Automatic Filling Values to Services: A Road Map

Shaohua Wang, Ying Zou
Queen's University, Kingston, Ontario, Canada
e-mail: shaohua@cs.queensu.ca, ying.zou@queensu.ca

Joanna Ng, Tinny Ng
IBM CAS Research, Markham, Canada
e-mail: {jwng, tng}@ca.ibm.com

***Abstract*—It is common for users to explicitly or implicitly compose on-line services to accomplish daily tasks, such as shopping for a pair of shoes on-line. However, unnecessary and repetitive data typing into the services would negatively impact the user experience and decrease the efficiency of service composition. Recent studies have proposed several approaches to help users fill in values to services automatically. However, existing approaches suffer the following two drawbacks: 1) poor accuracy of filling values to services; 2) not designed for service composition. In this position paper, we first present the recent approaches improving the process of filling values to services. We then present our recent achievements and results of using our proposed approaches to help users fill values to services. Lastly, we discuss the various opportunities and challenges in the research field of filling values to services.**

***Keywords*-information reuse; service composition; input parameter value recommendation**

I. Introduction

With the rapid advance of web services technologies, users can conduct daily tasks on-line, such as shopping groceries. To accomplish such daily tasks, users have to visit websites and on-line services to compose services together repeatedly [1]. To invoke a service, users are required to enter values to input parameters of the service. For example, buying a pair of shoes on-line could involve multiple services. First, a user needs to search for a suitable pair of shoes on various e-commerce websites, such as Ebay[1], Amazon[2] and Hudson-Bay[3]. The user then needs to enter various search criteria, such as shoe color and size, to services for selecting shoes. Second, the user needs to enter payment information, such as shipping address and contact phone number, to purchase the selected pair of shoes. Such a simple task of buying a pair of shoes requires, at least, a dozen inputs. Filling values into services can be tedious, especially when the number of services is high and the previously entered values are required by similar services repetitively. There is an urgent need to call for approaches preventing the users from interruptions caused by unnecessary data entry to services during the service composition.

Recently, researchers from industry and academy have proposed a set of approaches (*e.g.*,[2][3][4][5]) to help users fill in values to input parameters of services. However, the existing approaches face the following three challenges:

- **Limited support of collecting and analyzing user inputs.** Over a period of time, a vast amount of user inputs can be collected and served as an excellent source for filling values to services automatically. Such a vast amount of information should be exploited for helping users fill in parameters. However, the existing approaches (*e.g.*, [2][3][5][11][12]) do not store and organize all of the available user inputs entered previously by the users. For example, Mozilla Autofill Addon [3] only support users to manually create their personal data and preferences records. Google Chrome Autofill [2] only collects basic information, such as name, credit card information and addresses.
- **Poor accuracy of filling values to input parameters of services.** Without a proper mechanism of storing user information, the existing approaches (*e.g.*, [4]) perform badly. Because they are likely to fill in wrong or out-of-context values to input parameters of services. With the rapid of growth of the population of mobile application users, it is even more frustrating for mobile application users to modify the wrongly filled values due to the restrictions in screen size and typing. Accurate filling or recommending values to parameters becomes a critical step to enhance the user experience.
- **Limitation in propagating user inputs across services.** During the service composition, the semantically related parameters should be linked, because the information can be propagated among the related parameters. For example, a user, living in Kingston, Canada, plans to attend an NBA Raptors[4] game in Toronto. The user could buy two tickets from the Raptors' website and two bus tickets from Megabus[5], a bus service. Both the Raptors' and Megabus' websites require the user to enter the number of tickets. The input parameters requiring the number of tickets from both websites should be linked. When one of the services is filled with a numeric value of the number of tickets, the value should be propagated to another service as well. However, none of the existing approaches (*e.g.*, [6][7][8]) is not designed for composed services.

1. http://www.ebay.com/
2. http://www.amazon.com/
3. http://www.thebay.com
4. http://www.nba.com/raptors
5. http://ca.megabus.com/

Recently, we developed a set of approaches to advance the process of auto-filling values to services.

Paper Organization. Section II introduces our recent achievements of assisting users in filling values to services automatically. Section III sketches the future research opportunities. Section IV concludes our paper.

II. Overview of Our Approaches

In this section, we introduce our recently developed approaches helping users fill in values automatically.

Characterizing parameters to improve the auto-filling process. To start filling an input parameter of a service, the characteristics of the input parameter should be understood. Various input parameters require to be filled under different conditions. We conducted an empirical study [13] on input parameters from 50 websites and 100 Android applications to investigate the characteristics of input parameters. We classify the collected input parameters into 4 categories:

- **Volatile Parameters:** have no change of values. For example, an on-line voucher number can only be redeemed once.
- **Short-time Parameters:** have values which change with a high frequency. For example, search criteria can be changed dynamically during a task of searching for clothes on different e-commerce websites.
- **Persistent Parameters:** require values which do not change over a long period of time. For example, the gender of a user may not change for a long time.
- **Context-aware Parameters:** needs context-dependent values, such as the user's current location.

Each parameter category should be collected, analyzed and reused differently. In [13], we propose an ontology-based approach to categorize parameters automatically and pre-fill values to input parameters of services. We compare our approach of filling values to parameters with a baseline approach proposed in [4]. The results show that our approach can improve the baseline approach significantly, i.e., on average 19% in terms of precision.

Leveraging user contexts and usage patterns into the auto-filling process. The user contexts and usage information play an important role in the process of filling values to services [15]. In [15], we propose a framework that automatically detects user inputs and builds user profiles. Then our framework organizes and analyzes the user inputs to generate patterns of user inputs for auto-filling. During the course of the auto-filling process, our approach links similar user interface (UI) components by clustering UI components into semantically related groups. Then, our approach pre-fills values to services by leveraging the user contexts and usage patterns. We conducted an empirical study on web interfaces [16] to compare our approach with Mozilla Firefox Autofill addon and Google Chrome Autofill. Our results show that our approach outperforms both Firefox and Chrome.

Maximizing the propagation of values among services. During a service composition, a set of semantically related parameters should be linked together to maximize the value propagation among services. Only using textual information to chain parameters is not adequate for accurately linking parameters. In [14], we propose a user input model which stores user inputs with more information, such as task information. To identify similar parameters and chain them together, we propose a parameter model that stores three aspects of a parameter: 1) textual information; 2) task information; 3) the information of neighbors of the parameter. Furthermore, our approach identifies 4 scenarios of supplying values to an input parameter of a service: 1) sharing values to the similar input parameters; 2) reusing values of output parameters to the input parameter; 3) using external data sources to assign values to input parameters; and 4) physical typing. Our automatic approach of filling values to services uses the input model and the parameter model for context-aware matching between user inputs and parameters during the service composition, and maximizes the value propagation among services by understanding the 4 scenarios. We conducted an empirical study on 640 WSDL services, 60 Restful Services and 50 web forms to evaluate the performance of our approach. On average, our approach reduces 44.5% and 29.25% of the number of input parameters on web forms and WSDL & Restful services, respectively.

Multi-user auto-filling process. In some cases, one user's previous inputs or usage information may not be sufficient to identify a proper value for pre-filling an input parameter due to the lack of history. To increase the chance of discovering a suitable value for a user, the user information of other users, who share similar activities with the user, is taken into account [17]. In [17], we extended the earlier work [14] in the following way:

- We propose an approach to identify similar users who could reuse their user inputs. We conducted an empirical study to verify our approach of identifying similar users. Our empirical results showed that our approach achieves an accuracy of 83%.
- We redefined the pre-filling model by adding the dimension of multi-user into the model. The extended approach can leverage and consume user inputs from multiple users. We conducted an empirical study on the effectiveness of our overall approach for reducing the number of input parameters by leveraging user inputs from different users. Our empirical results confirmed that including user inputs of other users similar to a user can help reduce the parameter filling for the user.

Ranking user inputs dynamically by learning user contexts. With the accumulation of user history, the interactions between users and parameters should be analyzed and learned automatically to improve the accuracy of filling values to parameters of services. In [18], we propose a ranking approach to rank a list of user inputs for filling in a parameter. We propose five ranking features derived from various types of user information that could affect the ranking of user inputs, such as user contexts. We adopted learning-to-rank [19], a supervised machine learning approach that automatically builds a ranking from training data, and integrated our ranking features into a state-of-the art ranking model named RankSVM [20]. Our approach analyzes and learns all the information of the past interactions between user inputs and input parameters to reuse user inputs for input parameter filling. Through an empirical study, we compared our approach with two baselines: Bayesian Belief Network [21] based ranking approach and frequency-based ranking approach. Our results showed that our approach outperforms the two baselines.

III. Road Ahead

Although we have advanced the process of filling values to services, the following aspects of the auto-filling process deserve more research efforts:

- **Heterogeneous structures of services.** Heterogeneity has a negative impact on auto-filling approaches, especially the varied web form structures. The current web form should be annotated with more meta-data to overcome the heterogeneity problem.
- **Wide range of machine learning algorithms and too much configuration.** A wide range of learning-to-rank (LTR) algorithms, or machine learning algorithms, have been proposed in other research fields, such as information retrieval and web page ranking. It has been proven that machine learning based ranking approaches outperform the conventional ranking models. We have only tested one LTR approach. More LTR approaches and different combinations of configuration variables should be tested to identify the best performing approaches under different circumstances.

IV. Conclusion

Unnecessary interruptions caused by repetitively typing the same information to services decreases the efficiency of service composition and negatively impacts the user experience. Automatically filling in values to services is a critical means to free users from repetitive typing. In this paper, we summarized the past achievements and discussed the drawbacks and limitations of the existing approaches. Then, we introduced our recent approaches on helping users fill in values to services. Lastly, we outlined possible research opportunities for auto-filling values to services.

References

[1] H. Xiao, Y. Zou, R. Tang, J. Ng and L. Nigul, *Ontology-driven Service Composition for End-Users*, Service Oriented Computing and Applications 5(3), pp.159-181,Springer,2011.

[2] Google Chrome Autofill forms,*https*://support.google.com/chrome/answer/142893. Accessed on Jan 1, 2015.

[3] Firefox Autofill Forms add-on, *https*://addons.mozilla.org /en-US/firefox/addon/autofill-forms. Accessed on Jan 1, 2015.

[4] S. Araujo, Q. Gao, E. Leonardi, J. Houben, *Carbon: domain-independent automatic web form filling*, Proc. of the Int'l Conference on Web Engineering, pp. 292-306, Vienna, 2010.

[5] S. Firmenich, V. Gaits, S. Gordillo, G. Rossi and M. Winckler, *Supporting Users Tasks with Personal Information Management and Web Forms Augmentation*, Proc. of Int'l Conference on Web Engineering, pp. 268-282, Berlin, July 23-27, 2012.

[6] M. Winckler, V. Gaits, D. Vo, F. Sergio and G. Rossi, *An Approach and Tool Support for Assisting Users to Fill-in Web Forms with Personal Information*, in Proceedings of the 29th ACM International Conference on Design of Communication (SIGDOC 2011), pp. 195-202, October 03-05, 2011, Pisa, Italy.

[7] S. Firmenich, M. Winckler, G. Rossi, S. Gordillo, *A Framework for Concern-Sensitive, Client-Side Adaptation*, in Proceedings of the 11th International Conference on Web Engineering (ICWE 2011) pp. 198C213. Springer, Heidelberg (2011).

[8] O. Diaz, I. Otaduy and G. Puente, *User-Driven Automation of Web Form Filling*, in Proceedings of the 13th International Conference on Web Engineering (ICWE 2013) pp. 171C185, July 8-12, 2013, Aalborg, North Denmark.

[9] R.L. Chambers, and Skinner, *Analysis of Survey Data, Wiley*, ISBN 0-471-89987-9. 2003

[10] B. Xiang, D. Jiang, J. Pei, X. Sun, E. Chen and H. Li, *Context-aware ranking in web search*, Proc. of the Int'l SIGIR conference on Research and development in information retrieval, pp.451-458, Geneva, Jul. 19-23, 2010.

[11] RoboForm, www.roboform.com. Accessed on Feb 15, 2015.

[12] LastPass, www.lastpass.com. Accessed on Feb 15, 2015.

[13] S. Wang, Y. Zou, B. Upadhyaya, I. Keivanloo, and J. Ng, *An Empirical Study on Categorizing User Input Parameters for User Inputs Reuse*, Proceedings of the 14th International Conference on Web Engineering (ICWE 2014), July 1 - 4, 2014, Toulouse, France.

[14] S. Wang, B. Upadhyaya, Y. Zou, I. Keivanloo, J. Ng and T. Ng, *Automatic Propagation of User Inputs in Service Composition for End-users*, Proc. of the IEEE Int'l Conference on Web Services, pp. 73-80, Anchorage, June 27-July 2 2014.

[15] S. Wang, Y. Zou, I. Keivanloo, B. Upadhyaya, J. Ng (in press), *An intelligent framework for auto-filling web forms from different web applications*, International Journal of Business Process Integration and Management.

[16] TEL-8 Query Interfaces, http://metaquerier.cs.uiuc.edu/repository/datasets/tel-8/.

[17] S. Wang, Y. Zou, I. Keivanloo, B. Upadhyaya, J. Ng (in press), *Automatic Reuse of User Inputs to Services among End-users in Service Composition*, IEEE Transactions on Services Computing. DOI: 10.1109/TSC.2014.2378278.

[18] S. Wang, Y. Zou, J. Ng and T. Ng, *Learning to Reuse User Inputs in Service Composition*, Proc. of the 22nd IEEE International Conference on Web Services, application track. June 27th - July 2 2015. New York City, USA.

[19] A. Trotman, *Learning to rank*, Information Retrieval Journal, 8(3): pp.359-381, 2005.

[20] T. Joachims, *Optimizing search engines using clickthrough data*. In KDD'02, 2002.

[21] M. A. P. de Cristo, P. P. Calado, M. D. L. da Silveira, I. Silva, R. Muntz, and B. Ribeiro-Neto, Bayesian Belief Networks for IR, Intl Journal of Approximate Reasoning, 34(2), 163-179.