WOK – User Interaction in a Crowdsourcing Application

Lourenço Francisco Costa Palma
lou.pal.1995@gmail.com

Instituto Superior Técnico, Universidade de Lisboa, Portugal

November 2019

Abstract

The multidisciplinary project IST SCOPE WOK, made in partnership with Unbabel, consists of a Crowdsourcing application for the Facebook Messenger Platform based on the Wisdom of the Crowd concept. Users perform tasks supplied by a task provider bot represented in the form of Co-Workers of the user. Co-Workers have different personalities and interact with the users providing them tasks and incentivizing them to perform said tasks, with the goal of increasing user productivity. Tasks can have two types: Quality Estimation or Categorization. In Quality Estimation tasks the users must evaluate the correctness of a short text with a yes or no answer. In Categorization tasks, the users must assign a category and sub-category to a short text. Tasks are performed in *Webviews*, embed browser engines inside the Facebook Messenger Platform.

Keywords

Bot, Client, Co-Worker, Crowdsourcing, Task, User, Webview

1 Introduction

This project followed a Capstone Model [1] approach, a diverse project that culminates in a rich academic and intellectual experience for the students involved.

The SCOPE project by Instituto Superior Técnico (IST) [2], having the Capstone Model as a base, attempts to introduce an innovative model based on interdisciplinary collaboration among different engineering fields at IST, enabling dissertation and project students to collaborate in small teams to solve real problems set by companies, while also being integrated in those companies environment.

1.1 Motivation

This project and its concept were proposed by *Unbabel* [3], a company that provides AI-powered human translation software. The concept consists of a Crowdsourcing [4][5] task solving service where human users would perform several tasks submitted by clients of this service; once the tasks are completed, the clients would obtain the completed tasks. The idea originates from the Wisdom of Crowd [6][7] concept, that serves as

the baseline for this project, hence the name of the project is *WOK*.

1.2 Team

This project is performed by a team of two Information Systems and Computer Engineering students (myself included) from IST, a Physics Engineering student from IST and a Communication Design and New Media student from Faculdade de Belas Artes da Universidade de Lisboa (FBAUL) [8].

1.3 Goal

The goal of this project is the development of a Crowdsourcing task solving service, emphasizing the user interactivity aspect of it.

2 Related Work

The main concept for this project is Crowdsourcing [4][5][43], a sourcing model where individuals or organizations use the contributions of large, open and rapidly-evolving groups of internet users to obtain services or goods. Crowdsourcing can be divided into four different strategies [44]: Crowdfunding, Crowdcreation, Crowdvoting, and, more notably, the strategy Crowd Wisdom, popularized as Wisdom of the Crowd, used in this project for the development of the task provider service, which states that groups of people collectively are smarter than an individual expert in problem-solving, decision-making and predicting.

2.1 Micro Tasks

Another important concept closely related to Crowdsourcing is Micro Tasks or Micro Work [42], consisting of spitting larger and more complex tasks into smaller tasks that can be performed within a matter of seconds or minutes.

When using Micro Tasks in a Crowdsourcing application, the application benefits from the user performing simple tasks or actions, while the user is usually rewarded by doing so.

An example of a Crowdsourcing application is *Mechanical Turk* [13], Amazon's take on the Crowdsourcing market. Mechanical Turk serves as an intermediary between requesters that have tasks they need to be completed and workers who want to earn money by performing tasks. The task availability depends on the requesters and can be quite diverse, ranging from transcriptions to surveys, commenting on images or writing product reviews, etc.

2.2 Chatbots

Most Crowdsourcing applications utilize bots to interact with the users. Bots or Internet Robots [45] are software applications that run automated tasks (scripts) over the internet.

Chatbots [46] communicate with the users using auditory or textual messages and may handle many tasks such as reporting the weather, sports scores or converting currency.

Chatbots offer different forms of user interaction that take different complexity levels [47]. For example Contextual Chatbots have a higher complexity level by using Machine Learning and Artificial Intelligence to self-improve, while Menu or Button-Based Chatbots, used in this project as a way for the task provider service to supply tasks, are less complex and use decision tree hierarchies for the interaction, presented to the user in the form of menus and buttons.

3 Solution

The solution developed is named WOK – Wisdom of the Krowd, a Crowdsourcing application where users perform tasks and get rewarded by doing so. The base concept of WOK is Wisdom Of the Crowd [7][8], hence the name WOK (with *K* instead of *C*), which states that groups of people collectively are smarter than an individual expert in problem-solving, decision-making and predicting. If we apply this concept in the context of WOK, the mode of the users' answers for a certain task will most likely be the correct answer.

The user interaction is performed via *Facebook* [14] and the *Messenger* [15] platform. Tasks are supplied to the users by a task provider bot (Menu and Button-based Chatbot), that takes the form of Co-Workers of the user.

There are two types of tasks the users can perform on WOK, categorization tasks and quality estimation tasks, see figure 1. In the categorization tasks, the user has to assign a category and sub-category to a short text. In the case of the quality estimation tasks, the user will have to determine the correctness of a short text with yes or no answer. The task assigned to the user depends on the availability of the task in the system.

3.1 User-research and Design

An innovation of WOK is the introduction of Co-Workers with different personality traits that provide tasks to users.

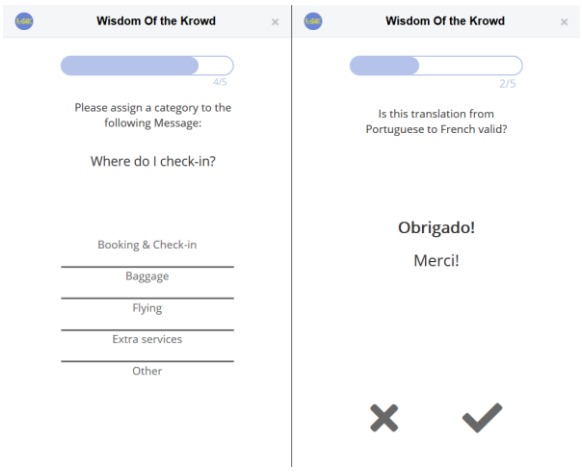

Figure 1: examples of a categorization task (left) and a quality estimation task (right).

User research was performed to identify the target demographic for WOK; from this research was concluded that a younger composed a majority of the user base of Crowdsourcing [4][5] applications; we can assume this will be the user base of WOK. More user research was made that helped define personas to represent the users of WOK and from these user personas Co-Workers, were created.

The task provider *bot* also offers five Co-Workers for the user to choose from. It was created a persona for each of these five Co-Workers. Each of these personas has a name, a short description and personality traits based on the Big Five Personality Theory [17][18], varying from openness to neuroticism in order to match and complement the personas of the users. The goal of the Co-Workers is to incentivize the user to perform more tasks, thus increasing user productivity and helping the user create a bond with WOK; this can be accomplished by interacting with the user via text where each Co-Worker has a different speech dependent of its persona.

The available Co-workers are the following: Gal (Openness), Julian (Agreeableness), Eli (Con-

scientiousness), Jesse (Extraversion) and Sam (Neuroticism).

For the Co-Workers avatars, geometric shapes and color contrast were used to represent the persona of the Co-Worker instead of human faces.

Only one Co-Worker may be active at a time, with Gal being the default.

3.2 Architecture

Regarding the architecture, WOK follows a microservices architecture [19], as shown in the diagram of figure 2, with three services: Facebook Handler, Server, and Client-side Handler.

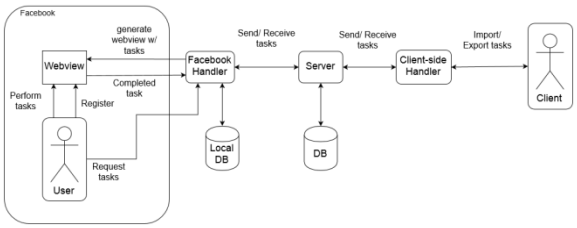

Figure 2: diagram representing WOK's architecture.

The Facebook Handler service manages the interaction with the users on the *Facebook Messenger* platform [20]. This interaction is performed by the task provider *bot* in the *Messenger* platform itself or by custom web pages, inside *Webviews* [21], where the tasks are presented to the users. This service accesses a local database where *Facebook* users' information is stored.

The Server service acts as the central point of the architecture, managing the requests from the user-interface (Facebook Handler) and the client-interface (Client-side Handler). This service accesses a local database containing all the information regarding the tasks.

The Client-side Handler service generates a website where the clients submit new tasks to WOK and export the completed tasks, while also having access to the information of each task.

3.3 Technologies and Frameworks

Regarding the technologies and frameworks utilized in WOK, the task provider bot uses the Messaging framework [24] by *Facebook for Developers* [16] and was implemented using the Python [25] programming language. The Python language was also used to implement the Facebook Handler, Server, and Client-side Handler services. The Flask module framework [26] and the Flask-RESTful API [27] were used for communication among the services.

Both the Facebook Handler local database (where the user information is stored) and the Server database (where the tasks information is stored) consist of documental databases and are represented using the JSON [22] format.

The tasks are presented to the human user in a *Webview*, using the *Webview* framework [21] by *Facebook for Developers*. Several APIs by *Facebook for Developers* were also utilized to improve the user experience, such as the Persona API [29], Send API [30] and Messenger Profile API [31].

All the services are hosted in Heroku [32] and both databases are hosted in MongoDB mLab [33]. The task provider *bot* is hosted on the *Facebook for Developers* platform.

3.4 User-interface and Interaction

The business model of the user-interface of WOK is rather simple, with the user communicating with the Co-Worker (task provider *bot*) inside the *Facebook* platform [14] via *Facebook Chat* or the *Messenger App* [15].

The task provider *bot* then communicates with the Facebook Handler Service, serving as an intermediary between the service and the user. The communication with the Co-Worker is performed via buttons. The Co-Worker replies to the user with a text message containing buttons the user can press to trigger a text response of the Co-Worker that may contain subsequent buttons.

3.4.1 Messenger Profile API – Persistent Menu

The *Messenger Profile API* [31] is used to set properties that define several aspects of the *Messenger Platform* [14][15] and features, such as the Persistent Menu [34], used in WOK.

One of the challenges of using a task provider *bot* in the *Facebook Messenger* platform was that the *bot* can't explicitly initiate a conversation with the user. The user must first interact with the *bot* for it to respond. So a scenario where the user overloads the *bot* by spamming messages would be a possibility. Therefore, we felt a need to control the interaction between the user and the bot. A solution found was the usage of a Persistent Menu.

A Persistent Menu is a pop-up menu with buttons through which the user interacts with the task provider *bot*. The most important feature provided by the Persistent Menu is the option to disable the *Message composer*, with the property *composer_input_disabled*, forcing the user to interact with the *bot* using only buttons (either the ones on the Persistent Menu or the ones sent by the Co-Worker).

3.4.2 Webview

Webviews [21] are a feature of the Messenger Platform [14][15] where web pages can be loaded inside the Messenger.

A challenge faced was how to represent tasks and where the user would perform said tasks. Since it would be difficult to represent tasks with message bubbles or buttons in the text chat, *Webviews* were chosen to help with this representation.

Webviews consist of embed browser engines where pages are loaded inside the *Facebook Messenger Platform*. The tasks themselves are presented to the user as web pages inside the W*ebview*, as well as the register page, settings page, and the tasks completion page. The usage of *Webviews* allows more flexibility and freedom in the tasks' representation.

The web pages available to the user are the following:

Categorization Task – where the user performs categorization tasks.

Quality Estimation Task – where the user performs quality estimation tasks.

Tasks Completion – is a web page that congratulates the user on completing a group of tasks with a *GIF* [35] file.

Settings – where the user can change the active Co-Worker as well as the nickname and notification options.

Registration – where the user must first register to start using WOK.

The web pages where the tasks are presented to the user, as well as the register, settings, and task completion pages don't consist of only plain HTML [28], other libraries, plug-ins, and frameworks were used.

Regarding the Categorization and Quality Estimation tasks web pages particularly, the generation of these web pages is dynamic using two templates, one for each task type. The templating engine Jinja2 [36] was used, allowing the writing of code with a syntax similar to Python [25] with the usage of special placeholders in the template.

CSS [37] was used to change and improve the visual aspect of the web pages. The JavaScript [38] development environment was used on all web pages to obtain the *recipient_id* of the user so that the Facebook Handler Service identifies the user.

JQuery [39], a JavaScript library, was also used on all web pages for obtaining the *recipient_id*. It was also used both the quality estimation and categorization web pages to create animations, as well as a progression bar that tracks the group of tasks progression.

The framework Bootstrap [40] was used on all the web pages to improve the overall visual aspect, making the web pages more appealing to the users.

Slick [41], a Javascript plug-in, was used to create a dynamic carousel where the user can select a Co-Worker.

3.4.3 Persona API

The Persona API [29] allows the creation of Personas [9] in a *bot* that carries messaging conversations with end-users in the Messenger Platform [14][15]. It is still the *bot* that sends the message in the platform API, but with the usage of this API, there is the option to send the message under the name of a Persona instead of the bot.

With the usage of this API, it was possible to very easily integrate each of the Co-Workers as Personas in the Messenger Platform.

For each Persona created, a new icon will be shown with the Co-Worker avatar and all messages sent by the Persona will be accompanied by an annotation above the message that states the name of the Persona and the name of the page the Persona belongs to.

3.4.4 Send API

The Send API [30] is the main API used by WOK to send text messages to users along with buttons and sender actions.

Sender actions [10] are used to simulate human interaction, either by marking messages sent by the user as seen or by simulating the Co-Worker is typing a message instead of replying instantly, thus making the task provider bot feel more responsive. With the usage of sender actions, a delay time is defined to simulate the Co-Worker typing.

The Send API is also responsible for the buttons [11] sent in the Co-Workers' replies. These buttons are sent inside the message as an attachment.

3.5 Client Interaction

The Client-side handler service is responsible for the interaction with the client. Using this service, the clients can submit new tasks to be performed by the users on the *Facebook* platform [14][15] and obtain the results of the completed tasks.

The interface for the task submission and retrieval consists of web pages and the Client-interface handler service is a server that manages these web pages. There are two web pages available: the task retrieval web page and the task submission web page. These web pages consist of HTML [28] pages that use JavaScript [38], CSS [37], Bootstrap [40] and JQuery [39].

3.5.1 Task Retrieval

The task retrieval page allows the user to visualize and filter task data by three different categories: time, description and response, as shown in figure 3. The tasks and these categories are displayed in the form of tables, generated from a template using the templating engine Jinja2 [36].

The task time category displays information of the task date and time of submission, average response time and date and time of completion. The task description category displays the task description and word count. And lastly, the task response category showcases information regarding the users' responses and completion such as the number of responses for the task, if the task is completed and the most chosen answer for the task.

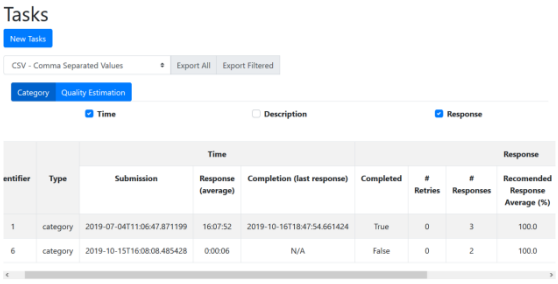

Figure 3: task retrieval web page.

Some task data varies depending on the type of the task. For quality estimation tasks, the response category will contain a true or false Boolean as the answer. For categorization tasks, the preferred category and sub-category will be the answer. For this reason, each type of task has its table where the tasks of that type are presented and the client can alternate between the tables.

In the task retrieval page, the client can obtain the information of the tasks presented in the tables using the export option. By using the "Export All" and "Export Filtered" buttons the client obtains a file with the data of the currently selected task category table in the desired format. The available formats are CSV [23], TSV [23] and JSON [22].

The "Export All" and "Export Filtered" actions are performed in runtime on the client browser by using JavaScipt functions. The task data, presented to the client in the web page tables, is converted to the desired format for the client to download. The conversion process utilizes HTML DOM [12] to create the downloadable file, as well as the templating engine Jinja2 [36] to cycle through the tasks.

3.5.2 Task Submission

The task submission page allows the client to submit new tasks to WOK by uploading a file with the tasks. On this page, the client can select the type of tasks to submit: Categorization or Quality Estimation; and the format of the tasks file: JSON, CSV, and TSV.

A Quality Estimation task is characterized by the task description, a source message, and a target message.

A Categorization task is characterized by a task message, a list of categories and, for each category of the list, an associated list of subcategories.

It is worth noting the task submission page allows Quality Estimation tasks to be uploaded in JSON, CSV, and TSV format, but for Categorization, only the JSON format is accepted. This limitation was imposed due to the fact that the number of categories and subcategories for this type of task can be highly variable. So by using CSV and TSV formats to represent this type of tasks due to this high variance and each task being represented in one line, it would be difficult to determine the number of categories and subcategories for each task.

4 Conclusion

The project developed, WOK, successfully integrates the concept of Wisdom Of the Crowd [6][7] in a Crowdsourcing service integrated into the *Facebook* platform [14][15]. Users can register to WOK and perform categorization and quality estimation tasks. Clients can submit new tasks to WOK and retrieve the completed tasks.

During the development of WOK, two user testing sessions were performed. However, due to the small sample size of five people during each session, the feedback obtained may not enough to fully determine what improvements need to be made to WOK. To fully grasp the public perception of WOK and what changes need to be performed to appeal to a larger audience larger-scale testing sessions are needed, along with alpha and beta testing.

The next step on the development of WOK consists on the implementation of several features and functionalities missing in the current version. These features consist of the user payment options by integrating payment platforms such as *Paypal* or *MbWay* with WOK, so that the coins the users receive can be converted into real currency. Another feature consists of the client-interface authentication so that the client could obtain and access the tasks he published to WOK after inserting his authentication credentials in task retrieval and submission web pages.

References

1. Capstone Model definition, https://www.edglossary.org/capstone-project/, last accessed 2019/10/27.
2. Instituto Superior Técnico Homepage, https://tecnico.ulisboa.pt/en/, last accessed 2019/10/27.
3. Unbabel Homepage, https://unbabel.com/, last accessed 2019/10/27.
4. Schenk, Eric; Guittard, Claude: Crowdsourcing What can be Outsourced to the Crowd and Why, https://www.researchgate.net/publication/40270166_Crowdsourcing_What_can_be_Outsourced_to_the_Crowd_and_Why.
5. Estellés-Arolas, Enrique; González-Ladrón-de-Guevara, Fernando: Towards an Integrated Crowdsourcing Definition, http://www.crowdsourcing-blog.org/wp-content/uploads/2012/02/Towards-an-integrated-crowdsourcing-definition-Estell%C3%A9s-Gonz%C3%A1lez.pdf.
6. Soll, Jack; Mannes, Albert; Larrick, Richard: The "Wisdom of Crowd" Effect, https://faculty.fuqua.duke.edu/~jsoll/Soll,%20Mannes,%20Larrick%202011.pdf.
7. On the Wisdom of Crowds: Collective Predictive Analytics, https://towardsdatascience.com/on-the-wisdom-of-crowds-collective-predictive-analytics-302b7ca1c513, last accessed 2019/10/27.
8. Belas Artes Lisboa Homepage, https://www.belasartes.ulisboa.pt/, last accessed 2019/10/27.
9. Facebook for Developers Personas, https://developers.facebook.com/docs/messenger-platform/send-messages/personas/, last accessed 2019/10/27.
10. Facebook for Developers Sender Actions, https://developers.facebook.com/docs/messenger-platform/send-messages/sender-actions/, last accessed 2019/10/27.
11. Facebook for Developers Buttons, https://developers.facebook.com/docs/messenger-platform/send-messages/buttons, last accessed 2019/10/27.
12. DOM documentation, https://dom.spec.whatwg.org/, last accessed 2019/10/27.
13. Mechanical Turk Homepage, https://www.mturk.com/, last accessed 2019/10/27.
14. Facebook Homepage, https://www.facebook.com/, last accessed 2019/10/27.
15. Messenger Homepage, https://www.messenger.com/, last accessed 2019/10/27.
16. Facebook for Developers Homepage, https://developers.facebook.com/, last accessed 2019/10/27.
17. Rothmann S, Coetzer EP: The big five personality dimensions and job performance, https://sajip.co.za/index.php/sajip/article/view/88.
18. Poropat AE: A meta-analysis of the five-factor model of personality and academic performance, https://www.ncbi.nlm.nih.gov/pubmed/19254083.
19. Microservice Architecture, https://microservices.io/patterns/microservices.html, last accessed 2019/10/27.
20. Facebook for Developers Messenger Platform page, https://developers.facebook.com/docs/messenger-platform/, last accessed 2019/10/27.
21. Facebook for Developers Webview page, https://developers.facebook.com/docs/messenger-platform/webview/, last accessed 2019/10/27.
22. JSON Homepage, https://www.json.org/, last accessed 2019/10/27.
23. CSV and TSV formats, https://help.gnome.org/users/gnumeric/stable/gnumeric.html#file-format-csv, last accessed 2019/10/27.
24. Facebook for Developers Messaging framework page, https://developers.facebook.com/docs/messenger-platform/send-messages/, last accessed 2019/10/27.

25. Python Homepage, https://www.python.org/, last accessed 2019/10/27.
26. Flask documentation page, https://flask.palletsprojects.com/en/1.1.x/, last accessed 2019/10/27.
27. Flask-RESTful documentation page, https://flask.palletsprojects.com/en/1.1.x/, last accessed 2019/10/27.
28. HTML W3C page, https://www.w3.org/html/, last accessed 2019/10/27.
29. Facebook for Developers Persona API page, https://developers.facebook.com/docs/messenger-platform/reference/personas-api/, last accessed 2019/10/27.
30. Facebook for Developers Send API page, https://developers.facebook.com/docs/messenger-platform/reference/send-api/, last accessed 2019/10/27.
31. Facebook for Developers Messenger Profile API page, https://developers.facebook.com/docs/messenger-platform/reference/messenger-profile-api/, last accessed 2019/10/27.
32. Heroku Homepage, https://www.heroku.com/, last accessed 2019/10/27.
33. mLab Homepage, https://mlab.com/, last accessed 2019/10/27.
34. Facebook for Developers Persistent Menu page, https://developers.facebook.com/docs/messenger-platform/reference/messenger-profile-api/persistent-menu, last accessed 2019/10/27.
35. GIF W3C documentation page, https://www.w3.org/Graphics/GIF/spec-gif87.txt, last accessed 2019/10/27.
36. Jinja2 documentation page, https://jinja.palletsprojects.com/en/2.10.x/, last accessed 2019/10/27.
37. CSS W3C page, https://www.w3.org/Style/CSS/, last accessed 2019/10/27.
38. JavaScript Homepage, https://www.javascript.com/, last accessed 2019/10/27.
39. JQuery Homepage, https://jquery.com/, last accessed 2019/10/27.
40. Bootstrap Homepage, https://getbootstrap.com/, last accessed 2019/10/27.
41. Slick Homepage, https://kenwheeler.github.io/slick/, last accessed 2019/10/27.
42. Microtasking definition, https://www.clickworker.com/crowdsourcing-glossary/microtasking-microjobs/, last accessed 2019/10/27.
43. Howe, Jeff: Why the Power of the Crowd is Driving the Future of Business, https://dl.acm.org/citation.cfm?id=1481457.
44. Crowdsourcing strategies, https://www.tricider.com/Crowdsourcing-Strategies, last accessed 2019/10/27.
45. Bot definition, https://techterms.com/definition/bot, last accessed 2019/10/27.
46. Chatbot definition, https://searchdomino.techtarget.com/definition/IM-bot, last accessed 2019/10/27.
47. Chatbot interaction, https://chatbotsmagazine.com/the-3-types-of-chatbots-how-to-determine-the-right-one-for-your-needs-a4df8c69ec4c, last accessed 2019/10/27.