates (x, y) of every fixated or clicked point in every URL (Media Name). Pupil diameter values for both eyes were also provided in detail.

Figure 4.2 depicts an excerpt of the raw data as exported in excel.

LocalTimeStamp	EyeTrackerTimestamp	GazeEven	GazeEventDuration	GazePointX (ADCSpx)	GazePointY (ADCSpx)	EyePosLeftX (ADCSmm)
10:21:44.409	1422520232513180			1744	577	184,51
10:21:44.418	1422520232521430	Fixation	258	1762	567	184,48
10:21:44.426	1422520232529800	Fixation	258	1722	572	184,86
10:21:44.434	1422520232538180	Fixation	258	1725	580	184,81
10:21:44.443	1422520232546430	Fixation	258	1722	587	184,55
10:21:44.451	1422520232554800	Fixation	258	1714	586	184,50
10:21:44.459	1422520232563050	Fixation	258	1762	579	184,25
10:21:44.468	1422520232571420	Fixation	258	1756	580	184,23
10:21:44.476	1422520232579800	Fixation	258	1715	594	184,46
10:21:44.484	1422520232588040	Fixation	258	1700	589	184,44
10:21:44.493	1422520232596420	Fixation	258	1698	585	184,45
10:21:44.501	1422520232604790	Fixation	258	1705	592	184,42
10:21:44.509	1422520232613040	Fixation	258	1705	585	184,32
10:21:44.518	1422520232621410	Fixation	258	1702	591	184,26
10:21:44.526	1422520232629660	Fixation	258	1693	597	184,29
10:21:44.534	1422520232638030	Fixation	258	1698	587	184,24
10:21:44.543	1422520232646410	Fixation	258	1705	585	184,14

Figure 4. 2 Raw eye and gaze data as exported by Tobii software

URL provision was very important since if URL data was missing it would be impossible to calculate some 'page-dependent' variables like the number of fixations that turned into clicks. We need to group by URL for every screen point (X, Y) to tell whether a fixation belongs to a specific page (URL) when it was clicked or not.

For instance, to calculate the average fixation duration for every participant on every EUD webpage (i.e. different URL) we needed to group the SQL result by the URL (MediaName) as the following code example depicts.

```
SELECT ParticipantName, COUNT( GazeEventType ),
COUNT( MouseEvent ), MediaName
FROM ` table_eyedata `
WHERE GazeEventType = 'Fixation'
AND MouseEvent = 'Left'
AND ((FixationPointX_MCSpx <= MouseEventX_MCSpx + $actarea)
OR(FixationPointX_MCSpx >= MouseEventX_MCSpx - $actarea))
AND ((FixationPointY_MCSpx <= MouseEventY_MCSpx + $actarea)
OR (FixationPointY_MCSpx >= MouseEventY_MCSpx - $actarea))
GROUP BY ParticipantName, MediaName
```

Similarly we built all the SQL queries to calculate and extract the measured variables since they were provided as raw data in excel format.

As explained in the eye tracking technology section, the eye tracker software also produced gaze video recording including gaze plots ('scanpaths'). Scanpaths are the 'paths' that the eye follows to retrieve information over the visual item (Jacob and Karn, 2003; Poole and Ball, 2005). Scanpaths describe a sequence of saccade-fixation-saccade depicting them as lines (saccades) and dots (fixations) toward the target (Goldberg and Kotval, 1999). Scanpaths can also give important qualitative eye metrics such as the scanpath length (e.g. short scanpaths reveal sufficient search strategy), duration, normality, etc.

Figure 4.3 below provides an example of two video gaze plots generated by the current experiment.

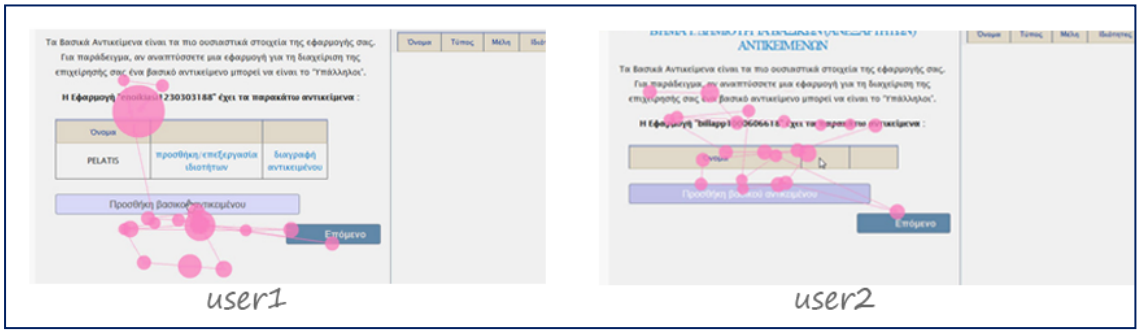

Figure 4. 3 Video gaze plots (scanpaths) of 2 different users in the same EUD system page

Performance measure design

The users' performance was calculated based on the success of five EUD sub-tasks related to the created database items required by the user's exercise. The five EUD sub-tasks were the following:

- Sub-task 1:objects construction (i.e. entities/tables);
- Sub-task 2: attributes construction (i.e. fields);
- Sub-task 3: relationship type construction (e.g. many-to-many);
- Sub-task 4: data types' description (e.g. decimal, integer, string, Boolean, etc.);
- Sub-task 5: relationships' description (using the appropriate verb. e.g. rents – is rented).

Based on the formula used in Protogeros and Tzafilkou (2015), in order to calculate the EUD performance for every participant we assigned a relative weight of importance to each item as the following equation (1) shows:

$$Pe = o *0.3+ a *0.2+ r*0.25+ d *0.15+ v *0.1 \quad (1)$$

Where, Pe=Performance (scale 1-5), o=Object-Performance (scale 1-5), a=Attributes-Performance (scale 1-5), r=Relationships-Performance (scale 1-5), d=Data Types-Performance (scale 1-5) and v=Verbs-Performance (scale 1-5).

Measured variables and questionnaire

The following list contains the set of user-oriented variables (a sub-set of the RULES attributes) measured for each user.

- Self-Efficacy (SE)
- Risk Perception (RP)
- Perceived Ease of Use (PEOU)
- Perceived Usefulness (PU)

The following list contains the set of eye tracking variables (metrics) measured for each user.

- Number of fixations (NumFixations)

- Number of fixations that turned into clicks (FixToClicks)
- Average duration of fixation (FixDuration)
- Pupil size -average increment from initial state (PupilAvg)

We also measured the following variables:

- Task duration time (TaskDuration)
- Performance (Pe)

The questionnaire survey was based on a questionnaire used in previous experiments in gender oriented research. This questionnaire consisted of 19 questions (items) which measure the four independent above-listed variables. A five point Likert-type scale with 1 = ‘‘strongly disagree’’ to 5 = ‘‘strongly agree’’ or 1 = ‘‘never’’ to 5 = ‘‘many times’’ was used to measure the items. The questionnaire’s internal validity and reliability has been evaluated in Tzafilkou et al. (2016) where the authors used this questionnaire in a larger EUD population for the purposes of a behavioral EUD experiment. The Questionnaire structure (the four of the five depicted items) is presented in Table I in Annex I.

4.4.2.3 Results

Table 4.4 presents the sample’s descriptive statistics results for the perception and acceptance variables that were measured via the questionnaire survey.

Table 4. 4 Descriptive statistics of user questionnaire measured items

Item	Mean (1-5)	Std. Deviation
SE	3,30	0,48
RP	3,20	0,63
PEOU	4,00	0,73
PU	3,00	0,81

Table 4.5 below shows the users’ overall performance score and their performance mean for every EUD sub-task for the user sample.

Table 4. 5 End-user's task performance (N=8)

EUD Task	Performance Mean (1-5)	Std. Deviation
Sub-task 1	5,00	0,00
Sub-task 2	3,25	0,88
Sub-task 3	4,50	1,41
Sub-task 4	3,37	1,06
Sub-task 5	4,25	0,70
Overall Performance	4,19	0,50

Diagram in Figure 4.4 gives a general overview of the measured eye-behavior of each user that was implicitly monitored by the eye-tracker.

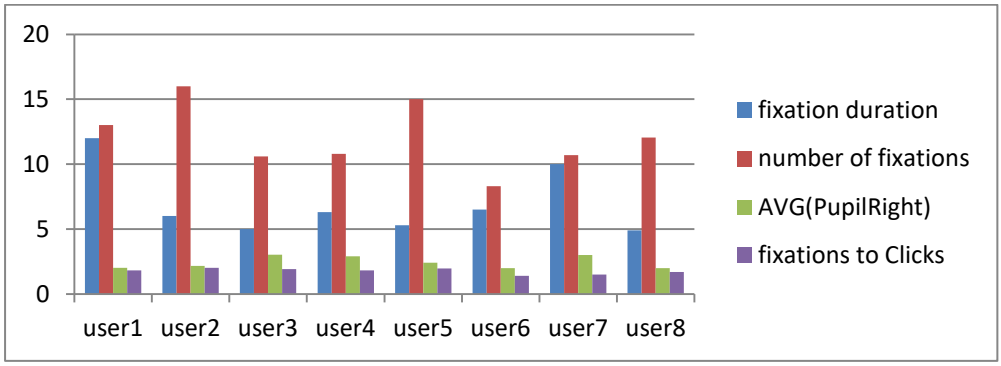

Figure 4. 4 Eye behaviors for every user

Table 4.6 shows the correlation coefficients between pairs of the measured variables for the users' sample. As the results show there is a number of significant correlations between eye behavior and perception or acceptance items.

Table 4. 6 Descriptive statistics of user questionnaire measured items

		FixToClicks	NumFixations	PupilAvg	FixDuration
SE	PearsonCorrelation	**0,65***	**0,66***	0,22	0,24
	Sig.	0,041	0,04	0,30	0,29
RP	PearsonCorrelation	0,41	0,4	**0,79****	0,35
	Sig.	0,16	0,17	0,01	0,20
PEOU	PearsonCorrelation	**0,86****	0,56	**0,80****	0,05

	Sig.	0,00	0,07	0,01	0,46
PU	PearsonCorrelation	0,44	0,02	0,07	**0,65***
	Sig.	0,13	0,48	0,43	0,04

*. Correlation is significant at the 0,05 level.

**. Correlation is significant at the 0,01 level.

Table 4.7 shows the correlation coefficients between the measured eye metrics and performance for the users' sample. As the results show there are a number of significant correlations between eye behavior (number of fixations and fixations that turned into mouse clicks) and performance.

Table 4. 7 Pearson correlations between eye metrics and performance (N=8)

		NumFixToClicks	NumFixations	AvgFixDuration
Pe	Pearson Correlation	0,707*	0,815*	-,012
	Sig. (2-tailed)	0,050	0,014	0,977

*. Correlation is significant at the 0.05 level.

Table 4.8 shows the correlation coefficients between task duration time and performance. As the results show there is a significant correlation between task duration and performance.

Table 4. 8 Pearson correlations between task duration time and performance (N=8)

		Task Duration
Pe	Pearson Correlation	0,754*
	Sig. (2-tailed)	0,031

*. Correlation is significant at the 0.05 level.

4.4.2.4 Discussion

The field test results are encouraging for the future of eye tracking integration in EUD behavioral analyses. Our exploratory study shows that eye tracking methodologies can be used to implicitly monitor the end-users and diagnose their perception and acceptance items in their web EUD activities. Our research hypotheses were confirmed

since there were a number of significant correlations between eye behavior and acceptance and perception while users interacted with a prototype web-based EUD environment for database-driven mobile applications.
Due to the small sample size we did not conduct any gender-oriented research in eye behavior.

Diagnosing Self Efficacy

Results in Table 4.4 confirm hypothesis H1.1since the correlation between Self-Efficacy and the number of fixations that turned into clicks is significant at the 0.05 level.

Results in Table 4.4 confirm hypothesis H1.2 since the correlation between Self-Efficacy and the total of fixations is significant at the 0.05 level.

Diagnosing Risk-Perception

Results in Table 4.4 confirm hypothesis H2.1 since the correlation between Risk-Perception and the average increment of the pupil's size (diameter) is significant at the 0.01 level.

Diagnosing Perceived ease of use

Results in Table 4.4 confirm hypothesis H3.1 since the correlation between Perceived ease of use and number of fixations that turned into clicks is significant at the 0.01 level.

Additionally, results showed that Perceived ease of use is also correlated to the average increment in pupil's size.

Diagnosing Perceived usefulness

Results in Table 4.4 confirm hypothesis H4.1 since the correlation between Perceived usefulness and the average fixation duration is significant at the 0.05 level.

Diagnosing Performance

Results in Table 4.5 and Table 4.6 show that there are some significant correlations between eye movements and performance. In particular, significant correlations were detected between number of fixation and performance, fixations that turned into

mouse clicks and performance and task duration time and performance. No significant correlation was found between average fixation duration and performance.

To sum up, the results reveal the following significant correlations:

- Self-Efficacy is significantly correlated to both, the Total Number of Fixations and to the Number of Fixations that Turned into Clicks.
- Risk-Perception is significantly correlated to the Average Increment of the Pupil Size.
- Perceived Ease-of-Use is significantly correlated to both, the Number of Fixations that Turned into Clicks and the Average Increment of the Pupil Size.
- Perceived Usefulness is significantly correlated to the Average Duration of Fixation.
- Performance is significantly correlated to the Total Number of Fixations, Fixations that Turned into Mouse Clicks and Task Duration Time.

What is also important to mention is that the lack of previous eye research in EUD triggered our interest and motivation to study and 'use' either user experience eye metrics or mouse behavioral patterns that already exist for detecting and measuring user perception and acceptance items. As already explained, the same measures can represent different phenomena in the context of different stimuli and goals (Bojo, 2013). Indeed, this exploratory work showed that previously recognized mouse tracking metrics can be useful in eye tracking research as well. Similarly, UX-centric eye tracking metrics can also be used to diagnose or measure user acceptance and perception items in various HCI areas such as EUD.

To this end, eye tracking seems to be a promising methodology in capturing and analyzing EUD behavior as expressed via perception and acceptance. Eye tracking implementation can be regarded as a contributing step in the deeper understanding of end-users' mental situations and hence in the design of more user-centered and efficient EUD tools and approaches in the near future. And this conclusion can be reflected on the basic truth that "we can build better End-User Development tools if we know how end-users think" (Rode et al., 2005).

Possible issues and limitations

This research is the first (to our knowledge) in the area of eye tracking analysis in web-based EUD environments for database driven mobile applications and there are some limitations.

First, the approach involves a limited number of variables and there are other important variables that could be taken into account in future studies.

A second limitation is imposed by the sample size. Even if Pearson correlation does not assume normal distributions, the sample size is quite small. The visual analysis though, based on gaze plots/scanpaths could be further used (as it was used in the case of fixation duration) to confirm and reinforce the statistical results. Additionally, it is admitted that in eye tracking research there is no "one sample size fits all". Especially in usability-targeted research, according to Bojko (2013), eye tracking as a research tool improves the problem discoverability and the more improved the problem discoverability, the fewer participants are needed.

Finally, there may be another possible limitation involved by the wizard-based design of the prototype tool. Wizard-logic has been proved to be preferred by female users (Beckwith et al., 2005; Burnett et al., 2010) and it can positively affect their perception and acceptance. Also, this can possibly lead to differentiated results in future web-EUD research that will be conducted on non-wizard like interface designs.

4.5 Summary and Contribution of the Chapter

In this chapter we examined the correlation between eye movements and end-user behavioral attributes and performance in a modern cloud-based EUD environment. Our main objective was to find out whether end-users' perception and acceptance attributes can be reflected on their eye behavior when interacting with cloud and/or web-based EUD environments.

The measured variables for perception and acceptance were the following four (out of five) items included in the RULES model presented in chapter 3: Self-Efficacy, Risk Perception, Perceived Ease of Use and Perceived Usefulness. The measured metrics for

the eye movements were the following: number of fixations, number of fixations that turned into clicks, average duration of fixations and average increment of pupil size.

To evaluate our research hypotheses we conducted a field test using our prototype cloud EUD tool (presented in chapter 2) to assist end-users in creating database-driven mobile applications. Eye tracking data from 8 participants were analyzed. Two participants (out of ten) were excluded due to calibration difficulties.

The conducted field test showed that there are several significant correlations between eye movements and acceptance and perception items. In particular, significant correlations were detected between Self-Efficacy and total number of fixations, Self-Efficacy and number of fixations that turned into clicks, Risk-Perception and average increment of the pupil size, Perceived Ease-of-Use and number of fixations that turned into clicks, Perceived Ease-of-Use and average increment of the pupil size, and finally Perceived Usefulness and average duration of fixation.

Also, results showed that there are some significant correlations between eye movements and performance. In particular, significant correlations were detected between number of fixation and performance, fixations that turned into mouse clicks and performance and task duration time and performance.

Some example analysis work on video-based gaze plots/scanpaths reinforced the confirmation of our hypotheses (e.g. regarding duration of fixation) and revealed that correlations between eye behavior metrics and perception can also be extracted using visual gaze plots/scanpaths. Hence, future EUD oriented works can use gaze videos as an analytical tool to examine more correlations by measuring e.g. the length of the saccades' path, normality, duration, etc.

This study can be perceived as a contributing work towards the design of user modeling methodologies in EUD environments. Also, this chapter's results confirmed the usefulness of including eye metrics as implicit user feedback in our end-user developers' modeling framework, presented in chapter 6.

CHAPTER 5

MOUSE AND KEYBOARD TRACKING IN END-USER DEVELOPMENT

In chapter 4 we presented an eye tracking methodology to reveal its usefulness for gathering implicit user feedback during EUD tasks. In particular, we showed the existence of several significant correlations between common eye metrics (fixation metrics and pupil size) and users' behavioral attributes of perception and acceptance. In this chapter we will investigate the role of mouse and keyboard monitoring in EUD environments. This feedback, along with the eye feedback presented in chapter 4, will be used for the construction of the final user modeling approach. The main contributions of this chapter have been published in Tzafilkou and Protogeros (2018) and in Tzafilkou et al. (2014).

5.1 Introduction

Understanding end-users' behavior under high cognitive loads, like developing and decision making activities, has become a major focus in Human Computer Interaction (HCI) research (Djamasbi et al., 2008; Payne, et al., 1988; Svenson, 1993) as it can provide insight into user performance. One class of behavior that is of interest within HCI research is mouse and keyboard behavior and therefore, techniques for studying mouse movements and keystroke dynamics have been considered an effective means of HCI. In the area of web-user interaction mouse and keyboard monitoring is a vital part since it can implicitly and dynamically provide useful information about the users' state of mind, their perceived user experience and the system usability as well.

Mouse movement can be thought of as a series of moves, i.e. gestures. Each gesture is a specific and continuous physical process initiated and concluded by the user (Arapakis et al., 2014). Hence, several mouse-tracking studies suggest the existence of a typology of users based on observed cursor behavior (Leiva et al., 2008; Rodden and Fu, 2007; Rodden et al., 2008). Researchers estimate that mouse features can reflect specific user behavior patterns regardless of the task and the environment, and can be used to model users' behavior (Hinbarji et al., 2015). Common mouse behavioral

patterns, such as the hesitation pattern, the straight pattern, the random pattern, etc. have been identified in many research works (e.g. Atterer and Lorenzi, 2008; Ferreira et al., 2010; Lee and Chen, 2007; Mueller and Lockerd, 2001; Rodden et al., 2008) but their characterization is generally vague. Their identification remains a largely visual process and is mainly based on qualitative analysis. Mouse pattern related metrics remain undefined and hence mouse patterns have not been quantitatively measured and analyzed in some important HCI behavioral research areas like EUD-user behavior.

Recently, keystroke dynamics (i.e. the timing information describing key press and key release events), an approach from user authentication research, has also been used in HCI research to detect user cognitive and affective states, like mood and emotions (e.g. in Khanna and Sasikumar, 2010, Epp et al., 2011). Similarly to mouse behavior, keystroke dynamics seem promising for modeling user behavior (Epp et al., 2011). However, despite their easy and useful applicability, keystroke dynamics have not been broadly studied in HCI behavioral research and particularly in EUD.

Triggered by the above-described situations, in the current research part we will examine whether end-users' mouse and keystroke behavior is associated with RULES behavioral attributes (presented in chapter 3) including perceived ease of use, perceived usefulness, self-efficacy, willingness to learn and risk perception.

The contribution of this chapter is to support the use of mouse and keyboard monitoring as implicit feedback gathering mechanisms for user modeling implementations and self-adaptive software in web-based EUD environments. Such adaptations could assist users (by providing appropriate feedback or changing the interface attributes) to enhance their performance in their developing activities.

5.2 Literature Background

5.2.1 Mouse Behavioral Patterns

Like most of web tracking technologies, the main goal of mouse tracking is a deeper understanding of the user behavior to infer user intentions. Mouse cursor movement is a vital part in HCI field and has been used in many studies.

One of the initial and main measurements used in HCI studies is Fitts' law for movement time (Fitts, 1954; Fitts and Peterson, 1964). This law originated from the desire to quantify the pointing behavior of humans and is currently most used to measure the efficiency of an interface's layout. The efficiency is measured in the time period that is needed to use the interface i.e. to click the buttons or to navigate within the interface. According to Fitts' law "a bigger and closer object (to the cursor), is easier to move to" (Fitts, 1954). Since Fitt's law many researchers have used mouse movement analysis to deeper understand users' cognition process, psychological aspects, perception, level of difficulty, etc.

Following we briefly present the most common mouse behavioral patterns as defined in the reviewed literature. We also attempt to conclude (based on previous research findings) which EUD-user behavioral attribute(s) every pattern could reflect on.

Straight Pattern

The defined 'straight pattern' (Lee and Chen, 2007) or else the 'direct mouse movements' (contradictory to the random ones) refers to the users' 'confident movements'. These are characterized by a pause before a direct movement towards a target since "once traced the desired feature users move the mouse straight to it" (Lee and Chen, 2007). According to Rodden et al. (2008), this pattern defines direct movements that occur once the user has decided which action to take and this could reveal certainty and task-oriented self-efficacy.

On presence of this pattern, researchers infer that there is an earlier decision and the user feels certain about his/her movement and hence they associate this pattern to the user's self-efficacy levels. A direct movement pattern may reveal that an activity is easy and simple, reflecting an earlier decision making.

In the context of interaction with web applications, direct movement is characterized by "a direct movement whit no big pauses" (Ferreira et al., 2010).

Hence, drawing from the afore-described, mouse trajectory (e.g. straight lines) or selected targets (i.e. clicks) after direct movements may measure the end-user's self-efficacy level or perceived ease of use when interacting with computer applications.

Hesitation Pattern

Hesitation pattern is among the most common and recognizable patterns of mouse behavior. 'Hesitation' pattern is first explicitly defined in Cheese project (Mueller and Lockerd, 2001). According to the authors, "hesitation on links or text could potentially provide information about what else interests the user on the page". In particular, users' second choice (the given task was to select and buy a product from Barnes&Noble.com and amazon.com) could be predicted by determining the link on which they hesitated longest before clicking their first choice. This hesitation could be expressed by the cursor as a mouse hover over the under decision item (i.e. element to click).

Other studies refer to the pattern of hesitation as movements between two or more answers while trying to decide which one to choose (Churruca, 2011). There have also been identified hesitation patterns during interaction with navigation menus (Atterer et al., 2006).

In commercial applications of mouse tracking, as Atterer et al. (2006), the pattern of hesitation is defined as "the average time from the beginning of a mouse hover to the moment of mouse click".

There is a clear distinction between those who believe that the hesitation may occur on a single target, as a pause before the click and among those studies that consider the hesitation as a pattern of doubt among many possible targets (Churruca, 2011).

In Reeder and Maxion (2006), user hesitation is defined as anomalously long pauses between events. Although such hesitations can occur for many reasons, they often indicate user difficulty, which according to the authors is generally caused by interface defects. Their designed hesitation detector algorithm takes as input a time-stamped, chronologically ordered data stream of mouse and keyboard events to detect anomalously long pauses. These are identified by computing the latency between every pair of consecutive events in the data stream, computing the average-length latency between events, and outputting those latencies that exceed a certain threshold which is specified as a number of standard deviations from the average-

length latency. Latency average and standard deviation are independently measured for each user. Thus, hesitations are defined relative to each user, and are still valid for users who use input devices unusually quickly or slowly.

Hesitation has also been studied in terms of content relevance on search results. Complementary to studies that focus on clickthrough rates to detect hesitation, Huang et al. (2011) examined also abandonment elements it terms of unhovered clicks to conclude that unclicked hovers and hover time are negatively correlated with relevance judgments. This appears to contradict previous findings which suggested that hesitation over a result is a positive indicator of relevance (e.g. Mueller and Lockerd, 2001).

Generally, via the hesitation pattern the user reflects a doubt about the option to choose and a difficulty in decision making. It is publicly assumed that the greater the degree of difficulty, the more user hesitation is observed during the user-system interaction (Belk et al., 2015; Reeder and Maxion, 2006).

Reading Pattern

The reading pattern can be divided into two different types of behavior: the horizontal reading pattern that is characteristic but not so common among users (Rodden et al., 2008) and the vertical reading pattern where the mouse path traces a vertical line, with small or large pauses. In Ferreira et al. (2010) the horizontal reading pattern is characterized by "a smooth mouse movement that trails horizontally across paragraphs". In comparative studies with eye-tracking it is defined as a movement that follows the eye horizontally (e.g. Rodden et al., 2008).

The vertical reading pattern is defined as a movement of "following the eye vertically" (Rodden et al., 2008). This pattern is observable in vertical lists like a menu where the users "use the mouse as a marker when they are looking through a list of links" (Mueller and Lockerd, 2001).

Random and Fixed Pattern

According to Ferreira et al. (2010) the random pattern is characterized by movements "without any specific intention, just playing around and doing random movements whit short pauses or not". The authors noted that the random movements often arise when the difficulty level of the task increases (in contrary to the straight pattern movements). Hence, on presence of a random pattern we can infer doubt, difficulty, low self-efficacy, anxiety, etc.

Described in Lee and Chen (2006), the fixed pattern refers to the areas where the users mostly used for "repose" the cursor. For Lee and Chen (2006) these areas usually are on the right-side of the page. This area is also mentioned as "White space" (Ferreira et al., 2010) and research indicates that a large number of users use whitespace to rest the mouse, thus avoiding accidentally clicking a link.

Fixed pattern could be expressed via long or shorter mouse pauses on ‘neutral’ web-page areas. During this time users may reflect on the task, read (eye tracking could only reveal reading behavior during mouse pauses) or evaluate the costs and benefits of making particular decisions (e.g. clicks, actions). Hence, fixed pattern could measure risk-perception or usefulness levels.

Guide Pattern

According to Lee and Chen (2006), the guide pattern defines a behavior of continuous movement of the cursor. According to the authors this pattern seems to reflex an «exploratory» role that suggests a relationship between mouse and eye movement" and hence this pattern may give us an idea of the user expectations. Hence, guide pattern may be associated with users’ acceptance, i.e. perceived usefulness and ease of use and willingness to learn as well.

Guide pattern though is very similar to the reading pattern since it can be expressed via ‘smooth’ (i.e. slow) cursor movements identifying horizontal or vertical reading.

5.2.2 Keystroke Dynamics in Human Computer Interaction

Keystroke dynamics have been typically used in biometrics and user authentication (Bergadano et al., 2003; Monrose and Rubin, 2000). As defined in Monrose and Rubin (2000) “keystroke dynamics is the process of analyzing the way a user types at a terminal by monitoring the keyboard inputs thousands of times per second in an attempt to identify users based on habitual typing rhythm patterns”.

In a more ‘HCI-friendly’ and generic approach, Epp et al. (2011) describes keystroke dynamics as ‘the study of the unique timing patterns in an individual’s typing, and typically includes extracting keystroke timing features such as the duration of a key press and the time elapsed between key presses”.

Recently, keystroke dynamics have been used in affective computing research to detect user affective states like mood and emotions (Eppet al., 2011; Khanna and Sasikumar, 2010).

In Khanna and Sasikumar (2010) the authors used keystroke dynamics to model three emotional classes (positive, neutral, negative). The attributes used in Khanna and Sasikumar (2010) include typing speed, mode, standard deviation, standard variance, range, total time taken, number of backspaces used and interval between typing (if any).

Epp et al. (2011) used keyboard dynamics in emotion detection and showed the classification of 15 emotional states and could identify six emotions (confidence, hesitance, nervousness, relaxation, sadness and tiredness). They also showed promising results for anger and excitement.

Nahin et al. (2014) have presented a method of determining user emotion by combining both keystroke dynamics and text pattern analysis.

There is not much research in analyzing keystroke dynamics and end-users behavioral states. Some related research may concern personality detection through keyboard and mouse input. For instance, in Khan et al. (2008) the authors suggest that some of the main traits and sub traits of personality can be measured from keyboard and mouse use. The keyboard attributes measured in two exploratory studies were ‘key-

up' and key-down'. Their findings revealed some limited significance since only recording the standard deviation of the average time between events gave some insight into a user's activity level, a sub trait of extraversion.

In Vizer et al. (2009) the authors used keystroke timing features of free text in conjunction with linguistic features to identify cognitive and physical stress.

In a more related study of Dijkstra (2013) the author examined the correlation between mouse or keyboard input and self-efficacy levels. Although mouse input showed some promising results for detecting self-efficacy levels, no correlations were found between typing keyboard behavior and self-efficacy.

5.3 Literature Review Concluding Remarks and Challenge

Although analyzing user behavior in HCI tasks has revealed its usefulness, mouse tracking methodologies have not been integrated yet in common everyday cloud and/or web-based activities like End User Development. Mouse behavior in EUD could be monitored and analyzed to measure user intention and engagement, diagnose perception and acceptance states, detect user emotions, predict user performance, etc. User modeling implementations based on implicit mouse and keyboard input could be used to personalize and adapt the EUD and assist end-users perform better in their developing tasks.

To this end, our challenge is to examine whether end-users' mouse and keystroke behavior is associated with most EUD behavioral attributes (RULES, chapter 3) including perceived ease of use, perceived usefulness, self-efficacy, willingness to learn and risk perception.

Our main objective is twofold:

i) to divide every mouse behavioral pattern (as defined in reviewed literature) in a set of measurable mouse metrics and examine the potential correlations between these 'pattern-derived' metrics and EUD-user behavioral attributes (RULES attributes), and
ii) to examine the potential association between a set of common keystroke dynamics and EUD behavioral attributes (RULES attributes).

The central goal of this research part is to identify possible connections between mouse movements and/or keystroke dynamics and a set of EUD user behavioral attributes, like self-efficacy, risk-perception, etc. and understand how mouse behavioral patterns and keystroke dynamics are related to user behavioral states during EUD task execution.

5.4 The proposed approach: Mouse and Keyboard Behavior as Implicit Feedback in End-User Development

Our aim is to investigate the usefulness of mouse and keyboard tracking as implicit feedback in web and/or cloud based EUD environments in order to include relative metrics in our final user modeling implementation.

In this section we first present the prototype tools we developed to conduct our mouse tracking experiment:

i) a mouse and keyboard tracking software, and
ii) a web EUD tool for the construction of simple web forms.

Then, we present the laboratory test we conducted to investigate potential correlations between mouse and keyboard actions and the EUD behavioral attributes (RULES) presented in chapter 3. To conduct the study we connected the mouse tracker tool to the prototype cloud EUD tool.

The study results are encouraging for our final user modeling implementation (see chapter 6) since several mouse and keyboard metrics seem useful to be included in our behavioral user input components (as implicit feedback).

2.1. Prototype Tools

For the purpose of the mouse and keyboard tracking experiment we designed and developed a prototype monitoring software (Mouse Tracker) and a new web EUD tool to facilitate the mouse tracking procedure.

Following, we present the main technical aspects and the interface layout of the developed tools.

5.4.1 Mouse and Keyboard Monitoring Software

Presentation

The prototype mouse tracker was designed to capture users' mouse and keyboard behavior and store every event in a MySQL database.

An important feature is that the tool displays graphs and statistics in real-time at any website connected via a JavaScript file in the central server. Hence, we connected the monitoring tool to the EUD prototype tool to implicitly monitor the end-users' mouse and keyboard behavior while working on their developing task.

After the storing of the mouse and keyboard events the administration panel (viewed in a new web page) the main mouse movements for every stored client id. In particular it displays in a graphical manner mouse moves and mouse clicks distinguishing also between left and right clicks.

Finally, to link the questionnaire answers to each user's mouse and keyboard behavior we generated a unique client id for every user and we stored it in every database table and in a questionnaire's answer field.

Technical Overview

Figure 5.1 illustrates the overall concept's architecture and underlines the used technologies.

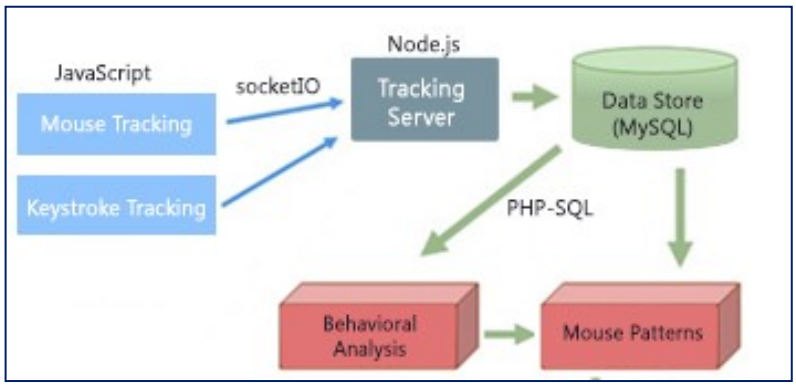

Figure 5. 1 Mouse Tracking Tool's Architecture

As depicted in Figure 5.1 the server-side part of the tool is based on Node.js while the client side is based on JavaScript and makes use of the socket.io library. The

transmitted data relates to mouse movements and user keyboard clicks that are recorded on the controlled web pages. Socket.io is fast enough to cope with near real time data transmission and allows no information to be lost.

Data is stored in a MySQL database in a relational schema composing of tables for mouse metrics' categories (e.g. mouse clicks, mouse hovers, mouse movements, keystroke events, etc.) and the user related data (e.g. session id, time, username, etc.)

The stored raw mouse data consists of mouse clicks, moves and hover events, their coordinates (x,y), their time intervals (in μs) and a timestamp (in μs) of when every mouse event occurred.

The raw keystroke data consists of key press and release events, unique codes for each key, their time intervals (in ms), and a timestamp (in ms) of when the key event occurred.

For example, the table of mouse clicks (ms_clicks) consists of the following fields: coordinate x, coordinate y, time since last mouse click, time the mouse click occurred, click element, element's content and client's id. The recorded time is very accurate since it is the client time in microseconds when the event has occurred. This time along with the coordinates and other event information is put in a message and queued with the help of the socket.io library, before being sent to the server and stored in the database. Thus, any delay in the transmission and storage process does not affect the actual data of the event and the reliability of the database.

Similar is the approach to creating the other tables and recording the time. As shown in the code snippets, there is an SQL query to create the database table and another one to store the mouse data in the appropriate fields. The inserted data represent the user actions and are stored through server side functions

After data storage, a PHP-SQL behavioral analysis is conducted on mouse feedback to recognize behavioral mouse patterns, according to the literature review.

In particular, to extract the mouse and keyboard metrics (see Table 5.1 and Table 5.2) we used a set of PHP algorithms and SQL queries. Thus, we could retrieve from the database the desired mouse and keyboard behavior for every user.

5.4.2 Web EUD Tool for simple forms construction

Presentation

A prototype web-based EUD tool was designed to assist end-users in creating simple web forms. The reason we decided not to use the simple-talking tool (presented in chapter 2) in the mouse experiment was mainly its wizard-like logic that does not allow the 'freedom of many and various mouse movements. The idea was to design a Single-Page Application (SPA) web EUD so that it would be 'easier' for the users to express their mouse behavior during the development task. Since SPA implies that all the interface elements are placed in a single layout, users would be able to move their mouse around the interface and select the desired items without leaving the area. This would allow us to store in the database a sufficient amount of mouse events.

The tool's interface, as depicted in Figure 5.1, is a user friendly drag and drop single-page application composing of various 'clickable elements' (buttons, links, fields, etc.) so that the mouse tracking implementation and later behavioral analysis would be easy and efficient. A simple drag-and-drop approach permits the composition of various form fields by simply selecting them and drugging them in the form panel. The dropped item automatically exposes its options and characteristics to choose (e.g. size, options, length, etc.).

Also, the interface features offer many opportunities for text typing (e.g. name of objects, form fields, options, etc.) so the keyboard behavior would be efficiently monitored as well.

Via the forms construction, the tool allows the composition of a set of objects that represent the stored entities (see Figures 5.2-5.3). It also offers the option to preview the constructed form and edit its basic design elements such as colors, font size, etc.

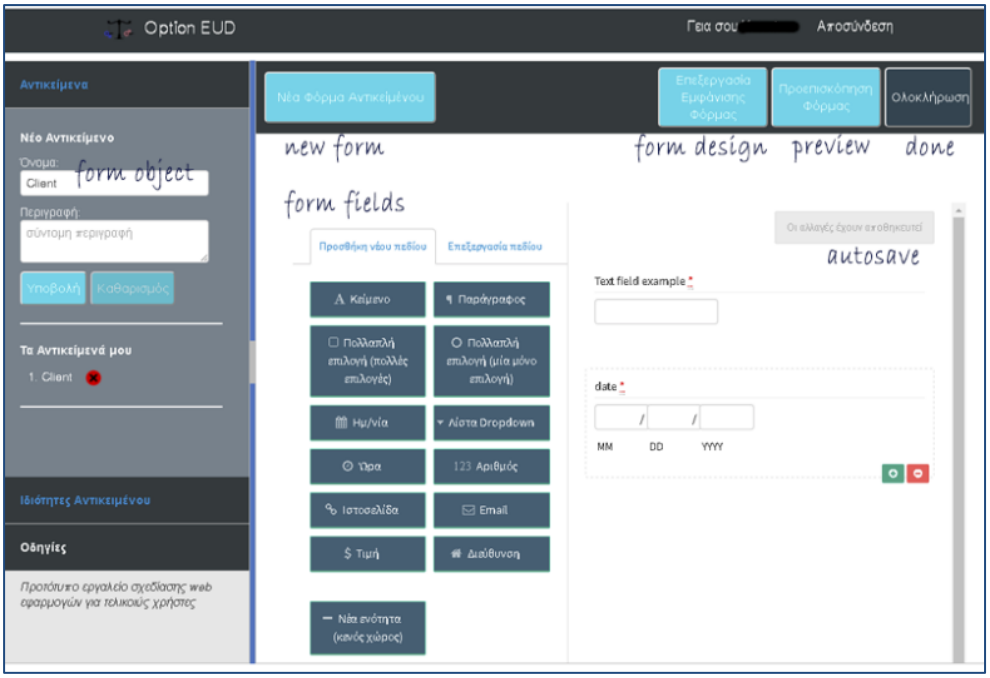

Figure 5. 2 EUD tool interface (some English translation is provided in hand-written fonts)

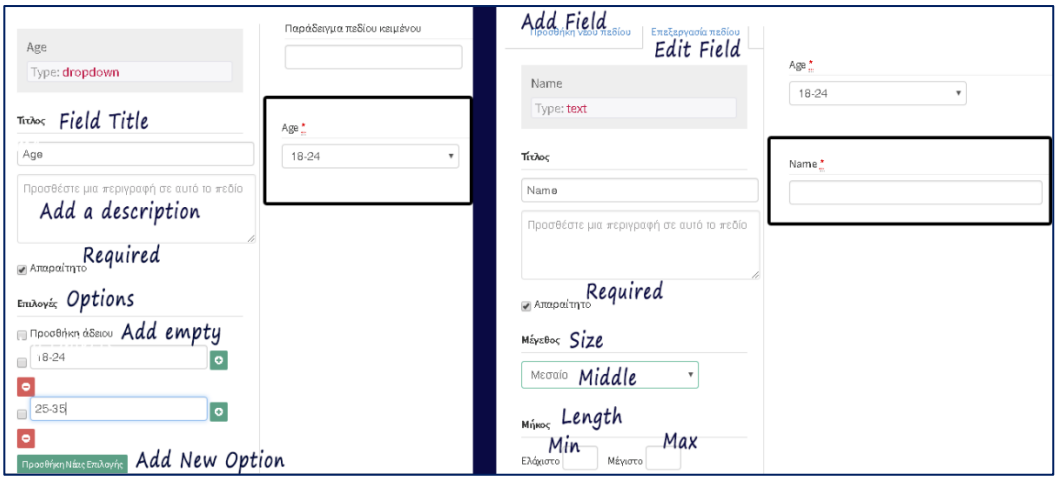

Figure 5. 3 Form construction interface (some English translation is provided in hand-written fonts)

As mentioned, the EUD tool stores the constructed forms as 'entities' and the form fields as entity fields. The EUD approach was initially designed to follow the simple-talking approach presented in chapter 2. However, the simple construction of a web form and its assignment to an object (i.e. entity) was implemented for the experiment purpose as the main focus was to capture a sufficient amount of the users' mouse and keyboard actions and hence a more complicated and time demanding EUD task would be unnecessary.

To evaluate the tool's ease of use and efficiency we measured the participants' score performance (as presented in subsection 5.4.3.2) and perceived ease of use. The participants showed high performance results (4,60 out of 5,00) and high levels of perceived ease of use (4,09 out of 5,00) as depicted in Table 5.4.

Technical Overview

The tool is developed using both native JavaScript code and the JavaScript libraries of JQuery and Bootstrap. For the drag and drop form builder functionality parts of Github source code are also used.

For the data storage we use MySQL database. The users' constructed items are stored in the database in a JSON format. JSON format allows the ease retrieval of every user's items. This makes it easy to calculate the users' performance score.

5.4.3 Mouse and Keyboard Tracking Experiment in End-User Development

5.4.3.1 Research Questions

Since EUD has some important differences from other HCI areas where mouse tracking has already been studied, we should consider some constraints before formulating our research objectives.

- A web-based EUD system does not consist of static web pages (like most mouse tracking studied environments); instead it is composed of dynamic web pages and in some cases it is a Single Page Application (SPA). Thus, we will not define any areas of interest (AOIs) in the web page and the suggested mouse metrics will be context and region-independent, ignoring the interface layout. Our approach will be suitable to be used in other similar EUD research works.
- We will measure the average mouse-behavior during the time needed by a user to complete a development task. This time may be different for each user and is defined as 'task duration time'. For this reason the time related mouse metrics (e.g. average pause time, average time between clicks, etc.) are divided by the task duration time for each user and this ratio is used as the main mouse metric.

This gives the analogy of the measured mouse time variable to each user's EUD completion time.

- In order to quantitatively examine mouse patterns in EUD activities we have first to find a way to measure. That is to subdivide each mouse pattern (from the ones described in the literature) into a set of measurable mouse metrics. In Churruca (2011) the authors subdivided each mouse pattern into a set of mouse movements and then measured them using only visual representation via video recording. In the current research we attempt to subdivide each mouse pattern into a set of measurable mouse metrics and examine them separately to find out which ones are met in end-user developers' mouse behavior and which ones are significantly correlated with one or more of their EUD behavioral attributes. Our main research objective is to examine the bivariate potential correlations between the pattern-related mouse attributes (direct moves, number of clicks, number of hovers, etc.) and end-users' behavioral attributes (perceived ease of use, perceived usefulness, self-efficacy, willingness to learn and risk perception).
- Also, we plan to measure the activity level of mouse movements, i.e. the relationship between movement of the cursor and the duration time of the EUD session. We will then examine the correlation between activity level and EUD behavioral attributes.
- Finally, we will examine the potential correlations between a set of main keystroke dynamics (dwell time, flight time, etc.) and the measured EUD behavioral attributes. Although web-based EUD activities do not include a lot of text typing tasks, users still have to provide some important input through the keyboard (e.g. form fields' names, options values, object's names and descriptions, etc.), and although limited, this input could still reveal a meaningful contribution.

The existence of significant correlations will undoubtedly reveal a promising research venue in the future of end-users' behavioral analysis and the usefulness of mouse and keyboard tracking methodologies as implicit feedback gathering mechanisms in web-based EUD systems.

Hence our main research questions are the following:

RQ1: Are straight pattern attributes associated with end-users' behavioral attributes?

RQ2: Are hesitation pattern attributes associated with end-users' behavioral attributes?

RQ3: Are random pattern attributes associated with end-users' behavioral attributes?

RQ4: Are fixed pattern attributes associated with end-users' behavioral attributes?

RQ5: Are guide pattern attributes associated with end-users' behavioral attributes?

RQ6: Is the activity level of mouse movements associated with end-users' behavioral attributes?

RQ7: Are any keystroke dynamics associated with end-users' behavioral attributes?

Since, this is an exploratory study we do not construct any hypotheses regarding which pattern reflects which behavioral attribute - every pattern is a composition of different mouse metrics and some metrics could reflect one behavioral state while some others could reflect another one. Also metrics of different patterns could be associated with the same behavioral states.

To examine the above-presented research questions, as explained, we have first to subdivide every behavioral mouse pattern into a set of measureable mouse attributes (mouse metrics). Following we present the mouse metrics for each one of the presented mouse patterns and argument on their selection. We exclude the reading pattern from our study since its metrics are really close to the guide pattern and there are not many reading tasks (except some instructions or wizard-based guidance text) in the EUD environments.

Straight pattern attributes

According to Lee and Chen (2006), "in straight pattern (or else direct pattern), the mouse movement starts from an initial region, while users are presumably visually browsing other regions of the page. It is surmised that once users spotted the link that they were asked to find, they moved the mouse straight to it, and usually terminated the action with a click".

Based on the above conclusion, in Churruca's (2011) visual interpretation, the straight pattern can be expressed through direct movements that may or may not include a pause and they usually move toward a target, that is, they end up in a mouse click.

Hence, the mouse metrics-attributes we plan to examine in regards to the straight pattern include the following:

- **Direct movements** (i.e. straight lines);
- **Clicks in the end of straight lines** (e.g. direct movements towards a target that was clicked) ;
- **Average time passed between straight lines** (e.g. time elapsed between direct movements; this could also show how often a user makes a decision and trace a direct trajectory).

We note that it is not necessary all the attributes to exist. The existence of only some of them could represent the straight pattern in a measurable way.

The same rule applies to all of the following patterns as well.

Hesitation pattern attributes

As mentioned, in ClickTale (2010) the pattern of hesitation is defined as "the average time from the beginning of a mouse hover to the moment of the mouse click".

In other research works hesitation may occur in a single target, as a pause before the click or in a dilemma between two or more choices and therefore can be expressed through mouse hovers between at least two items in the same sequence (Guo et al., 2008).

In a more generic way we could also link hesitation to mouse hovers, pauses before clicks and average time between clicks as well.

Hence, the mouse metrics-attributes we plan to examine in relation to the hesitation pattern include the following:

- Hovers that turned into clicks;
- All hovers in general;
- Average time from mouse hover to mouse click;
- Average time of pauses before clicks;
- Average time between clicks.

Random pattern attributes

According to Ferreira et al. (2010) a movement is random when it occurs without obvious intention, that is, it is not among targets. Also it may or may not include short pauses.

Random movements may reveal user anxiety and doubt (Churruca, 2011) and hence they should be clearly different from straight pattern movements.

The random related metrics we plan to examine are:

- **Curved (i.e. not direct or random) movements;**
- **Clicks that occurred outside direct movements;**
- **Total number of movements** (since a large amount of mouse movements could include a large percentage of movements with no intention).

Fixed pattern attributes

In Lee and Chen (2006), the fixed pattern refers to the areas where the users mostly used for "repose" the cursor. That is that except from the area where the mouse is reposing, it stays there for a 'long' time. In our case, this can be translated into 'long pauses'.

Different studies use different time thresholds to characterize a pause as long or short. Since this is a EUD exploratory study, and because of the nature and the cognitive load of a developing task, we consider as long pauses those over 4 seconds. Short pauses (or else pauses) are considered all the pauses that remain less than 200ms, as underlined in the related literature (Dijkstra, 2013).

Hence, our fixed related metrics include the following:

- **Long pauses;**
- **Average time of long pauses;**
- **Average time of all pauses** (including short pauses).

Guide pattern attributes

The guide pattern (Lee and Chen, 2006) defines a behavior of continuous movement of the cursor and it seems to plays an "exploratory" role that suggests a relationship between mouse and eye movement (Leiva and Hernando, 2008). Hence, guide pattern is similar to the reading pattern since it implies that the mouse traces Slow or Long Movements while following the eye.

In our study, long movements are considered as direct movements with distance more than 10 pixels and slow (or smooth) movements are considered as movements that their time interval exceeds 3 seconds.

Hence, the guided related metrics we plan to examine are the following:

- **Long movements;**
- **Slow (smooth) movements**.

In Figure 5.4 below we summarize the measurable mouse attributes we gathered for each mouse behavioral pattern that we plan to examine.

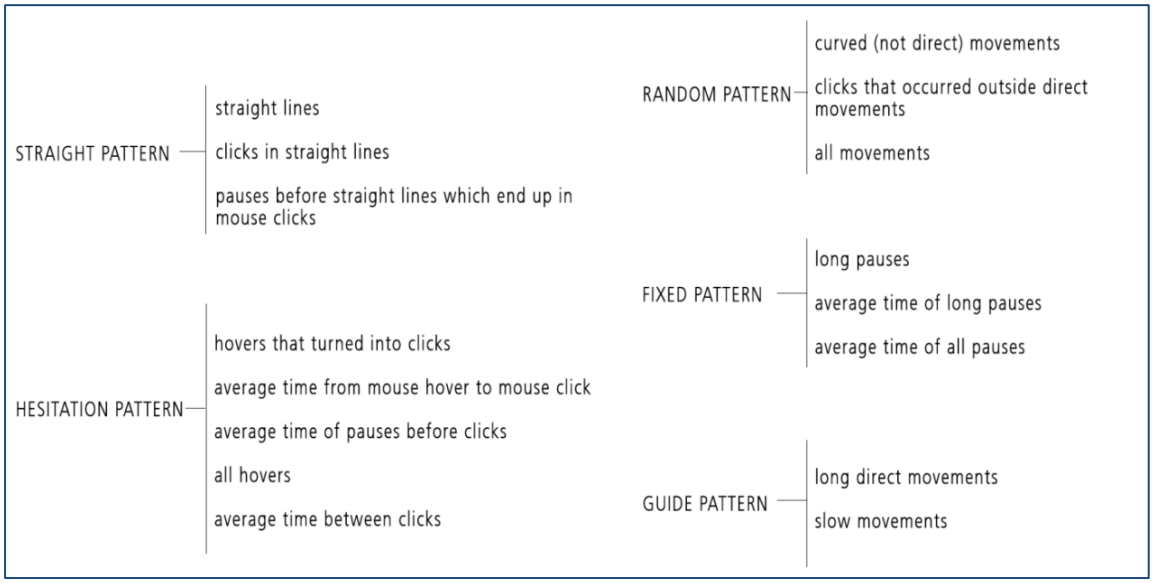

Figure 5. 4 Grouped measurable mouse attributes per mouse behavioral pattern

Keystroke attributes

Since there are not similar behavioral EUD or other related research on keystroke dynamics, we will 'borrow' some typical keystroke metrics from other HCI related studies like emotion recognition.

The main keystroke dynamics-attributes we plan to analyze are proposed in Khanna and Sasikumar (2010), Nahin et al. (2014) and in Epp et al. (2011). And as Figure 5.5 depicts they are based on:

- key press speed;
- duration (dwell time);
- latency (flight time);
- down-to-down time.

In our experiment we will examine the potential correlations between these typical keystroke dynamics and the measured EUD behavioral attributes of self-efficacy, risk-perception, willingness to learn, perceived usefulness and ease of use.

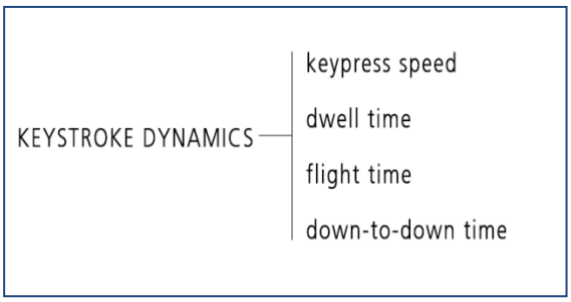

Figure 5. 5 Chosen measurable keystroke dynamics attributes

5.4.3.2 Evaluation Methodology

First we briefly present the data extraction methodology. Then, we describe the exploratory test including the sample and procedure details, the user-task, the performance calculation and the measured dependent variables and questionnaire. Finally, we present the data analysis process.

Mouse and Keyboard Data Extraction

As already mentioned, to extract the mouse and keyboard metrics we used a set of PHP algorithms and SQL queries. Thus, we could retrieve from the database the desired mouse and keyboard behavior for every user.

To build the data extracting algorithms we needed first to set some size and time thresholds necessary to decide whether a click relates to a particular application element or not. The thresholds were optimized after much experimentation. The time thresholds are all depicted in the following tables (Table 5.1 and Table 5.2) next to the relevant measures.

Table 5.1 below describes the mouse attributes that we plan to extract.

Table 5. 1 Extracted mouse attributes

Feature	Type	Meaning
num-mv	Count (units)	Number of mouse movements in a session.
num-crv	Count (units)	Number of non-direct (curved) movements in a session (slope > 0.01 and distance between two points > 3.5px).
num-outlineClks	Count (units)	Number of clicks outside direct movements (lines) in a session (slope > 0.01 and distance between two points >3.5px).
num-inlineClks	Count (units)	Number of clicks in the end of direct movements (lines) in a session (slope < 0.01).
num-longPauses	Count (units)	Number of mouse long pauses (time elapsed since last movement < = 4sec) in a session.
avg-longPauses	Time (ms)	Ratio of average time of long pauses (time elapsed since last movement >5sec) to the task duration time.
num-dirMv	Count (units)	Number of direct movements in a session (slope <0.01).
avg-timeBtwDir	Count (units)	Ratio of the average pauses time between direct movements to the task duration time.
num-hvrToClk	Count (units)	Number of mouse hovers that turned into mouse clicks (time interval from mouse hover to mouse click <=4sec) in a session.

avg-hvrToClk	Count (units)	Average time from mouse hover to mouse click on the same element (time interval from mouse hover to mouse click <=4sec).
avg-btwClk	Count (units)	Ratio of average time between clicks to the task duration time.
avg-pause	Count (units)	Ratio of average time of pauses (>200ms) to the task duration time.
avg-pauseBfrClk	Count (units)	Ratio of average time of pauses (>200ms and <10sec) before clicks to the task duration time.
num-longDirMv	Count (units)	Number of long (> 10px) direct movements in a session.
num-slowMv	Count (units)	Number of slow (> 2sec) movements in a session.
num-hvr	Count (units)	Number of all mouse hovers in a session.
task-dur	Time (ms)	The EUD task total duration (completion) time (which is different for every user).
acl-mv	Time (%)	Activity level for mouse movements (i.e. time of movements in a session/task-dur).

As already explained, the time mouse features (avg-longPauses, avg-hvrToClk, avg-btwClk, avg-pauseBfrClk, avg-pausesBtwDirand avg-pause) are taken into account along with the EUD task duration (task-dur) time which is different for every user. That is, where 'avg-shortPause' is the ratio avg-shortPause/task-dur, and so on.

Table 5.2 below describes the keystroke attributes we plan to extract. The main keystroke attributes we extracted were based on key press (character) speed, dwell, flight and down-to-down time as proposed in Khanna and Sasikumar (2010), Nahin et al. (2014) and in Epp et al. (2011).

Table 5. 2 Extracted keyboard attributes

Feature	**Type**	**Meaning**
speed-char	Count (units)	Number of characters pressed every 4 seconds.
dwell-time	Time (ms)	Time elapsed between key press and key release (time interval was set to 100ms).
flight-time	Time (ms)	Time elapsed between a key release and the next key press (time interval was set to 1 sec).

d2d-time	Time (ms)	Time elapsed between the characters pressed every 4 seconds.

Regarding speed-char (key press speed) we used a threshold of 4 seconds instead of 5 seconds proposed in the reviewed literature (e.g. Nahim et al., 2014), since as explained web-based EUD is not a typing task hence we need shorter time thresholds to measure users' keyboard activity.

Participants and Procedure

The experiment was conducted in an introductory e-commerce course, in the Department of Accounting and Finance in a Greek University. The participants' initial population was 42 volunteers and the final sample consists of 30 end-users, 18 male and 12 female. The sample size was reduced to 30 individuals because seven volunteers did not complete the EUD task and five did not submit the questionnaire form. Hence their data could not be used in the analysis process. There were no significant age and expertise differences among the participants. To confirm the sample EUD representation, i.e. the participants' similar level of computer and developing experience, an expertise pre-test was conducted.

The participants were given a EUD task to solve, in the form of an exercise demanding the construction of a simple web form. The tool's interface text was in Greek. Instructors made a basic presentation of the EUD tool at the beginning of the field test and did not give any other special instructions, apart from the exercise (i.e. the requested development task). Few students who were not very comfortable with the use of the system and asked help received some further information and instructions. In the end, participants could be informed about their performance and discuss their experience and difficulties.

Based on the EUD terminology given by Lieberman et al. (2006) that end-users act as non-professional developers to build or modify applications, we require from our participants to belong to the generic end-user developer population. That is, they should be familiar enough with web and computer software's and concepts but they should not be experts and of course they should not be programmers.

Drawing from all these, the targeted EUD group population is young (age 18-25) users from a European country (Greece), with no disabilities, having the regular computer skills and experience (e.g. computer use and web surfing) but they are not professional developers, meaning that they do not have any programming or web development skills.

To confirm these criteria, some personal information (sex, age, educational background etc.) was collected and as mentioned, prior to the EUD task the participants were also asked to answer a short questionnaire regarding their experience level on database concepts, programming, World Wide Web and overall computer use. Their experience level was measured in a scale from 1 to 5, as depicted in Figure 5.6.

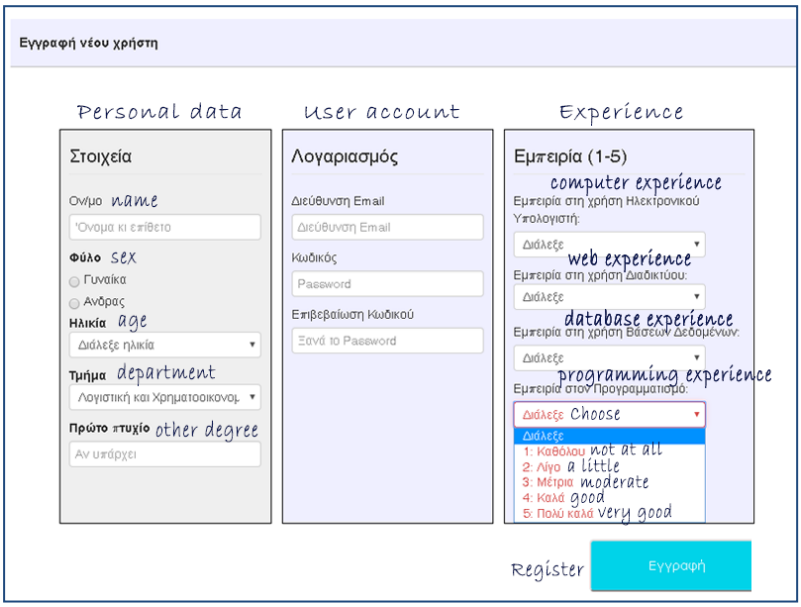

Figure 5. 6 Web questionnaire prior to EUD task, regarding the user personal info and computer experience (translation in English is provided in hand-written fonts)

The measured mean value of the participants' database familiarity was 1,45, meaning that they could safely be considered as non-professional/non-expert end-users. Additionally, their programming experience was 1,30, their familiarity with web was 2,97 and their general familiarity with computer use was 2,78 (see Table 5.3). These mean values satisfy our target group (end-user developers) requirements, i.e. users that are non-experienced programmers, with no or limited knowledge on database concepts but with efficient familiarity with web interaction and computer use in

general. This information could possibly allow us to generalize the results from a small sample and make broad claims about web-based EUD behavior of similar EUD populations.

Table 5. 3 Descriptive statistics of participants' experience level

Measured Item	Mean (0-5)	St. Deviation	St. Error
Computer Experience	2,78	0,81	0,11
Web Experience	2,97	0,80	0,11
Database Experience	1,45	0,75	0,32
Programming Experience	1,30	0,69	0,10

User Task and Performance Measure

The exercise (user task) needed to be small enough to be solved in a limited time by the participants, and as comprehensive as possible in order to capture a logic amount of mouse movements and keystroke events. The example of a web form is simple and comprehensive, since forms and fields are familiar to most end-users and its construction encompasses a set of items such as field types (text, number, list, etc.), choice options, length and design elements (fonts, colors, button styles, etc.) as well. So, the exercise given to the participants was to create a simple web form to store a set of customer data.

First, the end-users had to create an object named 'customer' and give it a description. Then, for this object they had to create a form and construct the following form fields:

- a text type field to store the customer name and last name;
- a dropdown type field with two options to store the customer age;
- an email type field to store the customer email address.

The participants could also set the fields as required or not, define their length and some other form aspects.

To calculate user performance for every participant we used a score scale from 1 to 5 and assigned a relative weight of importance to each constructed item as the equation (1) presented in chapter 4.

After completing the above-described user task, each participant had to answer a self-report questionnaire-based survey consisted of 22 items measuring a set of EUD behavioral variables (see Table I Annex I). As explained in previous chapters, the questionnaire was provided to the users as an online survey form, embedded in the last page of the EUD application.

Dependent Variables and Questionnaire

The following list contains the set of user-oriented (EUD behavioral) variables measured for each user from the post-task questionnaire survey.
EUD behavioral (dependent) variables:

- Self-Efficacy (SE)
- Risk-Perception (RP)
- Willingness To Learn (WL)
- Perceived Usefulness (PU)
- Perceived Ease of Use (PEOU)

Performance was excluded from the list of the dependent variables since the paper focuses on the variables of EUD behavioral attributes, and performance was calculated only to reinforce the tool's efficiency and the research validity.

Sample Characteristics

A normality distribution test was conducted to test whether the values of every measured dependent variable were approximately normally distributed for the whole sample.

A Shapiro-Wilk's test ($p > 0.05$) (Shapiro and Wilk, 1965) showed that the values of Perceived-Usefulness, Perceived-Ease of Use, Willingness to Learn, Self-Efficacy and Risk-Perception come from a normal distribution (see Table II in Annex II).

The fact that the sample size is normally distributed reinforces the validity of the statistical results.

Data Analysis Method

Since our data is approximately normally distributed we can use a parametric analysis method. Hence, in order to measure the bivariate correlations between the measured variables we used the Pearson correlation analysis as it is an appropriate method to define the strength of the association among a set of continuous variables.

To present the general results concerning every measured variable we used descriptive statistics.

To evaluate the questionnaire internal consistency we calculated the value of Cronbach alpha (α).

Later, in the Discussion section we highlight our main findings and discuss a number of potential limitations of the current experimental design.

5.4.3.3 Results

Table 5.4 below presents the descriptive statistics for the EUD behavioral measured items for the whole sample. Performance statistics are also included since performance calculation was used to reinforce the EUD prototype tool's efficiency.

Table 5. 4 Descriptive statistics of user questionnaire measured items and performance

Measured item	N	Mean (1-5)	Std. Deviation
Perceived Usefulness	30	3,78	0,60
Perceived EaseOfUse	30	4,09	0,70
Self-Efficacy	30	3,96	0,78
Risk-Perception	30	2,42	0,75
Willingness To Learn	30	3,70	0,82
Performance	30	4,60	0,66
Valid N (listwise)	30		

Diagrams in Figure 5.7 below show the mean values of the recorded mouse and keystroke attributes. Mouse time related metrics (average pauses, etc.) are not depicted.

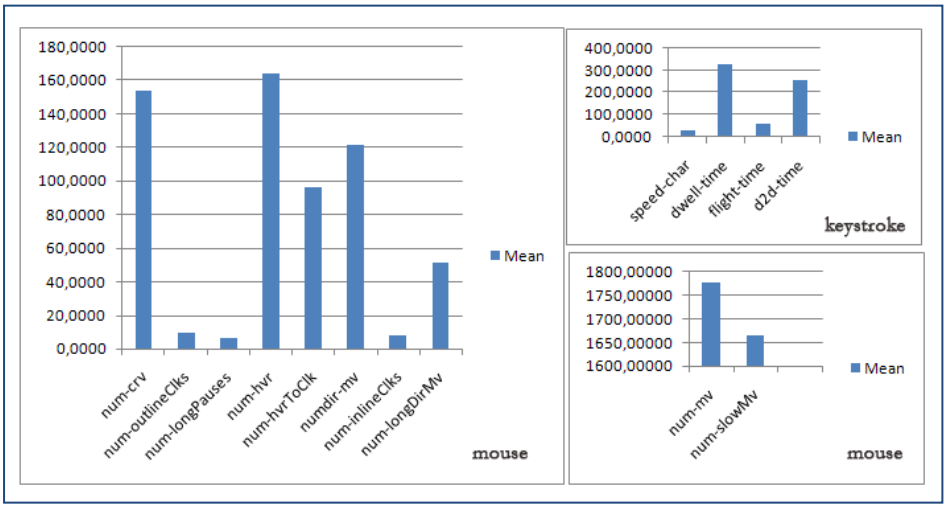

Figure 5. 7 Diagrams of participants' mouse and keyboard actions' values (N=30)

Table 5.5 below shows the Pearson correlations (r) between every extracted mouse metric and EUD behavioral attribute. As depicted, several significant correlations have been revealed. Last row depicts the Pearson correlation between activity level of mouse movements and behavioral variables.

Table 5. 5 Correlation analysis between mouse metrics and user behavioral attributes (N=30)

Pattern category	**Mouse attribute**	**PU** (r)	**PEOU** (r)	**SE** (r)	**WL** (r)	**RP** (r)
Random	num-mv	**-0,43***	**-0,41***	**-0,31***	-0,20	0,18
	num-crv	**-0,36***	**-0,38***	-0,29	-0,28	0,16
	num-outlineClks	-0,22	**-0.38***	-0,304	-0,18	0,16
Fixed	num-longPauses	-0,21	-0,32	**-0,43***	-0,44	0,27
	avg-longPauses	0,32	0,34	**0,44***	0,27	-0,87
	avg-pause	**0,39***	0,24	0,21	**0,32***	**-0,31***
Straight	numdir-mv	**-0,36***	**-0,37***	-0,24	-0,28	0,14
	avg-timeBtwDir	-0,25	-0,06	-0,12	**0,31***	-0,29
	num-inlineClks	-0,15	-0,28	-0,27	-0,28	-0,37

Hesitation	num-hvrToClk	-0,26	**-0,44***	-0,28	-0,16	0,15
	avg-hvrToClk	0,25	0,28	0,29	**0,50****	-0,17
	avg-btwClk	0,61	0,25	0,22	**0,35***	**-0,31***
	avg-pauseBfrClk	**0,38***	0,21	0,18	**0,33***	-0,27
	num-hvr	**-0,31***	-0,28	-0,16	-0,23	0,11
Guide	num-longDirMv	**-0,31***	**-0,34***	-0,19	-0,24	0,12
	num-slowMv	**-0,36***	**-0,36***	-0,25	-0,28	0,15
-	acl-mv	**-0,38***	-0,24	-0,20	-,0,30	-0,30

**.Correlation is significant at the 0.01 level

*.Correlation is significant at the 0.05 level

Table 5.6 below shows the Pearson correlations (r) between every extracted keystroke metric and EUD behavioral attributes. As depicted, some significant correlations have been found.

Table 5. 6 Correlation analysis between keystroke attributes and user behavioral attributes (N=30)

Keystroke Attribute	**PU** (r)	**PEOU** (r)	**SE** (r)	**WL** (r)	**RP** (r)
speed-char	-0,17	**-0,35***	**-0,34***	-0,14	0,23
dwell-time	0,19	0,24	0,22	0,23	-0,40
flight-time	0,13	0,01	0,08	0,28	-0,06
d2d-time	-0,23	-0,29	**-0,37***	-0,20	0,17

*.Correlation is significant at the 0.05 level

5.4.3.4 Discussion

The field test results are encouraging for the future of mouse and keyboard monitoring for EUD behavioral analyses. Initial exploratory findings have given us key insights since various correlations have been revealed between mouse/keyboard behavior and EUD behavioral attributes.

Most of our research questions have been answered in a contributing way as the findings reinforced the usefulness of mouse and keyboard behavioral analysis in web-based EUD activities.

Following we summarize the main findings that came up for every research question formed in section 3.

RQ1: Are straight pattern attributes associated with end-users' behavioral attributes?

Results in Table 5.5 show that one of the straight pattern's metrics (the number of direct movements) is significantly associated with perceived ease of use and perceived usefulness, and one metric (the metric of average time between direct movements) is significantly associated with willingness to learn. A possible reason for this latter correlation could be the fact that willingness to learn can be better 'expressed' via time related metrics. As we see in the rest of the patterns correlations, willingness to learn is mainly correlated to time-based metrics such as average time between clicks, average pause time before click and average time from mouse hover to click.

We also notice that the number of clicks inside direct movements showed no correlation to EUD behavioral attributes. One would also expect, based on the reviewed literature, the existence of some significant correlations between straight pattern attributes and self-efficacy since straight pattern implies user confidence. The absence of a self-efficacy correlation could mean that self-efficacy and straight pattern related mouse metrics should be studied more in the field of EUD and a deeper behavioral analysis is needed to understand which mouse behavior (if any) can clearly reveal EUD self-efficacy levels. Also, we see no correlation between straight pattern metrics and risk-perception. Risk-perception shows some correlation only with hesitation and fixed pattern. This could be possibly explained by the fact that risk-perception is a more 'time-prone' attribute since in the literature review it is clearly stated that it is related to time-demanding cognitive tasks where the end-users attempt to evaluate the costs and benefit of their next actions. Based on this, risk-perception might be more affected by mouse pattern metrics, most of which are time based, e.g. average pause time etc.

RQ2: Are hesitation pattern attributes associated with end-users' behavioral attributes?

As results in Table 5.5 show, hesitation pattern related metrics are significantly associated with all the measured behavioral attributes, i.e. perceived ease of use, perceived usefulness, willingness to learn, self-efficacy and risk-perception. We notice that no correlation appears between hesitation pattern metrics and self-efficacy. Also, self-efficacy is only correlated with fixed and random patterns. As mentioned earlier, self-efficacy and mouse correlation may need to be studied deeper in the EUD field.

In particular, hesitation pattern metrics revealed the following EUD correlations:

- Number of hovers that turned into clicks is significantly correlated to perceived ease of use.
- Average time from mouse hover to click is significantly correlated to willingness to learn and self-efficacy.
- Average time between clicks is significantly correlated to willingness to learn and risk-perception.
- Average pause time before clicks is significantly correlated to perceived usefulness and willingness to learn.
- Number of hovers is significantly correlated to perceived usefulness.

RQ3: Are random pattern attributes associated with end-users' behavioral attributes?

As results in Table 5.5 show, the random pattern related metrics are significantly associated with perceived ease of use, perceived usefulness and self-efficacy. We notice that no correlations are shown between random pattern metrics and willingness to learn and risk-perception.

In particular, random pattern metrics revealed the following EUD correlations:

- Number of movements is significantly correlated to perceived ease of use, perceived usefulness and self-efficacy.
- Number of curves (not direct movements) is significantly correlated to perceived ease of use and perceived usefulness.
- Number of clicks outside direct movements (straight lines) is significantly correlated to perceived ease of use.

RQ4: Are fixed pattern attributes associated with end-users' behavioral attributes?

Results in Table 5.5 show that fixed pattern related metrics are significantly associated with perceived usefulness, self-efficacy, willingness to learn and risk-perception. We notice that no correlations are shown between fixed pattern metrics and perceived ease of use.

In particular, random pattern metrics revealed the following EUD correlations:

- Number of long pauses (>5sec) is significantly correlated to perceived ease of use, perceived usefulness and self-efficacy.
- Average time of long pauses (>5sec) is significantly correlated to self-efficacy.
- Average time of all pauses (both long and shorter >200ms) is significantly correlated to perceived usefulness, willingness to learn and ease of use.

RQ5: Are guide pattern attributes associated with end-users' behavioral attributes?

Results in Table 5.5 show that the guide pattern related metrics are significantly associated only with the two acceptance attributes, i.e. perceived usefulness and perceived ease of use. We notice that no correlations are shown between guide pattern metrics and self-efficacy, willingness to learn and risk-perception.

In particular, random pattern metrics revealed the following EUD correlations:

- Number of long direct movements (>10px) is significantly correlated to perceived ease of use and perceived usefulness.
- Number of slow (smooth) movements (>2sec) is significantly correlated to perceived ease of use and perceived usefulness.

RQ6: Is the activity level of mouse movements associated with end-users' behavioral attributes?

As results in Table 5.5 show, activity level for mouse movements is associated only with perceived usefulness in web-based EUD activities.

RQ7: Are any keystroke dynamics associated with end-users' behavioral attributes?

As results in Table 5.6 show, only two out of four keystroke dynamics (key press speed and down-to-down time), revealed some significant correlations to EUD behavioral attributes. In particular, keystroke dynamics attributes revealed the following EUD correlations:

- Key press speed (threshold = 4sec) is significantly correlated to perceived ease of use and self-efficacy.
- Down-to-down time (threshold = 4sec) is significantly correlated to self-efficacy.

Table 5.7 summarizes the results of Table 5.5 and Table 5.6, presenting a clearer image of the bivariate correlations between the measured mouse/keyboard metrics and EUD behavioral attributes. As we see, every mouse behavioral pattern can be correlated to many user behavioral attributes and every mouse metric reveals its distinct particularities regardless the pattern it 'belongs to'. Thus, we notice that many mouse pattern metrics are correlated to more than one EUD behavioral attributes. This was the main reason we could not construct any particular hypotheses on the correlations between mouse patterns and particular behavioral attributes. However, we can notice some clear common behavioral correlations of patterns. For instance, we observe that fixed pattern is mainly associated to self-efficacy, whereas guide and random patterns are mainly oriented to perceived usefulness and ease of use. Straight pattern shows no correlation to willingness to learn and risk-perception, as we discussed these two attributes reveal a more 'time-prone' correlation and straight pattern is not mainly composed of time-based metrics. Hesitation pattern expresses a multi-item orientation since hesitation mouse metrics are more in number and not so close (in terms of mouse action) to each other.

Regarding keystroke dynamics correlations, we notice that both key press speed and down-to-down time are correlated to self-efficacy, revealing a strong relationship between user's self-efficacy and keystroke dynamics in EUD tasks. Key press speed also reveals a relationship with perceived usefulness. Dwell and flight time did not show any meaningful contribution in the current web-based EUD behavioral research.

Finally, no gender differences were detected in the participants' mouse movements. For this reason we do not analyze any gender-oriented research results. Also we found no significant correlation between mouse movements and users' performance, contrary to eye movements' results as presented in the previous chapter.

As a conclusion we could say that we cannot detect particular behavioral attributes from one single mouse behavioral pattern. Mouse pattern 'translation' and division in measurable mouse metrics is necessary before conducting a EUD-related correlation analysis. Additionally, different research works could enrich or differentiate these metrics resulting in some different or more/less correlations.

Table 5. 7 Summary of significant correlations between EUD behavioral attributes and mouse/keyboard metrics

Mouse/Keystroke Attribute	**Mouse Behavioral Pattern**	**User Behavioral Attribute (RULES Attributes)**
Mouse metrics		
num-mv	Random	PU, PEOU,SE
num-crv	Random	PU, PEOU
num-outlineClks	Random	PEOU
num-longPauses	Fixed	SE
avg-longPauses	Fixed	SE
avg-pause	Fixed	PU, SE, WL, RP
num-dirMv	Straight	PU, PEOU
avg-pausesBtwDir	Straight	WL
num-inlineClks	Straight	-
num-hvrToClk	Hesitation	PEOU
avg-hvrToClk	Hesitation	SE , WT
avg-btwClk	Hesitation	WL, RP,
avg-pauseBfrClk	Hesitation	PU, WL
num-hvr	Hesitation	PU

num-longDirMv	Guide	PU, PEOU
num-slowMv	Guide	PU , PEOU
al-mv	-	PU
Keystroke metrics		
speed-char		PEOU, SE
dwell-time		-
flight-time		-
d2d-time		SE

As already mentioned, based on the behavioral correlations found, mouse and keyboard tracking can be useful as implicit feedback gathering mechanisms in end-user profiling and modeling mechanisms. Mouse and keyboard monitoring allows indirect and real time data collection without interrupting the end-users' cognitive process during ongoing development tasks.

Unfortunately, mouse behavioral patterns have not yet matured in the HCI behavioral research and their mouse metrics remain vague. However, steps like the current study are contributing to the evolution of mouse behavioral patterns and their broader application in user-oriented HCI and EUD research.

5.5 Summary and Contribution of the Chapter

In this chapter we examined the correlation between a set of EUD behavioral attributes and mouse behavioral patterns and keystroke dynamics. Our aim was to find out whether end-users' behavioral attributes can be reflected on their mouse and keystroke behavior during web-based EUD activities.

The measured EUD behavioral attributes are the previously defined (in chapter 3) RULES attributes, including: Self-efficacy, Risk perception, Willingness to learn, Perceived ease of use and Perceived usefulness. The mouse behavioral patterns we examined include the Straight pattern, the Hesitation pattern, the Fixed pattern, the Random pattern and the Guide pattern. Reading pattern was excluded due to the dynamic nature of EUD activities and their limited 'content to read'. The measured

keystroke dynamics are the four main ones, as derived from the reviewed literature, including key press speed, dwell time, flight time and down-to-down time.

A set of research questions were formed to structure the research objectives and distinguish the measured variables. Except mouse pattern-related variables we also measured mouse activity level of movements and included it in our correlation research.

To explore our research questions we conducted a field test using a prototype web-based EUD tool to assist end-users in creating simple web forms, and we monitored their mouse behavior via a prototype mouse tracking mechanism. Mouse and keystroke monitoring data from 30 participants was analyzed.

The conducted filed test showed that there are several significant correlations between mouse measurable attributes and EUD behavioral states. Keystroke dynamics also revealed some significant correlations to perceived usefulness and self-efficacy. Summary conclusions in Table 5.7 show that mouse and keyboard monitoring is a useful mean of gathering real time implicit feedback in EUD activities.

This study can be perceived as a contributing step towards the design of user modeling methodologies in EUD environments. This study also contributes in the practical evaluation of the available mouse metrics, showing which ones are or can be useful for the EUD community. Finally, this chapter's results confirmed the usefulness of including mouse and keyboard metrics as implicit user feedback in our end-user developers' modeling framework, presented in next chapter.

CHAPTER 6

THE END-USER DEVELOPER MODEL: AN ONTOLOGY-BASED REPRESENTATION

The focus of this chapter is on behavioral modeling for end-user developers during their interaction with cloud and/or web-based EUD environments. Although designed for the thesis' prototype EUD environments (presented in chapters 2 and 4), the suggested approach could be applied to other EUD systems. In this chapter we employ an ontology-based representation of the End-User Developer Model (EUDM) based on both explicit and implicit input. Explicit input includes gender and domain knowledge level, and implicit input includes mouse, keyboard and eye behavior as being analyzed in the previous chapters. Within the OWL-defined user ontology we employ a rule-based reasoning methodology to define inferences on user domain knowledge and behavioral attributes and construct the final model which can be used for EUD adaptation and personalization services.

6.1 Introduction

One of the goals of software adaptation and personalization is to assist users in performing their tasks in the most efficient way. That implies, if possible, the system should not interfere with the main task of the users and should not demand from them extra efforts to maintain the effective adaptation (Sosnovsky, 2010). User modeling methodologies aim to provide a foundational solution to the aforementioned issue.

User modeling is based on the definition of user profiles to represent user characteristics and on the reasoning of users' behavior to provide them with personalized services. There are many different methods for user modeling including ontology-based user modeling, machine learning methodologies, user modeling via collaborative filtering, etc.

In our work we have adopted the use of semantic technologies and the ontology-based approach to represent the suggested user model. The main reason we decided on a semantic implementation is because semantic technologies enable interoperability across different platforms, support semantic reasoning and have the ability to reuse

information from several application domains (Janev et al., 2011) Various ontological user models have been developed for use within personalized web information retrieval systems, adaptive user interface design and for public services such as customized libraries (Mehta et al., 2005; Sutterer et al., 2008). Nevertheless, such models have not been adopted to implement user modeling methodologies in web-based EUD systems for the end-user developers' population.

For these reasons we designed an ontological-based user modeling framework to model end-user developers' profile characteristics (gender, knowledge-expertise), behavior (mouse, keyboard and eye movements) and behavioral/perceptional items (RULES: risk perception, self-efficacy, perceived ease of use, etc.) during their interaction with cloud or web EUD systems. Furthermore, the suggested modeling approach uses a rule-based reasoning within the ontology to assign property values to the final user model.

6.2 Literature Background

6.2.1 Ontology-based User Modeling

Ontology is a set of classes, properties connecting classes, restrictions on properties, and axioms (Maedche, 2002). Ontology-based modeling involves specifying a number of concepts related to a particular domain, along with any number of properties associated with those concepts (Lebib et al., 2016).

Ontologies are either storage ontologies or inference ontologies. As Fankam (2008) explains, storage ontologies are used for information storage, classification and reuse from heterogeneous sources. On the other hand, inference ontologies are especially well suited for domain representations and knowledge management applications, because they provide a rich representation of multi-view domains, with minimal commitment. According to Krima (2009), inference is the ability to make deductions about instances, regarding the classes, properties and restrictions defined. OWL (Web Ontology Language) is the only standard language, recommended by the W3C which expresses inference ontology-based models (Antoniou and van Harmelen, 20009).

With the advent of social and semantic web several ontological modeling approaches have been proposed. XML and RDF vocabulary has permitted the construction of multi-dimensional and cross-platform user profiles and ontological representations. To this end, OWL-defined ontology-based user modeling approaches have been mainly used to model context knowledge (Baldauf et al., 2007; Strang et al., 2004) and address problems of user model heterogeneity and aggregation (Goix et al., 2007). In fact, creating an ontology-based user model increases the probability that user characteristics will be shared among a range of web systems of the same domain, especially today that most ontologies are represented in OWL (Rath et al., 2009). According to Rath et al. (2009), the sharing of user models between web systems is considered as one of main advantages of using ontologies for user modeling.

Another advantage of ontology-based models is the fact that they support reasoning. That is, once the user characteristics are ontologically represented, the ontology and its relations, conditions and restrictions provide the basis for inferring additional user characteristics (Lebib et al., 2016).

Ontology-based modeling can be used for various purposes such as personalization and adaptation services. Many ontological user models have been developed so far in web information retrieval systems, adaptive user interface design and for public services such as digital museum guides or electronic libraries (e.g. Golemati et al., 2007; Mehta, 2005; Sutterer et al., 2008). However, their broad application, ontology user models have not been adopted in EUD environments so that personalization and adaptations for assistive services could be implemented.

Following we describe some popular ontology-based modeling approaches contributed in the related literature.

The first popular approach towards ontology-based user modeling is the General User Modeling Ontology (GUMO) created by Heckman et al. (2007; 2005). GUMO is represented in OWL and provides a collection of the user's dimensions that might be helpful for several information systems intending to provide personalization based on the user model. The user dimensions cover a range of user characteristics such as

contact information and demographics, abilities, personality and special information like mood, nutrition or facial expressions.

Another ontological modeling approach is OntobUM (Ontology based User Model) propsed by Razmerita et al. (2003). OntobUM is a generic ontology-based user modeling including both implicit and explicit user dimensions. Explicit dimension regard common user characteristics such as identity and preferences, whereas the implicit dimensions are related to user experiences on system usage.

The User-Profile Ontology with Situation-Dependent Preferences Support (UPOS) (Sutterer et al., 2008) is an OWL defined ontology addressing both static and context-aware aspects. UPOS allows the construction of situation-dependent and context-aware sub-profiles. A user has a profile and a context (location or activity) associated. The notion of condition is defined, which includes a user, an operator and a context-value. According to this condition, a corresponding sub-profile can be applied that contains all personalization states (Stan et al., 2008).

The ontology mIO! (Poveda, 2010) is a user-context ontology for mobile environments. This ontology is used to represent knowledge related to context as a whole, information on location and time, user information and their current or planned activities, as well as devices located in their surroundings.

Another quite recent and more closely related to our approach model is SERUM (Semantic Recommendations based on large unstructured datasets), proposed by Plumbaum et al. (2012). SERUM utilizes semantic technologies to collect implicit user behavior and to build semantic user models. These models are combined with large-scale semantic datasets and they are finally used to compute personalized news recommendations using graph-based algorithms. SERUM uses RDF annotations and collects implicit user behavior to build semantic user behavioral models.

The User Behavior Ontology (UBO) (Plumbaum, 2011), is possibly the ontology structure most closely related to our suggested end-user developer model (presented in section 6.3). Using OWL description it provides an ontological user model representation suitable for collecting and managing user behavior from websites. UBO

serves as a general behavior model for tracking user interaction data, including both explicit and implicit feedback such as clicks on links and search results. UBO is mainly based on the Behavior Ontology (Schmidt et al., 2007) which captures individual actions, such as mouse and keyboard events, timestamps and website elements of interaction. UBO uses simple log files to store implicit and explicit feedback and is limited to the implicit events it captures.

6.2.2 Semantic Technologies for Ontology-based User Modeling

The growing number of systems that model context data using ontologies and semantic web technologies make semantic interoperability imperative. For this reason the semantic web research community has devoted an important effort in defining common languages not only for ontology modeling but also for reasoning within ontologies. To this end, the Web Ontology Language (OWL) has become the recommended ontology language by the World Wide Consortium since 2004. OWL language contributes in constructing ontologies that provide high-level descriptions of web content. Regarding rule-based reasoning within owl-based ontologies, the Semantic Web Rule Language (SWRL) has been defined. SWRL is in fact a combination of the Rule Mark-up Language (RuleML) and OWL-DL (OWL Description Logics) providing both OWL-DL expressivity and RuleML rules (Horrocks, 2010). SPARQL has been emerged as the most popular query language appropriate to retrieve information from ontology repositories.

Following we briefly describe these three semantic technologies since they have been used to represent and evaluate our suggested End-User Developer Model (presented in section 6.3).

6.2.2.1 OWL-Web Ontology Language

The Web Ontology Language (OWL, 2004) has been emerged as the core semantic web language. OWL builds on RDF and is constructed from XML tags. It facilitates web systems' interpretability, whereas XML and RDF are limited to provide an additional vocabulary along with formal semantics.

The OWL tags that correspond to data modeling constructs include the following (Hay, 2014):

- *Entity Class: <owl: class rdf: id=" " >*
- *Attribute: <owl: data type Property rdf: id="….">*
- *Relationship: <owl: object Property rdf: id="….">*

OWL has three expressive sublanguages, each providing a different varying level of expressive power: OWL Lite, OWL DL, and OWL Full. An important characteristic of OWL-DL is that it provides strong decidability guarantees. However, OWL DL constructs can be used only under certain restrictions and cannot support reasoning with data values and certain types of interrelationships between multiple entities. OWL Full, although providing more expressiveness and significantly extend the RDF schema, shares these limitations.

Despite its limitations, its ability to represent explicitly semantics associated with knowledge and to provide reasoning capabilities used by intelligent systems and agents renders OWL the most preferred semantic language (Ngoc et al., 2005).

6.2.2.2 SWRL-Semantic Web Rule Language

The Semantic Web Rule Language (SWRL) has been proposed to add rules to OWL and expand its expressiveness (Horrocks et al., 2004). SWRL is built on the same description logic foundation as OWL and provides more expressive power than OWL alone, particularly when dealing with complex interrelationships between OWL individuals, or when reasoning with data values (O'Connor et al., 2009).

SWRL can be considered as a formal extension of the OWL language. Each SWRL rule is a sort of OWL axiom that interacts with existing OWL axioms in ontology. That is, SWRL rules are formal logical statements in OWL ontology (O'Connor et al., 2009).
The basic syntax of SWRL is of the following as decribed in Mun and Ramani (2011):

antecedent → consequent (2)

This syntax implies that the consequent must be true when the antecedent is satisfied. OWL expressions can occur in both antecedent and consequent (Hirankiti and Xuan, 2011).

SWRL however, does not support non monotonic reasoning and cannot modify the existing ontology. It is appropriate for monotonic reasoning since it shares OWL's Open World Assumption OWA. That is, deductions cannot be updated or retracted. Certain types of rules that assume a Closed World Assumption (CWA) may be difficult or impossible to write in SWRL (O'Connor et al., 2009).

The OWA implies that "if a proposition cannot be proved to be true with the current knowledge, the system cannot declare this proposition as false" (Drummond and Shearer, 2006). Unlike OWA, the CWA considers the empty knowledge is false rendering the available knowledge in the model as exhaustive (Wang, 2011).

For this reason, the SWRL rules cannot be used to modify existing information in an ontology (Rossell`o-Busquet et al., 2011) and they originally create new knowledge, i.e. new instances or new properties between existing instances (Fortineau et al., 2012).

6.2.2.3 SPARQL Query Language

SPARQL has been the standard query language for querying semantic web RDF data since 2008 when became a W3C Recommendation. The name SPARQL is a recursive acronym that stands for SPARQL Protocol and RDF Query Language (Prud' hommeaux and Seaborne 2008).

SPARQL is essentially a graph-matching query language of the following form (Perez et al., 2009):

$$H \leftarrow B \quad (3)$$

where:

- B is the body of the query, a complex RDF graph pattern expression that may include RDF triples with variables, conjunctions, disjunctions, optional parts, and constraints over the values of the variables.

- H is the head of the query, an expression that indicates how to construct the output.

The output of a SPARQL query can be of different types: Boolean queries, selections of values of the variables matching the constructed patterns, construction of new RDF data from these values, and resources' descriptions (Perez et al., 2009).

6.3 Literature Review Concluding Remarks and Challenge

The existing ontology-based user modeling approaches allow the specification of a great variety of static concepts, like user personal and demographic data, interests, preferences, etc. However, only a few modeling approaches have been traced to allow expressivity for real-time user behavior like mouse and keystroke events or eye related behavior. Although there are some modeling approaches that have used mouse data and other dynamic aspects (e.g. the appeared search results), they are reduced to the logging (in files) of the user activity and their collected data range is limited in very basic mouse actions (e.g. clicks on links) not including any sequential mouse behavior (e.g. to recognize mouse behavioral patterns like the ones presented in chapter 5).

What we also notice in the reviewed literature is the lack of user modeling implementations in EUD web environments. Although user modeling approaches have been implemented in web environments like for instance e-learning platforms and recommender systems, user modeling has not been implemented in web environments for EUD. As a consequence, end-user developers do not meet any representative user profiles and models aiming to express their individual and behavioral differences, which as discussed in chapters 2 and 3 are multifaceted and tend to influence their performance. To this end, personalization and adaptation services are not considerably implemented in EUD environments.

As we have already highlighted, modeling end-user developers' behavior is of crucial value since knowing their varying differences (that may be triggered by gender, expertise and other factors) a EUD system could provide the appropriate services to assist end-user developers in enhancing their developing performance and leverage their overall user experience. To this end, more efficient and useful EUD applications

would be developed and the EUD community would 'gain' a larger amount of 'successful' end-user developers enjoying their user-system interaction and producing software comparable to professional one.

To this end, our challenge is to propose an end-user developer model aiming to represent both user characteristics and real-time behavior while interacting with web and/or cloud EUD systems. To achieve our goal we use semantic technologies to build an ontology-based modeling framework.

Hence, our main modeling objective is to design, build and evaluate an OWL-based ontology modeling approach for end-user developers. This ontology will be structured in such a scheme to store both static and dynamic aspects of EUD related user-system interactions. The static aspects concern crucial EUD attributes like gender (as presented in chapter 3) and dynamic aspects target real-time user behavior such as mouse, keyboard (as presented in chapter 5) and eye (as presented in chapter 4) related actions.

Furthermore, to infer new knowledge on users' domain knowledge and behavioral traits like perceptional and acceptance items (i.e. the RULES attributes: self-efficacy, perceived-ease of use, etc.), we plan to implement a semantic rule based reasoning within the constructed ontology. That is, the final user model will represent new concepts and values derived (after reasoning) from the user data stored in the ontology.

6.4 The proposed approach: An ontological end-user developer modeling framework

In this section we present the methodological framework to construct a user model for the population of end-user developers. This model can be used in EUD systems to provide end-user developers with personalized services. We name our model 'the End User Developer Model' (EUDM) since it attempts to model end-user developers' behavior and other characteristics.

We employ an ontology-based representation of the EUDM and take advantage of RDF/OWL technologies to define classes and properties of user characteristics, task

information, and user behavior. Within the user ontology we employ a rule-based reasoning methodology to construct the final model.

Our user model aims to provide an abstract representation of the interaction between end-user developers and EUD applications. Interaction elements, like mouse clicks, mouse hovers, mouse trajectories, pauses, eye fixations and keystroke dynamics can be retained to have a comprehensive overview of a user's behavior. The final user model will represent user characteristics user behavior and EUD task related information. With this detailed information, EUD applications can build user models and provide the end-user developers with personalized and adaptive services to assist them in enhancing their performance and their overall user experience while interacting with cloud or web based EUD environments.

6.4.1 Description of the End-User Developer Model (EUDM)

In this section we introduce a new ontology-based model for the collection and management of end-user developer behavioral data, characteristics and domain knowledge, the End-User Behavior Model (EUDM).

We define our end-user developer modeling approach, EUDM as following:

The term *End-User Developer Modeling* refers to the process of generating a user model in the context of a EUD environment. A EUD model enables the system to adapt to the user who uses it and ideally includes all information about the user's behavior and knowledge that influences their developing performance.

To be able to create and semantically represent the EUDM, we first have to decide which user model dimensions should be part of the model and which attributes in the different dimensions should be supported.

First of all we need to define the provided user input. The EUDM approach incorporates both explicit and implicit user feedback. Explicitly provided feedback is typically collected by asking the user some questions during the registration phase. Such feedback regards the user's personal characteristics like gender, age background and also domain knowledge on EUD related fields like web development and database

logic. Task information like task name, description and constructed items are also explicitly provided via the corresponding forms during the developing task. EUD time-related information though, is dynamically calculated and stored in the end of every session. Implicit feedback regards the client-side monitored user behavior such as mouse, keyboard and eye movements. The expected user model is based on this feedback and is composed of three main components: the behavioral component (including RULES attributes), the user profile data (i.e. personal characteristics like gender and domain knowledge level) and some EUD task related information.

The feedback categories are summarized below:

Input (User Feedback):

- Explicit: gender, domain knowledge, static task information;
- Implicit: mouse actions, keyboard events, eye movements.

Output (composed/inferred user model):

- RULES behavioral attributes: {“Low”, “High”};
- User Profile: gender: {“Male”, “Female”}, domain knowledge level: {“Novice”, “Advanced”};
- Task information.

The EUDM dimensions need to be based on the above described input and the expected model components. As shown in GUMO (Heckman et al. 2007; 2005), several dimensions exist, but not all of them are required in the context of EUD. In particular, GUMO dimensions include personal characteristics, preferences and interest, needs and goal, knowledge and background, mental and physical state, context, behavior and individual traits. These dimensions build the basis for the selection of our model dimensions as presented in Table 6.1.

Table 6. 1 EUDM main dimensions

EUD Model Dimension	**Usage**
User	It includes personal characteristics that range from basic personal information like gender to knowledge background about EUD related

Profile	topics (e.g. web development, database concepts, etc.).
User Behavior	It refers to implicitly gathered user actions during the user-system interaction. These actions include mouse, keystroke and eye movements.
EUD Task Information	It includes EUD task information for every user session. This regards time related information and description of the EUD task (or subtasks) and the EUD constructed items by the end-user developer.
User Behavioral Traits	Inspired by GUMO's Individual Traits that usually refer to a broad range of user features that define the user as an individual, User Behavioral Traits refer to the EUD related user behavioral attributes including the RULES attributes presented in chapter 3. This dimension is different from the User Behavior dimension and is composed of different attributes. As explained later, Behavioral Traits can be inferred from User Behavior after a rule based reasoning within the constructed ontology.

As depicted in Table 6.1 have four basic model dimensions including User Profile, User Behavior, EUD Task Info and User Behavioral Traits.

The three first dimensions refer to the user context profile since they represent all the main user characteristics, task information and interaction (behavior) items during a EUD session. According to Rath et al. (2009) user context is "any information that can be used to characterize the situation of entities that are considered relevant to the interaction between a user and an application, including the user and the application themselves".

User Behavioral Traits are not included in the user context profile since they are assigned to values in the end of the session, after a rule based reasoning implementation. In simple words they infer new knowledge about the user and concern a basic part of the final user model.

After selecting the dimensions to be covered, we define the attributes that these dimensions should support. In particular:

- User Profile includes attributes of a user's personal characteristics such as gender and domain knowledge

- User Behavior includes client-side interaction elements that can be monitored in real time. These elements are about mouse movements, keyboard actions and eye behavior. The monitored mouse, keyboard and eye metrics are clustered in action-type related categories (e.g. mouse clicks, mouse hovers, eye fixations, etc.) representing the measurable items (metrics) presented in chapters 4 and 5.
- EUD Task Info refers to the EUD task related information such as the description of the tasks, subtasks and the items that need to be constructed (e.g. objects, form fields, relations, design elements, etc.). This information could also be used to evaluate user performance in EUD tasks.
- User Behavioral Traits include the five EUD behavioral attributes (RULES attributes) that, as been shown in chapter 3, tend to influence the end-user developers' performance and overall behavior during their interaction with EUD environments.

Figure 6.1 illustrates the overall concept of EUDM dimensions and attributes.

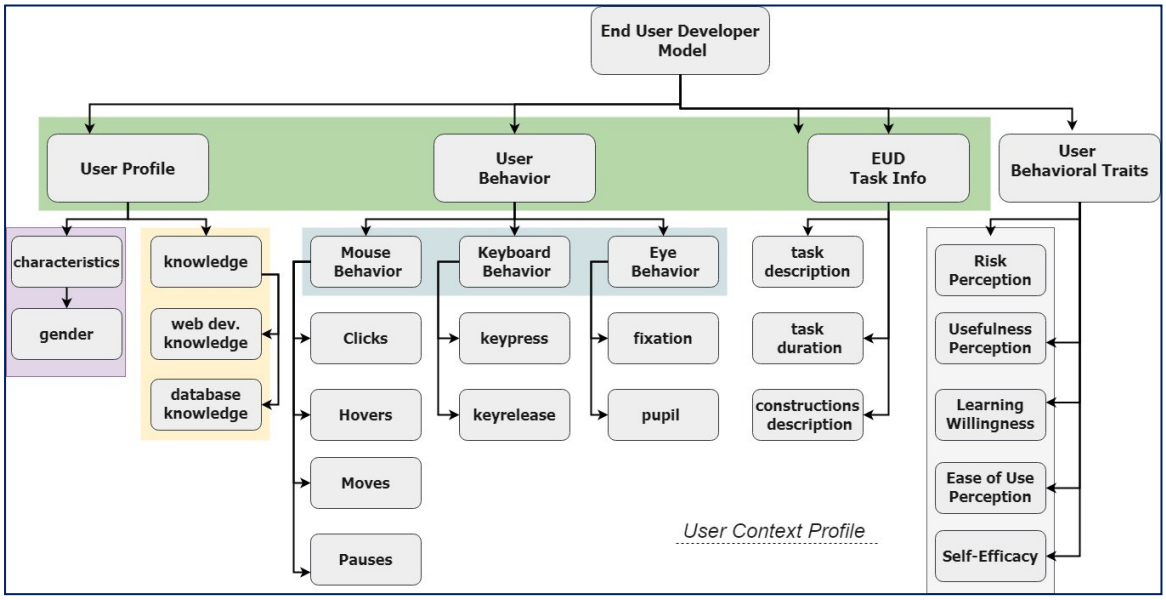

Figure 6. 1 Overall Concept of EUDM

As we see, the user context profile includes the main model dimensions of behavior (mouse, keyboard, and eyes), task info and user profile (gender, etc.). The user behavioral traits dimension includes the inferred values on RULES attributes after a rule-based reasoning.

Before dividing a user model into components, the stored information (raw data) about the user, needs to be analyzed and grouped for different types of information. In the following tables (Table 6.2-6.7) we present the type of information included in each dimension.

Table 6. 2 The dimension of User Profile

User Profile	**Attributes**
User Characteristics	gender
Knowledge	database knowledge (db_knowledgeLevel), web development knowledge (wb_knowledgeLevel)

Table 6. 3 The dimension of Mouse Behavior

Mouse Behavior	**Attributes**
Mouse Clicks	number of clicks outside direct movements (num-outlineClks), average time between clicks (avg-btwClk)
Mouse Hovers	number of all mouse hovers (num-hvr), number of mouse hovers that turned into mouse clicks (num-hvrToClk), average time from mouse hover to mouse click on the same element (avg-hvrToClk)
Mouse Moves	number of mouse movement (num-mv), number of non-direct movements (num-crv), number of all direct movements (num-dirMv), number of long direct movements (num-longDirMv), number of slow movements (num-slowMv), average time between direct movements (avg-timeBtwDir)
Mouse Pauses	average time of pauses (avg-pause), average time of pauses before clicks (avg-pauseBfrClk), number of mouse long pauses (num-longPauses), average time of long pauses (avg-longPauses)

Table 6. 4 The dimension of Keyboard Behavior

Keyboard Behavior	**Attributes**
Key Press	number of characters pressed every 4 seconds (speed-char), time elapsed between the characters pressed every 4 seconds (d2d-time)
Key Release	-

Table 6. 5 The dimension of Eye Behavior

Eye Behavior	**Attributes**

Fixations	number of fixations (num-fix), number of fixations that turned into clicks (num-fixToClk), average duration of fixation(avg-fixDur)
Pupil	pupil size average increment from initial state (avg-pupil)

Table 6. 6 The dimension of Task Information

EUD Task Information	**Attributes**
	task duration (task-duration), task description (task-desc), user constructed items, e.g. objects, forms, fields, relationships, etc. (constructed-item)

Table 6. 7 The dimension of Behavioral Traits

Behavioral Traits	**Attributes**
RULES	Self-Efficacy (SE), Risk-Perception (RP), Perceived Ease of Use (PEOU), Perceived Usefulness (PU), Learning Willingness (LW)

Overall, the EUDM is a collection of different linked entities that give a complete picture of the features needed to cover the main EUD behavioral, profile and task related aspects.

To semantically represent the EUDM we need to employ an ontology-based approach and take advantage of RDF/OWL technologies to define classes, instances and properties. Our ontology is named the “End-User Developer Ontology” (EUDO) and encompasses all the above mentioned EUDM dimensions and attributes in terms of distinct entities and properties.

EUDO ontology is both a storage and inference ontology, since some classes are responsible to store user data and behavior and some other classes are responsible to infer new knowledge and assign new values to their data type properties (e.g. property: Self-Efficacy = value: ‘High’).

In the following subsections we present the main steps we followed to build our User Developer Ontology and assign the end-user to the final model. These steps are the following:

- Modeling the end-user developer context profile and behavioral traits;

- Constructing the End-User Developer Ontology (EUDO);
- Using semantic rule-based reasoning to assign values to users' domain knowledge level;
- Using semantic rule-based reasoning to assign values to users' behavioral traits.

6.4.2 Modeling the User Developer Context Profile and Behavioral Traits

EUDO's user context profile is divided into different parts covering user profile aspects, task aspects and behavioral interaction aspects. In the remainder of this section, we describe the different entities that form the EUDO context profile, the data properties and the intended usage.

Representation of EUD user's context profile in OWL

Basis Steps:

1. Identification of Classes
2. Identification of Subclassses
3. Identification of Object Properties
4. Identification of Data Properties

Table 6.8 presents all the main EUDO entities (classes and subclasses) derived from every EUDM dimension previously described.

Table 6. 8 The EUDO entities as derived from the EUDM dimensions

Model Dimension	OWL Class	OWL Subclass
End-User Developer	End-user-developer	-
User Profile	User-profile	User-characteristics
User Profile	User-profile	User-knowledge
User Behavior	Mouse-behavior	Ms-clicks
User Behavior	Mouse-behavior	Ms-hovers
User Behavior	Mouse-behavior	Ms-moves
User Behavior	Mouse-behavior	Ms-pauses

User Behavior	Keyboard-behavior	Keypress
User Behavior	Eye-behavior	Eye-fixations
User Behavior	Eye-behavior	Pupil-size
EUD Task Information	EUDTask-info	-
User Behavioral Traits	Behavioral-traits	Self-Efficacy
User Behavioral Traits	Behavioral-traits	Risk-Perception
User Behavioral Traits	Behavioral-traits	Perceived-Ease-of-Use
User Behavioral Traits	Behavioral-traits	Perceived-Usefulness
User Behavioral Traits	Behavioral-traits	Learning-Willingness

Class: End-user developer

The OWL class *End-user-developer* is used to identify the users who interact with the EUD application. End-user-developer is the ontology's main class since it is responsible for 'hosting' all individuals (i.e. end-user developers). It is connected with all the other classes via object properties that define their relationship. The class of End-user-developer does not include any data properties (only session id) since all the necessary data properties are stored in the connected classes.

Table 6. 9 Properties of class End-user-developer

End-user-developer Object Properties		
Predicate	Range	Description
eudo:hasProfile	owl class: User-profile	The relation describes that every end-user developer individual has a user profile, where their personal and knowledge characteristics are stored.
eudo:hasEUDTaskInfo	owl class: EUD-TaskInfo	The relation describes that every end-user developer individual is related to a EUD task.
eudo:hasMouseBehavior	owl class: Mouse-behavior	The relation describes that every end-user developer individual has a particular mouse behavior during the EUD session. Mouse behavior is reflected on mouse related action metrics like hovers, clicks,

		pauses and moves.
eudo:hasKeystrokeBehavior	owl class: Keystroke-behavior	The relation describes that every end-user developer individual has a particular keystroke behavior during the EUD session.
eudo:hasEyeBehavior	owl class: Eye-behavior	The relation describes that every end-user developer individual has a particular eye behavior during the EUD session. Users' eye behavior is reflected on fixations and pupil size metrics.
eudo:hasTraits	owl class: Behavioral-traits	The relation describes that every end-user developer individual is characterized by a set of behavioral traits (the RULES attributes) where each trait is assigned to a value (e.g. Self-Efficacy='High'). These values are inferred after the evaluation of individuals' interaction behavior (mouse, keystroke and eye).
End-user-developer Data Properties		
Predicate	Range	Description
sessionid	xsd: int	This property defines the individual end-user developer's session id. Session id stores the session's id number that is generated in the beginning of the EUD system interaction.

Class: User-profile

The OWL class User-profile defines the end-user developer's personal and expertise related characteristics. In User-profile subclass User-characteristics we store user's gender as it has been shown to significantly influence end-user developers' performance and behavior (see chapter 2). Other personal user data may also be stored, such as: username, location, age, etc.

As described in chapter 2, another important factor for the end-user developers is their expertise level. Hence, in subclass User-knowledge we store users' knowledge-level (in a scale from 1 to 5) on EUD related concepts like database and web development experience.

Both gender and knowledge-level are explicitly provided by the user in the beginning of the user session.

Table 6. 10 Properties of class User Profile

User Profile Data Properties		
Predicate	Range	Description
eudo:gender	xsd:string	This property defines the user's gender (Male or Female). Gender is explicitly provided by the user.
eudo:db_knowledgeLevel	xsd:int	This property defines the user's experience/knowledge on database development concepts. It is explicitly provided in a scale from 1 to 5.
eudo:wb_knowledgeLevel	xsd:int	This property defines the user's experience/knowledge on web development tasks. It is explicitly provided in a scale from 1 to 5.
eudo:IsNovice	xsd:boolean	This property defines whether a user is Novice (value=True). The value is inferred after a rule-based reasoning is implemented on the user's database and web development knowledge level.
eudo:IsAdvanced	xsd:boolean	This property defines whether a user is Advanced (value=True). The value is inferred after a rule-based reasoning is implemented on the user's database and web development knowledge level.

Class: EUDTask-info

The OWL class *EUDTask-info* defines these characteristics that allow identifying the EUD task. EUDTask-info includes the EUD task duration in time, its description and the set of EUD constructed items. These items can be the user's constructed entities, relationships, forms, fields, design or other elements that compose the under development application. EUD constructed items can also be used by the system to evaluate user performance, as presented in previous chapters.

Table 6. 11 Properties of class EUDTask-info

EUDTask-info Data Properties		
Predicate	Range	Description
eudo:task-desc	rdfs:Literal	This property defines the EUD task's name and description that is explicitly provided by the user in

		the corresponding field forms.
eudo:task-duration	xsd:decimal	This property defines EUD task duration in time. Duration represents the duration of an active user session,
eudo:constructed-item	rdfs:Literal	This property defines the user's EUD constructed items (e.g. objects, relationships, forms, etc.). Constructed items can also allow the evaluation of user performance for a known task.

Class: Mouse-behavior

The OWL class *Mouse-behavior* defines the user's mouse actions during the user – EUD system interaction. The Mouse-behavior class stores all mouse actions shown to affect user's behavioral traits, as presented in chapter 5. It consists of the subclasses of Ms-hovers, Ms-clicks, Ms-moves and Ms-pauses. Every subclass stores the related mouse metrics in terms of data properties.

Table 6. 12 Properties of class Mouse-behavior

Mouse-behavior Data Properties		
Predicate	Range	Description
eudo:num-outlineClcks	xsd:int	This property defines the number of the mouse clicks that occurred outside straight lines (or else 'direct movements') during a EUD session.
eudo:avg-btwClcks	xsd:decimal	This property defines the average time (of the active user) between mouse clicks during a EUD session.
eudo:num-hvr	xsd:int	This property defines the number of all mouse hovers in a EUD session.
eudo:num-hvrToClck	xsd:int	This property defines the number of mouse hovers that turned into mouse clicks
eudo:avg-hvrToClck	xsd:decimal	This property defines the average time from mouse hover to mouse click on the same element.
eudo:num-crv	xsd:int	This property defines the number of non-direct (curved) movements in a EUD session
eudo:num-mv	xsd:int	This property defines the number of mouse

		movements in a EUD session.
eudo:longDirMv	xsd:int	This property defines the number of long direct movements in a EUD session
eudo:num-dirMv	xsd:int	This property defines the number of direct movements in a EUD session
eudo:avg-timeBtwDir	xsd:decimal	This property defines the ratio of the average pauses time between direct movements to the EUD task duration time
eudo:num-slowMv	xsd:int	This property defines the number of slow movements in a EUD session.
eudo:avg-pause	xsd:decimal	This property defines the ratio of average time of pauses to the EUD task duration time.
eudo:avg-longPauses	xsd:decimal	This property defines the ratio of average time of long pauses (time elapsed since last movement >5sec) to the task duration time
eudo:num-longPauses	xsd:int	This property defines the number of mouse long pauses in a EUD session

Class: Keystroke-behavior

The OWL class *Keystroke-behavior* defines the user's keyboard actions during the user – EUD system interaction. The Keystroke-behavior class stores all keystroke actions shown to affect user's behavioral traits, as presented in chapter 5. It consists of the subclass of Keypress. Keyup is not forming a subclass since keyup events were shown not to affect users' behavioral traits. However, this does not exclude its co-inclusion in other similar EUD ontologies.

Table 6. 13 Properties of class Keystroke-behavior

Keystroke-behavior Data Properties		
Predicate	Range	Description
eudo:d2d-time	xsd: decimal	This property defines the time elapsed between the characters pressed every 4 seconds.
eudo:speed-char	xsd: decimal	This property defines the number of characters pressed every 4 seconds.

Class: Eye-behavior

The OWL class *Eye-behavior* defines the user's eye movements during the user – EUD system interaction. The Eye-behavior class stores all eye related actions shown to affect user's behavioral traits, as presented in chapter 4. It consists of the subclass of Fixations and Pupil.

Table 6. 14 Properties of class Eye-behavior

Eye Behavior Object Properties		
Predicate	Range	Description
eudo:num-fix	xsd: int	This property defines the total number of a user's eye fixations that occurred during a EUD session.
eudo:num-fixToClk	xsd: int	This property defines the number of a user's eye fixations that turned into mouse clicks during a EUD session.
eudo:avg-fixDur	xsd: decimal	This property defines average time of fixation duration during a EUD session.
eudo:avg-pupil	xsd: decimal	This property defines average increment of pupil size in a EUD session.

Class: Behavioral-traits

The OWL class Behavioral-traits represents the end-user developers' behavioral items, like perception and acceptance items, that have been shown to affect end-user developers' performance and/or be affected by human factors such as gender. This class is composed of a set of subclasses based on the presented in chapter 2 RULES attributes (Risk-Perception, Usefulness-Perception, Learning-Willingness, Ease-of Use-Perception and Self-Efficacy). Every subclass includes a set of related data properties responsible to assign a value of 'Low' or 'High' to the corresponding attribute.

Mouse, keystroke and eye behavior metrics (data properties) that affect one or more of the RULES attributes (i.e. Behavioral-traits subclasses) are also assigned a value of 'Low' or 'High' in the corresponding affected subclass after a rule based evaluation that is later presented in this section. These sets of sub-properties are excluded from Table 6.15 below.

Table 6. 15 Properties of class Behavioral-traits

Behavioral-traits Data Properties		
Predicate	Range/Type	Description
SE	parent property	This property includes the sub-properties that define these behavioral attributes affecting Self-Efficacy Trait. It also includes the sub-property responsible to assign the final Self-Efficacy value ('Low' or 'High') to the user model.
SE-value	xsd:string, sub-property of SE	This property defines the user's final Self-Efficacy value ('Low' or 'High') for a EUD session.
RP	parent property	This property includes the sub-properties that define these behavioral attributes affecting Risk-Perception Trait. It also includes the sub-property responsible to assign the final Risk-Perception value ('Low' or 'High') to the user model.
RP-value	xsd:string, sub-property of RP	This property defines the user's final Risk-Perception value ('Low' or 'High') for a EUD session.
PEOU	parent property	This property includes the sub-properties that define these behavioral attributes affecting Perceived-Ease-of-Use Trait. It also includes the sub-property responsible to assign the final Perceived-Ease-of-Use value ('Low' or 'High') to the user model.
PEOU-value	xsd:string, sub-property of PEOU	This property defines the user's final Perceived-Ease-of-Use value ('Low' or 'High') for a EUD session.
PU	parent property	This property includes the sub-properties that define these behavioral attributes affecting Perceived-Usefulness Trait. It also includes the sub-property responsible to assign the final Perceived-Usefulness value ('Low' or 'High') to the user model.
PU-value	xsd:string, sub-property of PU	This property defines the user's final Perceived-Usefulness value ('Low' or 'High') for a EUD session.
LW	parent property	This property includes the sub-properties that define these behavioral attributes affecting Learning-Willingness Trait. It also includes the sub-property responsible to assign the final Learning-Willingness value ('Low' or 'High') to the user model.
LW-value	xsd:string, sub-property of LW	This property defines the user's final Learning-Willingness value ('Low' or 'High') for a EUD session.

6.4.3 Constructing the End-User Developer Ontology

To construct the user ontology and generate the complete OWL scheme we used the Protégé 5.2.0 desktop version ontology platform (Protégé). Protégé is a free and open-

source OWL ontology development environment. It provides a suite of tools to construct domain models and knowledge-based applications with ontologies.

Protégé offers a number of reasoning engines. The OWL Reasoner we used to evaluate and debug our Ontology is Pellet Reasoner. The Pellet Reasoner Plug-in, version includes data type reasoning, SWRL support, etc. Pellet (incremental) option automatically checks the consistency of the ontology every time an entity change is made to the ontology.

According to the Debugging Reasoner, the constructed EUDO ontology is coherent and consistent. This allows the further processing via rule-based reasoning mechanisms.
To build and represent EUDO a few more Protégé plugins have been used such as OntoGraf for the entities' graphic representation, as well as SPARQL, Snap-SPARQL (Horridge and Musen, 2015), and SWRL plugins for the rule-based reasoning presented in the next subsection.

The general proposed EUDO hierarchy is presented in Figure 6.2, which shows the class hierarchy and class connections as they are constructed in Protégé editor.

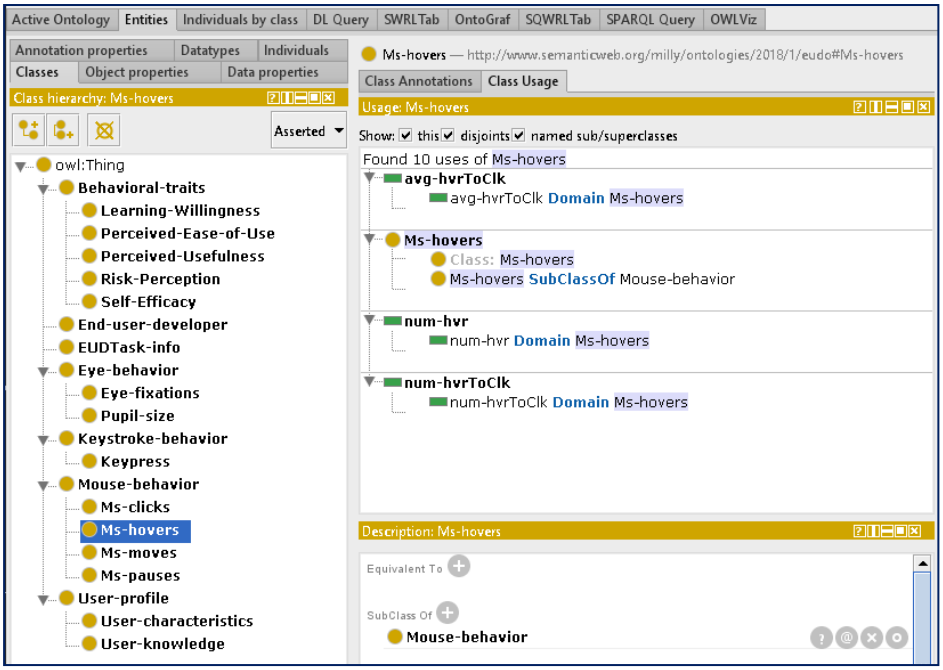

Figure 6. 2 EUDO's class hierarchy

In OWL ontologies, interrelationships between classes can be created using object properties. The object properties created for the EUDO concepts are shown in Figure 6.3.

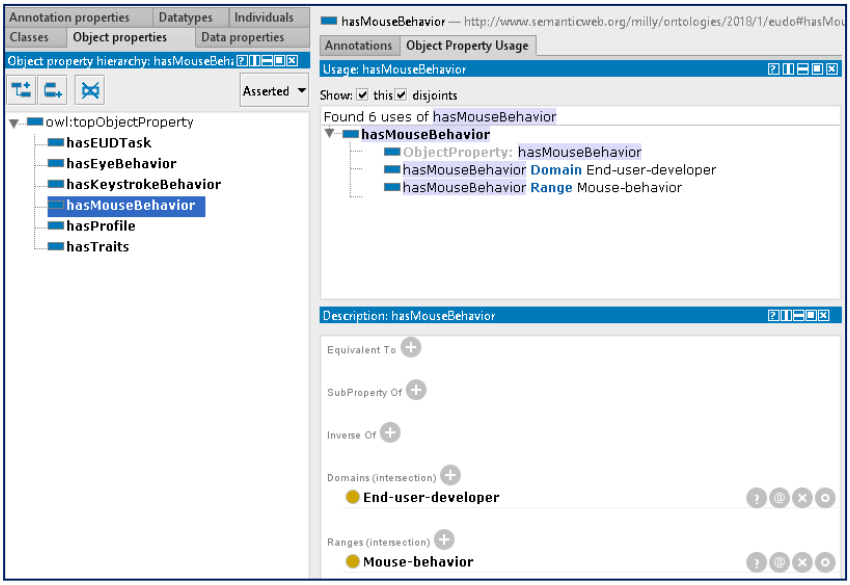

Figure 6. 3 EUDO's Object Properties

In Protégé environments, the Data Properties Tab displays the hierarchy of data properties based on subPropertyOf assertions. Figure 6.4 below depicts (most of) the EUDO's data properties. As we see they are organized in parent properties and sub-properties to distinguish the classes they belong to.

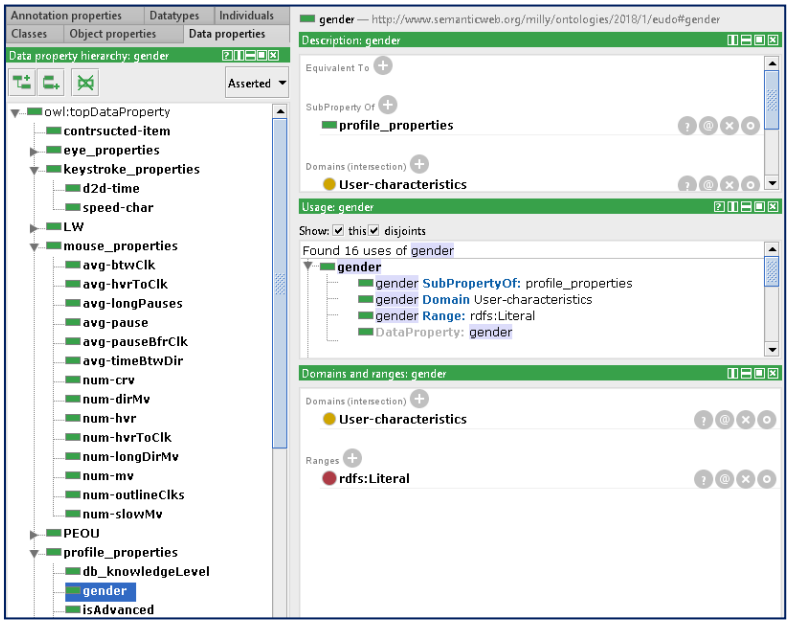

Figure 6. 4 EUDO's Data Properties

A fragment of the EUDO's OWL generated syntax is provided in Table II (Annex II).

Graph-based representation of EUDO

We used the Web-based Visualization of Ontologies (WebVowl) software (Lohmann, 2015) and ProtégéVOWL plugin to graphically present the constructed EUDO ontology. VOWL visualizations are automatically generated from JSON or XML files into which the ontologies need to be converted. Figure 6.5 depicts an excerpt of the ontology classes including object and data properties as well.

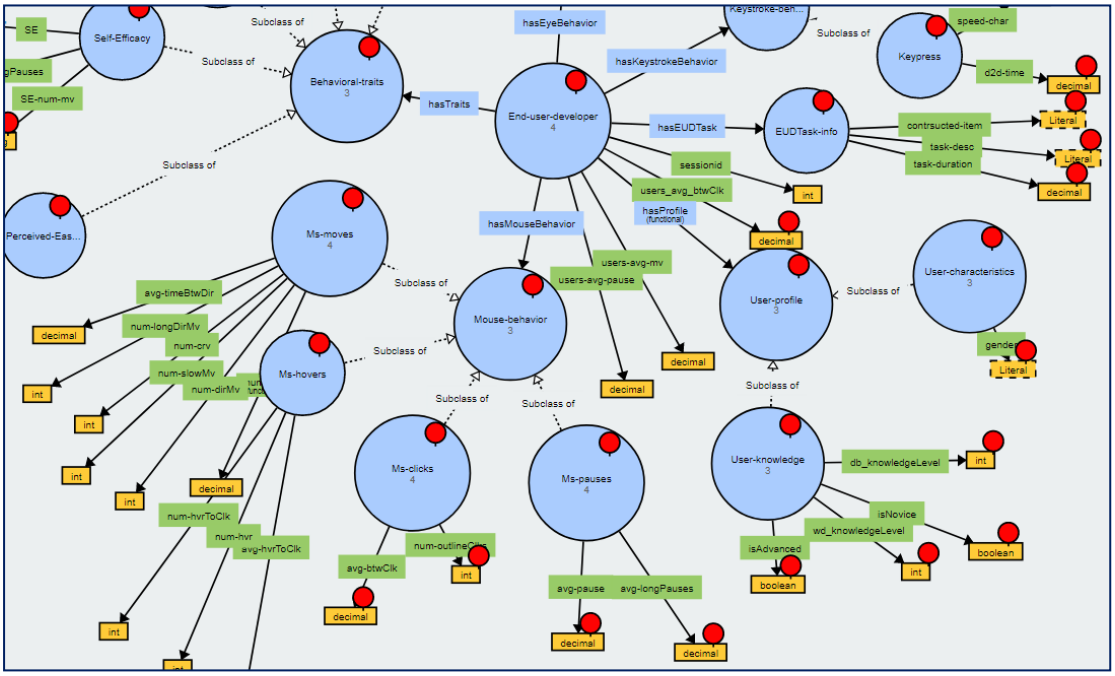

Figure 6. 5 Fragment of graph-based representation of EUDO using webvowl

6.4.4 Designing the Rule-Based Modeling Framework

To infer the final user model we have implemented a rule-based reasoning methodology within the ontology. In particular we have constructed a set of pre-defined rules to characterize users as 'Novice' or Advanced' according to their explicitly provided level of knowledge in some EUD related concepts (like web and database development). Rule-based reasoning has also been used to assign values of 'Low' and 'High' to the implicitly derived RULES attributes which are reflected in the Behavioral Traits items (self-efficacy, perceived ease of use, etc.)

To implement rule-based reasoning in Protégé we used both SWRL rules and SPARQL queries. SWRL is used for reasoning on the users' knowledge level, while SPARQL based reasoning is used to assign values to the users' Behavioral Traits items.

The decision to use both approaches was made because of the open world assumptions (OWA) limitations on our EUDO ontology. Since SWRL does not support non monotonic reasoning we could not infer any particular knowledge on the users' Behavioral Traits based on their mouse and keyboard feedback. SPARQL on the other hand and particularly the Snap SPARQL plug-in offers a non-monotonic reasoning functionality in the Protégé platform. SWRL could only be used for reasoning on users' knowledge since this assignment will be implemented only once (during the first user system-interaction) providing the system with new knowledge regarding the user's domain knowledge level. No updates or modifications in the ontology will be placed afterwards. However, the values of Behavioral Traits need to be updated in every user-system interaction (i.e. in every EUD session) possibly forming a different user model and SWRL, as explained, cannot be used for ontology modifications.

Following we describe the rule-based reasoning approach we followed to assign values in the data properties of both concepts, i.e. User Knowledge and User Behavioral Traits.

6.4.4.1 Rule-Based Reasoning for Modeling User Knowledge

In our User Knowledge reasoning implementation we adopt SWRL logic to handle OWL format user data. As explained earlier, to execute the SWRL reasoning we use Pellet Reasoner in Protégé desktop installation.

Our aim is to assign values to the Boolean type data properties of IsNovice and IsAdvanced and also to the domain specific data properties of web development and database development (wd_Advanced, wd_Novice, db_advanced, db_Novice).

To build numeric comparisons for the data type property values assigned to class of User-knowledge, we use SWRL built-ins (e.g. greaterThan()) to 'decide' weather a user is perceived as Novice (IsNovice=True) or Advanced (IsAdvancde=True) in terms of domain knowledge. Novice and Advance estimations derive from the explicitly

provided users' perceived level of expertise on particular EUD related concepts like web development and database logic.

Figure 6.6 depicts the overall concept of the rule-based approach (demonstrating a rule example) within the constructed owl defined ontology.

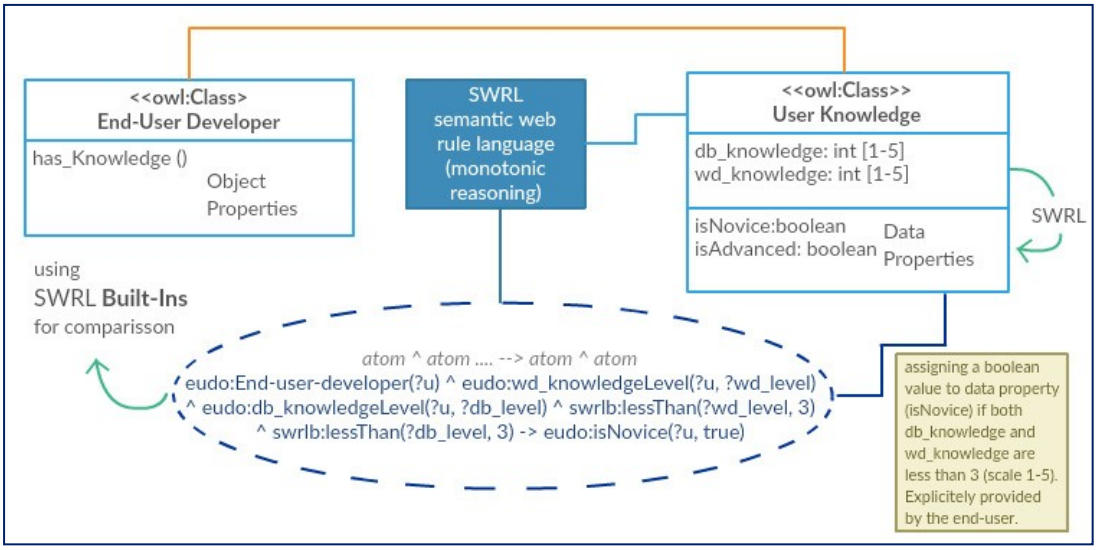

Figure 6. 6 Overall SWRL concept for modeling User Knowledge

As depicted in Figure 6.6 the explicitly provided user knowledge level on database and web development is rated on a scale of 1 to 5 and stored in the following data type properties: db_knowledge and wd_knowledge. Then, an SWRL rule controls the knowledge score in both domains and assigns a value of True to either IsNovice() or IsAdvanced() property.

In case of a significantly different knowledge level between the two domains a separate result is generated for the knowledge level about database and web development, according to the following Boolean type data type properties: db_Advanced, db_Novice, wd_Advanced, wd_Novice.

In Table 6.16 below we provide the SWRL rules we constructed to infer the above described knowledge on users' expertise level and assign it in EUDO's corresponding attributes.

Table 6. 16 SWRL rules to infer knowledge on users' expertise level

Data Property	SWRL Expression
isNovice	eudo:End-user-developer(?u) ^ eudo:wd_knowledgeLevel(?u, ?wd_level) ^ eudo:db_knowledgeLevel(?u, ?db_level) ^ swrlb:lessThan(?wd_level, 3) ^ swrlb:lessThan(?db_level, 3) -> eudo:isNovice(?u, true)
isAdvanced	eudo:End-user-developer(?u) ^ eudo:wd_knowledgeLevel(?u, ?wd_level) ^ eudo:db_knowledgeLevel(?u, ?db_level) ^ swrlb:greaterThanOrEqual(?db_level, 3) ^ swrlb:greaterThanOrEqual(?wd_level, 3) -> eudo:isAdvanced(?u, true)
db_Advanced; wd_Novice	eudo:End-user-developer(?u) ^ eudo:wd_knowledgeLevel(?u, ?wd_level) ^ eudo:db_knowledgeLevel(?u, ?db_level) ^ swrlb:greaterThanOrEqual(?db_level, 3) ^ swrlb:lessThan(?wd_level, 3) -> eudo:db_Advanced(?u, true) ^ eudo:wd_Novice(?u, true)
wd_Advanced; db_Novice	eudo:End-user-developer(?u) ^ eudo:wd_knowledgeLevel(?u, ?wd_level) ^ eudo:db_knowledgeLevel(?u, ?db_level) ^ swrlb:greaterThanOrEqual(?wd_level, 3) ^ swrlb:lessThan(?dd_level, 3) -> eudo:wd_Advanced(?u, true) ^ eudo:db_Novice(?u, true)

6.4.4.2 Rule-based Reasoning for Modeling User Behavioral Traits

Since SWRL does not support monotonic reasoning, we need to use SPARQL-based reasoning in order to update EUDO values such as the Rules attributes (in Behavioral traits class), in every user session. Protégé SPARQL-based reasoning is only supported via Snap SPARQL. Snap SPARQL allows for non-monotonic reasoning and ontology modification. However, the current version (5.2) does not support SPARQL sub queries and in our rule-based modeling we need to use sub queries to make comparisons between user's data properties and average values of data properties. For this reason we also need to use the basic SPARQL plug-in to query and update these average values. Hence, we use some non-monotonic reasoning via snap SPARQL to update the RULES' values for every user and basic SPARQL version to compare and edit average values between users.

Figure 6.7 below demonstrates the overall rule-based reasoning concept for the final definition of the Behavioral Traits values for every user.

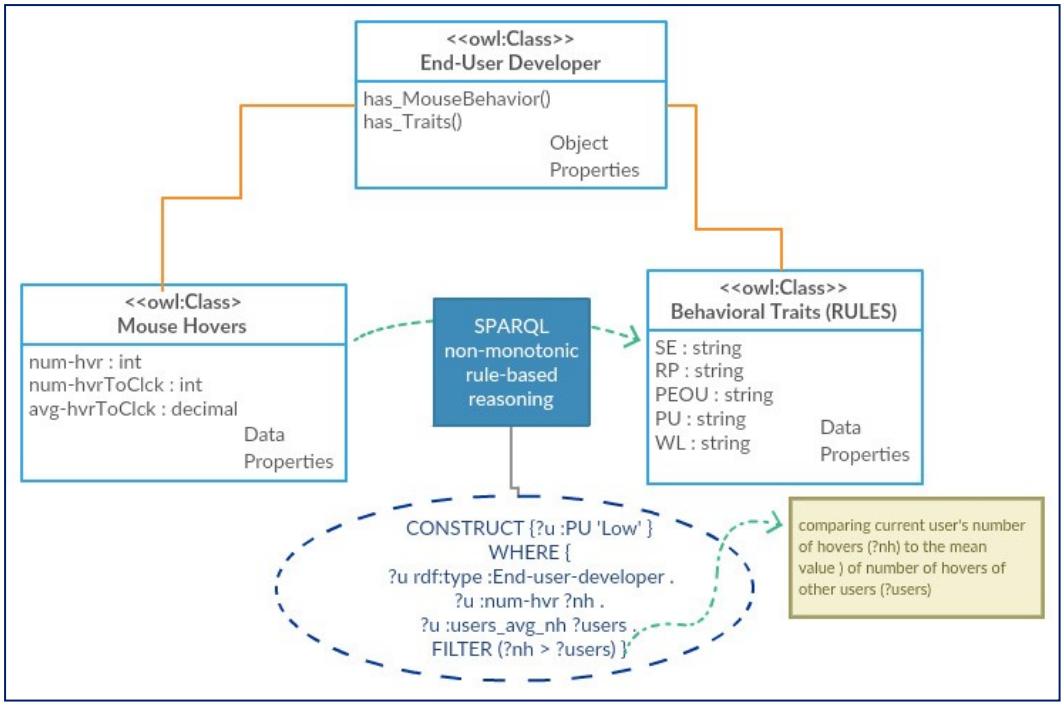

Figure 6. 7 Overall SPRQL reasoning for modeling Behavioral Traits

As depicted in the example provided in Figure 6.7, the aim of the SPRQL-based non monotonic reasoning is to assign a string value of "Low" or "High to every attribute in RULES model, as defined in the class of Behavioral Traits. This has to be conducted for every new user system interaction and the final model can be dynamically updated after every user-system interaction. Value assignment is based on the users' mouse and/or keyboard and/or eye behavior; the data type property values assigned to the behavioral classes are evaluated (compared to the average values of other users) and based on specific conditions (according to the correlation results presented in chapter 4 and 5) the RULES attributes are assigned to a string value of "Low" or "High". Thus the final user model, regarding Behavioral Traits can be defined.

Table 6.17 below demonstrates the reasoning logic for every RULES (Behavioral Trait) item as well as the behavioral metrics (mouse, keyboard and eye) that are taken into account as the main rules' parameters for the generated outcome.

Table 6. 17 Reasoning logic to assign data values to RULES attributes

Behavioral Trait Item	Assigned Value	Rule logic	User Behavior Item (mouse, keyboard, eye metric)
Self-Efficacy	High	If more or equal to	avg-longPauses (Ms-pauses), num-fix (Eye-fixations), num-fixToClk (Eye-fixations), avg-pupil

		all users' average	(Pupil-size)
	Low	If less than all users' average	num-mv (Ms-moves), num-longPauses (Ms-pauses), d2d-time (Keypress)
Risk-Perception	High	If more or equal to all users' average	avg-pupil (Pupil-size)
	Low	If less than all users' average	avg-btwClk (Ms-clicks), avg-pause (Ms-pauses)
Perceived-Ease-of-Use	High	If more or equal to all users' average	num-fixToClk (Eye-fixations)
	Low	If less than all users' average	num-outlineClks(Ms-moves), num-crv(Ms-moves), num-longDirMv(Ms-moves num-mv(-), num-dirMv(Ms-move), num-slowMv (Ms-moves), num-hvrToClk(Ms-hovers), speed-char(Keypress)
Perceived-Usefulness	High	If more or equal to all users' average	avg-pause (Ms-pauses), avg-pauseBfrClk (Ms-pauses), avg-fixDur (Eye-fixations)
	Low	If less than all users' average	num-hvr (Ms-hovers), num-crv(Ms-moves), num-longDirMv (Ms-moves), num-mv (Ms-moves), num-dirMv (Ms-moves), num-slowMv (Ms-moves)
Learning-Willingness	High	If more or equal to all users' average	avg-pause(Ms-pauses), avg-hvrToClk(Ms-hovers), avg-btwClk(Ms-clicks), avg-timeBtwDir(Ms-moves), avg-pauseBfrClk(ms-moves)
	Low	If less than all users' average	-

Following we demonstrate some code snippets of the steps to perform our SPARQL-based non monotonic reasoning.

To update the average values of all users' behavioral items (mouse, keyboard and eye) we have constructed the following SPARQL query:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://www.semanticweb.org/milly/ontologies/2018/1/eudo#>
CONSTRUCT {?u :avg-btwClk ?avg.}
WHERE {?u a :End-user-developer .
```

```
{SELECT (AVG(?avgpause) AS ?avg)
WHERE{?u :avg-pause ?avgpause }}
}}
```

As we see in the above update (construct) query we need to store all users' average (?avg) value as a data property, since Snap-SPARQL does not allow us to dynamically calculate it and use it in our construct query. For this reason we have defined a new data property for every mouse metric, responsible to store all users' average value for this mouse metric.

Alternatively, average values can also be calculated via SQWRL (O'Connor and Das, 2009) as depicted in the following rule snippet. However, SQWRL works only for querying the ontology and does not support ontology modifications.

To update the data properties values ('Low' or 'High') of the Behavioral Traits class we have built the following query to compare the active user's average behavioral values to the other users stored in the ontology.

```
CONSTRUCT {?u :RP_avg-pause 'Low' .}
WHERE { ?u rdf:type :End-user-developer .
   ?u :avg-pause ?avgpause .
   ?u :users_avg=pause ?users .
FILTER (?avgpause > ?users)}
```

In case no specific behavior is detected, a Behavioral Trait item can be characterized as 'Undefined' until updated in a later user session.

6.4.5 Use Case Scenario

To evaluate our ontology's reasoning approach we have constructed four end-user profiles as EUDO individuals in Protégé desktop environment. As presented in Table 6.18 below, the inserted data for each user refer to certain user characteristics and knowledge properties (gender, data base knowledge and web development knowledge) and to some mouse behavioral properties (average time between clicks and average pause time).

Table 6. 18 Description of user profiles evaluated in EUDO's reasoning–based modeling approach

user	gender	db_knowledge Level	wb_knowledge Level	avg-btwClk	avg-pause
UserA	Male	4	4	10147.25	82.00
UserB	Male	3	3	8736.08	7746.50
UserC	Female	2	1	86.00	51.00
UserD	Female	4	1	10289.45	49.00

All users' data (both profile and mouse behavioral) is assigned to the appropriate data type properties as depicted in Figure 6.8 below. The inserted data values represent real user knowledge and behavior as being captured through the mouse tracking experiment presented in chapter 5.

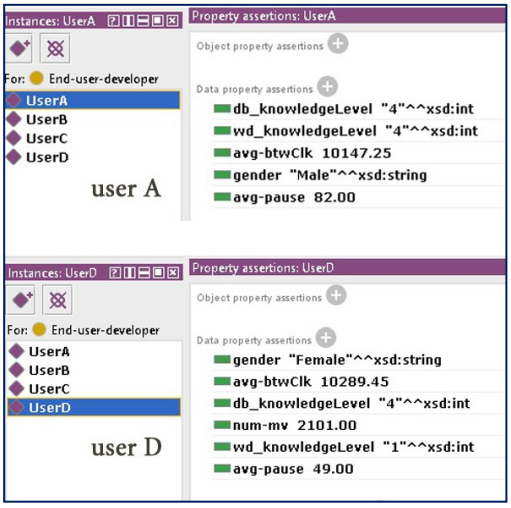

Figure 6. 8 Example of asserted property data to users A and D

Inferring EUDO Knowledge level

To infer knowledge from the asserted user profile OWL data properties we used Pellet Sirin (2007) as reasoning engine and ran SWRL statements via Protégé SWRL tab plugin.

Four SWRL statements were constructed to run a rule for every EUDO knowledge level:

- isAdvanced (True): In case user is advanced both in database and web development knowledge.

- isNovice (True): In case user is novice both in database and web development knowledge.
- wd_Advanced (True): In case user is advanced only in web development knowledge.
- wd_Novice (True): In case user is novice only in web development knowledge.
- db_Advanced (True): In case user is advanced only in database knowledge.
- db_Novice (True): In case user is novice both only in web development knowledge.

The constructed SWRL rules are depicted in Figure 6.9 below, where all profile properties are visible as well.

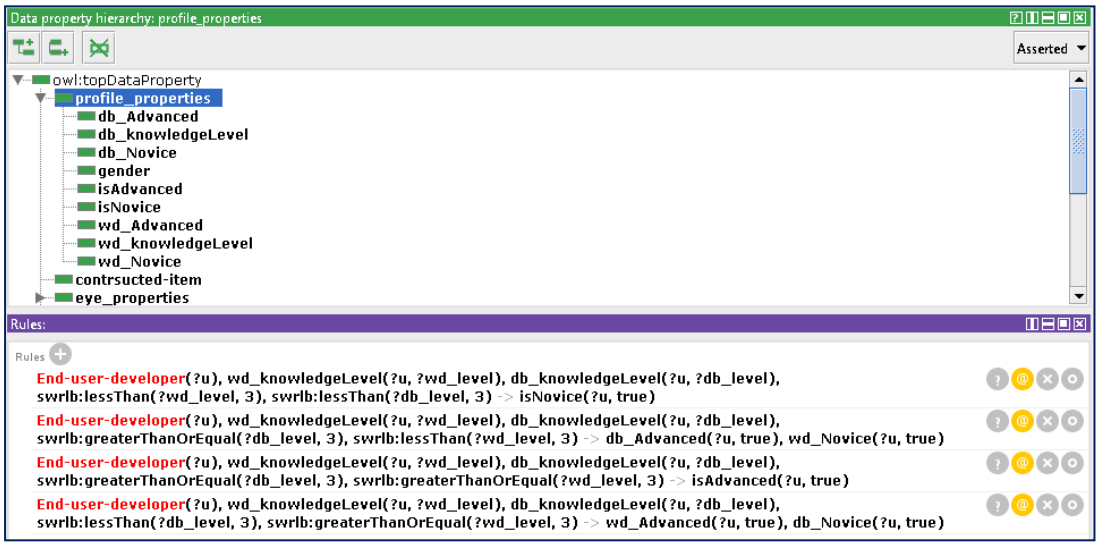

Figure 6. 9 EUDO SWRL rules

After executing the rules, the inferred EUDO knowledge is appeared in every individual's property assertion tab. As we see in Figure 6.10. Instances of users A, B, C and D have been assigned the following data properties' values:

- User A: isAdvanced (True)
- User B: isAdvanced (True)
- User C: isNovice (True)
- User D: wd_Novice (True); db_Advanced (True)

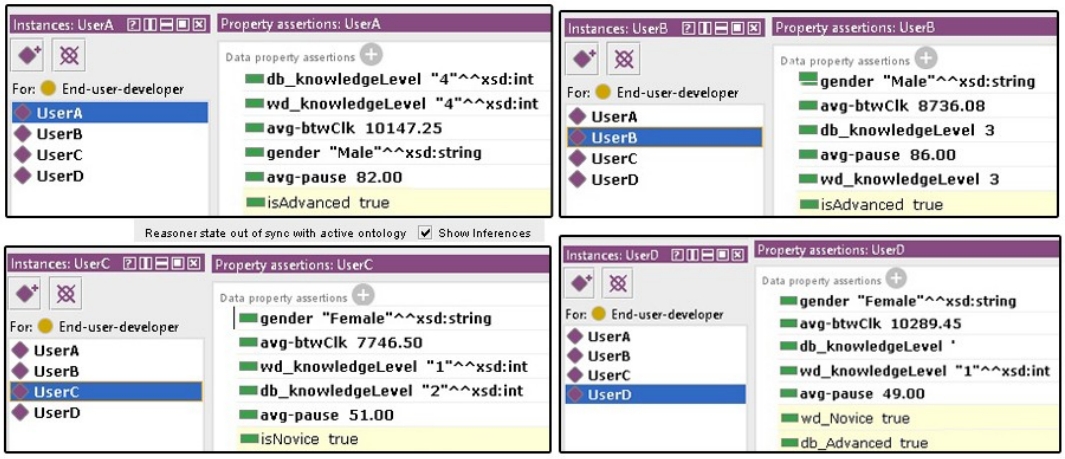

Figure 6. 10 Inferred Properties for EUDO user knowledge

Inferring EUDO Behavioral Traits' level

Concerning the reasoning logic presented in Table 6.17 we have built and executed a series of SPARQL queries in Protégé Snap-SPARQL and SPARQL tabs.

Figures 6.11 – 6.13 below depicts the SPARQL queries to calculate the average pause time (?avgpause=67.0ms) and the average time between mouse clicks for all users (?avgBtwClk=92229.82 ms).

Figure 6. 11 SPARQL query to select users' average time of pauses

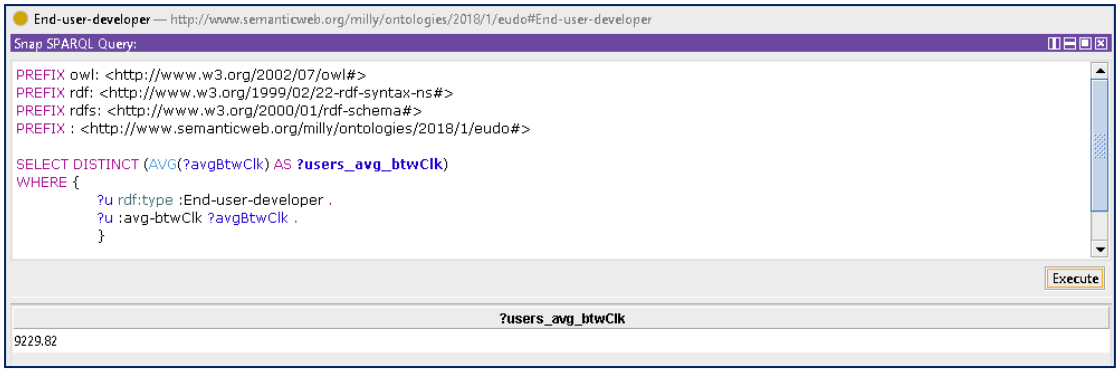

Figure 6. 12 SPARQL query to select users' average time between mouse clicks

Figure 6.13 below demonstrates the SPARQL query to assign a value of 'High' to users' Perceived Usefulness (PU), based on the comparison between their average mouse value and the average mouse value of the all users. As we see, users A and B have been assigned to a high level of Perceived Usefulness.

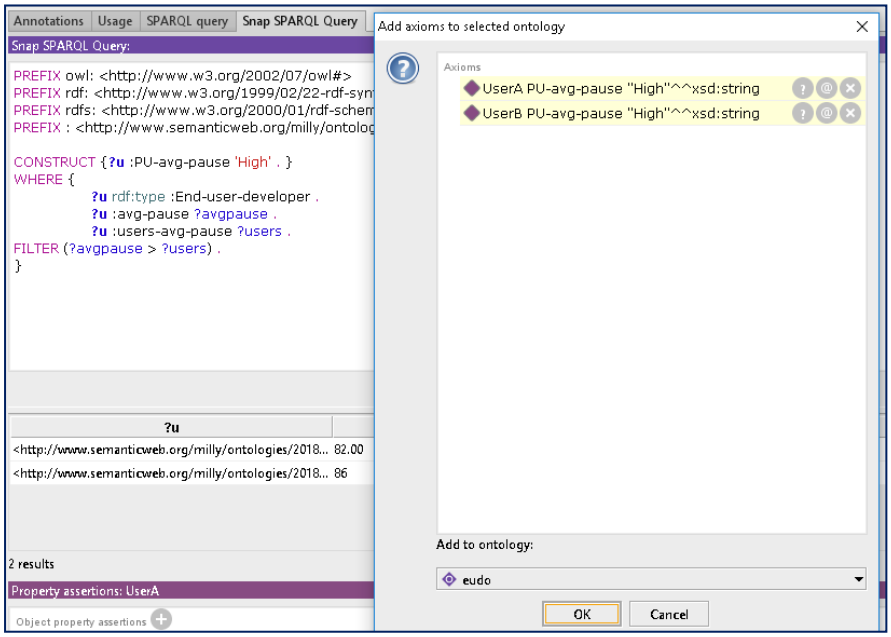

Figure 6. 13 SPARQL query to assign value to RULES property

After executing the query, the final property assignment automatically appears in each user's property assertion tab (see Figure 6.14)

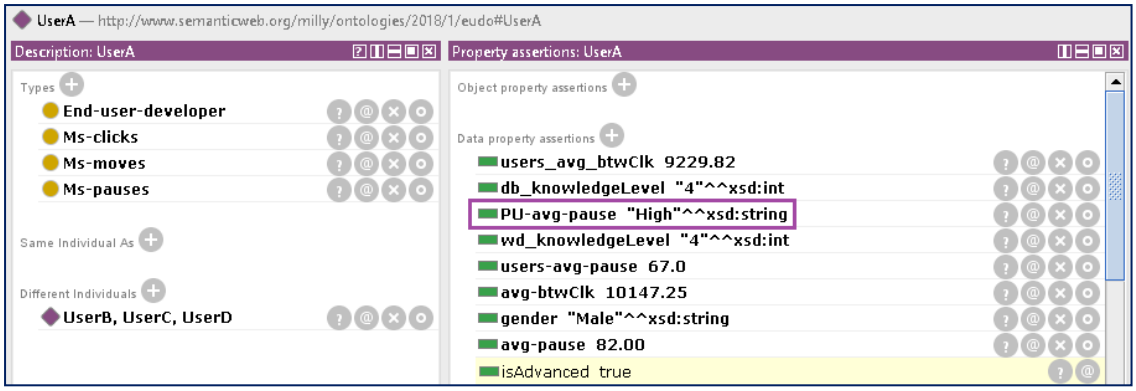

Figure 6. 14 Result of the executed query for value assignment

Similarly, Figure 6.15 shows the SPARQL query to update the value of users' Risk-Perception (RP). Users A and D have been assigned to a "Low" value of Risk-Perception. Result is visible in every user's property assertion tab (see Figure 6.16).

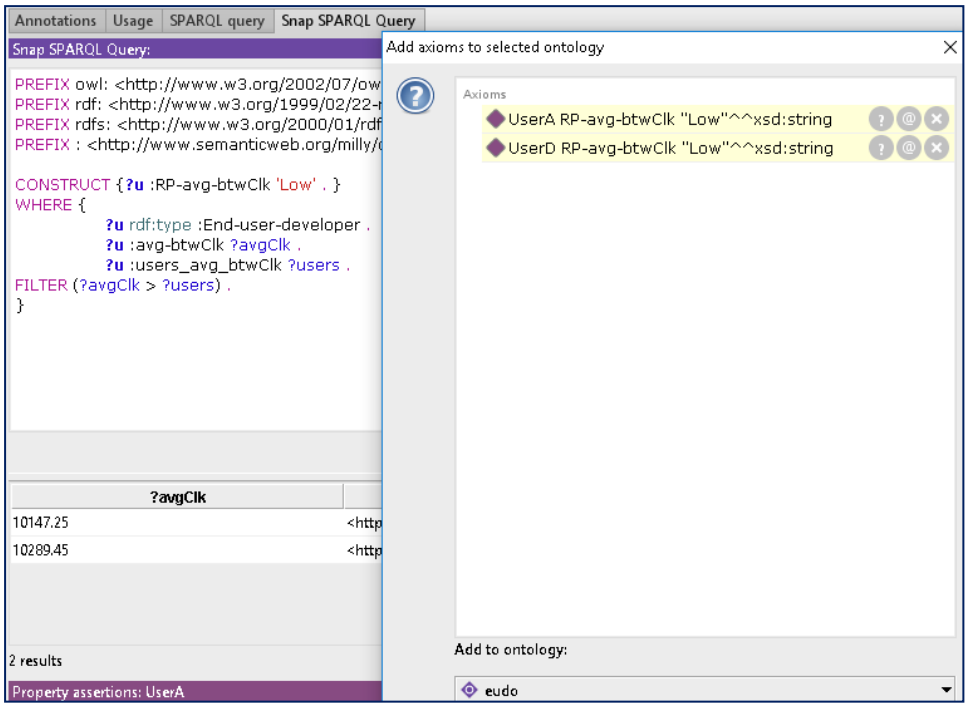

Figure 6. 15 Query to assign value to RULES property

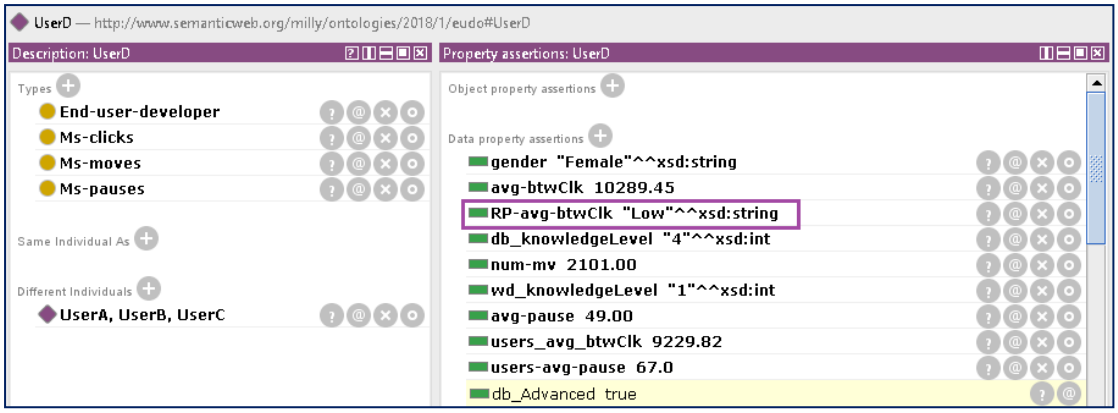

Figure 6. 16 Result of the executed query for value assignment

6.5.6 Results and Discussion

Use case scenarios showed that EUDO rule based reasoning for user modeling is efficiently working in Protégé environment. Users have been assigned to knowledge values according to their explicit feedback and to behavioral values according to their implicitly monitored mouse behavior.

Their profile properties have been successfully updated and can be visible for every user.

Table 6.19 presents the generated results for the use case users' profile properties. Gender property is included to provide a comprehensive user model presentation.

Table 6. 19 Description of user profile results from the EUDO's rule-based reasoning

User	Gender	Knowledge Level	Perceived Usefulness	Risk-Perception
UserA	Male	Advanced	High	Low
UserB	Male	Advanced	High	-
UserC	Female	Novice	-	-
UserD	Female	Advanced in web development; Novice in database knowledge	-	Low

As explained, SPARQL-based monotonic reasoning allows to regularly updating Behavioral Traits' values between user sessions.

6.5 Summary and Contribution of the Chapter

In this chapter we demonstrated our End-User Developer Modeling (EUDM) approach.

By defining the model dimensions we came up with the model structure and its main components including the user profile, the EUD overlay and the cognitive overlay. The model dimensions include the user profile (in terms of user characteristics), user behavior (as reflected in mouse/keystroke/eye movements), EUD task information (regarding EUD constructions) and behavioral traits (including the RULES attributes presented in chapter 3).

To semantically represent the model we employed an ontology-based approach and took advantage of RDF/OWL technologies to define classes, instances and properties. SWRL and SPARQL-based reasoning was used to infer the final data type properties' values regarding knowledge and behavioral traits. We used Protégé environment and Pellet reasoning engine to evaluate our approach.

The constructed and ontologically presented end-user model provides the EUD system with knowledge on the users' static and dynamic features. Static features include gender and expertise level, whereas dynamic features target behavioral aspects (RULES attributes) derived by mouse, keystroke and eye behavior. Such knowledge can be used to provide the users with personalized services and system adaptation. In this way, end-user developers' performance and user experience could be significantly enhanced.

CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH

7.1 Overview

This thesis is focused on user modeling for cloud and web-based systems, concerning End-User Development environments which are targeted at non-professional users who wish to customize or construct their own applications. The proposed user modeling framework suggests a continuous process of data collection and processing, which results in a model of user characteristics such as gender and domain knowledge as well as behavioral and perceptional characteristics like self-efficacy, perceived ease of use, etc. These characteristics can be used for subsequent personalization or adaptation of the EUD content and the system behavior. The suggested user model is finally represented via semantic technologies in the form of an ontological user modeling framework.

We present contributions in both data collection methods and user modeling structure. The suggested structure is composed of a set of gender-oriented user attributes which tend to play a significant role and affect the end-user developers' behavior and performance. The data collection methodologies suggested in the current thesis reveal for the first time the usefulness of eye, mouse and keyboard tracking to implicitly monitor the user behavior and collect valuable data during the user-system interaction.

In the current thesis we also propose a novel EUD approach to build database-centric applications, tracing the path for natural language ('simple-talking') usage to assist end-users in effectively constructing relational database schemes and use them in their applications.

Several prototype tools are also developed to contribute in the research requirements of the current thesis conducted experiments. In particular, we have built and experimentally used the following prototype tools:

- A cloud EUD system to integrate the 'simple-talking' approach for building database-centric applications.
- A web EUD system to assist users in constricting simple web forms and use them in their applications.
- A mouse monitoring mechanism to implicitly collect user behavioral data and store them in database while the user interacts with a web or cloud-based system's interface.

7.2 Research contributions

This Ph.D thesis is a first step towards the direction of integrating user modeling methodologies for system adaptation in EUD systems. In this thesis, we proposed methods which bring novel approaches to user modeling data collection and ontology representation in web and cloud EUD environments. Our main goals were i) to examine the role of gender in EUD and use it in the suggested model composition, ii) to evaluate implicit feedback monitoring techniques to collect user data in real time and iii) to represent the final user model as an ontology using semantic web technologies. The proposed methods represent a contribution in the field of user modeling for adaptive web-based systems, more specifically in cloud and web-based EUD systems. Furthermore we aimed to iv) suggest and evaluate new EUD approaches targeted at database-centric applications and use the prototype developed tools in our experiments. More specifically, we proposed and evaluated the following:

- The 'simple-talking' approach;
- The RULES attributes;
- Eye tracking in EUD
- Mouse and keyboard tracking in EUD;
- The EUDO ontology-based modeling approach

The 'simple-talking' approach

The 'simple-talking' approach presented in chapter 1 introduced a novel EUD methodology to assist the end-user in designing a database relational scheme and

use it in their database-centric applications. The approach 'guides' the user via a step-by-step methodology to unconsciously design a relational scheme by simple responding in real world questions. Through the user responses the system can tell between ontologies, data types and relationships without demanding from the user being familiar with such concepts.

The simple-talking approach was integrated in a prototype cloud-based EUD environment and its usefulness and validity has been experimentally evaluated.

Finally, the literature review and conclusions presented in chapter 1 highlighted the usefulness and benefits that Cloud based EUD environments can bring to end-users.

The RULES attributes

This Ph.D thesis is a first step towards the direction of including gender-oriented attributes in the composition of user models, especially in EUD systems. The extended literature review presented in chapter 3 assisted in the construction of the RULES model as a EUD user model structure composing of the following attributes: risk-perception, perceived usefulness, learning willingness, perceived ease of use, and self-efficacy.

According to the current thesis findings and suggestions the RULES attributes tend to affect the end-users' behavior and/or performance during their interaction with EUD environments. Furthermore, the RULES attributes can provide the system with valuable knowledge about the user perception and acceptance, and hence they should be taken into consideration when composing user models in EUD environments.

Eye tracking in EUD

The current research is a first approach of integrating eye tracking as an implicit feedback gathering methodology for the purposes of user modeling in EUD systems. An exploratory eye tracking study has been conducted to examine the potential correlations between eye movements and perceptional and acceptance (part of RULES) items during the user-system interaction. The study collects and analyses a

set of fixation based eye tracking metrics mainly derived from the field of user experience. The study outcomes reveal the usefulness of the suggested data gathering approach and contribute in the field of eye-tracking research for user modeling and system adaptation purposes.

Mouse and keyboard tracking in EUD

This Ph.D thesis also introduced a mouse and keyboard monitoring methodology to implicitly gather user behavioral input during the user-system interaction. This methodology shed light on the necessity for defining measurable mouse metrics which tend to compose common mouse behavioral patterns. A list of mouse pattern derived metrics has been presented and the conducted experimental study revealed a number of significant correlations between mouse behavioral metrics and users' RULES attributes. Keystroke dynamics were also examined in the conducted study and significant correlations were detected as well. The study outcomes significantly contribute in the research area of mouse tracking for diagnosing user perception and acceptance items, as well as for user modeling purposes.

Also, the developed prototype tools to monitor mouse data and store them in database in real time demonstrate the potential usefulness of similar technological approaches for mouse monitoring related projects in many other HCI research fields.

The End-User Developer Ontology (EUDO) modeling framework

For the first time an 'End-User Developer Ontology' is suggested to represent the end-user developers' characteristics and behavior and to dynamically build user models. Using semantic web technologies like OWL and SWRL, the current thesis introduced an ontological framework to collect user explicitly and implicitly provided data and based on a rule-based reasoning to infer new knowledge on the users' defined properties.

Gender, eye, mouse and keyboard behavior attributes have been integrated in the constructed ontology to propose a novel EUD user model according to the research distinct findings of the current thesis.

A use case scenario was implemented in a popular ontology editor (Protégé) to evaluate the functionality of the EUDO framework. The proposed framework traces the path towards semantically represented user modeling frameworks in EUD systems.

7.2 Future research directions

This thesis introduced a set of methodologies and approaches that can be used and extended to contribute to future research works in the fields of user modeling and End-User Development.

The ‘simple-talking’ methodology presented in chapter 1 suggests as a novel wizard-based EUD approach to assist end-users in creating database-centric applications. The ‘simple-talking’, if integrated in EUD environments, can assist end-users in building database driven applications and design relational schemes without the need to be trained in relational database principles.

The current version of ‘‘simple-talking’’ approach could be implemented in EUD tools that target novice end-users, while an extended and more optimized version needs to be developed to be adopted by EUD tools that target more expert end-users. Although the more experienced end-users can recognize entities, attributes and relationships types, the EUD approach should still abstract from them more specialized terminology (e.g. primary and foreign keys) and guide them through a user-friendly approach, similar to the ‘simple-talking’ logic, to build those items more efficiently without the need to deeper understand relational database philosophy.

In a future version the user could be provided with the option to use or not the wizard like logic and more features for control and test will be placed. The EUD tool for constructing simple web forms presented in chapter 5 can been considered as a step towards integrating the ‘simple-talking’ approach into a single-page application logic.

Also, the cloud-based EUD tool that was developed to integrate the ’simple-talking’ approach can inspire development of cloud-based architecture EUD systems.

The proposed RULES model presented in chapter 2 implies that gender behavioral attributes can be used as a stepping stone for the analysis of end-user behavior and the suggestion of particular end-user behavioral models.

Similar future works could use gender-based differences in the design of gender-neutral self-adaptive systems. In the future, self-adaptive cloud and web-based EUD environments could be developed to eliminate the 'gender-gap' both in user behavior and performance.

The implicit data collection methodologies designed and evaluated in the current thesis revealed the usefulness of integrating eye and mouse tracking mechanisms in user modeling frameworks. The experimental mouse, keyboard and eye behavioral results can provide the HCI and EUD research community with a basic research background and a motivation to study further the end-user behavior in web-based EUD environments, as well as the correlation between eye/mouse/keyboard movements and perception/acceptance items.

An interesting future research direction could also be to examine the users' behavior by combining mouse monitoring and eye tracking methodologies. Monitoring both eye and mouse movements could assist in detecting new perceptional and behavioral correlations and use them as implicit feedback in self-adaptive EUD systems. For instance, combined mouse and eye tracking data can help EUD researchers understand what happened in between clicks and infer deeper knowledge about the users' cognitive process.

Finally, the represented user ontology can be further evaluated and extended to semantically represent end-users' behavioral models cross multiple web and cloud EUD environments. The design of a self-adaptation and personalization mechanism is also an interesting future research direction towards the attempt to provide end-users with dynamically changed and personalized environments.

REFERENCES

Adi, A., Barnea, M., Guy, N., Kallner, S., Rubin, Y., & Shachor, G. (2008). End-User Programming for the Web. IADIS International Conference WWW/Internet, No 467-472.

Akesson, K.P., Crabtree, A., Hansson, P. et al., (2003). Playing with the Bits' User-Configuration of Ubiquitous Domestic Environments," in Proceedings of the 5th International Conference on Ubiquitous Computing (UbiComp '03), vol. 2864of Lecture Notes in Computer Science, pp. 256–263.

Akiyama, T., and Watanobe, Y. (2012). An advanced search interface for mobile devices. In Proceedings of the 2012 Joint International Conference on Human-Centered Computer Environments (HCCE '12). ACM, New York, NY, USA, 230-235.

Antoniou, G., van Harmelen, F.: Web Ontology Language: OWL. Handbook on ontologies, International Handbooks on Information Systems 1 (2009) 91–110

Arapakis, I., Lalmas, M., & Valkanas, G. (2014). Understanding Within-Content Engagement through Pattern Analysis of Mouse Gestures. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM '14). ACM, New York, NY, USA, 1439-1448.

Arroyo, E., Selker, T., Wei, W. (2006). Usability Tool for Analysis of Web Designs Using Mouse Tracks, CHI 2006 Work-in-Progress (2006), p. 22-27.

Atterer, R., Wnuk, M., & Schmidt, A. (2006). Knowing the User's Every Move – User Activity Tracking for Website Usability Evaluation and Implicit Interaction. Edinburgh, UK.

Baldauf, M., Dustdar, S., Rosenberg, F. (2007). A survey on context-aware systems, Int. J. Ad Hoc and Ubiquitous Computing 2 (4) (2007) 263-277.

Ball, L.J., Lucas, E.J., Miles, J.N.V., and Gale, A.G. (2003).Inspection times and the selection task: What do eye-movements reveal about relevance effects? Quarterly Journal of Experimental Psychology, 56A, 1053-1077.

Bandura, A. (1977). Self-efficacy: Toward a unifying theory of behavioral change. Psychological Review 8(2), 191-215.

Bandura, A. (1986). Social Foundations of Thought and Action: A social Cognitive Theory. Prentice-Hall, Englewood Cliffs, N.J.

Barroso, L.A, Clidaras J, Hölzle U. (2013). The datacenter as a computer: an introduction to the design of warehouse-scale machines. Synt Lect Comput Archit. 20132222 – What Are They Like?. In: Paternò F., Wulf V. (eds) New Perspectives in End-User Development. Springer, Cham

Batrinca, L., Lepri, B., Mana, N., Pianesi, F. (2012). Multimodal recognition of personality traits in human-computer collaborative tasks. In Proceedings of the 14th ACM international conference on Multimodal interaction (ICMI '12). ACM, New York, NY, USA, 39-46.

Bau, D., Gray, J., Kelleher, C., Sheldon, J., Turbak, F. (2017). Learnable programming: Blocks and beyond," Comm. of the ACM, vol. 60, no. 6, pp. 72-80, Jun, 2017.

Baudisch, P., Xie, X., Zang, C., Ma, W.Y. (2004). Collapse-to-zoom: viewing web pages on small screen devices by interactively removing irrelevant content," in Proceedings of the Annual ACM Symposium on User Interface Software and Technology (UIST '04), pp. 91–94.

Beckwith, L., and Burnett, M. (2004). Gender: An Important Factor in End-User Programming Environments? 2004 IEEE Symposium on Visual Languages - Human Centric Computing, 107–114.

Beckwith, L., and Burnett, M. (2007). Gender HCI Issues in End-User Software Engineering Environments. End user Software Engineering.

Beckwith, L., Burnett, M., Grigoreanu, M., Wiedenbeck, S. (2006). Gender HCI: What about the software? Computer, Nov. 2006, pp. 83-87.

Beckwith, L., Inman, D., Rector, K., & Burnett, M. (2007). On to the real world: Gender and self-efficacy in Exce. In Proc. VLHCC, IEEE (2007), 119-126.

Beckwith, L., Sorte, S., Burnett, M., Wiedenbeck, S., Chintakovid, T., Cook, C. (2005). Designing Features for Both Genders in End-User Programming Environments. In Proceedings of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VLHCC '05). IEEE Computer Society, Washington, DC, USA, 153-160.

Berenson, S., Slaten, K., Williams, L., Ho, C.-W. (2004). Voices of women in a software engineering course: reflections on collaboration. Journal on Educational Resources in Computing 4 (1).

Beyer, S., Rynes, K., Perrault, J., Hay, K., Haller, S. (2003). Gender Differences in Computer Science Students. SIGCSE: Special Interest Group on Computer Science Education, pp. 49–53.

Bickmore, T.W. and. Picard, R.W. (2005). Establishing and maintaining long-term human-computer relationships. ACM Trans. Comput.-Hum. Interact., 12:293–327.

Blackwell A.F. (2006) Psychological Issues in End-User Programming. In: Lieberman H., Paternò F., Wulf V. (eds) End User Development. Human-Computer Interaction Series, vol 9. Springer, Dordrecht.

Blackwell, A.F. (2002). First steps in programming: a rationale for attention investment models. In Proc. IEEE Human-Centric Computing Languages and Environments, 2-10.

Blackwell, A.F. and Hague, R. (2001). Designing a programming language for home automation. In: G. Kadoda (ed.), Proceedings of the 13th annual Workshop of the Psychology of Programming Interest Group (PPIG 2001). pp. 85-103

Blackwell, A.F. and Morrison, C. (2010). A logical mind, not a programming mind: Psychology of a professional end-user. In Proceedings of the 22nd Annual Workshop of the Psychology of Programming Interest Group (PPIG 2010). September 19-22, 2010. Universidad Carlos III de Madrid, Leganès, Spain. Published by Maria Paloma Díaz Pérez and Mary Beth Rosson.

Blackwell, A.F., Britton, C., Cox, A.L., T. R. G. Green, Gurr, C.A., Kadoda, G.F., Kutar, M., Loomes, M., Nehaniv, C.L., Petre, M., Roast, C., Roe,C., Wong, A. and Young, R.M. (2001). Cognitive Dimensions of Notations: Design Tools for Cognitive Technology. In Proceedings of the 4th International Conference on Cognitive Technology: Instruments of Mind (CT '01), Meurig Beynon, Chrystopher L. Nehaniv, and Kerstin Dautenhahn (Eds.). Springer-Verlag, London, UK, UK, 325-341.

Blackwell, A.F., Rode, J. A., Toye, E.F. (2009). How do we program the home? Gender, attention investment, and the psychology of programming at home. Int. J. Human Comput. Stud. 67, 324–341.

Bojko, A. (2013). Eye Tracking the User Experience: A Practical Guide to Research. Rosenfeld Media, Brooklyn, NY.

Borges, C.R., and Macías, J. A. (2010). Feasible database querying using a visual end-user approach. In Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '10). ACM, New York, NY, USA, 187-192.

Brdjanin, D., and Maric, S. (2013). Towards the Automated Business Model-Driven Conceptual Database Design, Advances in Databases and Information Systems Advances in Intelligent Systems and Computing Volume 186, 2013, pp 31-43

Broos, A., 2005. Gender and information and communication technologies anxiety: Male self-assurance and female hesitation. Cyber Psychology and Behavior, 8(1), 11–32.

Burnett, M. and Kulesza, T. (2015): End-User Development in Internet of Things: We the People. In International Reports on Socio-Informatics (IRSI), Proceedings of the CHI 2015 - Workshop on End User Development in the Internet of Things Era (Vol. 12, Iss. 2, pp. 81-86)

Burnett, M.M., Beckwith, L., Wiedenbeck, S., Fleming, S.D., Cao, J., Park, T.H., Grigoreanu, V., et al. (2011). Gender pluralism in problem-solving software. Interacting with Computers, 23(5), 450–460.

Burnett, M. and Scaffidi, C. (2011). End-User Development. In Soegaard, Mads and Dam, RikkeFriis (eds.) Encyclopedia of Human-Computer Interaction. Aarhus, Denmark: The Interaction-Design.org Foundation.

Burnett, M., (2009). What is end-user software engineering and why does it matter? End-User Development, 15–28.

Burnett, M., Chekka, S., Pandey, R. (2001). FAR: An End-User Language to Support Cottage EServices. In: Human-Centric Computing Languages and Environments, pp. 195–202. IEEE, Los Alamitos.
Burnett, M., Chekka, S., Pandey, R. (2001). FAR: An End-User Language to Support Cottage EServices. In: Human-Centric Computing Languages and Environments, pp. 195–202. IEEE, Los Alamitos.
Burnett, M., Cook, C., Pendse, O., Rothermel, G., Summet, J. and Wallace, C. (2003). End-user software engineering with assertions in the spreadsheet paradigm. In Proc. of International Conference on Software Engineering. 93-103.

Burnett, M., Fleming, S. and Iqbal, S. (2010). Gender differences and programming environments: across programming populations. Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement.

Burnett, M., Kulesza, T., Oleson, A., Ernst, S., Beckwith, L., Cao, J., Jernigan, W., Grigoreanu, V. (2017) Toward Theory-Based End-User Software Engineering (preprint), in New Perspectives in End-User Development (F. Paterno and V. Wulf, eds.), Springer.

Burnett, M., Stumpf, S., Macbeth, J., Makri, S., Beckwith,L., Kwan, I., Peters, A., Jernigan, W. (2016). GenderMag: A method for evaluating software's gender inclusiveness. Interacting with Computers, online January 2016.

Burnett, M., Wiedenbeck, S. Grigoreanu, V., Subrahmaniyan, N., Beckwith, L., Kissinger, C. (2008). Gender in end-user software engineering. In Proceedings of the 4th international workshop on End-user software engineering (WEUSE '08). ACM, New York, NY, USA, 21-24.

Byrnes, J., Miller, D., Schafer, W. (1999). Gender differences in risk taking: a meta-analysis, Psychological Bulletin, 125(3), 367-383.

Cao, J., Kwan, I., White, R., Fleming, S.D., Burnett, M., Scaffidi, C. (2012). From Barriers to Learning in the Idea Garden: An Empirical Study. IEEE Symposium on Visual Languages and Human-Centric Computing, Innsbruck, Austria, September 2012, 59-66.

Carroll, J.M. (1998). The Nurnberg Funnel. MIT Press, (Ed.) Minimalism Beyond Cambridge, MA.

Ceri S. et al. (2000). Web Modeling Language (WebML): A Modeling Language for Designing Web Sites. In Proceedings of the 9th international World Wide Web conference on Computer networks : the international journal of computer and telecommunications networking. Amsterdam, The Netherlands, pp. 137-157.

Chagas, B.A., Fuks, H., de Souza, C.S. (2015). Lessons Learned in the Design of Configurable Assistive Technology with Smart Devices," in Fifth International Symposium on End User Development, 2015, vol. 9083, pp. 180–185.

Chagas, B.A., Redmiles, D.F., de Souza, C.S. (2017). End-user development for the Internet of Things OR How can a (smart) light bulb be so complicated?," in 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2017, pp. 273–277.

Chang, Y. (2003). A Graphical Query Language for Mobile Information Systems, In Proceedings of at SIGMOD Record, pp. 20-25, 2003.

Chen, S and Epps, J. (2014a). Using Task-Induced Pupil Diameter and Blink Rate to Infer Cognitive Load, Human–Computer Interaction, 29:4, 390-413.

Chen, S., and Epps, J. (2013). Automatic classification of eye activity for cognitive load measurement with emotion interference. Computer Methods and Programs in Biomedicine, 110, 111–124.

Chen, S., and Epps, J. (2014b). Efficient and robust pupil size and blink estimation from nearfield video sequences for human–machine interaction. IEEE Transactions on Cybernetics Advance online publication.

Chen, S., Epps, J., and Chen, F. (2011). A comparison of four methods for cognitive load measurement. Proceedings of the Oz CHI 2011 Conference on Australian Computer-Human Interaction. New York: ACM.

Chen, Y. H., & Corkindale, D. (2008). Towards an understanding of the behavioral intention to use online news services: an exploratory study. Internet Research, 18(3), 286-312.

Churruca, S.L. (2011). Comparative study of cursor movement patterns between a touchpad and a mouse devices. Master Thesis. Department of information and communications Technologies, UPF.

ClickTale (2006). http://www.clicktale.com

ClickTale User Manual. (2010). V0.5 May 2010. www.clicktale.com

Colley, A., Comber, C. (2003). Age and gender differences in computer use and attitudes among secondary school students: what has changed? Educational Research 45 (2), 155–165.

Compeau, D., and Higgins, C. (1995). Application of social cognitive theory to training for computer skills. Information Systems Research, 6(2), 118-143. Computers in Human Behavior, 17, 21–33.

Cooke, L., (2006). Is the Mouse a "Poor Man's Eye Tracker?. Usability and Information Design magazine.

Costa, P.T., Jr. and McCrae, R.R. (1992). Revised NEO Personality Inventory (NEO-PI-R) and NEO Five-Factor Inventory (NEO-FFI) manual. Odessa, FL: Psychological Assessment Resources.

Costa, P.T., Terracciano, A., McCrae, R.R. (2001). Gender differences in personality traits across cultures: Robust and surprising findings. Journal of Personality and Social Psychology, 81(2), 322–331.

Costabile M.F., Fogli, D., Fresta, G., Mussio, P., Piccinnov, A. (2003b). Building environments for end-user development and tailoring, Human Centric Computing Languages and Environments, 2003. Proceedings. 2003 IEEE Symposium on, 2003, pp. 31-38

Costabile, M., Dittrich, Y., Fischer, G., Piccinno, A. (eds.) IS-EUD. (2011). MicroApps Development on Mobile Phones. LNCS, vol. 6654, pp. 289–294. Springer, Heidelberg.

Costabile, M.F., Fogli, D., Letondal, C., Mussio, P., & Piccinno, A. (2003a). Domain-Expert Users and their Needs of Software Development. Proc. Special Session on EUD, UAHCI Conference, Crete, Greece, pp. 532-536.

Costabile, M.F., Mussio, P., Provenza, L.P., and Piccinno, A. (2008). End users as unwitting software developers. In Proceedings of the 4th international workshop on End-user software engineering (WEUSE '08). ACM, New York, NY, USA, 6-10.

Cuccurullo S., Francese R., Risi M., Tortora G. (2011) MicroApps Development on Mobile Phones. In: Costabile M.F., Dittrich Y., Fischer G., Piccinno A. (eds) End-User Development. IS-EUD 2011. Lecture Notes in Computer Science, vol 6654. Springer, Berlin, Heidelberg

Cuddihy, E., Guan, Z. and J. Ramey, J. (2005). Protocol Considerations for Using Eye-Tracking in Website Usability Testing.

Cyr, D., Hassanein, H., Head, M., Ivanov, A. (2007). The role of social presence in establishing loyalty in eservice environments. Interacting with Computers, 19(1), 43-56.

Danado J. and Paternò F. (2014) Puzzle: A Visual-Based Environment for End User Development in Touch-Based Mobile Phones. In: Winckler M., Forbrig P., Bernhaupt R. (eds) Human-Centered Software Engineering. HCSE 2012. Lecture Notes in Computer Science, vol 7623. Springer, Berlin, Heidelberg.

Danado, J., Davies, M., Ricca, P., Fensel, A. (2010). An Authoring Tool for User Generated Mobile Services. In: Berre, A.J., Gómez-Pérez, A., Tutschku, K., Fensel, D. (eds.) FIS 2010. LNCS, vol. 6369, pp. 118–127. Springer, Heidelberg.

Davis, F.D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly, 13, 319–340.

Deng, Y., Churcher, C., Abell, W., McCallum, J. (2011). Designing a Framework for End User Applications, End-User Development, Lecture Notes in Computer Science Volume 6654, 2011, pp 67-75

Dey, A.K., Sohn, T., Streng, S., Kodama, J. (2006). iCAP: interactive prototyping of context-aware applications. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 3968, pp. 254–271, 2006.

Dijkstra, M. (2013). The Diagnosis Of Self-Efficacy Using Mouse And Keyboard Input. Master thesis, Utrecht University.

Djamasbi, S., Samani, A., and Mehta, D. (2012). Eye movements, perceptions, and performance, in Proceedings of the eighteenth Americas Conference on Information Systems (AMCIS), Seattle, Washington, August 2012, pp.1-7.

Djamasbi, S., Siegel, M., Skorinko, J., and Tullis, T. (2011). Online Viewing and Aesthetic Preferences of Generation Y and the Baby Boom Generation: Testing User Web Site Experience through Eye Tracking. International Journal of Electronic Commerce, 15, 4, 121-158.

Djamasbi, S., Tulu, B., Loiacono, E., & Whitefleet-Smith, J. A. (2008). Can a Reasonable Time Limit Improve the Effective Usage of a Computerized Decision Aid? Communications of the Association for Information Systems, 23, 22,393-408.

Drummond, N., Shearer, R. (2006). The Open World Assumption. Technical report, University of Manchester, UK.

Duchowski, A. (2007). Eye Tracking Methodology: Theory and Practice, Springer-Verlag New York Inc.

Eachus, P., Cassidy, P., Norgate, S., Marrow, L., Greene, L. (2008). Internet Self-Efficacy and Visual Search Strategies: The Use of Eye Tracking Technology in the Development of Web-

Based Learning Resources. Proceedings of the Informing Science & IT Education Conference (In SITE) 2008.

Eachus, P., Cassidy, S. (2006). Academic journal article from Issues in Informing Science & Information Technology, Vol. 3.

Economides A.A. (2017) User Perceptions of Internet of Things (IoT) Systems. In: Obaidat M. (eds) E-Business and Telecommunications. ICETE 2016. Communications in Computer and Information Science, vol 764. Springer, Cham.

Epp, C., Lippold, M., Mandryk, R.L. (2011). Identifying emotional states using keystroke dynamics. CHI conference on human factors in computing systems, 7–12 May, Vancouver, BC, Canada, 715–724.

Fan, J.G. Li, L. Zhou. (2011). Interactive sql query suggestion: Making databases user-friendly. In Proceedings of the 2011 IEEE 27th International Conference on Data Engineering (ICDE '11). IEEE Computer Society, Washington, DC, USA, 351-362

Fankam, C. (2008). Ontodb2: support of multiple ontology models within ontology based database, EDBT Ph.D. workshop, New York, USA Floch, J. (2011). A framework for user-tailored city exploration, in End-User Development, vol. 6654 of Lecture Notes in Computer Science, pp. 239–244, Springer, Berlin, Germany.

Farmer, R.a., Hughes, B. (2006). Towards a “personal cost” model for end-user development.. Proceedings of the 6th ACM SIGCHI New Zealand chapter’s international conference on Computer-human interaction design centered HCI - CHINZ ’06 ,New York, New York, USA: ACM Press. 75-82.

Featherman, M., Fuller, M. (2003). Applying TAM to eservices adoption: The moderating role of perceived risk. In Proc. of Hawaii International Conference on System Sciences. IEEE.

Ferreira, S., Arroyo, E., Tarrago, R. and Blat, J. (2010) Applying Mouse Tracking to Investigate Patterns of Mouse Movements in Web Forms. Universitat Pompeu Fabra.

Finucane, M.L., Slovic, P., Mertz, C.K., Flynn, J., Satterfield, T.A. (2000). Gender, race, and perceived risk: The 'white male' effect. Health, Risk, & Society 2(2), 159-172.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A.G. & Mehandjiev, N. (2004). Meta-design: a manifesto for end-user development. Commun. ACM 47, 9 (September 2004), 33-37.

Fitts, P.M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology. 47 (6): 381–391.

Fitts, PM. and Peterson, J.R. (1964). Information capacity of discrete motor responses. Journal of Experimental Psychology. 67 (2): 103–112.

Floch, J. (2011). A framework for user-tailored city exploration, in End-User Development, vol. 6654 of Lecture Notes in Computer Science, pp. 239–244, Springer, Berlin, Germany.

Fogli, D., Lanzilotti, R., Piccinno, A. (2016). End-User Development Tools for the Smart Home: A Systematic Literature Review," in Distributed, Ambient and Pervasive Interactions, 2016, pp. 69–79

Fortineau V., Paviot T., Louis-Sidney L., Lamouri S. (2012) SWRL as a Rule Language for Ontology-Based Models in Power Plant Design. In: Rivest L., Bouras A., Louhichi B. (eds) Product Lifecycle Management. Towards Knowledge-Rich Enterprises. PLM 2012. IFIP Advances in Information and Communication Technology, vol 388. Springer, Berlin, Heidelberg.

Fraternali, P., Comai, S., Bozzon, A. and Toffetti C. G. (2010). Engineering rich internet applications with a model-driven approach. ACM Trans. Web 4, 2, Article 7 (April 2010), 47 pages.

Fraternali, P., Ceri,S., Tisi, M. (2006). Developing eBusiness solutions with a Model Driven Approach. IMP 2006.

Freeman, J., Dale, R. and Farmer, T. (2011) Hand in motion reveals mind in motion. Frontiers in Psychology, 2.

Froschl, C. (2005). User Modeling and User Profiling in Adaptive E-learning Systems An approach for a service-based personalization solution for the research project AdeLE (Adaptive e-Learning with Eye-Tracking). Master's Thesis at Graz University of Technology.

Ghiani, G., Paterno, F., Spano, L.D. (2009). Cicero designer: an environment for end-user development of multi-device museum guides," Lecture Notes in Computer Science (including subseries LectureNotes in Artificial Intelligence and LectureNotes in Bioinformatics), vol. 5435, pp. 265–274.

Goix, L.W., Valla, M., Cerami, L., Falcarin, P. (2007). Situation Inference for Mobile Users: A Rule Based Approach. In Mobile Data Management, 2007 International Conference On; pp. 299-303.

Goldberg, H. J. and Wichansky, A. M. (2003). Eye tracking in usability evaluation: A practitioner's guide. In J. Hyönä, R. Radach, & H. Deubel (Eds.), The mind's eye: Cognitive and applied aspects of eye movement research (pp. 493-516). Amsterdam, Elsevier.

Goldberg, H.J., and Kotval, X.P. (1999). Computer interface evaluation using eye movements: Methods and constructs. International Journal of Industrial Ergonomics, 24, 631-645.

Golemati, M., Katifori, A., Vassilakis, C., Lepouras, G., Halatsis, C. (2008). Creating an Ontology for the User Profile: Method and Applications. In Proceedings of the First RCIS Conference; pp. 407-412.

Grigoreanu, V., Beckwith, L., Fern, X., Yang, S., Komireddy, C., Narayanan, V., Cook, C., Burnett, M.M. (2006). Gender differences in end-user debugging, revisited: What the miners found. In Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing. 19–26.

Grigoreanu, V., Cao J, Kulesza, T., Bogart, C., Rector, K., Burnett, & M., Wiedenbeck, S. (2008). Can feature design reduce the gender gap in end-user software development environments? In Proceedings of the 2008 IEEE Symposium on Visual Languages and Human-Centric Computing (VLHCC '08). IEEE Computer Society, Washington, DC, USA, 149-156.

Guo, Qi and Eugene Agichtein (2008). Exploring mouse movements for inferring query intent. Annual ACM Conference on Research and Development in Information SIGIR'08, July 20-24, 2008, Singapore. ACM 978-1-60558-164-4/08/07

Gwizdka, J. (2014). Characterizing Relevance with Eye tracking Measures. In Proceedings of the 5th Information Interaction in Context Symposium (pp. 58–67). New York, NY, USA: ACM.

Gwizdka, J. and Zhang, Y. (2015). Towards Inferring Web Page Relevance — An Eye tracking Study, iConference 2015 Proceedings, iSchools. California, USA, 15/03.

Haapalainen, E., Kim, S., Forlizzi, J. F., and Dey, A.K. (2010). Psycho-physiological measures for assessing cognitive load. Proceedings of the UbiComp 2010 Conference on Ubiquitous Computing. New York: ACM.

Harper, Allen V.R (2015). Eye Tracking and Performance Evaluation: Automatic Detection of User Outcomes (2015). CUNY Academic Works.

Hartzel, K. (2003). How self-efficacy and gender issues affect software adoption and use. Comm. ACM 46, 9. 167-171.

Hay, D.C. (2014). Data Modeling, RDF, & OWL - Part One: An Introduction To Ontologies, David C. Hay, Published: April 2018, http://www.tdan.com/ view-articles/5025.

Heckmann, D., Kruger, A. (2003). A user modeling markup language (UserML) for ubiquitous computing. Lecture Notes in Artificial Intelligence 2702 393–397

Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., von Wilamowitz-Moellendorff, M. (2005). Gumo – the general user model ontology. In User Modeling, pages 428–432, 2005.

Heckmann, D., Schwarzkopf, E., Mori, J., Dengler, D., Kroner, A. (2007). The user model and context ontology gumo revisited for future web 2.0 extensions. In Paolo Bouquet, Jrme Euzenat, Chiara Ghidini, Deborah L. McGuinness, Luciano Serafini, Pavel Shvaiko, and Holger Wache, editors, Proceedings of the International Workshop on Contexts and Ontologies: Representation and Reasoning, volume 298 of CEUR Workshop Proceedings. CEUR-WS.org.

Henze, N., Herder, E. (2011). User Modeling and Personalization User Modeling - Introduction, (April).

Hill, C.G., Haag, M., Oleson, A., Mendez, C., Marsden, N., Sarma, A., Burnett, M. (2017). Gender-Inclusiveness Personas vs. Stereotyping: Can We Have it Both Ways?. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17). ACM, New York, NY, USA, 6658-6671.

Hinbarji, Z. Albatal, R., &Gurrin, C. (2015). Dynamic User Authentication Based on Mouse Movements Curves. MultiMedia Modeling: 21st International Conference, MMM 2015, Sydney, NSW, Australia, January 5-7, 2015, Proceedings, Part II 111 - 122 Springer International Publishing.

Hirankitti, V., Xuan, T.: (2011). A meta-reasoning approach for reasoning with SWRL ontologies, Internation Multiconference of Engineers.

Holmqvist, K., Nystrom, M., Andersson, R.D., Jarodzka, H., van de Weijer, J. (2011). Eye Tracking: A Comprehensive Guide to Methods and Measures. London: Oxford University Press.

Hornbæk, K. and Frøkjær, E. (2003) Reading patterns and usability in visualizations of electronic documents. ACM Transactions on Computer-Human Interaction (TOCHI),\ 10(2):119–149, 2003.

Horridg, M., and Musen, M. (2015). Snap-SPARQL: A Java Framework for Working with SPARQL and OWL. In Revised Selected Papers of the 12th International Experiences and Directions Workshop on Ontology Engineering - Volume 9557, Valentina Tamma, Mauro Dragoni, Rafael Gonçalves, and Agnieszka Ławrynowicz (Eds.), Vol. 9557. Springer-Verlag New York, Inc., New York, NY, USA, 154-165.

Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B. Dean, M. (2010). SWRL: A Semantic Web Rule Language combininig OWL and RuleML.

Horvitz, E., Breese, J., Heckerman, D., Hovel, D., Rommelse, K. (1998). The Lumiere project: Bayesian ` user modeling for inferring the goals and needs of software users. In In Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 256–265. Madison, WI: Morgan Kaufman.

Hubona G.S. and Shirah G.W. (2004). The Gender Factor Performing Visualization Tasks on Computer Media. In Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4 - Volume 4 (HICSS '04), Vol. 4. IEEE Computer Society, Washington, DC, USA, 40097.3.

Hubona, G. (2004). The gender factor performing visualization tasks on computer media. Proceedings of the 37th Annual Hawaii International Conference on System Sciences 2004 Proceedings of the (2004),00(1), 1–9.

Hui, B., Boutilier, C. (2006). Who's asking for help?: a bayesian approach to intelligent assistance. In IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces, 186–193. New York, NY, USA: ACM Press

Ikehara, C. S., and Crosby, M. E. (2005). Assessing cognitive load with physiological sensors. Proceedings of the 38th annual hawaii international conference on system sciences, January 3-6, 295.

Iqbal, S. T., Adamczyk, P. D., Zheng, X. S., and Bailey, B. P. (2005). Towards an index of opportunity: Understanding changes in mental workload during task execution. Proceedings of the CHI 2005 Conference on Human Factors in Computer Systems. New York: ACM.

Jacob, R. J. K. and Karn, K.S. (2003).Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research. Hyona, Radach&Deubel (eds.) Oxford, England, Elsevier Science BV.

Janev, V. and Vraneš, S. (2011). Applicability Assessment of Semantic Web Technologies. Information Processing & Management 2011, 47, 507-517.

Jason, B., Calitz, A., Greyling, J. (2010). The evaluation of an adaptive user interface model. In Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists (SAICSIT '10). ACM, New York, NY, USA, 132-143.

Just, M. A., and Carpenter, P. A. (1976). Eye fixations and cognitive processes. Cognitive Psychology, 8, 441-480.

Just, M.A., and. Carpenter, P.A. (1980). A theory of reading: from eye fixations to comprehension, Psychol. Rev. 87 (4) (1980) 329.

Kay, J., and Lum, A. (2005) Ontology-based user modeling for the semantic web. In 10th Int. Conf. on User Modeling, UM'05, Workshop 8, pages 11{19, Edinburgh, Scotland, UK, 2005.

Khan, I.A., Brinkman, W-P., Fine, N. & Hierons, R.M. (2008). Measuring personality from keyboard and mouse use. In Proceedings of the 15th European conference on Cognitive ergonomics: the ergonomics of cool interaction (ECCE '08), Julio Abascal, Inmaculada Fajardo, and Ian Oakley (Eds.). ACM, New York, NY, USA, Article 38, 8 pages.

Khanna, P. and Sasikumar, M. (2010). Recognising emotions from keyboard stroke pattern. International Journal of Computer Applications, 11 (9), 1–5.

Kim, Y.M. (2010). Gender role and the use of university library website resources: A social cognitive theory perspective. Journal of Information Science, 36,5, 603–617.

Kissinger, C., Burnett, M., Stumpf, S., Subrahmaniyan, N., Beckwith, L., Yang, S., & Rosson, M. (2006). Supporting end user debugging: What do users want to know? Advanced Visual Interfaces, ACM, 135-142.

Ko, A. J. and Myers, B. A. (2004). Designing the Whyline: A Debugging Interface for Asking Questions about Program Failures. Proceedings of the SIGCHI conference on Human factors in computing systems. Vienna, Austria, pp. 151-158

Ko, A. J., Myers, B., Rosson, M. B., Rothermel, G., Shaw, M., Wiedenbeck, S., Abraham, R., et al. (2011). The state of the art in end-user software engineering. ACM Computing Surveys, 43(3), 1–44.

Kobsa, A. (2001). Generic user modeling systems. User modeling and user-adapted interaction (pp. 136–154).

Kobsa, A. (2007). Generic user modeling systems. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, The Adaptive Web, volume 4321 of Lecture Notes in Computer Science, chapter 4, pages 136–154. Springer Verlag.

Kozsa, J.E. (2011). A potential application of pupillometry in web-usability research. Social and Management Sciences, 18, 109–115.

Krima, S., Barbau, R., Fiorentini, X., R., S., Foufou, S., Sriram, R. (2009). OntoSTEP: OWL-DL ontology for STEP, International Conference of Product Lifecycle Management, Bath, UK.

Kulenza, T., Wong, W., Stumpf, S., Perona, S., White, R., Burnett, M. M., Oberst, I., and Ko, A. J. (2009). Fixing the program my computer learned: barriers for end users, challenges for the machine. In Proceedings of the 14th international conference on Intelligent user interfaces (IUI '09). ACM, New York, NY, USA, 187-196

Lakiotaki, K., Matsatsinis, N. F., Tsoukias, A. (2011). Multicriteria User Modeling in Recommender Systems. IEEE Intelligent Systems, 26(2), 64–76.

Lao, N. (2017) Developing cloud and shared data capabilities to support primary school students in creating mobile applications that affect their communities, M.Eng thesis, Elect. Eng. Comput. Sci., Massachusetts Inst. of Technol., Cambridge, 2017.

Lebib, F-Z., Mellah, H. and Mohand-Oussaid, L. (2016). Ontological Interaction Modeling and Semantic Rule-based Reasoning for User Interface Adaptation. In Proceedings of the 12th International Conference on Web Information Systems and Technologies (WEBIST 2016) - Volume 1, pages 347-354

Lee, G. and Chen, Z. (2007) Investigating the Differences in Web Browsing Behavior of Chinese and European User Using Mouse Tracking. N. Aykin (Ed.): Usability and Internationalization, Part I, HCII 2007, LNCS 4559, pp. 502–512, 2007. Springer-Verlag Berlin Heidelberg.

Lee, Y. C. (2008). The role of perceived resources in online learning adoption. Computers & Education, 50(4), 1423–1438

Leiva A.L. and Hernando V.R. (2008). A gesture Inference Methodology for user evaluation based on mouse activity tracking. Proceedings of the IADIS International Conference on Interfaces and HCI. ISBN (Book): 978-972-8924-59-1.

Leiva, T. and A. Hernando, R. (2007). (SMT) Real Time Mouse Tracking Registration and Visualization Tool for Usability Evaluation on Websites. ISBN 978-972-8924-44-7

Li Qian, Kristen LeFevre, and H.V. Jagadish. (2010). CRIUS: user-friendly database design. Proc. VLDB Endow. 4, 2 (November 2010), 81-92.

Li, G., Fan, J., Wu, H., Wang, J., & Feng, J. (2011). DBease: Making Databases User-Friendly and Easily Accessible, In Proceedings of CIDR. 2011, 45-56.

Lieberman, H. (ed.). (2001). Your Wish Is My Command: Programming By Example. Morgan Kaufmann Publishers, San Francisco.

Lieberman, H., Paternò, F., Wulf, V. (2006). End User Development: An emerging paradigm. ser. Human-Computer Interaction Series. Germany: Springer, Nov. 2006, vol. 9, pp: 1-8.

Liscombe, J., Venditti, J., Hirschberg, J. (2003). Classifying subject ratings of emotional speech using acoustic features. In Proc. of Eurospeech - Interspeech 2003, p. 725–728.

Little, G., Lau, T., Cypher, A., Lin, J., Haber, E., Kandogan, E. (2007). Koala: Capture, Share, Automate, Personalize Business Processes on the Web. In: ACM Conference on Human Factors in Computing Systems, pp. 943–946. ACM, New York.

Litvinova, E. (2010). XIDE - a Visual Prototype application for End User Development of Web Applications, Master's Thesis, Joensuu, University of Eastern Finland.

Liu, H. and Lieberman, H. (2005). Programmatic semantics for natural language interfaces. In Proceedings of the ACM Conference on Human Factors in Computing. 1597–1600.

Lizcano, D., Alonso, F., Soriano, J., L´opez, G. (2011). End-User Development Success Factors and their Application to Composite Web Development Environments, In Proceedings of the Sixth International Conference on Systems, ICONS 2011, St. Maarten, Antillas Holandesas.

Lizcano, D., Soriano, J., Reyes, M., & Hierro, J. J. (2008). Ezweb/fast: Reporting on a successful mashup-based solution for developing and deploying composite applications in the upcoming 'ubiquitous soa. Mobile Ubiquitous Computing, Systems, Services and Technologies, International Conference on, vol. 0, pp. 488–495.

Loewenstein, G. (1994). The psychology of curiosity: A review and reinterpretation. Psychology Bulletin 116(1), 75-98

Luce, R.D. (1986). Response times: their role in inferring elementary mental organization: Oxford University Press, New York, USA.

Maedche, A. (2002). Ontology learning for the semantic web. Journal of Intelligent Systems, IEEE 16 (2), pp.72-79.

Magnuson, J. (2005). Moving hand reveals dynamics of thought. Proceedings of the National Academy of Sciences of the United States of America, 102(29):9995.

Margolis, J., Fisher, A. (2003). Unlocking the Clubhouse. MIT Press.

Martins, A. C., Faria, L., Vaz de Carvalho, C., Carrapatoso, E. (2008). User Modeling in Adaptive Hypermedia Educational Systems. Educational Technology & Society, 11 (1), 194-207.

Martinson, A.M. (2005). Playing with technology: Designing gender sensitive games to close the gender gap. Working Paper SLISWP-03-05, School of Library and Information Science, Indiana University.

Mastor, K.A. (2003). Personality traits and gender differences in the selection of academic major among Malay students. Journal Pendidikan (28), 3–13.

McCrae, R.R., Costa, P.T., Jr. (1999). A five-factor theory of personality. In L. A. Pervin, O. P. John (Eds.), Handbook of personality: Theory and research (2nd ed., pp. 139–153). New York: Guilford Press.

McIlroy, D., Bunting, B., Tierney, K., Gordon, M. (2001). The relation of gender and background experience to self-reported computing anxieties and cognitions. Computers in Human Behavior. 17, 1, (1 January 2001), Pages 21–33

McKinstry, C. Dale, R. and. Spivey, M. (2008). Action dynamics reveal parallel competition in decision making. Psychological Science, 19(1):22.

McLeod, L., and MacDonell, S.G. (2011). Factors that affect software systems development project outcomes: A survey of research. ACM Comput. Surv. 43, 4, Article 24 (October 2011), 56 pages.

Mehta, B., Niederee, C., Stewart, A., Degemmis, M., Lops, P., Semeraro, G. (2005). Ontologically-Enriched Unified User Modeling for Cross-System Personalization. User Modeling, 151-151.

Mejía, A., Juárez-Ramírez, R., Inzunza, S., Valenzuela, R. (2012). Implementing adaptive interfaces: a user model for the development of usability in interactive systems. In Proceedings of the CUBE International Information Technology Conference (CUBE '12). ACM, New York, NY, USA, 598-604.

Miller, R.C., Bolin, M., Chilton, L.B., Little, G., Webber, M., Chen-Hsiang,Y. (2010). Rewriting the web with chickenfoot. Iin No Code Required: Giving Users Tools to TransformtheWeb, pp. 39-62, Elsevier, Burlington, Mass, USA, 2010.

Monrose, F., and Rubin, A.D. (2000). Keystroke dynamics as a biometric for authentication, Future Generation Computer Systems, Volume 16, Issue 4, 2000, Pages 351-359, ISSN 0167-739X.

Moon, J. and Kim, Y. (2001). Extending the TAM for a world-wide-web context. Information and Management, 38(4), 217–230

Mueller, F. and Lockerd, A., (2001). Cheese: Tracking Mouse Movement activity on Websites, a Tool for User Modeling. Ext. Abstracts CHI. Seattle, Washington, USA, pp1-2.

Mun, D., Ramani, K. (2011). Knowledge-based part similarity measurement utilizing ontology and multi-criteria decision making technique. Advanced Engineering Informatics 25 (2011) 119–130.

Myers, B. A., Hudson, S. E, Pausch, R. (2000). Past, present and future of user interface software tools. ACM Transactions on Computer Human Interaction, vol. 7, no. 1, pp. 3–28.

Nahin, A.F.M. NazmulHaque, Alam, J.M., Mahmud, H & Hasan, K. (2014). Identifying emotion by keystroke dynamics and text pattern analysis, Behavior& Information Technology, 33:9, 987-996.

Nakayama, M., and Katsukura, M. (2007). System usability evaluation for input operation using oculo-motors. Proceedings of the AUIC 2008 Conference on Australasian User Interface. Darlinghurst: Australian Computer Society.

Namoun, A., Daskalopoulou, A., Mehandjiev, N. and Xun, Z. (2016). Exploring Mobile End User Development: Existing Use and Design Factors. IEEE Trans. Softw. Eng. 42, 10 (October 2016), 960-976.

Nass, C., Brave, S., 2005. Wired for Speech: How Voice Activates and Advances the Human-Computer Relationship. MIT Press.

Navalpakkam, V., and Churchill, E. (2012). Mouse tracking: measuring and predicting users' experience of web-based content. SIGCHI, 2012.

Navalpakkam, V., Jentzsch, L., Sayres, R., Ravi, S., Ahmed, A., &Smola, A.J. (2013). Measurement and modeling of eye-mouse behavior in the presence of nonlinear page layouts. WWW.

Newman, M.W. et al. (2003). DENIM: An Informal Web Site Design Tool Inspired by Observations of Practice. In proceedings of the CHI conference. Fort Lauderdale, Florida, pp. 259-324.

Ngoc K.A.P., Lee Y.K., Lee SY. (2005) OWL-Based User Preference and Behavior Routine Ontology for Ubiquitous System. In: Meersman R., Tari Z. (eds) On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE. OTM 2005. Lecture Notes in Computer Science, vol 3761. Springer, Berlin, Heidelberg

Nichols, J. and Lau, T. (2008). Mobilization by demonstration: using traces to re-author existing web sites," in Proceedings of the 13th International Conference on Intelligent User Interfaces (IUI '08), pp. 149–158, January 2008.

Nili, G. et al. (2009). Freedom - End User Programming for the Web, IBM Programming Languages and Development Environments Seminar 2009 on Innovations in Software Development April 22, 2009 Organized by IBM Haifa Research 7Lab.

Nunes, M. A. S. N., Cerri, S. A. Blanc, N. (2008). Towards user psychological profile. In Proceedings of the VIII Brazilian Symposium on Human Factors in Computing Systems (IHC '08). Sociedade Brasileira de Computação, Porto Alegre, Brazil, Brazil, 196-203.

Nunnally, J. (1967). Psychometric Theory. New York: McGraw.

O'Connor, M.J. and Das, A. (2009). "SQWRL: a Query Language for OWL" OWL: Experiences and Directions (OWLED), 6th International Workshop, Chantilly, VA, 2009.

Oliveira, F. T. P., Aula, A., and Russell, D. M. (2009). Discriminating the relevance of web search results with measures of pupil size. In Proceedings of the 27th international conference on Human factors in computing systems (pp. 2209–2212). Boston, MA, USA: ACM.

Ong, C., Lai, J. (2006). Gender differences in perceptions and relationships among dominants of e-learning acceptance. Computers in Human Behavior, 22(5), 816–829.

Ong, K.W. (2010). Web Application Creation Made Easy: A Sql-Driven Rapid Development Framework and a Do-It-Yourself Platform. Ph.D. Dissertation. University of California at San Diego, La Jolla, CA, USA. Advisor(s) Alin Deutsch. AAI3422419.

Onorati, F., Barbieri, R., Mauri, M., Russo, V., and Mainardi, L. (2013). Characterization of affective states by pupillary dynamics and autonomic correlates. Frontiers in Neuroengineering, 6, 9.

Osvalder, A.L., Ulfvengren, P. (2009). Human-technology systems. In Bhgard, G et al.. (Eds.) Work and technology on human terms, Prevent, Sweden.

OWL 2 Web Ontology Language Document Overview (Second Edition) W3C Recommendation 11 December 2012, accessed April 2018, https://www.w3.org/TR/2012/REC-owl2-overview-20121211/.

OWL Web Ontology Language-Overview, W3C Recommendation 10th February 2004, accessed April 2018, https://www.w3.org/TR/2004/REC-owl-features-20040210/.

Pajares, F. (2002). Gender and perceived self-efficacy in self-regulated learning, Theory into Practice. The Ohio State University, on behalf of its College of Education, 2002 41, 2.

Pan, P.J.D., et al. (2010). University students' perceptions of a holistic care course through cooperative learning: implications for instructors and researchers. Asia Pacific Education Review, 11 (2), 199–209.

Pane, J. F., Ratanamahatana, C. A., and Myers, B. A. (2001). Studying the language and structure in non-programmers' solutions to programming problems. International Journal of Human-Computer Studies 54 (2): 237-264.

Pane, J., Myers, B., Miller, L. (2002). Using HCI Techniques to Design a More Usable Programming System. In: Proc. IEEE Human-Centric Computing Languages and Environments, pp. 198–206. IEEE, Los Alamitos

Papamitsiou, Z. and Economides, A. (2014) students' perception of performance vs. actual performance during computer based testing: a temporal approach. INTED2014 Proceedings, pp. 401-411.

Park, S. (2009). An Analysis of the Technology Acceptance Model in Understanding University Students' Behavioral Intention to Use e-Learning. Education Technology & Society, 12(3), pp.150-162.

Paternò, F. (2013). End User Development: Survey of an Emerging Field for Empowering People, ISRN Software Engineering, vol. 2013, Article ID 532659, 11 pages.

Payne, J. W., Bettman, J. R., & Johnson, E. J. (1988). Adaptive strategy selection in decision making. Journal of Experimental Psychology: Learning, Memory, and Cognition, 14, 3, 534-552.

Pérez, J., Arenas, M., and Gutierrez, C. (2009). Semantics and complexity of SPARQL. ACM Trans. Database Syst. 34, 3, Article 16 (September 2009), 45 pages

Plumbaum T., Lommatzsch A., De Luca E.W., Albayrak S. (2012) SERUM: Collecting Semantic User Behavior for Improved News Recommendations. In: Ardissono L., Kuflik T. (eds)

Advances in User Modeling. UMAP 2011. Lecture Notes in Computer Science, vol 7138. Springer, Berlin, Heidelberg.

Plumbaum, T. (2010). Semantically-enhanced ubiquitous user modeling. In Paul De Bra, Alfred Kobsa, and David N. Chin, editors, UMAP, volume 6075 of Lecture Notes in Computer Science. Springer.

Plumbaum, T., Wu, S., De Luca, W.W., Albayrak, S. (2011). User modeling for the social semantic web. In Proceedings of the Second International Conference on Semantic Personalized Information Management: Retrieval and Recommendation - Volume 781 (SPIM'11), Marco de Gemmis, Ernesto William De Luca, Tommaso Di Noia, Aldo Gangemi, and Thomas Lukasiewicz (Eds.), Vol. 781. CEUR-WS.org, Aachen, Germany, Germany, 78-89.

Poole, A. and Ball, L.J. (2005). Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future. Prospects, Chapter in C. Ghaoui (Ed.): Encyclopedia of Human-Computer Interaction. Pennsylvania. Idea Group, Inc.

Privitera, C.M., and Stark, L.W. (2000). Algorithms for defining visual regions-of-interest: comparison with eye fixations, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 970–982.

Protégé, https://protege.stanford.edu/, accessed April 2018

Protogeros, N. and Tzafilkou, K. (2015). Simple-talking database development: Let the end-user design a relational schema by using simple words, Computers in Human Behavior, Volume 48, July 2015, pp. 273-289.

Prud'hommeaux, E. and Seaborne, A. (2008). SPARQL query language for RDF. W3C recommendation. http://www.w3.org/TR/rdf-sparql-query/.

Qian, L., LeFevre, K. and H.V. Jagadish. (2010). CRIUS: user-friendly database design. Proc. VLDB Endow. 4, 2 (November 2010), pp.81-92.

Rath, A.S, Devaurs, D., Lindstaedt, S.N. (2009). UICO: an ontology-based user interaction context model for automatic task detection on the computer desktop. Proc. Workshop on Context Information and Ontology, ESWC '09, Jun 2009, Heraklion, Greece.

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research, Psychol. Bullet. 124 (3) 372.

Razmerita L., Angehrn A., Maedche A. (2003) Ontology-Based User Modeling for Knowledge Management Systems. In: Brusilovsky P., Corbett A., de Rosis F. (eds) User Modeling 2003. UM 2003. Lecture Notes in Computer Science, vol 2702. Springer, Berlin, Heidelberg.

Razmerita, L., Angehrn, A., Maedche, A. (2003) Ontology-based user modeling fo knowledge management systems. In 9th Int. Conf. on User Modeling, UM'03, pages 213-217, Johnstown, PA, USA, 2003. Springer, LNCS 2702.

Repenning, A., Ioannidou, A. (2006). What Makes End-User Development Tick? 13 Design Guidelines. In End-User Development: Empowering people to flexibly employ advanced information and communication technology, edited by Lieberman, H., Paternò, F., Wulf, V. Dordrecht: Kluwer.

Rich, E. (1989). Stereotypes and user Modeling. In User models in dialog systems, Berlin: Springer Verlag.

Robertshon, T.J., Prabhakararao, S., Burnett, M., Cook, C., Ruthruff, J. R., Beckwith, L., Phalgune, A., 2004. Impact of interruption style on end-user debugging. In Proceedings of the ACM Conference on Human Factors in Computing Systems. 287–294.

Rodden, K.X., Fu, A. A., & Spiro, I. (2008). Eye-mouse coordination patterns on web search results pages. In CHI ’08 Extended Abstracts on Human Factors in Computing Systems, CHI EA ’08, pages 2997–3002, New York, NY, USA, ACM.

Rode, J. and Rosson, M. B. (2003). Programming at Runtime: Requirements & Paradigms for Nonprogrammer Web Application Development. IEEE HCC 2003. Auckland, New Zealand. Oct. 28-31.

Rode, J., Rosson, M. B., M. A. Pérez-Quiñones, J. (2004). End Users’ Mental Models of Concepts Critical to Web Application Development. Processing of Visual Languages and Human Centric Computing. Rome, Italy, pp. 215-222

Rode, J., Bhardwaj, Y., P´erez-Qui˜nones, M. A., Rosson, M. B., Howarth, J. (2005). As Easy as “Click”: End-User Web Engineering. International Conference on Web Engineering. Sydney, Australia. July 27–29.

Rode, J., Rosson, M. B., and Pérez-Quiñones, J. (2006). End User Development of Web Applications. End User Development, Human-Computer Interaction Series Volume 9, 2006, pp 161-182

Rode, J.A. (2008). An ethnographic examination of the relationship of gender & end-user programming, Ph.D. Thesis, University of California Irvine.

Rossell`o-Busquet, A., Brewka, J., Dittman, L. (2011). OWL ontologies and SWRL rules applied to energy management, International Conference on Modeling and Simulation.

Rubio, S., Díaz, E., Martín, J., and Puente, J. M. 2004. Evaluation of Subjective Mental Workload: A Comparison of SWAT, NASAıTLX, and Workload Profile Methods. Applied Psychology, 53,1, 61-86.

Ruthruff, J.R., Phalgune, A., Beckwith, L., Burnett, M., Cook, C. (2004). Rewarding “good” behavior: End-user debugging and rewards. In Proceedings of the IEEE Symposium on Visual Languages and Human-Centered Computing. 115–122.

Saadé, R.G., Kira, D., Otrakji, C.A., 2012. Gender Differences in Interface Type Task Analysis. International Journal of Information Systems and Social Change, 3(2), 1–23.

Saleem, H., Beaudry, A., Croteau, A.M. (2011). Antecedents of computer self-efficacy: A study of the role of personality traits and gender, Computers in Human Behavior, Volume 27, Issue 5, September 2011, Pages 1922-1936.

Scaffidi, C., Myers, B. &Shaw, M. (2008). Topes: Reusable Abstractions for Validating Data. International Conference on Software Engineering (ICSE 2008), Leipzig, Germany.

Scaffidi, C., Bogart, C., Burnett, M.M. Cypher, A., Myers, B. & Shaw, M. (2010). Using Traits of Web Macro Scripts to Predict Reuse, Journal of Visual Languages & Computing, vol. 21, issue 5, pp. 277 - 291, 12/2010.

Schiller, J. Turbak, F., Abelson, H., Dominguez, J., McKinney, A., Okerlund, J., Friedman, M. (2014). Live Programming of Mobile Apps in App Inventor. In Proceedings of the 2nd Workshop on Programming for Mobile & Touch (PROMOTO '14). ACM, New York, NY, USA, 1-8.

Schmidt, K.U., Stojanovic, L., Stojanovic, N., Thomas, S. (2007) On enriching ajax with semantics: The web personalization use case. In 4th European Semantic Web Conference, June 2007.

Schroder, H. M., Driver, M. J., and Streufert, S. (1967). Human information processing: New York: Holt, Rinehart and Winston, 54-61.

Seifert, J., Pfleging, B., Bahamóndez, E., Hermes, M., Rukzio, E., Schmidt, A. (2011). Mobidev: A tool for creating apps on mobile phones. In: Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI 2011), pp. 109–112. ACM, New York.

Shapiro, S.S. and Wilk, M.B. (1965). An analysis of variance test for normality (complete samples). Biometrika 52 (3–4), pp.591–611.

Sharafi, Z., Soh, Z., and Guéhéneuc, Y-G. (2015). A systematic literature review on the usage of eye-tracking in software engineering. Inf. Softw. Technol. 67, C (November 2015), 79-107.

Shea, P. and Bidjerano, T. (2010). Learning presence: towards a theory of self-efficacy, self-regulation, and the development of a communities of inquiry in online and blended learning environments. Computers & Education, 55 (4), 1721–1731.

Shneiderman, B., 1980. Software Psychology: Human Factors in Computer and Information Systems. Winthrop Publishers.

Silveira, C., Eloy, L. & Montiero, J.M. (2010). A Query Language for Data Access in Ubiquitous Environments, In Proceedings CILEI electronic journal, vol. 13, No. 3, 2010.

Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A. and Katz, Y. (2007). Pellet: A Practical Owl-Dl Reasoner. Web Semantics: science, services and agents on the World Wide Web, 5, pp. 51-53.

Sosnovsky, S. & Dicheva, D. (2010). Ontological technologies for user modeling. International Journal of Metadata Semantics and Ontologies.

Spahn, M., Dörner, C., & Wulf, V. (2008). End User Development : Approaches towards a Flexible Software Design Questions to Answer. Information Systems, (June).

Srivastava, S., John, O. P., Gosling, S. D., Potter, J. (2003). Development of personality in early and middle adulthood: Set like plaster or persistent change? Journal of Personality and Social Psychology, 84(5), 1041–1053.

Stan, J., Egyed-Zsigmond, E., Joly, A., Maret, P. (2008) .A User Profile Ontology For Situation-Aware Social Networking. 3rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI2008) J.C. Augusto, D. Shapiro and H. Aghajan. 21st-22nd of July 2008. Co-located event of ECAI 2008.

Stipek, D., and Gralinski, J.H. (1991). Gender differences in children's achievement-related beliefs and emotional responses to success and failure in mathematics. J. of Educational Psychology, 83(3).

Strang, T., Linnho_-Popien, C. (2004). A context modeling survey, in: Workshop on Advanced Context Modeling, Reasoning and Management, UbiComp '04, Nottingham, England.

Subrahmaniyan, N., Beckwith, L., Grigoreanu, V., Burnett, M., Wiedenbeck, S., Narayanan, V., Bucht, K., Drummond, R., Fern, X. (2008). Testing vs. code inspection vs. ... what else? Male and female end users' debugging strategies, In Proc. CHI, ACM, 617-626.

Sun, H., & Zhang, P. (2008). An exploration of affect factors and their role in user technology acceptance: mediation and causality. Journal of the American Society for Information Science and Technology, 59(8), 1-12.

Sutterer, M.; Droegehorn, O. (2008). David, K. UPOS: User Profile Ontology with Situation-Dependent Preferences Support. In Advances in Computer-Human Interaction, 2008 First International Conference On; pp. 230-235.

Svenson, O. (1993). Time pressure and stress in human judgment and decision making: Springer, Pletinum Press, New York, USA.

Sweller, J. (1988) Cognitive load during problem solving: effects on learning, Cognitive Science 12 (1988) 257–285.

Terzis, V., and Economides, A.A., (2011). Computer based assessment: Gender differences in perceptions and acceptance. Computers in Human Behavior, 27, 6, November 2011, 2108-2122.

Terzis, V., Moridis, N.C., Economides, A.A. (2012). How student's personality traits affect Computer Based Assessment Acceptance: Integrating BFI with CBAAM. Comput. Hum. Behav. 28, 5 (September 2012), 1985-1996.

Tetteroo D. and Markopoulos, P. (2015). A review of research methods in end user development," in International Symposium on End User Development, Madrid, Spain, 2015, pp. 58–75.

Tetteroo D. and Markopoulos, P. (2015). A review of research methods in end user development," in International Symposium on End User Development, Madrid, Spain, 2015, pp. 58–75.

Tetteroo, D., et al. (2015). Lessons Learnt from Deploying an End-User Development Platform for Physical Rehabilitation," in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, New York, NY, USA, 2015, pp. 4133–4142.

Thagard, P., (2006). Hot Thought: Machanisms and Applications of Emotional Cognition. A Bradford Book- MIT Press, Cambridge, MA, USA.

Thompson, R. L., Higgins, C. A., & Howell, J. M. (1991). Personal computing: Toward a conceptual model of utilization. MIS Quarterly, 15(1), 124–143.

Trappl, R., Payr, S., Petta, P. (2003). Emotions in Humans and Artifacts.MIT Press, Cambridge, MA, USA.

Tuomela, U., Kansala, I., Hakkila, J., Mantyjarvi, J. (2003) Context-Studio? Tool for Personalizing Context-Aware Applications in Mobile Terminals. In: Proceedings of 2003 Australasian Computer Human Interaction Conference, OzCHI 2003 (Nokia Research Center), p. 292.

Turau, V. (2002). A Framework for Automatic Generation of Web-based Data Entry Applications Based on XML. Proceedings of the 2002 ACM symposium on Applied computing. Madrid, Spain, pp. 1121--1126.

Turkle, S., Papert, S. (1992). Epistemological pluralism and the revaluation of the concrete. Journal of Mathematical Behavior, 11, 3-33.

Tzafilkou K., Protogeros, N., Yakinthos, C. (2014). Mouse Tracking for Web Marketing: Enhancing User Experience in Web Application Software by Measuring Self-Efficacy and Hesitation Levels. International Journal of strategic and Innovative Marketing, 1, 4.

Tzafilkou, K. and Protogeros, N. (2017a). Diagnosing user perception and acceptance using eye tracking in web-based end-user development. Computers in Human Behavior. 72, C (July 2017), 23-37.

Tzafilkou, K. and Protogeros, N. (2017b). Examining gender issues in perception and acceptance in web-based end-user development activities. Oct 2017, Education and Information Technologies.

Tzafilkou, K. Protogeros, N., Karagiannidis, C., Koumpis, A. (2017). Gender-based behavioral analysis for end-user development and the `RULES' attributes. Education and Information Technologies 22, 4 (July 2017), 1853-1894.

Tzafilkou, K., Protogeros, N., Koumpis, A. (2017). Eye movements and end-user performance in web development tools November 2017 Conference: Sixth International Conference on Virtual and Networked Organizations Emergent Technologies and Tools, ViNOrg'17

Tzafilkou, K., Protogeros, N., Yakinthos, C. (2015). End-User Development of CRM Systems: Towards a Behavioral End- User Profiling based on Gender and Expertise. International Journal of Electronic Business 01/2015; 12(3).

Ur, B., McManus, E., Pak Yong Ho, M., Littman, M.L. (2014). Practical Trigger-action Programming in the Smart Home, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, NY, USA, 2014, pp. 803–812.

Van Raaij, E. M. and Schepers, J. J. L. (2008). The acceptance and use of a virtual learning environment in China. Computers & Education, 50(3), 838–852.

Venkatesh, V. and Morris, M. (2000). Why don't men ever stop to ask for directions? Gender, social influence, and their role in technology acceptance and usage behavior, MIS Quarterly, 24(1), pp. 115-139.

Venkatesh, V., Morris, M.G., Davis, G.B., & Davis, F.D. (2003). User acceptance of information technology: Toward a unified view. MIS Quarterly, 27(3), 425–478.

Vizer, L.M., Zhou, L., and Sears, A. (2009). Automated stress detection using keystroke and linguistic features: An exploratory study. Int. J. Hum.-Comput. Stud. 67, 10 (2009), 870-886.

Wang, Q., Yang, S., Liu, M., Cao, Z., Ma, Q. (2014). An eye-tracking study of website complexity from cognitive load perspective, Decision Support Systems, Volume 62, June 2014, Pages 1-10, ISSN 0167-9236.

Wang, Q., Yu, X. (2011). Reasoning over OWL/SWRL ontologies under CWA and UNA for industrial applications. Advances in Artificial Intelligence 7106 .789– 798.

Wang, Y.S. and Shih, Y.W. (2009). Why do people use information kiosks? A validation of the unified theory of acceptance and use of technology, Government Information Quarterly, Vol. 26 No. 1, pp. 158-165.

Weinberg, G.M. (1998). The Psychology of Computer Programming (Silver Anniversary Ed.). Dorset House Publ. Co., Inc., New York, NY, USA.

Wilson, A., Burnett, M., Beckwith, L., Granatir, O., Casburn, L., Cook, C., Durham, M., Rothermel, G. (2003). Harnessing Curiosity to Increase Correctness in End-User Programming. In ACM Conference on Human Factors in Computing Systems. ACM, New

Wolber D. et al. (2002). Designing dynamic web pages and persistence in the WYSIWYG interface. Proceedings of the 7th international conference on Intelligent user interfaces. San Francisco, CA, pp. 228--229.

Wong, J. and Hong, J.I. (2007). "Making mashups with marmite: towards end-user programming for the web," in Proceedings of the 25th SIGCHI Conference on Human Factors in Computing Systems (CHI '07), pp. 1435–1444, May 2007.

Wu, H.G. Li, C. Li, & Zhou, L. (2010). Seaform: Search-as-you-type in forms. PVLDB, 3(2):1565–1568

Wulf, V., Pipek, V., & Won, M. (2008). Component-based tailorability: Enabling highly flexible software applications. International Journal of Human-Computer Studies 66(1), 1–22.

Yoon, D. and Narayanan, N.H. (2004). Mental imagery in problem solving: An eye tracking study. In Proceedings of the Eye Tracking Research and Applications Symposium 2004 (pp. 77-83). NY: ACM Press.

Zardas. G., Moschidis, O., Mavridis, I., & Manitsaris, A. (2011). A methodology for evaluating web-based educational systems using statistical multidimensional analysis, Int. J. of Learning Technology, 2011 Vol.6, No.4, pp.409 – 432

Zdun, U. (2002). Dynamically Generating Web Application Fragments from Page Templates. Proceedings of the 2002 ACM symposium on Applied computing, Madrid, Spain, pp. 1113--1120.

Zhang, Y., Chen, W., Wang, D., Yang, Q. (2011). User-click modeling for understanding and predicting search-behavior. Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11 (p. 1388). New York, New York, USA: ACM Press.

Technological documents

Tobii Technology. 2011. Accuracy and precision test method for remote eye trackers. Test Specification Version: 2.1.1

Tobii Technology. 2015. User Manual Tobii Studio. Version: 3.3.1

ANNEX I

Table I Questionnaire survey and results for validity of the measurement model

Construct Item		Cronbach α (>=0.70)
Self-Efficacy		0,89
SE1	I felt confident while I was using the system	
SE2	I believed that I could perform well	
SE3	I felt I had the control of the task	
SE4	I felt that everyone else knew what to do but me	
SE5	I felt confused while using the system	
Risk Perception		0,70
RP1	It was taking me time to decide how to move while using the system	
RP2	I felt nervous every time I took an action (e.g. pressed a button)	
RP3	I checked well my actions before moving to the next steps	
RP4	I had no hesitation to take an action	
RP5	I had no difficulty to try which feature (among other) to use	
Willingness to Learn		0,81
WL1	I wanted to learn how to use the system while I was using it	
WL2	I would like to learn more how to use the system	
WL3	I would like to learn how to use other similar systems too	
Perceived Ease of Use		0,76
PEOU1	The system is easy to use	
PEOU2	I do not need to try too hard to use the system effectively	
PEOU3	I can use the system without written instructions	
PEOU4	I can learn how to use the system easily and fast	
PEOU5	I can easily correct my mistakes while I use the system	
Perceived Usefulness		0,71
PU1	The system is useful	
PU2	The system makes me more productive	
PU3	The system makes me save time	
PU4	The system satisfies my needs and requirements	

ANNEX II

Table II Test of Normality (N=30)

	Shapiro-Wilk		
	Statistic	df	Sig.
Perceived Usefulness	0,956	30	0,248
Perceived EaseOfUse	0,934	30	0,065
Self-Efficacy	0,933	30	0,057
Willingness toLearn	0,959	30	0,289
Risk-Perception	0,967	30	0,453

ANNEX III

Following we present some excerpts of the OWL syntax for the Protégé generated EUDO ontology. OWL excerpts present some of the constructed EUDO subclasses, object and data properties.

```
<owl:ObjectProperty
rdf:about="http://www.semanticweb.org/milly/ontologies/201
8/1/eudo#hasEUDTask">
<rdfs:domain
rdf:resource="http://www.semanticweb.org/milly/ontologies/
2018/1/eudo#End-user-developer"/>
<rdfs:range
rdf:resource="http://www.semanticweb.org/milly/ontologies/
2018/1/eudo#EUDTask-info"/> </owl:ObjectProperty>
<owl:ObjectProperty
rdf:about="http://www.semanticweb.org/milly/ontologies/201
8/1/eudo#hasEyeBehavior">
<rdfs:domain
rdf:resource="http://www.semanticweb.org/milly/ontologies/
2018/1/eudo#End-user-developer"/>
<rdfs:range
rdf:resource="http://www.semanticweb.org/milly/ontologies/
2018/1/eudo#Eye-behavior"/></owl:ObjectProperty>
<owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/milly/ontologies/201
8/1/eudo#LW">
<rdfs:subPropertyOf
rdf:resource="http://www.w3.org/2002/07/owl#topDataPropert
y"/>
<rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/></
owl:DatatypeProperty>
<owl:Class
rdf:about="http://www.semanticweb.org/milly/ontologies/201
8/1/eudo#Risk-Perception">
<rdfs:subClassOf
rdf:resource="http://www.semanticweb.org/milly/ontologies/
2018/1/eudo#Behavioral-traits"/> </owl:Class>
```