International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 2, April 2021, pp. 1549∼1560
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i2.pp1549-1560

Help through demonstration and automation for interactive computing systems: A survey of recent works

Pedro Rodrigues[1], **José Luís Silva**[2]
[1]Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal
[2]ITI / LARSyS / Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR-IUL, Lisboa, Portugal

Article Info

Article history:

Received Jan 17, 2020
Revised Aug 25, 2020
Accepted Sep 10, 2020

Keywords:

Automation
Demonstration-based help
Help systems
Learning
Picture-driven computing

ABSTRACT

Usability is very important however, it is still difficult to develop interactive computing systems that meet all user's specificities. Help systems should be a way of bridging this gap. This paper presents a general survey on recent works (building upon previous surveys) related to improving applications' help through demonstration and automation and, identifies which technologies are acting as enablers. The main contributions are, identifying: i) which are the recent existing solutions; ii) which aspects must be investigated further; and iii) which are the main difficulties that are preventing a faster progress.

Corresponding Author:

José Luís Silva
ITI / LARSyS / Instituto Universitário de Lisboa (ISCTE-IUL)
ISTAR-IUL, 1649-026, Lisboa, Portugal
E-mail address: jlcsa@iscte-iul.pt

1. INTRODUCTION

User interfaces (UIs) have a major role in any interactive computing system (ICS). The life cycle of technology proposed by Norman [1] state that when technology satisfy basic user needs then they become more interested in efficiency, convenience and pleasure. Therefore, usability and user experience became crucial. Indeed, usability is vital for websites, products and services to achieve success. However, although all advances on user-centered approaches and design, some interfaces are still difficult to use, causing frustration to users [2]. Managers and developers often recognize it but still leave it apart from the product-development process [3, 4].

To address user's difficulties some ICS provide help systems however, it is still verified a major difficulty in making people use them. Dworman *et al.*, [5] state that this behaviour is explained by either users not recognizing help systems existence, not wanting to pause their current tasks to search help or because they want to find the solution on their own. Nevertheless, the design of effective user-centered help systems is still seen as a weak negotiator point by the business because it is faced as a secondary functionality [4, 6] as shown in Figure 1.

To combat user's resistance to use help systems and still make users familiar with tools, especially the ones with some complexity, training programs focused on task-modelling were proposed [7]. They aim to understand which are the key aspects that depend on the user. This modelling has, yet, the potential to improve systems design if considered in the implementation process [8]. However, this modelling has a certain level of complexity and some tools propose to overlap it by analyzing the task being accomplished and defining it [9].

Objectifying the improvement of aid systems, contextual help proved to be effective in learning user interfaces. Contextual-help defines the paradigm of help given in the context where the doubt has risen up,

e.g. a certain tool-tip associated to a button [10]. This concept is similar to the situated action theory which states that knowledge can only be interpreted in its context. From this concept help tools have arose, like AIDE which through communication with users, understanding the context of the doubt, discovers and solves the issues [11]. Similar approaches are demonstration-based help that in addition to working in the context of the application demonstrates how a certain task is to be executed. For instance, through automation scripts [12] and, model-driven help [10] that firstly models the system and details interactions on it and then starts the automation/explanation design.

Different approaches to provide help have been developed. Either using video, tutorials, functionalities as automation and picture-driven computing (PDC), among others [11, 13-16]. Some also suggest approaches based on crowd learning, like help forums, arguing that people learn better when discussing subjects and an many-to-one support is more efficient [17, 18]. Many current help systems are still based on user manuals or do not provide inter-application support. The major barriers to an overall improvement are the increase on systems' complexity [19] and impossibility to access and change source codes.

Overcoming user's issues, this paper reviews how existing help systems are being implemented and which technologies are being used. The revision aims to identify existing approaches to develop help systems and what can still be improved on them. Additionally, understanding how far help tools are connected to the application (e.g. can be separated), whether approaches like continuous learning [20] either supervised or unsupervised are applicable and, how far could a crowd-sourced system be beneficial. The review includes applications that implement help mechanism or automate certain (sub)tasks to overcome user's issues (e.g. usability).

This paper is organized as follows. Section 2 presents topics related to most recent help solutions (e.g. picture-driven computing, automation). Section 3 reviews existing help solutions and sort them into several high-level categories. In Section 4, both comparison between identified help solutions and critical analysis are made. Finally, conclusions and directions for future work are presented in Section 5.

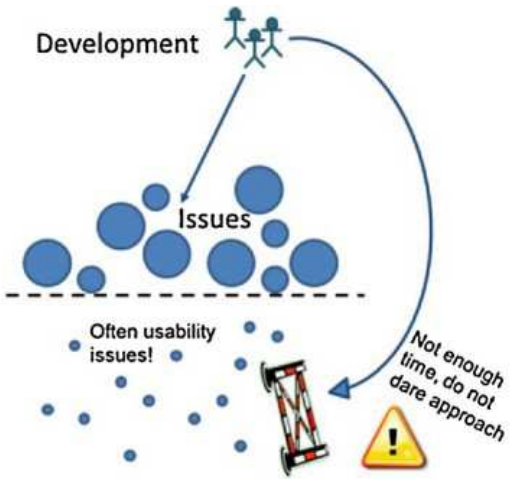

Figure 1. Usability study in a large company focusing on problems and barriers to usability [4]

2. BACKGROUND

This section introduces important concepts associated with most recent help and automation tools. The section starts by describing picture-driven computing (PDC), a paradigm that enables computer programming based on screen shots, not just code. The two main approaches based in this paradigm are presented (sikuli and robotic process automation). Then automation, a mechanism to reduce users' effort by automating steps to accomplish some task and, task-models are presented.

2.1. Picture-driven computing

Kourousias and Bonfiglio present the visual information in a 2-dimensional spatial domain as nuclear data of the PDC paradigm. This visual information is what intends to be visible on a graphical output device [21]. This paradigm is useful for task automation of systems already developed and/or without access to their

source code [11].

One way to implement PDC is by using the Sikuli [21] tool. This is an MIT open-source programming environment developed using python and using OpenCV (OpenCV: https://opencv.org (last accessed: 14 January 2020)) packages for finding images on the screen. Sikuli was developed to empower help designers in the general public (computer-skill independent) to create contextual help [15]. The tool allows for standard programming functions (e.g. loops, conditions, variables) but with the addition of target patterns identification on the screen. To use the tool, users can take advantage of the SikuliX integrated development environment (IDE) that is a basic script editor to load, edit, save and run scripts. In what regards non-picture related functions users must have previous basic computer programming basis. On Figure 2 an example of a Sikuli script developed on the IDE.

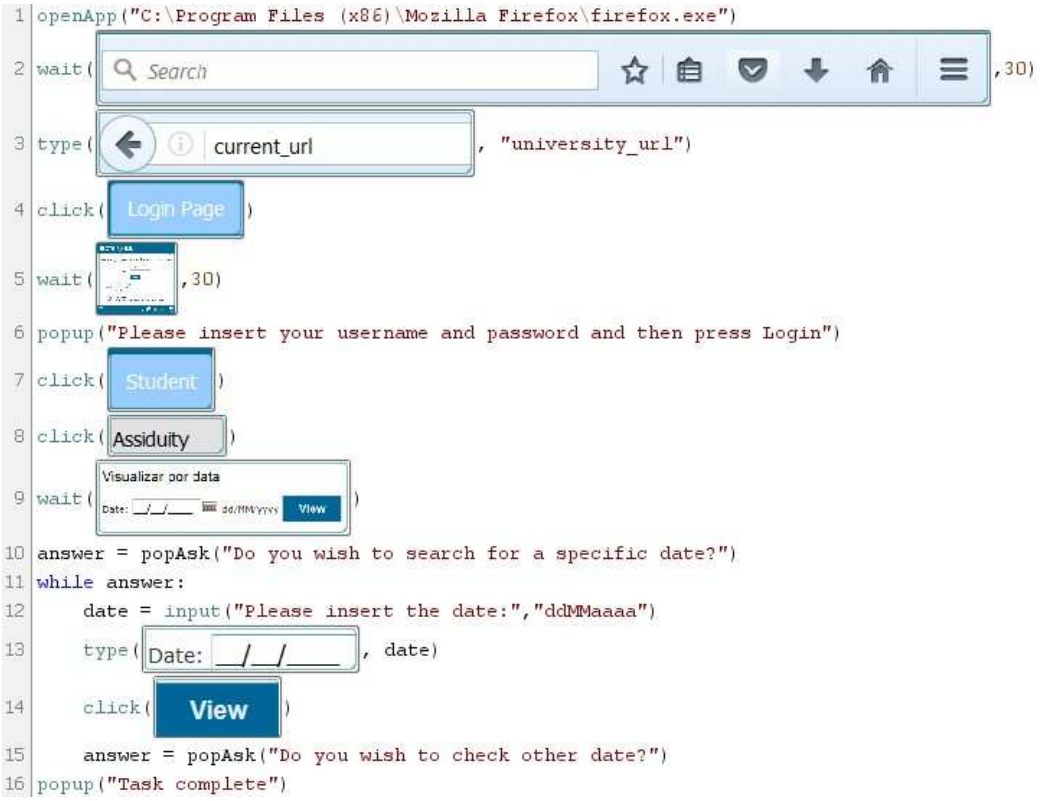

Figure 2. Sikuli script, example of a script developed using sikuli IDE

In the context of developing help systems, sikuli's user can select an element from the interface by capturing its' screenshot and selecting the type of contextual help to add (available action to perform on specified UI elements) to that element. Some examples are highlights, clicks, and type into (writes a specified message when an element is pressed). When presenting the resulting help system to users, pixels on the screen are searched, elements located and the corresponding action reproduced. See the work of Yeh *et al.* [15] for additional information about creating contextual help using screenshots.

Another way to implement PDC solutions is done using robotic process automation (RPA). As stated by Schatsky *et al.* [22] RPA software automate repetitive, rule-based processes usually performed by people sitting in front of computers. The interaction with computers is done by mimicking the interaction between humans and computers and therefore robots can open email attachments, complete e-forms, and record and replay data [22]. To mimic user interactions, current robots operate in two main modes. Either they are familiar with applications, usually through communication plugins, and can interact directly with all UI elements (e.g. buttons, text boxes, drop menus, combo boxes) or they operate through PDC using also screenshots and image recognition techniques.

On Figure 3 a synthesis of RPA goals and its description is presented. This information was gathered

from one of the market RPA applications. The goals emphasized are increased customer satisfaction, better resource utilization, increased accuracy and improved productivity.

Figure 3. RPA goals, Synthesis of RPA goals retrieved from Nice [23]

Most common RPA applications are UIPath [24], BluePrism [25], and Automation Anywhere [26]. They are designed mainly for companies which recognize the existence of tasks that pose as good candidates to be automated [27].

2.2. Automation and task-models

This section describes automation and concepts related to it (e.g. tasks and task-models). Before automating it is important to properly understand the task that will be automatically performed. This can be achieved with a detailed design and analysis. Adequate notations for this purpose are also described in this section.

Automation consists on the transformation of systems' functions into automatic processes to affect it following a specific target [19]. Some basic concepts associated to automation are:

- Business process-consists of sequences of activities
- Activity-discrete process step performed either by machines or humans (activities may consist of one or more tasks)
- Worklist-set of tasks to be performed by a user in a workflow system
- Workflow-is the automation of a business process in whole or in part, during which documents, information or tasks are passed through participants following a set of rules [28, 29]

One way to describe automation level of a system corresponds to the identification of the proportion of automatic actions [19]. They are three different types, full automation, semi-automation or manual. Fully automated are fully conducted by a computer and all possible outcomes are predicted. Semi-automated requires an interaction between the computer and a human that validate, compare, or add information to the workflow. Manual are usually complex and requires comprehensive knowledge and skills to take adequate decisions [28]. Other way to describe automation level is using a scale from one to ten. In this scale a value of one represents business process on which the computer offers no assistance and humans take all decisions and actions. A value of ten means that the computer decides everything, acting autonomously and ignoring human inputs [30].

On critical workflows a semi-automated approach is best suited making the system to work on a multi-agent basis (an human agent interacting with a software agent) [31]. However, some problems and concerns might arise:

- When the automation requires user to input some variables to set the session the automation can be put at risk. This happen either because the input parameters are not the expected or because the responsible might forget that the automation program was waiting for the parameter [19]
- The fact that people cannot recognize repetitive tasks worthy of automating leading to bad automation task selection. User interviews revealed that tasks were identified as repetitive if manually performed numerous times in a row or if complex but infrequent [32]

Those problems might be mitigated with an accurate process definition that can be accomplished with task-models and proper task definition [7, 33].

Powerful analytical tools such as Petri [34] or concur task trees (CTT) [35] provide a thorough process (specifying tasks) definition and enable formal workflow analyze (e.g. correctness and consistency). A valuable aspects of these tools is the decomposition of tasks in its sub-tasks enabling a proper understanding of required steps to accomplish them [7]. To define a task, it is necessary to identify the objects and actions that define the communication. Objects consist on entities that are manipulated to perform tasks (e.g. menus, icons, state requests of some application, etc.) [35]. Those tools also provide hierarchical logical structure allowing for different abstraction level and process refinement. A variety of temporal relationships operators supporting interactive behaviors and representation of relevant relationship is also provided. To express temporal relationship among tasks LOTOS [36] concurrent notation is used. CTT editor and HAMSTERS [37] enable the definition of task-models being possible to specify the type of each task:

- User task-performed entirely by the user
- Application task-completely executed by the system
- Interactive task-performed by user interactions with the system
- Abstract task-represents high level task including a set of tasks of different types

The possibility to represent concurrent tasks makes CTT different from unique sequential relationship available on GOMS [38], despite their similarity on hierarchical tasks disposition [30, 39]. Figure 4 presents a partial task-model (in the context of a flight software board) developed with HAMSTERS. Task-models might represent a barrier for task automation mainly because the additional effort to the overall development [29, 32]. Aiming at reducing the effort required for the development of task models, Machado *et al.* proposed a tool that constructs CTT task models from the execution of tasks [9].

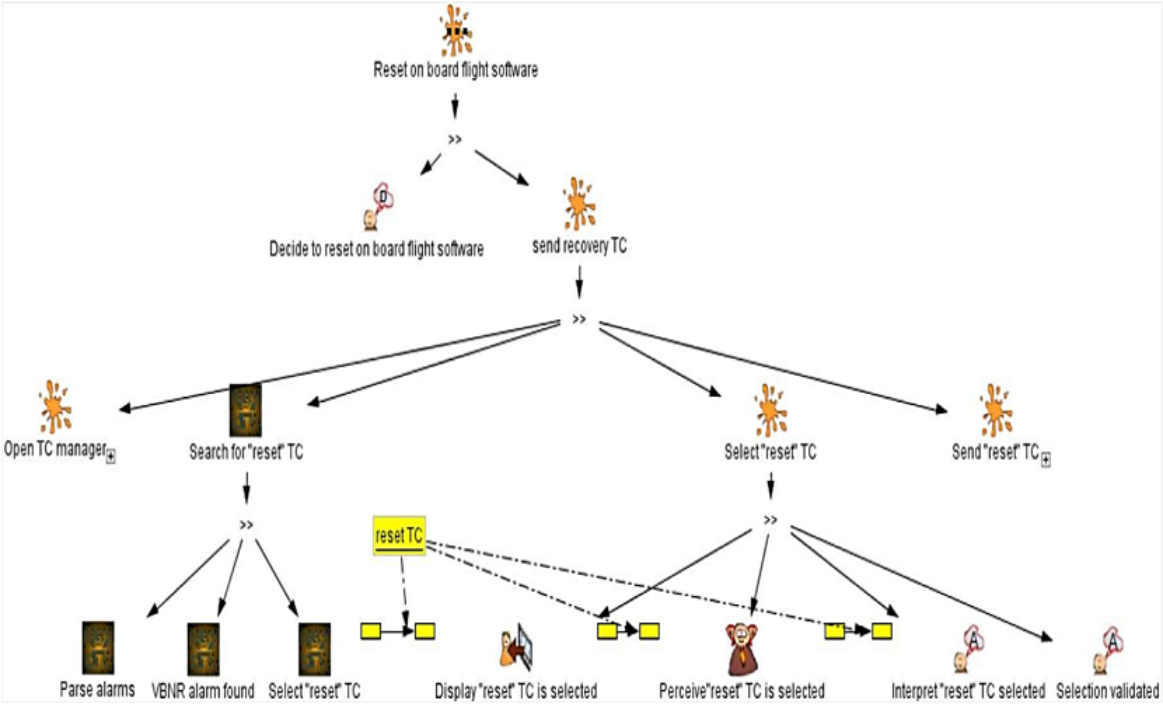

Figure 4. Task-model, Example of a task-model developed with HAMSTERS [30]

3. LITERATURE REVIEW

This section presents approaches which aim to improve daily interactions between users and computers. The section starts by presenting the methodology used in the selection process and then presents the identified work grouped by interaction core used by the approaches, i.e. how interaction with applications is performed (e.g. script and automation-based, video-based, GUI elements based). The intent was to perform a (not systematic) literature review of recent work adding to existing surveys.

3.1. Methodology for tool selection

The search for this review was made through digital repositories. The repositories from which most information was taken are ACM Digital Library, Springer, IEEE Xplore and Elsevier. Additionally, information from AAAI Digital Library, Wiley Online Library, Taylor and Francis Online and Scientific Research publisher was also taken.

The range of presented tools focus mainly in the 2010-2018 period. This choice was due to the identification of a review made by Grossman and Fitzmaurice [40] that presents help tools previous to 2010 based on video and animated documentation, contextual assistance, and contextual video assistance. Two additional works where included because they present interesting approaches which with the addition of current technology capabilities would pose as good candidates on the help-tools umbrella.

The keywords used on the research were 'action repetition', 'contextual help', 'crowdsourced help', 'demonstration help', 'GUI automation', 'knowledge sharing', 'model-driven help', 'programming by demonstration', 'software support', 'workflow automation', 'picture-driven computing', 'tutorial' and 'user support'. Relevant cited papers were also included.

3.2. Scripts and automation-based

Sheepdog [20] is a programming-by-demonstration system that works in different phases. First, it learns from multiple experts, each performing the same procedure (set of steps to achieve a goal) directly on a Windows desktop. With the recording, Sheepdog models the procedure allowing different configurations. These models can be reproduced on an end-user device via an executable file. When generating the models, Sheepdog abstracts from low-level Windows events into clean high-level representations and uses input/output hidden Markov models to determine the odds of occurrence of the next node. The interface where the system automation is mapped should be collaborative to allow the user (expert) to add extra-input. When replaying the script, step by step, the system takes a snapshot and from it determines and indicates what is the next step.

CoScripter, previously called Koala [41], is a collaborative scripting environment for recording, automating and sharing web-based processes. It has two main modules, a plug-in that allows users to record and play actions and a repository where users can share their scripts, rate and comment other scripts. The actions recorded are recognized and replayable via HTML interpretation.

ActionShot [42] is an extension to Firefox web browser built on top of the CoScripter web recording/playback platform. Unlike CoScripter, ActionShot continuously records users' browser history grouping actions by pages' host name. It also provides a visual interface that allows users to use or share any action they have ever performed. The description of each module is based on text taken from HTML code (usually the text displayed on widgets).

CoCo [43] is a system that automates web tasks. It takes advantage of a repository with previous web scripts and users' personal web browsing history. People ask questions via Twitter and the system can either respond asking for more information or with an answer. The system can cover each user needs.

TaskTracer [44] is able to track users behavior when using applications like Microsoft Office, e-mail, and Internet Explorer (an extra add-in has to be installed) being able to recover the full context of the task if desired. To classify the task, the user must indicate its beginning and ending. Authors acknowledge it as an extra burden. They aim at the ability to restore all applications associated with a process, the documents used and even the indication of the last changes.

Smart Web Tutor [18] proposes an approach to enhance massive open online courses (MOOCs) with education resources development for instructors and students, allowing realtime collaborative learning for students. As for the recording phase, it is based on the DOM structure, common on web-sites, and works on three main phases, recording, optimizing, replaying. A similarity formula that considers platforms variability is also considered as different systems may require different steps to accomplish a task. For real-time collaborative learning, authors propose a communication protocol to send each step on a synchronous way, making possible to a group of people to watch a procedure executing on real-time.

3.3. Video-based

Ambient Help [45] works on a secondary monitor. While people are performing a task on the primary monitor Ambient Help is automatically displaying videos or/and textual information relevant to the process. Whenever people need help or want to see Ambient Help suggestions they focus on the secondary monitor and search the desired help-content.

Pause-and-Play [14] proposes a learning improvement via video tutorials. The approach consists on receiving a video-tutorial and recognizing its main steps via computer-vision algorithms. Then to trace user behavior, it is possible (with the addition of plugins) to pause and play the video tutorial according to user performance on the application. In addition, a functionality which allows easy navigation through the most important parts of the video was developed.

ToolClips [13] is a contextual-based help system that aims to replace tooltips associated to UI elements. Instead of standard tooltips (e.g. with icons' name), when the cursor pass over an UI element, the user can access video or text with information about the elements' use. With this approach users are presented with extra-information inside the application.

3.4. GUI elements based

LemonAid [17, 46] is a crowdsourced contextual help system. It provides technical help based on the selection of GUI elements (label, widget, link, and image) instead of keywords. People get help by selecting an UI element and, from there, LemonAid presents the five most asked questions by the community. Users can use the search bar to enter keywords, refine the questions shown, make new questions or see the answers to existing questions. Technically, it works as a layer above a web application UI, independent from back and front-end implementations. Authors claim that one-to-many support is more efficient and provides greater cost savings.

DemoHelp [47] is an help system that takes advantage of PDC and automation (using Sikuli scripts) and is to be used side by side with an application. Users select an action to execute in the help system and the procedure is triggered on the application. To facilitate the creation of Sikuli scripts users can enrich CTT task models with applications' UI elements and select possible scenarios. Based on this work, an help tool (FEL) [48] was developed. An user study was performed comparing the help provided by the developed tool and guidebooks. Results shown benefits using FEL on first-time task execution.

Interactive systems integration tool (ISI) [12] is a tool that also works with enriched task models (CTT) and scenario selection (using a tool from the Human Interfaces in Information Systems Laboratory [49]) to automatically create Sikuli scripts. The tool removes complexity from original GUI, presenting a simplified GUI with complex tasks automated and ready to be reproduced. The tool can merge multiple interfaces from different applications into only one GUI. When the user selects a task, all steps are executed by Sikuli on a virtual machine being presented to the user only the final result (hidding the original GUI). The approach works with any application/platform.

Help, it looks confusing (HILC) [50] is a tool that relies on computer vision techniques rather than accessibility APIs being therefore, domain independent. It focus on personal automation and script generation from non-programmer users and automates procedures by demonstration. When demonstrating, the tool produces the needed screenshots and corresponding mouse-keyboard events. Scripts can be made from video records, with specialized screen casting software, or with a sniffer program. Inputs are converted to log files that pass through a classifier algorithm. However, the authors state that pure programming-by-demonstration is still unrealistic. They verified that for small procedures the classifier algorithm does not have great performance yet.

Contextual help for adaptive interfaces (CHAIN) [10] was developed to provide model driven contextual help to adaptive user interfaces (interfaces that change their presentation at runtime). The adaptation to change is made by an association between concrete elements and scene elements. Concrete elements are defined by specific parameters like the concrete *id* on the code, and scene elements are associated to them (e.g. a menu is a concrete element and its 'inner' options are the scene elements). CHAIN was developed using JavaScript for the web, C# with .Net Framework for desktop and C# with Xamarin for mobile. When source-code is not available it is able to use the PDC paradigm interacting in different ways depending on where the application is running. For web applications it uses Selenium capabilities and for desktop applications it makes use of an UI automation framework. The help is given by the definition of a model on Cedar Studio [51], which also allows for the importation of Selenium scripts. The focus of CHAIN is providing proper annotations on the target interface, being also able to insert some values-see Figure 5. The authors state (based on their study) that one-shot learning is something appreciated. Help across several applications was something that they did not reach and crowd-sourcing would be an interesting trait.

Table 1 (see in appendix) summarizes and sorts into three categories (automation, automation/help, and help) the stated works. The categorization is based on the goal of the work (i.e. to improve processes

through automation or to help users). Each work was then classified into six factors considered important to describe and understand their purpose. They are: i) domain; ii) interaction core (how interaction with applications is performed); iii) crowdsourced-based; iv) uses picture-driven computing interaction; v) enables demonstration-based learning (learn with user demonstration); vi) enables continuous learning (improves with user interactions).

Those factors emerged from the interconnection between our goals and identified works. From them, one can identify, solutions independent of application's source code, how interaction is performed, check continuous learning viability, identify interface interaction challenges, and if it works on a crowdsourced-based way.

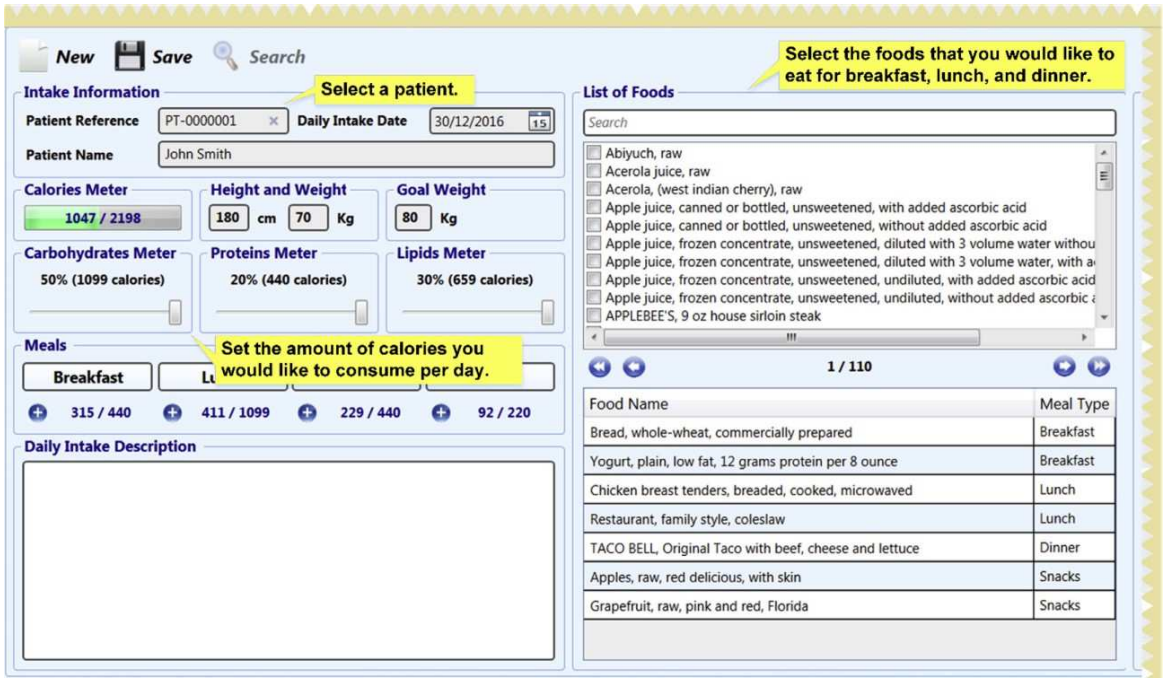

Figure 5. CHAIN, example of an interface with annotations made with CHAIN [10]

4. DISCUSSION

This section describes how the presented work addresses the three contributions presented in the abstract. Considering the identification of recent solutions (1), several works were found and presented. They were categorized and classified into different factors to facilitate their comparison as shown in Table 1 (see in appendix). From the table many differences and similarities between works can be identified but more interesting are the patterns that seem to emerge.

Starting with the main goal(s) of solutions, only one goal tends do be addressed (only two recent works address both automation and help). In relation to domain, no pattern stands out, being all web, plugin dependent and domain independent present with the same frequency. However, in what concerns automation solutions, a prevalence for web stands out. One reason for this might the DOM structure availability making development easier. Another reason might be related to the higher applicability as typical users use to utilize mainly web applications. A similar conclusion can be draw regarding interaction core, i.e. a pattern seems to emerge only for solutions with focus on automation-the use of automation scripts. Crowdsourced is being frequently used in automation but more frequent usage should be reached when the goal of solutions is help. PDC and one-shot learning started to be used in solutions to provide help. Trends were not identified regarding continuous learning.

From the comparison made some problems and possible improvements were identified (2). Some tools present benefits of both source code and domain independence but overcoming all aspects still needs research. PDC is a recent technology that facilitates the overall implementation of extra-application capabilities however,

after the decomposition of an GUI (see the work of Dubrovina *et al.* [52] or Prefab [53] for further details) it is still difficult to properly interact with its elements atomically (e.g. Dixon *et al.* [54] reports that clickable "target" are often ambiguous and cannot be reverse engineered without human intervention). To accomplish full domain independence icons and widgets should be fully identified even without accessing source code. Solutions based on artificial intelligence (AI) might be able to address this challenge.

Being able to work in groups and share knowledge is also a valuable aspect that most tools still leave apart (crowdsourced-help). Continuous learning, through the implementation of learning algorithms, may also be an advantage improving the interaction of the user as the steps that are chosen to automate can be better selected. Additionally, the ability to implement continuous learning, allowing the tools to keep learning from users, is another interesting aspect still missing. The joint between PDC, automation, crowdsourced-help, with the ability to capture tasks from a single demonstration are desired.

The main barrier for faster development (3), is related to domain independency, either from source code and/or extra-plugins. Not requiring the user to install extra-plugins or to restrict the automations/ demonstrations to a set of applications is something that brings many benefits like inter-application help possibilities. To achieve this independency a system that does not depend on applications' code or structure is a must. PDC has the capabilities required however, we believe it is still not being largely used because it has some limitations yet (e.g. element's identification ambiguity). The development of solutions merging PDC with AI will overcome this challenge. For instance, the work of Dubrovina *et al.* [52] presents a step in this direction presenting a solution for GUI object classification using a support vector machine classifier. Solutions like this are essential (a US patent already exists [55]) for image-based software automation/demonstration tools however, this is an area that has still not got the deserved attention.

5. CONCLUSIONS

This paper presents a literature review of recent tools that aim to facilitate the way tasks are accomplished via help and/or automation. The most relevant solutions were identified, described, compared and sorted into categories. The categories that came up are related to the way tools implement help and/or automation. The categories are, Scripts and automated-based, Video-based, GUI elements based.

From the comparison made as shown in Table 1 it can be concluded that both automating tasks to provide help is something being addressed in many recent works and, most works are still restricted to a specific domain. In what concerns continuous-learning it is an aspect that has not received much attention yet. This is true also for crowdsourced concepts allowing people to easily work and share their developments in groups. PDC and one-shot learning are implemented in some tools and show interesting results but are concepts still on a rising phase.

Improving the way people get help and/or making tasks' accomplishment easier are major advantages. This is true for both corporate workers in their daily activities where they must be productive and, for singular computer users that many times feel they cannot execute some tasks (usually attributing wrongly the failure to their lack of skills instead of recognizing software deficiencies). Developing a tool that would consider the implementation of crowdsourced collaboration, interaction through picture-driven computing, that would easily capture each steps of a task and, that could learn with users' executions seems to be an interesting path to follow.

APPENDIX

Table 1. Identified works categorized regarding their main goal and classified into six factors

Main goal	Domain	Interaction core	Crowd sourced-based	uses PDC interaction	enables Demonstration-based Learning	enables Continuous Learning	Name
Automation	Web	Automation scripts	Yes	No		No	CoScripter
	Web	Automation scripts	Yes	No		No	ActionShot
	Web	Automation scripts question-answer analysis	Yes	No		Yes	CoCo
	Plugin dependent	Automation scripts	–	No		No	TaskTracer
Automation /Help	Independent	Learn-replay procedure	–	No	Yes	Yes	Sheepdog
	Independent	Enriched task models and PDC	No	No	No	No	ISI
Help	Web	Dialogues with users	No	No	No	Yes	AIDE
	Web	Scripts-Web' DOMs	Yes	No	Yes	No	Smart Web Tutor
	–	Videos or/and textual help	No	No	No	No	ToolClips
	Plugin dependent	Videos	No	No	No	No	Pause-and-Play
	Plugin dependent	GUI Elements	Yes	No	No	No	LemonAid
	Independent	Videos or/and textual help	No	No	No	No	Ambient Help
	Independent	Enriched task models and PDC	No	Yes	No	No	DemoHelp
	Independent	GUI Elements	No	Yes	Yes	Yes	HILC
	Independent	Interaction models (through Cesar Studio)	No	Yes*	No	No	CHAIN

ACKNOWLEDGEMENTS

This work was supported by Fundação para a Ciência e a Tecnologia (FCT, Portugal), through projects UIDB/50009/2020, UIDB/04466/2020 and UIDP/04466/2020. The authors would like to thank Rúben Pereira for the given support.

REFERENCES

[1] Norman, D. A., "The invisible computer: why good products can fail, the personal computer is so complex, and information appliances are the solution," *MIT press*, 1998.

[2] Lazar J, Jones A, and Shneiderman B, "Workplace user frustration with computers: An exploratory investigation of the causes and severity," *Behaviour and Information Technology*, vol. 25, no. 03, pp. 239-251, 2006.

[3] Rosenbaum S, Ramey J, and Redish JG., "Current issues in assessing and improving information usability," *In CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pp. 3155-3158, 2010.

[4] Winter J, Rönkkö K, Rissanen M., "Identifying organizational barriers-A case study of usability work when developing software in the automation industry," *Journal of Systems and Software*, vol. 88, pp. 54-73, 2014.

[5] Dworman G, Rosenbaum S., "Helping users to use help: improving interaction with help systems," *CHI'04 extended abstracts on Human factors in computing systems*,pp. 1717-1718, 2004.

[6] Dworman G, "Arbitration of a help system," *Interactions*, vol. 14, no. 1, pp. 39-42, 2007.

[7] Martinie C, Palanque P, Navarre D, Winckler M, Poupart E., "Model-based training: an approach supporting operability of critical interactive systems, *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*, pp. 53-62, 2011.

[8] Navarre D, Palanque P, Ladry JF, Barboni E., "ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 16, no. 4, 2009.
[9] Machado V, Lopes N, Silva JC, Silva JL., "Picture-Based Task Definition and Parameterization Support System," *World Conference on Information Systems and Technologies*, pp. 592-601, 2017.
[10] Akiki PA., "CHAIN: Developing model-driven contextual help for adaptive user interfaces," *Journal of Systems and Software*, vol.135, pp. 165-190, 2018.
[11] Vouligny L, Robert JM., "Online help system design based on the situated action theory," *Proceedings of the 2005 Latin American conference on Human-computer interaction*, pp. 64-75, 2005.
[12] Silva JL, Ornelas JD, Silva JC., "Make it ISI: interactive systems integration tool, "Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, pp. 245-250, 2016.
[13] Grossman T, Fitzmaurice G., "ToolClips: an investigation of contextual video assistance for functionality understanding," *Proc., of the SIGCHI Conf., on Human Factors in Computing Systems*, pp. 1515-1524, 2010.
[14] Pongnumkul S, Dontcheva M, Li W, Wang J, Bourdev L, Avidan S, Cohen MF., "Pause-and-play: automatically linking screencast video tutorials with applications," *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp. 135-144, 2011.
[15] Yeh T, Chang TH, Xie B, Walsh G, Watkins I, Wongsuphasawat K, Huang M, Davis LS, Bederson BB., "Creating contextual help for GUIs using screenshots," *Proceedings of the 24th annual ACM symposium on User interface software and technology.*, pp. 145-154, 2011.
[16] Yeh T, Chang TH, Miller RC., "Sikuli: using GUI screenshots for search and automation," *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 183-192, 2009.
[17] Chilana PK, Ko AJ, Wobbrock JO, Grossman T., "A multi-site field study of crowdsourced contextual help: usage and perspectives of end users and software teams," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 217-226, 2013.
[18] Sun Y, QiaO Z, Chen D, Xin C, Jiao W., "An Approach to Using Existing Online Education Tools to Support Practical Education on MOOCs," *IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1, pp. 696-705, 2016.
[19] Vogel-Heuser B, Diedrich C, Fay A, Jeschke S, Kowalewski S, Wollschlaeger M, Göhner P., "Challenges for software engineering in automation," *Journal of Software Engineering and Applications*, vol. 7, no. 5, 2014.
[20] Lau T, Bergman L, Castelli V, Oblinger D., "Sheepdog: learning procedures for technical support," *Proceedings of the 9th international conference on Intelligent user interfaces*, pp. 109-116, 2004.
[21] Kourousias G, Bonfiglio S., "Picture-Driven Computing In Assistive Technology And Accessibility Design," *1st International ÆGIS Conference*, vol. 7, 2010.
[22] Schatsky D, Muraskin C, Iyengar K, "Robotic process automation: A path to the cognitive enterprise," [ONLINE], Avaible: https://www2.deloitte.com/insights/us/en/focus/signals-for-strategists/cognitive-enterprise-robotic-process-automation.html.
[23] [ONLINE], Avaible:https://www.nice.com.
[24] [ONLINE], Avaible:https://www.uipath.com.
[25] [ONLINE], Avaible:https://www.blueprism.com.
[26] [ONLINE], Avaible:https://www.automationanywhere.com.
[27] Utermohlen K., "All the Robotic Process Automation (RPA) Stats You Need to Know," [ONLINE], Avaible:https://towardsdatascience.com/all-the-robotic-process-automation-rpa-stats-you-need-to-know-bcec22eaaad9.
[28] Shi, JJ, Lee DE, Kuruku E., "Task-based modeling method for construction business process modeling and automation," *Automation in Construction*, vol. 17, no. 5, pp. 633-640, 2008.
[29] Stohr EA, Zhao JL., "Workflow automation: Overview and research issues," *Information Systems Frontiers*,
vol. 3, no. 3, pp. 281-296, 2001.
[30] Martinie C, et al, "Task-model based assessment of automation levels: application to space ground segments," *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 3267-3273, 2011.
[31] Boy GA., "Cognitive function analysis for human-centered automation of safety-critical systems," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 265-272, 1998.

[32] Amershi S, Mahmud J, Nichols J, Lau T, Ruiz GA., "LiveAction: Automating web task model generation," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 3, no. 3, pp. 1-23, 2013.
[33] Instructional Systems Development (ISD)., Air Univ, [ONLINE], Avaible: http://www.au.af.mil/au/awc/awcgate/doe/isd/paper.htm.
[34] Murata T., "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541-580, 1989.
[35] Paternò F, Mancini C, Meniconi S., "ConcurTaskTrees: A diagrammatic notation for specifying task models," *Human-Computer Interaction INTERACT'97*, pp. 362-369, 1997.
[36] Bolognesi T, Brinksma E., "Introduction to the ISO specification language LOTOS," *Computer Networks and ISDN systems*, vol. 14, no. 1, pp.25-59, 1987.
[37] iCS, HAMSTERS, Inst. Rechearch En Inform. Toulouse. [ONLINE], Avaible: https://www.irit.fr/recherches/ICS/softwares/hamsters/index.html.
[38] Card SK., "The psychology of human-computer interaction," *CRC Press*, 2017.
[39] Paternò F., "ConcurTaskTrees: an engineered notation for task models," *The handbook of task analysis for human-computer interaction*, pp. 483-503, 2004.
[40] Grossman, T., and Fitzmaurice, G., "ToolClips: An investigation of contextual video assistance for functionality understanding," *Proc., of the SIGCHI Con., on Human Factors in Computing Systems*, pp. 1515-1524, 2010.
[41] Leshed G, Haber EM, Matthews T, Lau T., "CoScripter: Automating and sharing how-to knowledge in the enterprise," *Proc., of the SIGCHI Conf. on Human Factors in Computing Sys.*, pp. 1719-1728, 2008.
[42] Li I, Nichols J, Lau T, Drews C, Cypher A., "Here's what i did: sharing and reusing web activity with ActionShot," *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pp. 723-732, 2010.
[43] Lau T, Cerruti J, Manzato G, Bengualid M, Bigham JP, Nichols J., "A conversational interface to web automation," *Proc., of the 23nd annual ACM symposium on User interface software and technology*, pp. 229-238, 2010.
[44] Dragunov AN, Dietterich TG, Johnsrude K, McLaughlin M, Li L, Herlocker JL., "TaskTracer: A desktop environment to support multi-tasking knowledge workers," *Proceedings of the 10th international conference on Intelligent user interfaces.*, pp. 75-82, 2005.
[45] Matejka J, Grossman T, Fitzmaurice G., "Ambient help," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2751-2760, 2011.
[46] Chilana PK, Ko AJ, Wobbrock JO., "LemonAid: selection-based crowdsourced contextual help for web applications," *Proc., of the SIGCHI Conf. on Human Factors in Computing Sys*, pp. 1549-1558, 2012.
[47] Ornelas JD, Silva JC, Silva JL., "Demonstration-based Help for Interactive Systems," *Proceedings of the 2nd International Conference in HCI and UX Indonesia*, pp. 125-128, 2016.
[48] Rodrigues P, Silva JL, Pereira R., "DEMONSTRATION-BASED HELP: A CASE STUDY," *10th annual International Conference on Education and New Learning Technologies*, pp. 4367-4376, 2018.
[49] [ONLINE], Avaible:http://hiis.isti.cnr.it:4500/home.
[50] Intharah T, Turmukhambetov D, Brostow GJ., "Help, It Looks Confusing: GUI Task Automation Through Demonstration and Follow-up Questions," *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, pp. 233-243, 2017.
[51] Akiki PA, Bandara AK, Yu Y., "Cedar studio: an IDE supporting adaptive model-driven user interfaces for enterprise applications," *Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems*, pp. 139-144, 2013.
[52] Dubrovina, A., Kisilev, P., Freedman, D., Schein, S., and Bergman, R., "Efficient and robust image descriptor for GUI object classification," *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 3594-3597, 2012.
[53] Dixon M, Nied A, Fogarty J., "Prefab layers and prefab annotations: extensible pixel-based interpretation of graphical interfaces," *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pp. 221-230, 2014.
[54] Dixon M, Fogarty J, Wobbrock J, "A general-purpose target-aware pointing enhancement using pixel-level analysis of graphical interfaces," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 3167-3176, 2012.
[55] Freedman, D., Kisilev, P., Dubrovina, A., Schein, S., and Bergman, R., "Graphical object classification," *U.S. Patent No. 9,213,463. Washington, DC: U.S. Patent and Trademark Office*, 2015.