DEMONSTRATION-BASED HELP: A CASE STUDY

P. Rodrigues[1], J. Silva[2 3], R. Pereira[3]

[1]Instituto Universitário de Lisboa (ISCTE-IUL) (PORTUGAL)
[2]Madeira-ITI (PORTUGAL)
[3]Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR-IUL (PORTUGAL)

Abstract

The usability of today's applications is of utmost importance, and to fully achieve it some changes are yet to be adopted. Help systems are part of the applications and allow users to understand it and use it. However, this is one of the fields that some corporates still neglect. Moreover, every system has its characteristics and customizations and being able to explain how something works on a generic but efficient way is a major challenge.

Studies have been conducted and applications, especially on a daily-use context, are still cause of frustration to its users. Contextual and demonstration-based strategies are examples of beneficial approaches to improve the status. Additionally, current technologies like picture-driven computing and automation are enablers facilitating the interaction with "closed" applications, whose source codes are not available, and allowing for the process to be reproduced automatically on any platform.

Our solution deploys fast easy learning. It allows users to learn how to perform tasks by watching them get done on their own system. This was done by the manual creation of picture-driven scripts with the use of a tool named Sikuli.

This paper presents our tool and a preliminary case-study. A preliminary evaluation with users was made and positive results were shown. When tracking metrics relative to a first interaction with a system/realization of a specific tasks, users achieved the goal five times faster with the demonstration-based help tool. On second time executions, the performance from users that used our tool and users that learned through exploration and manual readings were similar, indicating that learning by observation does not detract the learning process.

The solution is now to be implemented in a technology corporation where problematic tasks and platforms will be identified, automation scripts developed, and an evaluation with its employees will be conducted. From this evaluation, the usability of the tool will be measured, as learning improvements and increased task performance will be tracked. The availability to help other users, developing and sharing scripts is also one of the research points.

Keywords: Help by demonstration, Usability, Picture-Driven Computing, Automation, Learning process analysis

1 INTRODUCTION

The age range of corporate computing system users is wide [1]. This fact must be considered, especially for the systems used in day-to-day jobs, and, therefore, adaptive user interfaces shall be developed. User interfaces should pay particular to all types of users, such as elderly [2], novice [3] or other extreme users [4]. It is hard to design an interface that fits all users and faults in it are cause of frustration and anxiety [5]. Several advances have been made in several domains (e.g. usability, contextual help, interaction style) but better solutions are still required [2].

Sometimes, users do not interact with help systems the way they should neither help systems are their first option solving a problem, so alternative approaches should be used to provide contextual help [6]. Reducing the number of steps/actions on task execution and improved help systems can be alternatives to make systems more user friendly. These two alternatives can be done with the use of automation [7]. However, automation should be used with care [8] because an excess might lead to inadequate results [9].

With this paper, the authors claim that help tools to existing applications can be improved without accessing their source code, and with little effort. Therefore, the authors present FEL, illustrated with users of a university's online platform. FEL is a tool that provides demonstration-based help for both intra system and inter-system task execution. For this task automation and picture-driven computing

(PDC) concepts have been used. Results that suggest a trend in which the apprenticeship is improved with step by step visualization instead of illustrated manuals readings and following are presented. The paper makes the following contributions:

1. Supports the addition of help based on demonstration to existing interactive computing systems without access to their source codes;
2. Presents preliminary evaluation results indicating that the approach makes interactive computing systems and inter-systems tasks easier and faster to learn;
3. Demonstrates the approach with an example.

This article is structured as follows: on section 2 both related backgrounds and research are presented. Section 3 describes the university's online platform, and on section 4 the approach is presented. Section 5 presents the evaluation of the tool and on section 6 and 7 a discussion and conclusions respectively presented.

2 BACKGROUND

Automation is a valid concept when trying to make an interface user friendly. In order to automate it is important to properly understand all tasks associated to the system and the operator, being possible to assign an automation feasibility level to each [10]. It is important to ensure the communication between the human agent and the autonomous program guaranteeing that the level of automation is not exceeded, leading to bad results [9]. Palanque et al. [11] introduced techniques related to task modeling aiming to help in the design of interactive systems with usable automation.

Considering the learning process and following universal usability concepts, Shneiderman promotes the idea of multi-layer interfaces in which as the user acquires knowledge about a level he/she can move to a new one with increased complexity [12].

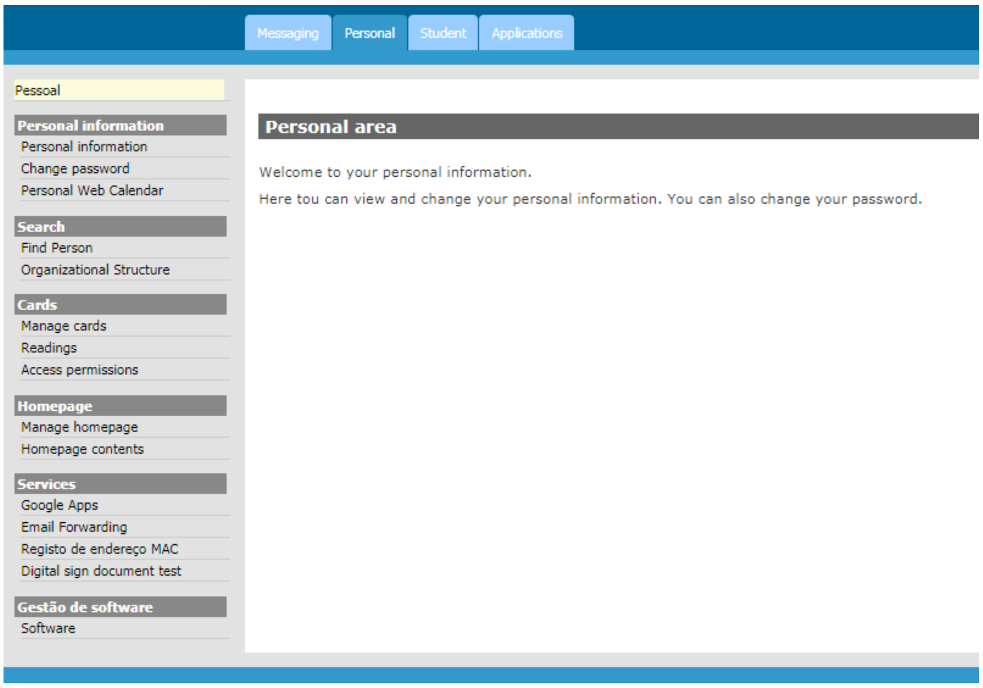
If to define a training program, it is proposed to focus in a model that properly describes all tasks in a complete and unambiguous manner. This is a modeled behavior of the system and includes interactions between the system and the operator [13]. With temporal ordering constraints, using operators of a concurrent formal notation (e.g. LOTOS [14]), it is easy to generate task-oriented help from the abstract formal specification that is focused on user interactions rather than aspects presentation alone [15]. *ConcurTaskTrees* (CTT) is a recognized notation for task modeling [16]. Palanque et al. present a contextual help system for user-driven applications that in a specific context provides help about "what, why and how". On it, it is claimed that the most effort is on the development of the systems with this help system embedded [17].

PDC allows automation following a certain model since events are triggered in response to presented graphical user interfaces (GUI)/images. *Sikuli* is a tool that allows automation of GUIs interaction with the use of screenshots [18] and with it, contextual help systems are possible by showing instructions and highlights on the actual interface rather than in a separate viewer [19]. Some alternatives to Sikuli are Automa [20] and RIATEST [21].

Applying automation on a non-intrusive way is possible using PDC. It consists on programs that allow people to write scripts using screenshots of GUIs [22] and, not having the need to access source codes. Tools have been developed using the previous concepts aiming to reduce the effort that users face while trying to learn how to perform tasks [23] and to simplify user interaction and integration of independent interactive systems [24]. This way an abstraction from the way things were done is created.

3 THE UNIVERSITY ONLINE PLATFORM

The university's management platform (see *Figure 1*) illustrating this work is used by a vast number of users and with a large age range. Every year there are (new) students, teachers and institution collaborators that in their daily activity require this tool. For that reason, a help system capable of satisfying everyone's needs in the best way is valuable.



Welcome to your personal information.

Here tou can view and change your personal information. You can also change your password.

Figure 1. University's online platform.

As for many applications, it has its help support in a separated page where tasks are textually described, and some images are presented as guide. Adding difficulty some tasks require to use other applications and task support is spread between those application's help systems.

The authors have chosen to automate the following three tasks as means of example to illustrate the proposed approach (described in the next section):

- Check attendance ("*Verificar assiduidade*") – allows users to quickly check their attendance (overall view or day search). For the specific date check, users must insert the date on a dialog box on a pre-defined format. This task was chosen because it is a very useful feature that is used by all students. This way, we are automating a frequent task and providing a learning opportunity via contextual help for those new to the interaction.
- Check evaluations sign up ("*Verificar abertura de inscrições para avaliação*") – directs users to an online page from the university platform where all the evaluations are registered and some of them may require inscription. With the goal of making all tasks complete, beyond directing to the accurate page it also checks the page for any evaluation where the inscription is still not made notifying the user whether there is any sign up. This one was chosen because it is a new feature and most of the users (students) still need to discover it.
- Synchronizing scholar calendar with Google Calendar ("*Adicionar calendário ao Google*") – has as challenge the interaction between multiple-applications (inter-systems task). With this task, it is possible to incorporate the university calendar with the personal calendar of each user. To perform this task, users not using FEL must check university online platform guidebook where they took the first steps, and then had to check Google help books to finalize the synchronization.

4 APPROACH

The FEL tool allows to perform tasks with a different kind of help on a university's online platform. It allows people to perform tasks while they are still learning. It was developed to provide help as an external system.

To reach help, users only need to select the task they are interested and execute it. From there the interactions with the online platform are automatically triggered. Interactions consist of clicking on GUI elements, text selection, text insertion, between others. The goal is that by viewing all the steps required to accomplish the task users learn by observation and get their task complete.

4.1 Task definition

To properly understand tasks and what should be automated the first step was to model them using CTT.

The enriched notation presented by Ornelas et al. in similar work was adopted [25]. With this enrichment, beyond the decomposition of the main task and the choice of which sub-tasks to automate some keywords are given to the subtasks name with the goal of smoothing the mapping between the task model and script commands. Additional information is also encouraged to be inserted in this phase with the target of complementing even more the information contained on the models; this information can be URLs to be used and even the images that Sikuli will search for when executing a picture-based script.

Alternatively to this process Machado et al. [26] present a tool that aims to reduce the complexity of Sikuli script definition. In it, the user performs the task, while running the tool, which records the steps and, with the use of Windows library user32.dll, captures mouse and key interactions with the interface. Then, it can generate a Sikuli script with screen recordings and commands mapped from the inputs.

4.2 Sikuli

By choosing to use Sikuli and without the use of any translation tool from CTT models, a better understanding of the Jhyton-based visual scripting API is required.

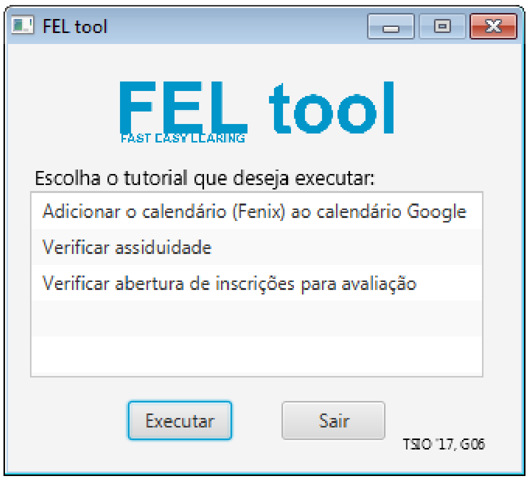
One of its advantages is the ability to, not only, use the visual scripting options that allows for all kind of interactions with the interface, by pixel recognition, but also the possibility to use standard programming commands that allow for more input-based checks, and the ease of repetition of tasks with the use of loop commands.

4.3 Wrapping up

When developing a contextual help system with the approach described before, we cannot think of CTT and Sikuli as unrelated. Modelling the task in an enriched way is the biggest effort as it already considers all the steps/actions to complete a task. The scripting phase is only the translation of the model into a language that computers can understand. Therefore, we first created the enriched task models that served as basis for the creation of the scripts.

4.4 FEL production

To make a single application that would concentrate all Sikuli scripts, a simple GUI was developed with use of JavaFX [27]. In it users can select a task to execute, from a list with the available ones. This GUI is shown in *Figure 2*.



Escolha o tutorial que deseja executar:

Adicionar o calendário (Fenix) ao calendário Google
Verificar assiduidade
Verificar abertura de inscrições para avaliação

Figure 2. FEL tool user interface.

Since FEL was developed only as means of an example, the program calls the scripts via command line, where the local Sikuli program is launched with the local script file. However, it is possible to completely integrate Sikuli with a Java application, writing the programs on the Integrated Development Environment (IDE) and not being necessary to access external files [28].

The first steps of all tasks were to open Firefox, browser the university online platform login page and ask for the insertion of sign in data. In *Figure 3* is shown the corresponded Enriched Task Model. On it there are two things: the types of tasks and the operators. For this modelling, we can see Abstraction Tasks (☁) that indicate an abstract task that is defined by its sub-tasks, Application Tasks (🖥) which denote that the task is performed independently by the computer/program and Interaction Tasks (🧑) consisting on the tasks where the user must interact with the system. As for the operators in this case were used the *SequentialEnabling* (>>), representative that the task on the left must be performed to execute the other one, *OrderIndependence* (|=|), significant that both tasks have to be executed but the one who is performed first is insignificant, and *SequentialEnablingInfo* ([]>>), where the task on the left has to be performed first but the information on it is of importance for the task in the right.

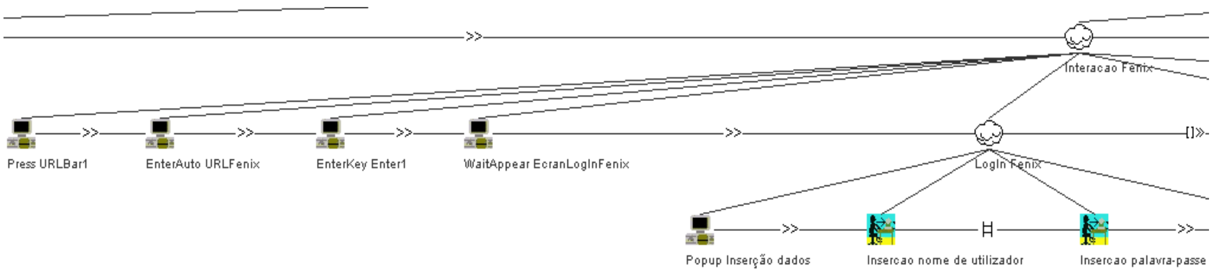
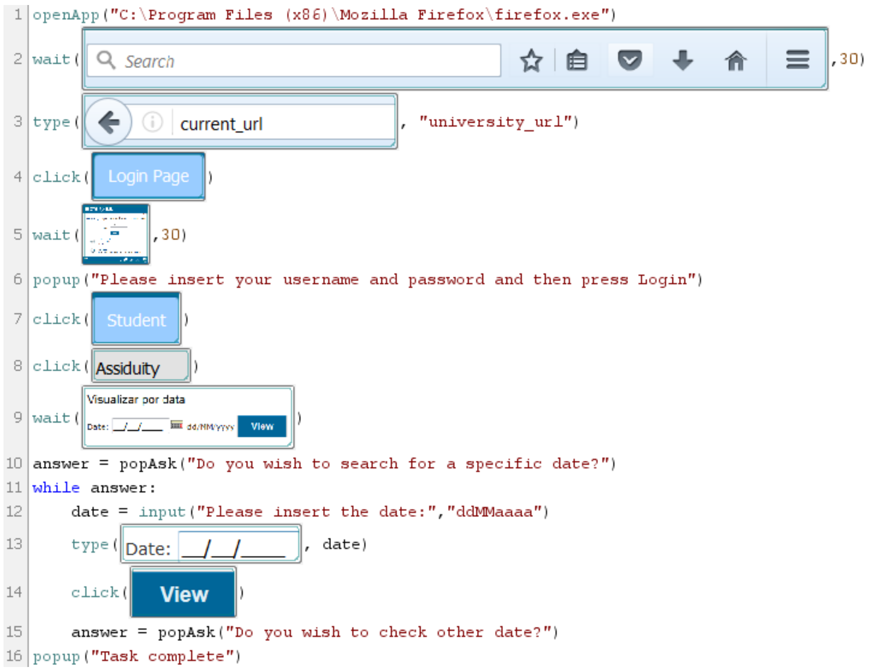


Figure 3. Browsing university online platform and asking for sign in - partial Enriched Task Model

Observing the task model, we can see that the task interaction was considered abstract and over-decomposed in other tasks. These were, browsing the site by its URL on the first four application tasks, followed by one more abstract task where through a popup, the user is prompted for the login and password entry. It was considered that the insertion of the login and the password were order independent and that this data is important information for the load of the online platform, as the login and password are the ones who will define the content displayed on the personal page.

Figure 4 is presented with the goal of demonstrating the ease of visual-based programming as for the use of standard programming structures, in this case the while loop. It corresponds to the check attendance task.



```
openApp("C:\Program Files (x86)\Mozilla Firefox\firefox.exe")
wait( Search ,30)
type( current_url , "university_url")
click( Login Page )
wait( ,30)
popup("Please insert your username and password and then press Login")
click( Student )
click( Assiduity )
wait( Visualizar por data Date: __/__/____ dd/MM/yyyy View )
answer = popAsk("Do you wish to search for a specific date?")
while answer:
    date = input("Please insert the date:","ddMMaaaa")
    type( Date: __/__/____ , date)
    click( View )
    answer = popAsk("Do you wish to check other date?")
popup("Task complete")
```

Figure 4. Checking attendance – Sikuli script

On the script, it was possible to automate a task using only seven different commands: *openApp*, *wait*, *type*, *popup*, *click*, *popAsk*, and *while*. Passing an executable file path to *openApp* it will suffer there and run it, wait allows to pass an image and a window of waiting time, type allows text and character insertion, popup displays a message to the user, click presses a given image, *popAsk* displays a message and gives the user a choice (yes or no) and while works as in standard programming being in loop until the variable turns false. Firstly, Firefox is launched and by recognition of its search bar the link for the university online platform is inserted. When the login screen is shown, the user is asked to insert his login and password. Then, the user is directed to the proper screen, automatically, and finally it is asked if there is a specific date to check attendance. There either the script ends or if inserted a date of interest it searches the attendance on that day.

5 EVALUATION

To understand the potential impacts of using a contextual help system a preliminary evaluation to the FEL tool was made. Ten users participated. There were three females and the age range went from fifteen to fifty-four. They were all familiar with the use of computers, using them at least three times per week, but without previous interaction with the university online platform. Users were divided into two groups. At the end, all users had to answer an adapted version of the USE questionnaire[1] regarding the help system that was used (https://goo.gl/mxNxdz).

Both groups had to perform twice the three tasks (described on section 3), i) Check attendance, ii) Check evaluations sign up, and iii) Synchronizing scholar calendar with Google Calendar. First, the task had to be carried out using a help tool. Group one used the current university platform guidebook and group two used FEL. Then they had to perform again the task without recurring to any kind of help.

User's interaction was recorded and then analyzed. Notes were taken during the studies. When analyzing, special attention was given to the time needed to complete tasks, and the number of errors and wait times superior to five seconds on the second time performing a task. Participant characterization was made, and users filled a questionnaire that addresses four aspects: Usefulness, Ease of Use, Ease of Learning and Satisfaction (as defined in the standard USE questionnaire). Answers were on a 7-point Likert scale with values from 1 (strongly disagree) to 7 (strongly agree). The questionnaire included open question on the strengths and weaknesses of the tool.

5.1 Results

On *Figure 5* is presented a comparison between the average times on performing the tasks. The left ones represent the first time performing each task, in which the task was performed with a help system (current guidebook or FEL depending on the group users were sorted into) and the ones on the right represent the second time performing the task, recurring to no kind of help. From these results, we can see that using the FEL (users from group two), on a first-time interaction with the system, tasks are performed much faster (5,71 times faster) and performing tasks on a second time, without any help system, users from group one (without FEL) performed only 1,26 times faster (average from the three tasks).

Regarding errors and wait times superior to five seconds (indicating potential hesitancies), on *Figure 6* is presented the difference between the number of wrong clicks and wait times superior to five seconds when performing tasks, on a second time, being on group two, and the same numbers regarding users sorted on group one. To calculate the difference, average values from the experiments were used; regarding wrong clicks group 1 $\bar{x}$ = 0,2 σ = 0,56 and group 2 $\bar{x}$ = 0,67 σ = 1,05; for wait times group 1 $\bar{x}$ = 0,33 σ = 0,90 and group 2 $\bar{x}$ = 0,47 σ = 1,06. As for the wait times superior to five seconds, the occurrences are very similar whether in the group one or two. Concerning wrong clicks performing a task, group two presented an average of 0,46 more wrong clicks than group one.

One of the differences between both cases is that users learning from observation, group 2 users, are exposed to more distractions. Whilst having to execute the task while reading a page of text and viewing images is done entirely by the user, when observing an automated interaction with little intervention the attention required is diminished.

[1] http://garyperlman.com/quest/quest.cgi?form=USE. Accessed on: 2018-05-08

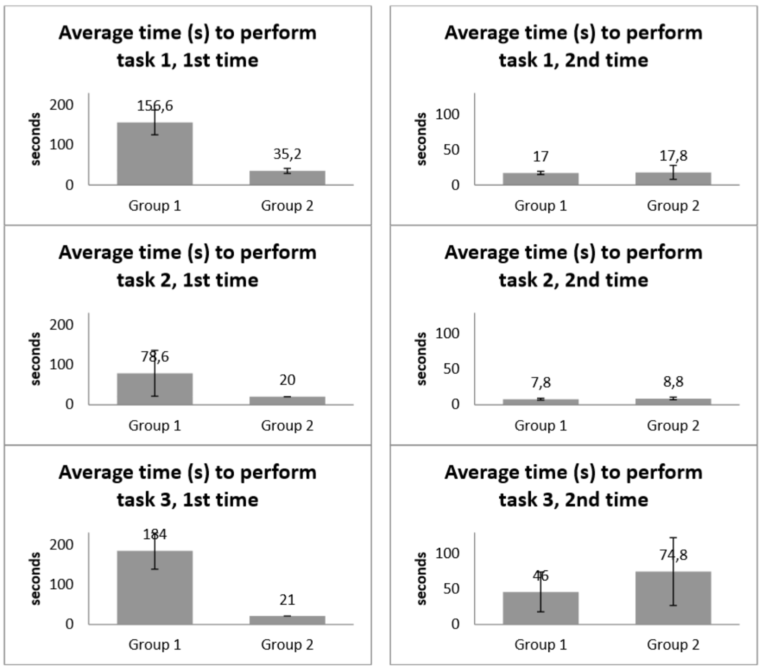


Figure 5. Average time, in seconds, displayed by interactions. All plots have two bars representing the two different groups.

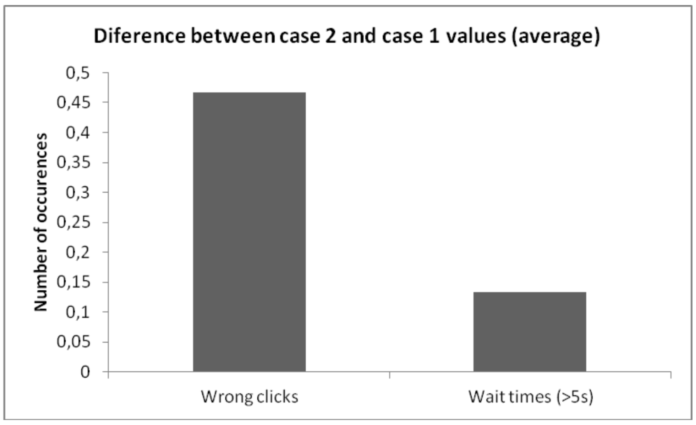


Figure 6. Difference between the number, average from tasks performed on a second time, of wrong clicks and wait times superior to five seconds between case of study number two (using FEL tool) and number one (using current guidebook).

On *Figure 7* the results from the USE questionnaires are summed up. It is shown that in all aspects the FEL tool was scored higher than the current guidebook. Little variation was shown on the Ease of Learning category but as for Usefulness and Satisfaction they are higher for more than 1 point. It was emphasized by group one users that there were some text inconsistencies and that the reading was too extensive. As for group two users, one user referred that some automated steps were too quick.

6 DISCUSSION

On a technological developed society like ours, with a wide range of applications in our hands, where almost everyone must deal with computerized interactive systems and the learning process is complex, our society should have realized that most current help systems are inefficient and may bring frustration to people. Optimization and improvement of all processes should be the priority regardless of the industry.

PDC on its non-intrusive way may be the key to improve learning processes and improve both intra-system and inter-system task execution.

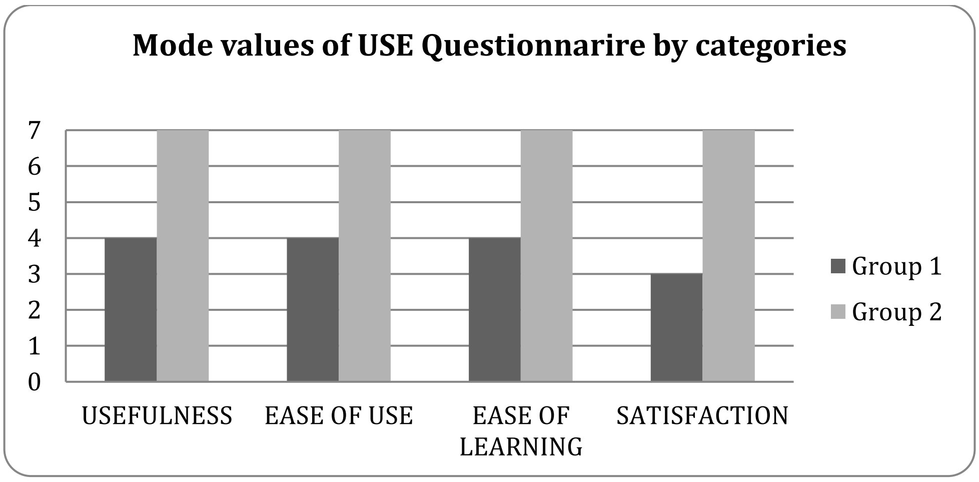


Figure 7. Average results of USE Questionnaire divided by the four aspects and filtered by experiment group.

The application of the concepts described on this paper may contribute for a higher efficiency day-to-day work, enabling accelerated learning. This approach also allows users to learn at the same time as the tasks are performed.

The presented results must be interpreted carefully due to the reduced number of participants. Nevertheless, the results indicate that the approach enables users to perform tasks easier and faster without compromising the learning (as users tend to learn better by doing instead of just watching). With regards to acceptability, very good results were obtained but user satisfaction (5 out of 7) can still be improved. Based on the comments collected we believe this can be further improved by reducing (partially) the speed of tasks execution. Overall, the results seem to indicate clear benefits of the approach over traditional help.

With many programs and functionalities where some are performed sporadically, tools like FEL allow to quickly get things done using a light tool rather than searching the several guidebooks or online videos. An experimental comparison with works such as Pause-and-Play [29] aiming to facilitate linking between video tutorials with applications should also be performed. On Pause-and-Play it is proposed an improvement on learning via video tutorials. The improvements consist on receiving a video-tutorial that recognizes the main steps on it, using computer-vision, and with the inclusion of plugins on applications it is possible to track user behavior, adapting the tutorial to the tracked performance. Additionally, it is provided a functionality which allows easy navigation through the most important parts of the video. We believe that FEL's automatic and inter-systems task execution present and important advantage over similar tools.

7 CONCLUSIONS

Recurring to PDC and automation, system producers may improve substantially their tools usability and users' satisfaction. The conjunction of an enriched task model and a PDC tool like Sikuli is a good one. The thought and the deep specification of the process are made in the first and an easy mapping with simple and intuitive commands is allowed in the second.

A major advantage of this approach is the abstraction from the source codes, which on many cases is inaccessible. This kind of demonstrated-help is available for everyone with an interest and willingness.

From our case study, the results are clear. Performing tasks for a first time or in cases that we are not certain of all steps a contextual help like the one our tool offers, with picture-based computing and automation, is time beneficial and provides basis of learning like the apprenticeship with the current method, a guidebook.

In future research, it is planned to improve the tool using more complex examples and applying it to different software. It is also planned to run further evaluations with a larger number of users.

ACKNOWLEDGEMENTS

This work was supported by ISTAR-IUL (UID/MULTI/0446/2013). José Luís Silva acknowledges support from Fundação para a Ciência e a Tecnologia (FCT, Portugal), through project UID/EEA/50009/2013.

REFERENCES

[1] G. Adams and B. Rau, “Job Seeking Among Retirees Seeking Bridge Employment,” *Personnel Psychology*, vol. 57, no. 3, pp. 719–744, Sep. 2004.

[2] N. Schneider, S. Schreiber, J. Wilkes, M. Grandt, and C. M. Schlick, “Foundations of an age-differentiated adaptation of the human-computer interface,” *Behaviour & Information Technology*, vol. 27, no. 4, pp. 319–324, Jul. 2008.

[3] J. Tidwell, *Designing interfaces: patterns for effective interaction design*, 2. ed. Beijing: O’Reilly, 2011.

[4] J. Huh and M. S. Ackerman, “Designing for All Users: Including the Odd Users,” in *CHI ’09 Extended Abstracts on Human Factors in Computing Systems*, New York, NY, USA, 2009, pp. 2449–2458.

[5] B. Shneiderman, “Universal Usability,” *Commun. ACM*, vol. 43, no. 5, pp. 84–91, May 2000.

[6] G. Dworman and S. Rosenbaum, “Helping Users to Use Help: Improving Interaction with Help Systems,” in *CHI ’04 Extended Abstracts on Human Factors in Computing Systems*, New York, NY, USA, 2004, pp. 1717–1718.

[7] G. W. Paynter, “Automating iterative tasks with programming by demonstration,” Thesis, The University of Waikato, 2000.

[8] P. Palanque, C. Martinie, and C. Fayollas, “Automation: Danger or Opportunity? Designing and Assessing Automation for Interactive Systems,” in *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, New York, NY, USA, 2017, pp. 1257–1260.

[9] G. A. Boy, “Cognitive Function Analysis for Human-centered Automation of Safety-critical Systems,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 1998, pp. 265–272.

[10] C. Martinie, P. Palanque, E. Barboni, and M. Ragosta, “Task-model based assessment of automation levels: Application to space ground segments,” in *2011 IEEE International Conference on Systems, Man, and Cybernetics*, 2011, pp. 3267–3273.

[11] P. Palanque, C. Martinie, and C. Fayollas, “Automation: Danger or Opportunity? Designing and Assessing Automation for Interactive Systems,” 2017, pp. 1257–1260.

[12] B. Shneiderman, “Promoting Universal Usability with Multi-layer Interface Design,” in *Proceedings of the 2003 Conference on Universal Usability*, New York, NY, USA, 2003, pp. 1–8.

[13] C. Martinie, P. Palanque, D. Navarre, M. Winckler, and E. Poupart, “Model-based Training: An Approach Supporting Operability of Critical Interactive Systems,” in *Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, New York, NY, USA, 2011, pp. 53–62.

[14] T. Bolognesi and E. Brinksma, “Introduction to the ISO specification language LOTOS,” *Computer Networks and ISDN Systems*, vol. 14, no. 1, pp. 25–59, Jan. 1987.

[15] S. Pangoli and F. Paternó, “Automatic Generation of Task-oriented Help,” in *Proceedings of the 8th Annual ACM Symposium on User Interface and Software Technology*, New York, NY, USA, 1995, pp. 181–187.

[16] F. Paternò, “ConcurTaskTrees: an engineered notation for task models,” *The handbook of task analysis for human-computer interaction*, pp. 483–503, 2004.

[17] P. Palanque, R. Bastide, and L. Dourte, “Contextual help for free with formal dialogue design,” in *HCI (2)*, 1993, pp. 615–620.

[18] T. Yeh, T.-H. Chang, and R. C. Miller, “Sikuli: Using GUI Screenshots for Search and Automation,” in *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*, New York, NY, USA, 2009, pp. 183–192.

[19] T. Yeh *et al.*, “Creating Contextual Help for GUIs Using Screenshots,” in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, New York, NY, USA, 2011, pp. 145–154.

[20] Bugfree Software, “Next Generation GUI Automation | Automa.” [Online]. Available: http://www.getautoma.com/. [Accessed: 19-Mar-2017].

[21] Cogitek, “RIATest - Web application automation tool.” [Online]. Available: http://www.cogitek.com/riatest.html. [Accessed: 19-Mar-2017].

[22] L. Hardesty, “Picture-driven computing,” *MIT News*, 2010. [Online]. Available: http://news.mit.edu/2010/screen-shots-0120. [Accessed: 19-Mar-2017].

[23] J. D. Ornelas, J. C. Silva, and J. L. Silva, “Demonstration-based Help for Interactive Systems,” in *Proceedings of the 2Nd International Conference in HCI and UX Indonesia 2016*, New York, NY, USA, 2016, pp. 125–128.

[24] J. L. Silva, J. D. Ornelas, and J. C. Silva, “Make It ISI: Interactive Systems Integration Tool,” in *Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, New York, NY, USA, 2016, pp. 245–250.

[25] J. D. A. Ornelas, “Improving the use of interactive systems,” 2016.

[26] V. Machado, N. Lopes, J. C. Silva, and J. L. Silva, “Picture-Based Task Definition and Parameterization Support System,” in *Recent Advances in Information Systems and Technologies*, 2017, pp. 592–601.

[27] “Java SE | Oracle Technology Network | Oracle.” [Online]. Available: http://www.oracle.com/technetwork/java/javase/overview/index.html. [Accessed: 10-Sep-2017].

[28] “How to use Sikuli Script in your JAVA programs — Sikuli X 1.0 documentation.” [Online]. Available: http://doc.sikuli.org/faq/030-java-dev.html. [Accessed: 10-May-2018].

[29] S. Pongnumkul *et al.*, “Pause-and-play: Automatically Linking Screencast Video Tutorials with Applications,” in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, New York, NY, USA, 2011, pp. 135–144.