PALTask Chat:
A Personalized Automated Context-Aware Web-Resources Listing Tool

Pratik Jain, Andreas Bergen, Lorena Castañeda, Hausi A. Müller
Department of Computer Science
University of Victoria, Victoria, Canada
{*pratik,andib,lcastane,hausi*}*@cs.uvic.ca*

***Abstract*—With the constant evolution of the Internet, a repetitive and ordinary task such as searching online resources has become more complex due to the amount of web services and formats available (e.g., video, audio, text or images). In order to obtain resources within a specific domain, a user manually performs several tasks, such as navigating through different web services, filtering according to various criteria and selecting the relevant results. However, the insufficient contextual information of the underlying application hampers the user's experience. In this paper, we propose a tool to improve a user's experience by automating the task of retrieving interesting resources in a multi user setting. We apply this approach to a chat scenario where users are exposed to resources that are of common interests by exploiting the users' personal context information.**

***Keywords*-Automated Web-Tasking, Personal Web, Smart Internet, Smart Applications, Context-Aware.**

I. Introduction

Personalized web-tasking aims to improve the user's experience through the automation of repetitive and ordinary tasks to fulfill personal goals [1], [2]. To achieve this automation in software solutions we focus two elements: 1) *uncertainty of the context*, which refers to the unpredictable changes in a user's goals as well as in the elements of the domain, and 2) *dynamic self-adaptation* as a response to context changes to guarantee the achievement of the user's goals. In the personalized web-tasking scope, this automation is highly dependent on the user's personal context (i.e., interests, preferences, historical usage, internet behavior, devices, and personal information).

An example of an ordinary task is the search and selection of proper web-resources based on a specific matter of concern. During the execution of this task, a user might previously know what is the objective of the search. The user also elaborates a plan that includes: a set of preferred web sites based on the nature of the user's personal goals, a set of keywords that describe the user's interest, and the desired formats for the resources the user expects to retrieve (e.g., video, audio, text, or images).

However, in a dynamic situation, such as an online conversation between two or more users, the topics are dynamic and uncertain. This implies for the users to constantly switch among web sites, formats and keywords, in order to search relevant information while simultaneously carrying out the conversation. Indeed, users switch constantly from task to task while risking of loosing precious information and control over the execution of such tasks. For example, during this web resources search, the user might loose track not only of the conversation but also of the multiple websites being explored. To overcome these situations, the user usually scrolls back into the conversation, or saves the resourses as bookmarks. In addition, if the user gets interested in certain resource—either because of the format, topic, or a web source—the user manually updates the whole search including this particular new interest. Finally, the user is possibly unaware of new sources that could provide better or more interesting resources that might enrich the task performance.

Ordinary and repetititve tasks such as the one already described are often performed manually by the user and could be automated to provide the user with a better experience. Automating the user's personalized web tasks in this scenario is a challenge due to the lack of understanding of the user's dynamic context. Moreover, the user's feedback during the execution of the task is required to refine the search and the resources retreival.

Some existing approaches take into consideration key elements suitable for our scenario; Ranganathan demonstrates with ConChat that the user experience of chat clients can be improved by augmenting chat messages with contextual information to prevent semantic ambiguity between the chat participants [3]. Similarly, the SemChat tool proposes that context analysis can be carried out offline after a chat session has ended to improve context awareness [4]. GaChat is based on automatically annotating chat conversation on the fly, then extracts context from each chat message and then integrates the discovered related items into the chat [5]. GaChat performs analysis on a server where chat messages are analyzed for context. This limits the available context and centralizes the processing burden.

Within the scope of online conversations we propose a tool to automate the web-resources listing (e.g., videos, music, images or documents) based on the contextual personal information of the user, with context-aware self-adaptive capabilities. In brief, our tool provides assistance to the user

by automating the search of resources through different web services enhancing users' experiences. Our proposal takes advantage of the Villegas' Personal Context-Sphere [6] to incorporate the user's personal information into the automation of the task as well as the previous approaches discussed above as the basis for our application.

The remainder of this document descibes the main elements of our approach. Section II describes the user interface elements and the automatic operations, and Section III presents the application architecture, with the elements and the interactions among them, and the technical details of our proposal such as the search sources, and the API explored to build this application.

II. Proposed Solution

We illustrate the functionality of our proposed tool with a common chat application scenario. To achieve automation of the web-resources listing task, our proposed tool relies heavily on context analysis. This context is obtained from several sources: 1) the users' personal context spheres, and 2) the chatting contents itself. Firstly, the context of each users' is crucial. This includes aspects such as previous browsing history, search preferences, and interests as presented by Villegas as the personal Context-Sphere [7]. Secondly, the conversation between users can be analyzed dynamically to extract context information. For example, during an online conversation, context analysis determines that one of the users is referring to a particular video found online at YouTube. The tool displays relevant videos from YouTube and other sources, through context matching. This eliminates manual steps such as opening a browser, connecting to a website, searching for the proper video, copying and pasting the URL into the chat.

A. User's Interface Look and Feel

Figure 1 shows a conceptual GUI of our prototype. It contains two main elements. 1) The conventional chat window, and 2) the web-resources list display. This second element, contains a tab navigation that represents the web-resource format list (i.e., video, images, text, documents, or audio).

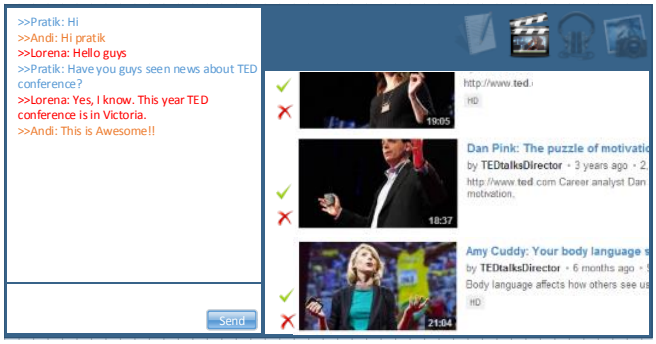

Figure 1. The PALTask's GUI Illustration

Figure 2 illustrates the architecture of our solution. The key elements of this architecture are: 1) The dynamic resource engine consists of two parts: a) the keyword extractor that uses known methods (cf. Sect. III) to obtain the keywords and phrases from the chat; and b) the analyzer that merges these extracted keyword with information from the personal context-sphere to discover the conversation context as it pertains to each user. 2) The extensible module of web-resources, which makes queries based on the personalized keywords to different sources.

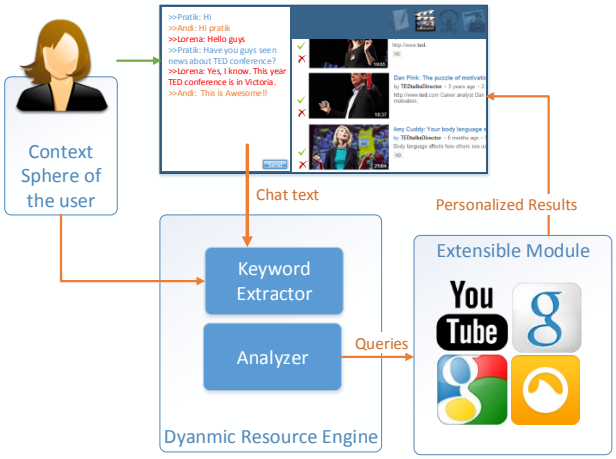

Figure 2. High-level Interactions

B. Operations

The operations of this tool are as follows: 1) *Drag and share,* in which a user will be able to drag and drop web-resources through the window chat, and dynamically updates another user's web-resources list. This operation enhances the user's experience by sharing interest with other users. Once a user shares a resource, the other users in the chat get input on the personal interests of that user. Sharing this information might reveal a previously unknown common interest and spark new discussions.

2) *Automatic refreshing* of the web-resources list based on a user's feedback and chat behavior (e.g., links on the chat window, likes or dislikes on a video). With this feedback the personal context sphere automatically updates and effects future results.

3) *Source switching* that automatically displays the appropriate type of resource based on the personalized context of the user. From the point of view of the user, this operation eliminates the overhead of maintaining multiple tabs with different contents.

III. Design and Implementation of PALTask

We compare our design and the tool's component architecture to related approaches and highlight our tool features

including the following artifacts: keyword extractors, communications API for web resources, and Villegas' personal context-sphere [6]. We evaluated over a dozen keyword extractors based on the quality of results, availability of remote API, cost, license, and availability of source code. Keyword extractor tools often apply statistical algorithms and morphological analysis. Some keyword extractors considered in our evaluation include: Yahoo API Term Extractor [8] which uses morphological analysis, Word Finder Extractor [9], Sketch Engine [10] and Alchemy [11]. We selected a Python implementation of the Rapid Automatic Keyword Extraction (RAKE) algorithm [12], [13]. Using this implementation of RAKE facilitates the integration with PALTask's code base and the extraction of keywords from a person's context sphere.

Websites such as YouTube, Google Web Search and Google Scholar, provide public APIs that enable the extraction of resources with customized queries [14]–[16]. Similarly, Tinysong [17] offers an API for Grooveshark's music catalog. PALTask's queries to these sites are based on the keyword extractor's output.

Our approach includes five software components as depicted in Figure 3: GUI, Keyword Extractor, Personal Context Sphere(PCS), Server, and Client.

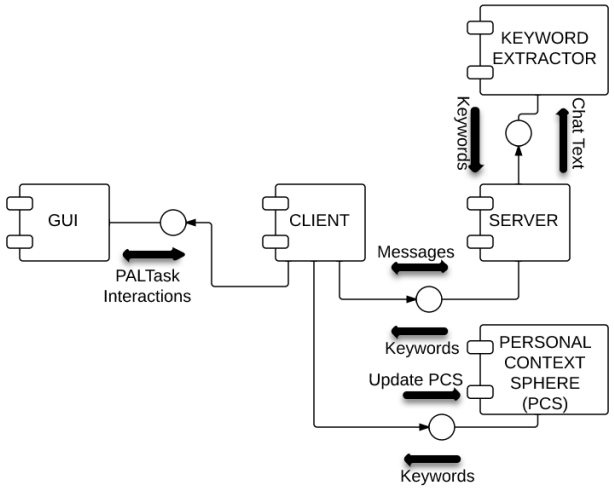

Figure 3. PALTask Component Architecture

A. Graphical User Interface component

The GUI's main functions are to allow user interactions and display automatically retrieved web resources. The GUI provides operations such as: chat console, web-resource list and filtering buttons (a resource like, delete, and for different formats: audio, video, text, and images).

B. Keyword Extractor component

Keyword Extractor component ranks and analyses keywords obtained from text. The keyword extractor is deployed at the server to leverage greater computational power and resources. Moreover, having extensible modules at the server side allows us to add resource intensive services (such as automated video processing). The keyword extractor is external to the server component and connected via an API. In our prototype, chat messages pass through keyword extractor. The keyword extractor receives two messages as input each time a chat message is sent. To obtain significant information from a conversation, we analyze the most recent message sent, which is based in the 140-160 character length of SMS [18] and Twitter [19] statement for meaningful messages. Keywords obtained from the single most recent message are weighted greater in importance and a combined average is returned as a ranked list of keywords using an API.

C. Personal Context-Sphere component

The Personal Context-Sphere (PCS) component represents the user's personal interests and contextual information. It provides a representation of this information in the form of keywords. A user's personal context sphere is accessed through the client component. This information along with the keywords provided by the server component are matched to retrieve the proper web-resources. This increases the accuracy of results and improves the automation of related personalized web tasks.

D. Server component

Our server component is a traditional management system of chat conversations. It includes connecting users, exchanging messages and controlling the users' chat sessions. PALTask adheres to the traditional centralized client server architecture. Clients are connected to a central server component via a network. All clients' messages pass through the central server which controls all message passing. Furthermore, the server is responsible for relaying the text which is to be analyzed by the keyword extractor. The retrieved keywords are passed to the respective client.

E. Client component

Lastly, the client connects to the server as an ordinary chat application, which includes login and communication interactions. Ranked keywords, which represent the context of the conversation, are obtained from the server. Then, it matches these results with contextual keywords from the user's PCS. Matches between keywords receive a modified higher ranking to increase the degree of personalization of retrieved web resources. The client follows the traditional model with small additions. The well known text input field, chat message display window. Additionally, the client component includes the analyzer element that queries results of web resources based on the chat and the personal context-sphere keywords, and are displayed on the GUI.

ACKNOWLEDGMENT

This work is funded in part by University of Victoria (Canada), the National Sciences and Engineering Research Council (NSERC) of Canada under the NSERC Strategic Research Network for Smart Applications on Virtual Infrastructure (SAVI - NETGP 397724-10).

REFERENCES

[1] J. W. Ng, M. Chignell, J. R. Cordy, and Y. Yesha, "Overview of the smart internet," in *The Smart Internet*, M. Chignell, J. Cordy, J. Ng, and Y. Yesha, Eds. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 49–56.

[2] IEEE SERVICES 2013. 1st international workshop on personalized web tasking (pwt 2013). Retrieved April 2013. [Online] http://www.servicescongress.org/2013/pwt.html

[3] A. Ranganathan, R. H. Campbell, A. Ravi, and A. Mahajan, "ConChat: A context-aware chat program," *Pervasive Computing, IEEE*, vol. 1, no. 3, pp. 51–57, 2002.

[4] C. Abela and K. Cortis, "SemChat: Extracting Personal Information from Chat Conversations," in *Workshop on Personal Semantic Data, EKAW*, 2010.

[5] S. Horiguchi, A. Inoue, T. Hoshi, and K. Okada, "GaChat: A chat system that displays online retrieval information in dialogue text," in *Workshop on Visual Interfaces to the Social and the Semantic Web (VISSW 2009), Sanibel Island, Florida*, 2009.

[6] N. M. Villegas, "Context Management and Self-Adaptivity for Situation-Aware Smart Software Systems," Ph.D. dissertation, University of Victoria, Canada, February 2013.

[7] N. M. Villegas, H. A. Müller, J. C. Muñoz, A. Lau, J. Ng, and C. Brealey, "A dynamic context management infrastructure for supporting user-driven web integration in the personal web," in *2011 Conference of the Center for Advanced Studies on Collaborative Research (CASCON 2011)*, 2011, pp. 200–214.

[8] Yahoo. Yahoo Term extractor. Retrieved April 2013. [Online] http://developer.yahoo.com/search/content/V1/termExtraction.html

[9] World finder extractor. World finder extractor. Retrieved April 2013. [Online] http://wordsfinder.com/api_Keyword_Extractor.php

[10] Sketch engine extractor. Sketch engine extractor. Retrieved April 2013. [Online] http://trac.sketchengine.co.uk/wiki/SkE/KeywordsAPIl

[11] Alchemy API extractor. Alchemy APIi extractor. Retrieved April 2013. [Online] http://www.alchemyapi.com/api/keyword/

[12] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic keyword extraction from individual documents," *Text Mining*, pp. 1–20, 2010.

[13] Aneesha. Rake implementation. Retrieved April 2013. [Online] https://github.com/aneesha/RAKE

[14] Google. YouTube API. Retrieved April 2013. [Online] https://developers.google.com/youtube/v3/getting-started

[15] Christian Kreibich. Python API for google scholar. [Online] http://www.icir.org/christian/scholar.html

[16] Google scholar. PHP API for google scholar. Retrieved April 2013. [Online] https://code.google.com/p/bioguid/source/browse/trunk/www/scholar_ris.php?spec=svn112&r=112

[17] Grooveshark. Grooveshark API for music. Retrieved April 2013. [Online] http://developers.grooveshark.com/

[18] L. Barkhuus and A. Vallgårda, "Saying it all in 160 characters: Four classes of sms conversations," *The IT University of Copenhagen, Technical Repport: ITUTR-2004-45*, 2004.

[19] S. Rybalko and T. Seltzer, "Dialogic communication in 140 characters or less: How fortune 500 companies engage stakeholders using twitter," *Public Relations Review*, vol. 36, no. 4, pp. 336 – 341, 2010. [Online] http://www.sciencedirect.com/science/article/pii/S0363811110000792