Towards Personalized Web-Tasking: Task Simplification Challenges

Lorena Castañeda *†‡, Hausi A. Müller *†, Norha M. Villegas‡*
* *Department of Computer Science. University of Victoria. Victoria, Canada*
email: {lcastane,hausi}@cs.uvic.ca
‡ *Dept. of Information and Communication Technologies. Icesi University. Cali, Colombia*
email: nvillega@icesi.edu.co
† *IBM Canada Software Laboratory, CAS Research. Markham, Canada*

***Abstract*—Despite the increasing use of the web to support human activities, most web interactions required to accomplish personal goals are performed manually by users. Even though users can easily transform a goal into multiple web interactions, the manual governance of these interactions diminishes the user experience. Personalized web-tasking seeks to improve the user experience by automating personal web tasks. This automation is driven by user needs, matters of concerns, and personal context. An important concern in personalized web-tasking is task simplification, the process of decomposing a personal web task into simpler tasks that can readily be composed into bigger tasks. This position paper characterizes a set of task simplification challenges intended as starting points for advancing the field of personalized web-tasking.**

***Keywords*-Web—Tasking, Task Simplification, Personal Web, Self-Adaptive Systems, User-Centric, Smart Internet, Semantic Web, Context-Awareness**

I. Introduction

Users increasingly rely on web applications to complete a variety of every day tasks that used to be performed offline. Concerns related to these tasks include gathering information on a research topic or a personal interest, executing transactions (e.g., for performing banking operations), communicating with other people (e.g., through email, chat, or social networks) [1].

Despite the increasing use of the web to support human activities, most personal web operations are performed manually by the user. This implies that the user understands her own concerns and can readily identify web operations to accomplish the task and ultimately her goals. For this, the user must be capable of translating a particular personal goal into a sequence of web tasks. We define a *web task* as the integration of web services, interactions and sessions, from which the user benefits to achieve a personal goal. If the user classifies the task as complex, she must decompose it manually into smaller and simpler tasks logically sequenced. Hence, the user is responsible for managing the state of the task, that is, interrupting, resuming, and measuring the execution of these subtasks. Moreover, the user is also in charge of, without any support from the system, evaluating whether her goal has been achieved successfully. Furthermore, the user must be aware of changes in her concerns, available services and resources, and relevant environmental situations, as well as the opportunity of interacting with other users to adapt the task sequence accordingly.

As an example imagine a user browsing the latest news. This ordinary main task implies the accomplishment of a set of smaller tasks. First the user must load a set of local and international news websites that she regularly visits. Then she filters manually those news within her personal interest (e.g., politics, technology and worldwide top news). Afterwards, while reading the selected news, she can use external applications to save important news she wants to investigate later, as well as other resources that are relevant to them (e.g., videos, audio recordings, pictures). Finally, the user also might share some of these news with other users using social media or email.

Manual task simplification is impractical. For instance, at some point of the execution the user can forget the task sequence, and then spend significant time trying to recover it. (e.g., the user may forget the news related to a video that she saved for watching later). Moreover, the user is generally unaware of newer and better services that may become available to improve the task sequence, and trying to find them is time consuming and relies on the user's expertise. (e.g., the user is unaware of the existence of a web application that groups several news from different websites such as News360). Similarly, the manual management of the simplification process becomes an issue, given that the user must keep track of the task life cycle manually. Task decomposition may require the interaction of multiple web applications. Therefore the user would have to control the sessions and permissions of each web service. (e.g., independent login processes by the same user for interacting with multiple applications such as the actual news websites, her email client, or her social network). Finally, the user will have to deal with runtime conflicts such as unexpected exceptions and unavailable services (e.g., the video the user is trying to watch may be unavailable for her country), and collaboration with other users will depend on the user's

social networking and personal skills. In other words, there is no automatic way to share a task simplification plan with other users who may have the same concerns.

Personalized web-tasking can be defined as the automation of personal web interactions that are usually repetitive [2]. To improve the user's web experience, this automation must exploit the user's personal context, including her social sphere, as well as historical information from previous interactions. This position paper focuses on a crucial aspect of personalized web-tasking, task simplification. We define *task simplification for personalized web-tasking* as the automatic decomposition of a task into an ordered set of simpler subtasks that must all be executed independently, to advance towards the fulfillment of the user's goal. This decomposition must be performed at runtime, using web technologies, and according to changing context situations. These context situations may affect not only the user's goals but also the resources required to complete the main task.

Task simplification has been an important concern in different application domains. In particular, in industry people seek the simplification of daily operations to save resources and time. Task simplification is also relevant in software-based systems, due to the need to increase productivity and efficiency, to reduce the task complexity, or to substitute tasks for more appropriate tasks.

The contributions of this position paper are as follows. First, in Sect. II, we summarize selected findings of our analysis of task simplification methods applied in several domains. Second, from these findings, we characterize in Sect. III a set of task simplification challenges that provide valuable starting points for advancing personalized web-tasking. As part of the identified challenges, (1) we propose a definition for task simplification and a characterization of the life cycle of web tasks, both in the context of personalized web-tasking; (2) identify a list of key elements that must be modeled to support the self-adaptation of web tasks according to context situations; and (3) propose a set of attributes useful to measure the complexity of web tasks.

II. An Overview of the State of the Art

The concept of *task* refers to the actions that are required and performed by humans, and therefore has been studied in different domains including software engineering. Moreover, the concept of *task simplification* is an important concern to enhance performance and efficiency of the execution of a task. With the goal of characterizing research challenges for task simplification in personalized web-tasking, we analyzed methods and techniques applied to task simplification in several domains. In this section we present selected findings of our analysis.

A. Organizations

Lynch argues that task simplification is important in organizations for increasing the efficacy of employees in completing assigned tasks [3]. In this domain, the ultimate goal of reducing the complexity of a particular task is to simplify the skills required to perform it. For this a task must be supported with two elements: (1) a simpler, and easy to remember, representation of the elements of the task (e.g., numbers or symbols in the fast-food business), and (2) technological devices that equip employees with cognitive support to perform the tasks more effectively (e.g., a cash register machine).

B. Human Behavior

Sohn and Anderson focus on the capability of the human brain to switch tasks and keep track of them [4]. Moreover, they argue that task preparation and task repetition are important for performing tasks successfully. Task preparation refers to the strategic planning of the task that includes having previous knowledge and understanding of its requirements, restrictions and structure. In other words, preparation concerns the cognitive control of the process. Task repetition concentrates on reducing the possibility of error by constant practice, and also provides a baseline for performance. According to Sohn and Anderson, task switching is improved with both systems, the foreknowledge gained with preparation, holding the information and mapping of the task, and the fluency after repetition.

C. Business Processes

Shi *et al.* define a business process as the representation of a sequence of tasks and the dependencies among them, and propose a task-based modeling method for constructing businesses processes [5]. In their approach, independence, encapsulation, completeness and consistency are key elements to enable task re-usability. Relevant aspects of their proposal include: (1) The management of a task, which includes the specification of the actions to be executed, the sequence to follow to complete the task, and the object on which the task is performed; (2) the degree of involvement of the user in the execution of the task (i.e., completely manual, semi-automated, and fully automated); and (3) the assessment of the outcome of a task to decide whether it requires a follow up action (or task) to continue the sequence, or is completed and assessed as either failure or success.

D. Robotics

Tsujiuchi *et al.* apply task simplification to teach a robot arm movements [6]. In their approach, they perform manual actions over the robot arm and record them to create a memory of movements. Later, the movements are matched to objectives that correspond to human requests (e.g., pick something from a surface, or wave to an audience). If a new objective request is made with no movement associated, the robot arm tries to infer it based on previously learned movements. This process is performed by decomposing the

already known movements and creating new sequences, and can be enhanced by gathering information from other sources, such as sensors to track the objective, or cameras to sense the environment.

E. Healthcare

The healthcare domain seems to have two different views for task simplification—human resources and medical processes. Aylward and Linkins highlight the importance of task simplification based on limitations and available resources, hence decomposing and adapting tasks according to different scenarios [7]. Another important element is the use of technological innovations to simplify the execution of complex tasks.

F. Software Engineering

Improving software developer experience and optimizing software development processes—while dealing with complexity, uncertainty, runtime constraints, and diverse users, technologies and requirements—have been important challenges for software engineering research communities. In this domain the definition of *task* is used to represent two concepts. On the one hand it refers to the functionality performed by the software system. On the other hand it represents the user's intention over the software system that can be seen as the functional requirements that fulfill the expectations of the user.

Cousin and Collofello use task analysis to improve the software maintenance process by identifying generic tasks and the information required to perform them [8]. In their approach, task analysis for any domain implies: (1) determining the subtasks of a task domain, (2) determining how these subtasks interrelate, and (3) determining the information necessary to perform each task. Also, this proposal defines six steps including creating a network of tasks and identifying, representing and obtaining the knowledge required to execute them.

Guoqi *et al.* address task simplification by decomposing tasks using weighted and/or tree and AOE-Network (Activity On Edge Network), which is a graph where the edges represent the tasks to be performed [9]. This approach classifies tasks into three categories. *Simple task*, which corresponds to a basic work unit that can be completed by a single agent; *complex task*, which generally requires multi-agent participation; and *subtask*, which corresponds to an element of a set of tasks whose completion is required to fulfill a complex task.

Erdem *et al.* propose an interesting approach to understand software based on user expertise, that is by expressing users goals as computational tasks [10]. In their study they recorded user actions over predefined questions and identified patterns. With these patterns they built a taxonomy of software engineering tasks associated with user goals. Relevant elements of their approach include: the *who* and *how* questions and their relation with the task (i.e., the *who* is related with the owner, the *how* is related with the steps), the complexity of the questions to measure the answers, the relevance of the task to the user and vice-versa, the expertise of the user for performing a specific task, and the composition of a task (i.e., start and end points, structure of the task, and partial goals).

Feldman *et al.* annotate task models to represent user interactions that must be performed before the execution of the annotated task [11]. These annotations contain all the information required for the operation of the task (e.g., specifications of the data being exchanged, the order and the conditions of the executions of the tasks). Task simplification and automation can benefit from the inclusion of preconditions into the annotated tasks.

Task simplification techniques applied in other domains provide valuable insights for task simplification in personalized web-tasking. From the analyzed techniques, we identified the following elements as relevant to the characterization of task simplification research challenges in personalized web-tasking: task complexity measurement, resource planning, expertise and knowledge required to execute the task, task interdependency, output assessment, proper task preparation, task learning (e.g., through task repetition to record useful information), and task equivalence to substitute tasks (e.g., when a particular task cannot be performed due to unavailable resources required to complete it).

III. Research Challenges on Task Simplification

According to our definition of task simplification, the decomposition of a complex task into ordered and simpler subtasks must be performed automatically, at runtime and taking into account changing context situations. Task simplification is a fundamental concern for realizing personalized web-tasking. From our analysis of task simplification techniques used in different application domains, we characterize selected research challenges for personalized web-tasking as follows.

A. CH-1: Life Cycle Management

The life cycle of a task defines the key stages of its execution, including the invocation and termination actions, as well as the subtasks that perform the task. Life cycle modeling is crucial for the management of web tasks at runtime. Relevant management operations include the elements identified in the previous section (i.e., complexity measurement, resource planning, expertise determination, task interdependency, task preparation, output assessment, task learning, and task equivalence). In our news example, we identify different operations performed by the user, such as browsing a specific set of preferred websites, filtering interesting news, saving news and related resources, and sharing news with other users. All these operations concern the access to the latest news, but each operation can be seen

as an independent task on its own. Figure 1 depicts our proposal for the life cycle of a web task. Each phase is characterized as follows:

Preparation: This phase determines the beginning of the execution of the task, and establishes the goal to be achieved, as well as the decomposition of the task. This phase involves the following management operations: task simplification, input gathering, resource planning, complexity measurement, expertise determination, and task interdependency and equivalence analysis. It is worth to mention, that this phase is recursive for each task within the decomposition. In our example, getting the latest news includes preparation actions such as defining the time frame for selecting recent news, and getting the different news sources. Moreover, after the decomposition one of the resulting subtasks is to filter the news according to the user's preferences. Thus, this filtering subtask executes its own preparation actions such as gathering the user's interests and map them to the categories of the news websites. This recursion is to get all the required inputs to execute the main task as a whole, while avoiding unnecessary interruptions and preventing errors.

Execution: This phase manages the activities of the task. During this phase task simplification might be re-evaluated and the subtasks in the sequence might change. The execution of a subtask is represented by their corresponding life cycles (i.e., the *Subtasks life cycles* box in Fig. 1). It is important to mention that even though the preparation phase was previously executed for the subtasks, runtime changes in a subtask might imply the execution of a new preparation phase. However the results of the previous preparation should remain available if needed. For example, the subtask of sharing the news with the user's social network could be replaced by a new subtask consisting in sending an email. This new subtask requires other inputs different than the ones previously gathered (a new preparation phase will be required), however, the recipients remain the same. This information was already gathered during the previous subtask preparation). As shown in Figure 1, the decision node labeled as (1) evaluates the completion of the subtasks as part of the main task goal. This decision node evaluates whether the output of the subtask is the expected according to its specific goal, and then decides whether or not to continue with the execution of the sequence. If the decision is to continue, the execution of the task sequence is resumed (e.g., for example to perform additional activities based on the outputs of the subtasks). Once the execution is finished the output is sent to the next phase for analysis.

Output analysis: This phase measures and evaluates the results of the task execution according to the goal and success conditions. In figure 1 this decision node, labeled as (2), determines whether the task can be terminated or needs to be executed again (in such case, the output of the evaluation is returned to the start point of the execution as an input for adaptation decisions). In our example, the desired output is to provide the user with a personalized list of the latest news. The evaluation of this output depends on the fulfillment of the requirements (i.e., the news listed are recent, from the user's preferred websites, and matches her interests) as well as the feedback from the user (e.g., total acceptance, modifications into the list).

Termination: This phase ends the task by sending as the output the results of the task according to the goal specifications applied in the Preparation phase. Also, some additional activities might occur in this phase such as notifications, knowledge base modification, and resources liberation. For example, the feedback of the user about the completion of the task is provides key information for decision making in future task simplification processes. An example of feedback from the user is the inclusion of additional news associated with topics of interest that were not in the original list. n

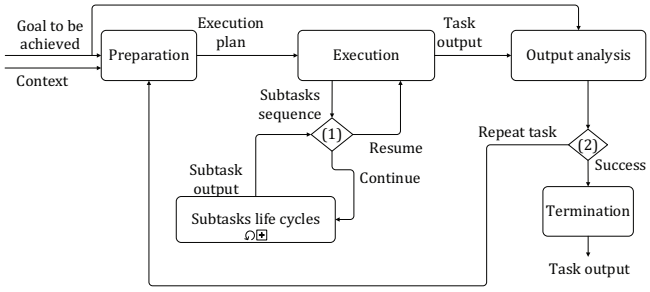

Figure 1. The proposed life cycle for a web task. Decision (1) triggers the independent life cycle of the subtasks in the sequence, and decision (2) decides weather the task goals were successfully achieved or not in order to terminate the task or repeat it.

B. CH-2: Task Dependencies management

An important concern in task simplification is the management of the sequence of subtasks and the proper communication among them. An important element to manage the dependency among tasks, and realize dynamic composition, is the definition of standard interfaces that not only act as subtask connectors, but also as manageability endpoints that allow subtask controlling. Properties of task connectors include type constraints (e.g., the task type to be plugged according to the right sequence), inputs, outputs, data types including contextual data, and communication protocols. Our news example illustrates a clear dependency among two subtasks of the filtering task: the subtask responsible for extracting the news from a website strongly depends on the results of the subtask responsible for matching each news website's category with a user's interest. Certainly the output of one subtask becomes the input of the next one, and in this case could be in the form of a simple data type such as a vector of strings. Moreover, in this specific example the preparation phase of the extracting subtask would define the requirement of this type of information, but

it will not be solved until the matching subtask is completed. Therefore, this specific task dependency is managed from the preparation phase through the end of the life cycle of the extracting subtask.

C. CH-3: Logic and Structural Self-Adaptation

Tasks may require to self-adapt by changing their business logic, replacing an entire subtask, or recomposing the subtasks sequence. These adaptations must be performed automatically, at runtime, and according to context situations, which include the user's needs and the availability of resources. For this, self-adaptation models must be defined at design time and manipulated at runtime [12]. Important elements that must be specified in these task self-adaptation runtime models include: (1) the user's goal or matter of concern, (2) the subtasks sequence, (3) the set of equivalent tasks that may be used to accomplish the next step towards the ultimate goal, (4) the quality attributes that determine the fulfillment of the task, and (5) the context dimensions involved in the execution of the task (e.g., the user's location), including the corresponding context sources (i.e., sensors to be consumed). One example in our news scenario is a recent user's interest in a new worldwide topic. This new interest can be inferred from a recent and recurrent web navigation activity of the user. Consequently, the latest news task should be updated during runtime to provide the user with the latest news after including this new input. Another example is when a regular website changes its structure and the categories are no longer recognizable. In this case, the task simplification must adapt during runtime, to include a new activity in the filtering subtask to be responsible for identifying the category. Nevertheless, a self-adaptive action might include a change in the execution of the life cycle as well.

D. CH-4: Governance of Task Simplification

Task simplification implies management activities that include: the control of the life cycle of the task and its subtasks, the resolution of runtime exceptions and tasks conflicts, the management of dependencies and communication mechanisms among tasks, the management of resource availability, the control of task sequences, and the assurance of context-aware task self-adaptation. In other words, if during the execution of the sharing subtask the web service responsible for assisting this task becomes unavailable, the governance process indicates that it is necessary to trigger proper activities to solve the conflict. Such as activities may include finding a new equivalent service, and readjusting the parameters to incorporate this new service into the subtask.

E. CH-5: Measurement for Complexity

Task complexity measures are required to determine whether a task is candidate to simplification, as well as the simplest task. As mentioned in our example the main task is recovering the latest news. We identified a set of subtasks associated with it such as saving news and its related multimedia resources. However, this subtask could not be easily qualified as the simplest if we do not know what and where to save it, and in what format. This means that we could simplify this task into smaller subtasks to solve those questions. Likewise, if we have enough information to predict the answers to these questions, the complexity of the subtask is reduced. We propose a set of attributes that are relevant to the complexity measure of a task: (1) number of web interactions, (2) available knowledge about the task, (3) information available about previous simplifications, (4) number of inputs that can be inferred and number of inputs that require user's intervention, availability of resources including web services and context sources, and level of dynamics of the relevant context information (i.e., whether the relevant context dimensions static (e.g., the user's birthday) or change frequently (e.g., the user's location, preferences).

F. CH-6: Measurement for Automation

After translating a user's personal goal into a set of web tasks, it is equally important to identify the degree of intervention required from the user. We argue for the importance of this measure for the identification of simplification opportunities (i.e., those sections of the task that require less user intervention).

We identify three levels of automation: *(1) No automation*, which implies full intervention of the user for the completion of the task; *(2) partial automation*, which requires the intervention of the user for activities such as authorizations, input provisioning, and decisions making; and *(3) Complete Automation*, where the user is unaware of the execution of the task and its simplification, and have provided all the inputs required for the execution of the task during the preparation phase.

G. CH-7: Context-Awareness capability

The management of context information is crucial for the realization of personalized web-tasking. This is because both user matters of concern (i.e., task goals) and the resources required to accomplish user web tasks are highly affected by environmental situations (e.g., top news may imply new interests for the user, or better services that provide the news content appear).

Therefore, context information must be managed at runtime along the phases of the life cycle of the user's tasks. Dynamic context information differs from static context in aspects related to its modeling and management. Concerning context modeling, static context specifies, at design time, relevant context entities and the interactions among them, which remain immutable at runtime. The birthday and gender of a user are instances of static context. Therefore, monitoring mechanisms based on static context keep track

of entities specified at design-time. Once the system is in execution, the addition of new entities is not supported by the static context specification. On the contrary, dynamic context requires modeling techniques that support changes in the specification of context entities and corresponding monitoring requirements at runtime.

For example, location, and news topic and website preferences are instances of highly dynamic context. With respect to context management, monitoring strategies to keep track of static context are well known at design time and remain fixed at runtime, whereas monitoring strategies to manage dynamic context are required to change over time [13]. Dynamic context management is key to provide users with personal web tasks that are really relevant to their matters of concern.

IV. Conclusions

Personalized web-tasking aims to assist users in the achievement of their personal goals through the automation of web tasks and thereby enhancing the user's experience. We define task simplification as the automatic decomposition—at runtime while being aware of the various relevant contexts—of a personal web task into an ordered sequence of simpler and smaller web tasks, that eventually will fulfill the user's goal.

To address task simplification for personalized web-tasking, we reviewed selected approaches form different domains to highlight key elements with the goal of characterizing a set of research challenges on task simplification. Key elements include: the representation and structure of a task, the conceptualization of a task and its simplification, the set of activities that compose a task, and the interpretation of a task as an understanding of a user's intention.

The challenges we characterize in this paper are: (1) the need for managing the life cycle of a task based on the following proposed stages: preparation, execution, output analysis, and termination; (2) the standardization of task connectors, with corresponding properties, as manageability endpoints that allow the proper control of task sequences; (3) the need for runtime adaptation models and mechanisms to support logical and structural self-adaptation in personalized task simplification; (4) the governance of the life cycle of tasks and subtasks; (5) the measurement of task complexity to determine the required degree of simplification; (6) the need for an automation metric to evaluate the degree of user intervention required in the execution of a task; and (7) effective context-awareness support to tailor task sequences according to changes in the user's context and system's execution environment.

To the best of our knowledge, this position paper is the first to characterize research challenges for personalized web-tasking. By identifying this initial set of challenges, we aim to motivate researchers to investigate personalized web-tasking in the smart internet domain.

Acknowledgments

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) and Collaborative Research and Development program (CRDPJ 320529-04 and CRDPJ 356154-07), IBM Corporation, and Icesi University (Colombia).

References

[1] B. MacKay and C. Watters, "An examination of multisession web tasks," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 6, pp. 1183–1197, 2012.

[2] IEEE SERVICES 2013, "1st International Workshop on Personalized Web Tasking (PWT 2013)." [Online]. Available: http://www.servicescongress.org/2013/pwt.html

[3] M. Lynch, "The art of task simplification." *Modern Machine Shop*, vol. 75, no. 1, p. 148, 2002.

[4] M.-H. Sohn and J. R. Anderson, "Task preparation and task repetition: Two-component model of task switching," *Journal of Experimental Psychology*, 2001.

[5] J. J. Shi, D.-E. Lee, and E. Kuruku, "Task-based modeling method for construction business process modeling and automation," *Automation in Construction*, vol. 17, no. 5, pp. 633 – 640, 2008.

[6] N. Tsujiuchi, T. Koizumi, T. Hiroshima, H. Oshima, and R. Fujikura, "Simplification of task teaching for human-friendly small robot arm," in *Proc. 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2009, pp. 1400–1404.

[7] R. Aylward and J. Linkins, "Polio eradication: mobilizing and managing the human resources." *Bull World Health Organ*, vol. 83, no. 4, pp. 268–73, 2005.

[8] L. Cousin and J. S. Collofello, "A task-based approach to improving the software maintenance process," in *Proc. 1992 Conference on Software Maintenance*, 1992, pp. 118–126.

[9] X. Guoqi, P. Jun, Z. Xiaoyong, and L. Kuo-Chi, "A dynamic task planning based on task subcontracting and dynamic re-decomposition," in *Proc. 29th Chinese Control Conference (CCC)*, 2010, pp. 4518–4523.

[10] A. Erdem, W. L. Johnson, and S. Marsella, "Task oriented software understanding," in *Proc. 13th IEEE International Conference on Automated Software Engineering.*, 1998, pp. 230–239.

[11] M. Feldmann, G. Hubsch, T. Springer, and A. Schill, "Improving task-driven software development approaches for creating service-based interactive applications by using annotated web services," in *Proc. 5th International Conference on Next Generation Web Services Practices (NWESP 2009)*, 2009, pp. 94–97.

[12] N. Bencomo, G. Blair, R. France, F. Munoz, and C. Jeanneret, "Proc. 4th international workshop on models@run.time," in *4th International Workshop on Models@run.time (MODELS 2009)*, ser. LNCS, vol. 6002. Springer, 2010, pp. 173–188.

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