POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Designing the Integration of Conversational AI with Web Architectures

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Gianluca Spadone

Advisor: Prof.ssa Maristella Matera

Co-advisors: Marcos Baez, Emanuele Pucci

Academic year: 2020-2021

1. Introduction

Everyone should be able to access the Web in the same way, with no exceptions. As a result, different types of users (blind or visually impaired people, individuals who are unable to use their hands or eyes in a certain situation such as while cooking or driving, elderly people) should be able to access information as they were using a mouse or a keyboard. Now that natural language interaction for the Web has become a reality [1], the notion of Conversational Web Browsing emerges to increase Web accessibility via voice commands.

A recent study [3] has investigated the existing challenges for reaching the notion of Conversational Web, with a focus on visually impaired people. Capitalizing on the results of this work, this thesis has designed the *Conversational Web Framework* (ConWeb), a software platform to assist users in gaining access to Web information with a new conversational paradigm. The thesis in particular investigates the different challenges that this new paradigm poses [2], and aims to establish the groundwork for a new way of accessing information, which is focused on accessibility, as also demonstrated by an evaluation activity that we conducted to test the validity of the resulting interaction paradigm for blind and visually impaired individuals.

2. State of the Art

Several efforts have been devoted in the last years to granting universal access to the information on the Web. The World Wide Web Consortium (W3C), in particular, started projects for the Web that promoted a high degree of usability for people with disabilities. The W3C Web Accessibility Initiative (WAI) has defined technical specifications, guidelines, strategies, and supporting materials to standardize Web development under principles that aim to make websites accessible. However, Web developers are either unaware of these accessibility principles or ignore them given the efforts they require. As a result, Web accessibility is not always fulfilled and users may encounter a range of issues while browsing.

2.1. Accessibility approaches

Web accessibility is a severe issue. Screen readers are commonly-adopted assistive tools, but they are incapable of filtering content in a practical manner, offering adequate navigation control, or enabling a quick identification of the

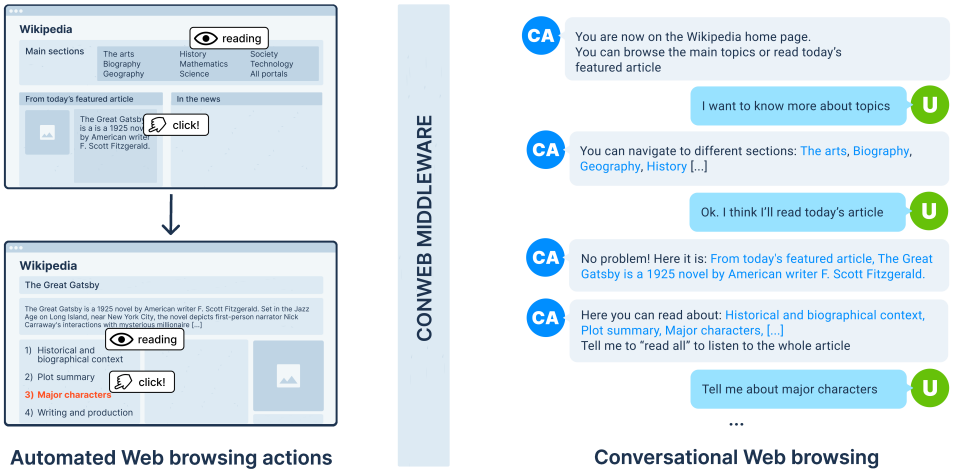

Figure 1: ConWeb scenario

available content that may be gained when visually inspecting a Web page (even involuntarily). Recently proposed voice assistants, such as Google Assistant, Amazon Alexa, and Apple's Siri, can also be used to obtain Web content. However, these devices can only respond to simple information queries or functions (such as setting alarms, timers, or asking the weather); they cannot assist actual browsing through Web pages. For the motivations described thus far, there is a need for new technologies and new paradigms to make the Web truly accessible.

2.2. Non-visual browsing

New, disruptive paradigms can exploit the concept of *non-visual browsing*. It is a way of browsing websites without relying on a graphical user interface (GUI), which aims to assist users in understanding the content through various strategies such as `segmentation` and `summarization`. The former seeks to extrapolate content from the Web into separate pieces of information, to allow the final user to understand the general content of a page through a mental view, whilst the latter aims to summarize content to understand its meaning without reading it in its entirety.
Non-visual browsing is an important advancement that we have explored to provide access to Web content in a way that is detached from the visual layout.

2.3. Web Exploration methods

In the last years, some research works have focused on Web automation tools to retrieve content from a Web page using various commands, also exposed in natural language; an example of these tools is Firefox Voice [4], which retrieves Web content upon requests in natural language. To understand natural language commands, a natural language understanding (NLU) engine is needed. Natural language processing (NLP) is a class of machine learning methods that analyze unstructured text and convert it to structured data. Natural language understanding, then, is a subset of NLP that classifies the intents, or meaning, of text based on the message's context and content.
Rasa Open Source[1] is the NLU library that is used in ConWeb to interpret users' requests to extract intents (goals of the user) and entities (extracted words relevant in a selected context).

3. ConWeb - approach

To solve some accessibility challenges in this thesis we followed the non-visual browsing paradigm, by providing a framework that could perform actions on websites (such as Web automation tools) in response to user requests exposed via voice and processed by an NLU (in our case Rasa).
Consider the following scenario: a person with visual impairments wants to browse information on Wikipedia. Our approach aims to allow her to explore a Wikipedia page while conversing with a Conversational Agent (CA, e.g., a smart speaker, a Web-client or a voice-based browser plugin). As illustrated in Figure 1, starting from

[1] https://rasa.com/open-source/

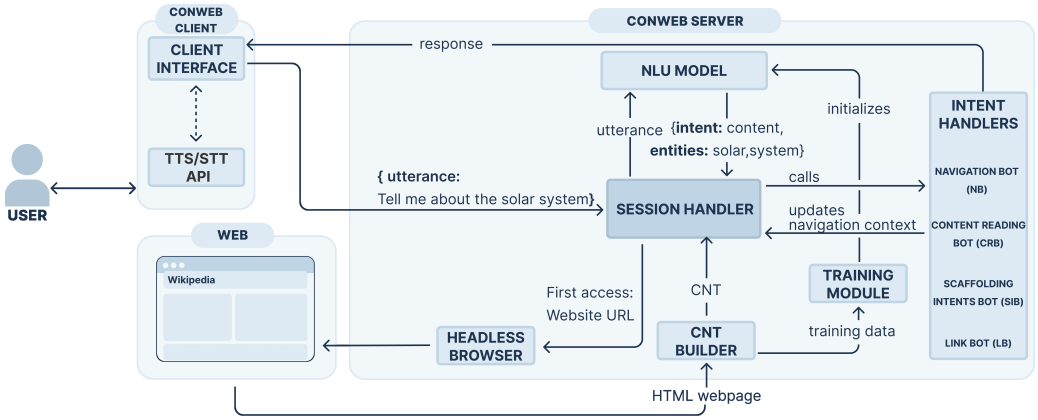

Figure 2: ConWeb architecture

the home page, the user can be introduced with a brief summary of the website's basic structure. The user might also get oriented at any time by querying about the content accessible in a certain context by saying "What can I find on this page?". The user can then navigate the website by selecting one of the available options (for example, "I want to go to [...]"). These queries can cause the website to navigate inside or across pages (e.g., from the Home to an article page).
To enable such interaction, as shown in Figure 1, ConWeb acts as a middleware between the CA and the website, being able to recognize the offerings and content of the website that could be accessed through it, interpret user requests, and automatically take corresponding actions (e.g., click, extract information).
To achieve these goals, we must deal with three major aspects: retrieving and handling data from the Web, interpreting user requests, and performing appropriate actions. The first aspect could be managed using a Conversation-oriented Navigation Tree (CNT), which is a data structure that represents the content of a Web page along various nodes. The second element could be solved using an NLU (such as Rasa), while the last one is addressed with Intent Handlers (specific components responsible of performing the proper action on the interpretation of user requests). Figure 2 illustrates the core architectural components that cover the three aspects described above. It includes two major modules: *ConWeb client* and *ConWeb server*, which are described in the following.

3.1. ConWeb Client

The ConWeb client component is the front-end exposed to the final user, who can utilize it by speech (via a Web browser such as Google Chrome or Firefox). We can translate a user's spoken request into text and send it to the ConWeb Server using Speech Recognition/Synthesis APIs (`Google APIs` are used given their accuracy in speech recognition and synthesis). Using the same APIs, we parse the ConWeb Server answer and present it to the user by voice.

3.2. ConWeb Server

The ConWeb Server is a Python module that manages the framework's main logic. The Session Handler component analyzes the user request and determines if it is for accessing a new URL or it is a simple request within the current URL the user is browsing. Requests for a new URL (i.e., URL first access requests) imply loading a new page with its data, whereas requests for content within the current URL are for new content segments (nodes) within the page the user is visiting.
Navigation context initialization. When the Session Handler receives a request for a new URL, through a Headless Browser (in our case Selenium APIs[2]) it simulates navigation and creates a tree data structure, the CNT described in the next section, that represents the content of the page allowing navigation through it. It also trains an NLU model with Web page data extrapolated from the CNT. The NLU model will thus be able to understand the user's utterance on the given page and provide informa-

[2]https://www.selenium.dev/

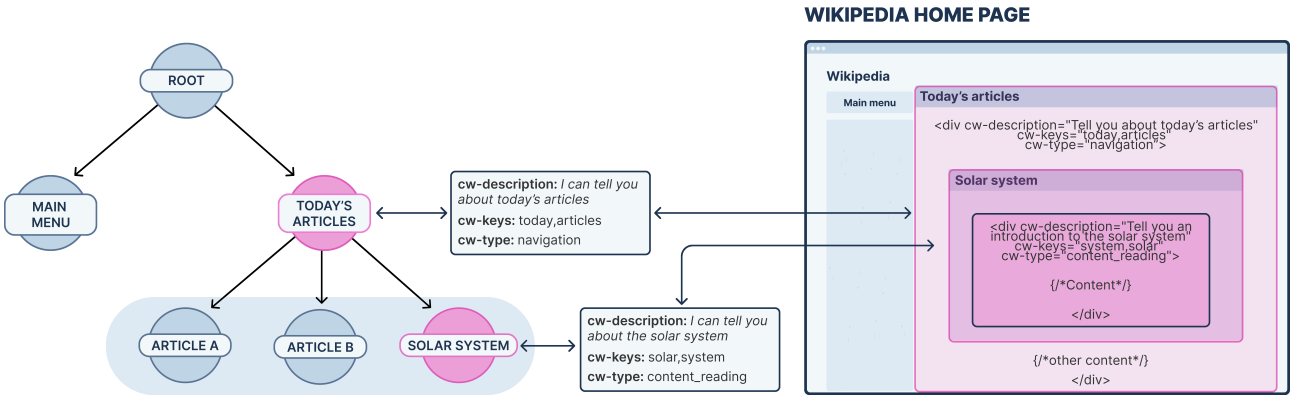

Figure 3: CNT structure

tion that will be utilized to perform the appropriate actions by the Intent Handlers. These are components (bots) responsible of performing navigation actions on the CNT of the current page. Consequently, a response to the user is returned, providing the content requested for the new page.

Navigation context management. If the user request is within the current URL (the navigation context has been initialized), the Session Handler will send the user utterance to the NLU model to retrieve necessary information to call a specific Intent Handler and perform the proper action on the CNT; subsequently, it will build and provide the answer to the user.

The main characteristics of the ConWeb approach can be summarized by the following three aspects already briefly introduced:

- **Conversation-oriented Navigation Tree (CNT)** - use of a data structure to represent the Web page content.
- **Intent Recognition** - interpretation of user utterances through a NLU model trained by using the website content.
- **Intent Handlers** - components responsible of taking automated navigation actions to fulfill the user requests.

In the following we are going to describe their characteristics in detail.

3.3. CNT

We want to develop a framework that can not only get information from Web pages but also support navigation and browsing inside their content. As a result, we need an auxiliary navigation structure that allows the user to simulate Web navigation. A Conversation-oriented Navigation Tree (CNT) is the structure used to represent the content of a page. The framework creates it by utilizing all of the relevant information to maintain the meaning and hierarchy of the content. In fact, the structure of the CNT is made up of many nodes, each with its own content. The CNT's key feature is that each node represents its own content; consequently, the page follows a hierarchy between parent and child nodes; in other words, the page is divided into separate sections, each of which is represented by a node. Only leaf nodes hold the actual content of the page (that could be a link, a text, an image and so on). The other nodes represent navigation indices, providing for navigation towards the other nodes.

In the current ConWeb prototype, the creation of the CNT for a particular URL is made possible by the presence of extra annotations (additional HTML attributes) within the page HTML, therefore leveraging an Headless Browser we are able to built it when needed.

Figure 3 illustrates an example of CNT related to a generic Web page.

Every CNT node contains different information that our framework could utilize to manage it provision of its related content, such as a description of the content, its position, and so on. All of this information is collected through HTML annotations or by building the CNT. The most significant data in the nodes are the `keys`, which are words (entities) that represent the content of the node. A node describing the content of a solar system, for example, will have the entities `solar` and `system` as keys. These keys are critical for allowing our framework to retrieve the specific content requested by the user.

Intent	Description
content	Reaching a specific content
read	Reading a specific text in a node
read_paragraphs	Knowing how many paragraphs are present inside a text
read_links	Reading all the links inside a node
help	Requesting help by the assistant
page_info	Requesting information on the content of the page
location	Requesting information on the current position
repeat	Asking the assistant to repeat what it has just said
back	Asking to return to the previous content (i.e. node)
top	It means the will to return to the starting point of the user navigation
affirmative	It means an affirmative answer to the question asked by the assistant
negative	It means a negative answer to the previous question asked by the assistant

Table 1: List of Intents that ConWeb can recognize and that support the browsing of a Web page.

3.4. Intent Recognition - NLU

To specify the user goal, in NLU we refer to `intent`, whereas with `entities` we refer to words relevant in the context of a Web page (`keys`) and also to words relevant for reading content such as `first paragraph`, `second` and so on. The user utterance is provided to the NLU model which has been already trained; in particular it is trained with Web page relative data (`keys`) and other data to handle default intents such as `back`, `help` and so on. Table 1 summarizes all of the ConWeb supported intents, along with a brief description of each. The NLU model upon receiving the user utterance will provide as output (interpreted result) `intents` (with a confidence) and `entities`. As described in the following section, the Intent Handlers use these intents and entities to identify and perform the necessary actions. `Rasa`, an open-source Conversational AI library, is used as NLU component in charge of parsing the user utterances and extract the needed data.

3.5. Intent Handlers

The Session Handler component calls the appropriate Intent Handler based on the intents and entities extracted by the NLU engine and the user's current navigation context. Each handler eventually builds a response and forwards it back to the ConWeb Client, which will render it via voice.

Navigation Bot. The Navigation Bot (NB) is invoked when a user requests content (`content` intent). Its primary function is to direct the user to the desired content via a search of the related node in the CNT. Depending on the node content (text, link or another parent node), it calls the bot in charge of serving that specific content.

Content Reading Bot. The intents `read`, `read_paragraphs`, and `read_links` are handled by the Content Reading Bot (CRB). If a user lands on a node that has text, the CRB is called and the text content is exposed (which is divided in paragraphs). Following that, the user could request to read a specific paragraph (`read`), to read the links that are present (`read_paragraphs`), or to read the number of paragraphs (`read_links`). The CRB manages entities such as `first paragraph`, `second` and so on, to select which text the user wants to read.

Link Bot. When a user lands on a node representing a link, the Link Bot (LB) is invoked to redirect the user to the new page represented by the intercepted URL. The navigation context will be updated, as well as a new CNT and a new NLU model will be built and trained.

Scaffolding Intents Bot. The Scaffolding Intents Bot (SIB) is in charge of handling general (i.e., not content-specific) navigation requests such as `back`, `repeat`, `help` and so on. Depending on the intent, the appropriate action will be conducted also considering the current navigation context.

4. Evaluation

Some evaluation activities provided us with useful hints on the perception of the ConWeb

paradigm by visual impaired users and on technical aspects that validate our approach and expose further challenges for future work.

User-based evaluation. The user-based evaluation was conducted with visually impaired users. The users appreciated the structure of the conversation and they judged positively the new paradigm. They also highlighted some interesting extensions, such as how ConWeb should be made customizable to meet specific needs and preferences, and how the framework should be somewhere between a typical voice assistant and a screen reader. In fact, our assistant is designed to help individuals explore websites, but our target users also include people who are experienced with using screen readers; therefore, through customization, we want to meet their needs for a device which can be also integrated with the technology they are accustomed with, with the advantage of exposing content in a more structured way (highly admired aspect).

Technical evaluation. The technical evaluation then allowed us to discover that the response time for each request to our assistant (from the user's spoken utterance to the received answer) is acceptable and never exceeds 1 second. The only challenge we encountered is when we need to train the NLU model on a newly visited page (only done once per page). Training took up to 15 seconds in our testing, but the Virtual Machines where we tested our framework did not have a GPU (necessary for training NLU dataset and could minimize the response time). In particular, we tested our framework with the following configuration: CPU Intel(R) Core(TM) i7-10875H 2.30GHz, 8 physical cores, RAM 32GB, GPU not present, medium test page size: 250KB.

5. Conclusion

This thesis aims to demonstrate how Conversational AI combined with Web technologies can provide an additional channel for accessing websites via an natural language interaction. The thesis has validated the designed framework primarily on content-oriented websites. Even if additional elements are required to meet the requirements of other website categories, we are certain that the flexibility provided by the ConWeb architecture can favor the extensions needed to handle other types of intents and content. However, there are some limitations that can be overcome in future work.
First of all, the performance for training the NLU model on the fly must be improved. Also, the current ConWeb platform requires augmenting the website HTML with annotations for building the CNT, while future research can concentrate on automating the annotation process and content extraction from Web pages. In addition, future work can be devoted to new user studies that will also include sighted users to determine whether the assumptions developed with and for visually impaired people may be applied to other types of users. Indeed, we are confident that the resulting approach can benefit people universally, and has a potential that will impact Web Engineering in the coming years.

References

[1] M. Baez, F. Daniel, and F. Casati. Conversational web interaction: Proposal of a dialog-based natural language interaction paradigm for the web. In *Chatbot Research and Design*, pages 94–110. Springer, 2020. ISBN 978-3-030-39540-7.

[2] M. Baez, C. Cappiello, C. M. Cutrupi, M. Matera, I. Possaghi, E. Pucci, G. Spadone, and A. Pasquale. Supporting Natural Language Interaction with the Web. In *Proc. of ICWE 2022 (in print)*. Springer, 2022.

[3] M. Baez, C. M. Cutrupi, M. Matera, I. Possaghi, E. Pucci, G. Spadone, C. Cappiello, and A. Pasquale. Exploring Challenges for Conversational Web Browsing with Blind and Visually Impaired Users. In *CHI'22 Extended Abstracts*. ACM, 2022. ISBN 978-1-4503-9156-6/22/04.

[4] J. Cambre, A. C. Williams, A. Razi, I. Bicking, A. Wallin, J. Tsai, C. Kulkarni, and J. Kaye. Firefox voice: An open and extensible voice assistant built upon the web. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.