Research Article

User-Centric Design for Mathematical Web Services

Adlin Sheeba^{1,2} and Chandrasekar Arumugam³

¹ Department of Computer Applications, St. Joseph's College of Engineering, Chennai 600119, India

² Research Scholar, Centre for Information Technology and Engineering, Manonmaniam Sundaranar University, Tirunelveli 627012, India

³ Department of Computer Science and Engineering, St. Joseph's College of Engineering, Chennai 600119, India

Correspondence should be addressed to Adlin Sheeba; adlinsheeba@yahoo.com

Received 30 October 2013; Revised 13 January 2014; Accepted 31 January 2014; Published 6 March 2014

Academic Editor: Francesco Bellotti

Copyright © 2014 A. Sheeba and C. Arumugam. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A web service is a programmatically available application logic exposed over the internet and it has attracted much attention in recent years with the rapid development of e-commerce. Very few web services exist in the field of mathematics. The aim of this paper is to seamlessly provide user-centric mathematical web services to the service requester. In particular, this paper focuses on mathematical web services for propositional logic and set theory which comes under discrete mathematics. A sophisticated user interface with virtual keyboard is created for accessing web services. Experimental results show that the web services and the created user interface are efficient and practical.

1. Introduction

Traditionally, the web has been conceived as a distributed source of information. The emergence of the web service technology permitted to extend it to a distributed source of functionality. Thus, web services connect computers and devices with each other using the internet to exchange and combine data in new ways. Many web services exist for various domains such as travel and e-commerce. Only few web services are available for computing mathematical functions such as calculator and so on. But to large extent, software is available for computing mathematical data.

Although software available for computing functions in mathematics has produced promising results that are certainly useful, it is not very much suitable for user-centric internet environment. First, software has to be installed in the system and users must be aware of the built-in functions available. Second, those built-in functions are rarely available as web services. Third, user interfaces with mathematical symbols are limited in use. To address these issues, in this paper, mathematical web services with learning material and user interface are created that will be more useful for the service requester to use and compute the desired services.

This work aims to provide MAThematical Web Services (MATWS) for Propositional logic (MATWS-P) and Set

theory (MATWS-S). Beyond the web services, our approach provides sophisticated user interface with virtual keyboard to assist the service requester.

The contribution of the work reported here is summarized as follows.

- (1) A framework is proposed for the creation of MATWS-P, MATWS-S, and user interface.
- (2) Various experiments have been done to measure the performance of web services and usage of user interface. Initial results show that the proposed strategy is efficient and practical.

The rest of the paper is organized as follows: Section 2 presents a survey on existing system, while Section 3 provides a conceptual framework for our approach. Sections 4 and 5 describe the creation of MATWS-P and MATWS-S, respectively. Section 6 presents the creation of user interface with virtual keyboard. In Section 7 results are discussed and Section 8 gives some concluding remarks.

2. Related Work

Boyatt and Joy [1] developed learning materials for mathematical logic that includes set theory and propositional logic.

TABLE 1: Sample web service operations.

WS1: Web Service: Principle Conjunctive Normal Form	
Operation: PCNF	
Input: expression	Data type: string
Output: lit	Data type: literal
WS2: Web Service: Symmetric Difference	
Operation: SymmetricDifference	
Input: operandVal	Data type: array of string
operandName	Data type: array of string
expression	Data type: string
Output: lit	Data type: literal

4. Web Services for Propositional Logic

Propositional logic is a system of symbolic logic using symbols [11] which has important applications in computer science such as to verify that computer programs produce the correct output for all possible input values and to show that algorithms always produce the correct results. Hence it is necessary to create web services for propositional logic concept such as negation, conjunction, disjunction, implication, tautology, equivalence, normal forms, and expression evaluation. The WSDL description of a tautology web service is given in Table 2.

Letters/sequence of letters/numbers is used to denote propositional variables, that is, variables that represent propositions. The truth value of a proposition is true (denoted by T) if it is a true proposition; the truth value of a proposition is false (denoted by F) otherwise.

Any given logical expression entered using virtual keyboard can be evaluated. The validity of the expression is verified to avoid wrong calculation of results. After verifying the validity of the logical expression, it is converted to postfix expression. During the conversion of logical expression into postfix expression, the operands are identified. The operand may be a letter or a sequence of letters or numbers. A count is made on the number of distinct operands (N) and the distinct operand names are stored in an array.

Partial truth table values are framed for identified operands. If there are N different operands, then there are 2^N different lines in the truth table and $N * 2^N$ truth table values. For each column in the table, $2^N/2$ true (T) values and $2^N/2$ false (F) values are computed to get all combinations. Then table head is created based on operator precedence and the operand values are substituted and evaluated.

User interface with mathematical symbols, identified operands, partial truth table values, table head, and the resultant truth table framed using HTML are shown in Figure 2 for the logical expression $OP1 \wedge (OP2 \rightarrow OP5) \leftrightarrow OP3$.

5. Web Services for Set Theory

In computer science, a set is an abstract data structure that can store certain values, without any particular order, and no repeated values and set theory are most commonly utilized

FIGURE 2: MATWS-P screenshot.

FIGURE 3: MATWS-S screenshot.

in science and mathematics fields as well as in computer and electrical engineering.

In the web-centric environment if the various set theory concepts such as union, intersection, symmetric difference, Cartesian product, and power set are available as web services, it would be useful for service requester to access and use them effectively. Also a virtual keyboard is needed for mathematical symbols which are shown in Figure 3.

The service requester may give any set expression. The N number of distinct operands in the expression is identified and N dynamic text boxes are created to get set elements. The expression is evaluated with operand names and operand values given by client in the dynamically created text box as input and the result is displayed as a literal.

TABLE 2: WSDL description of a tautology service.

<pre> <?xml version="1.0" encoding="utf-8"?> <wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tm="http://microsoft.com/wsdl/mime/textMatching" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://tempuri.org/" xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" targetNamespace="http://tempuri.org/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"> <wsdl:types> <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/"> <s:element name="Tautology"> <s:complexType> <s:sequence> <s:element minOccurs="0" maxOccurs="1" name="expression" type="s:string"/> </s:sequence> </s:complexType> </s:element> </s:sequence> <s:element name="TautologyResponse"> <s:complexType> <s:sequence> <s:element minOccurs="0" maxOccurs="1" name="TautologyResult" type="s:string"/> </s:sequence> </s:complexType> </s:element> </s:schema> </wsdl:types> <wsdl:message name="TautologySoapIn"> <wsdl:part name="parameters" element="tns:Tautology"/> </wsdl:message> <wsdl:message name="TautologySoapOut"> <wsdl:part name="parameters" element="tns:TautologyResponse"/> </wsdl:message> <wsdl:portType name="ServiceSoap"> <wsdl:operation name="Tautology"> <wsdl:input message="tns:TautologySoapIn"/> <wsdl:output message="tns:TautologySoapOut"/> </wsdl:operation> </wsdl:portType> </pre>	<pre> <wsdl:binding name="ServiceSoap" type="tns:ServiceSoap"> <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/> <wsdl:operation name="Tautology"> <soap:operation SoapAction="http://tempuri.org/Tautology" style="document"/> <wsdl:input> <soap:body use="literal"/> </wsdl:input> <wsdl:output> <soap:body use="literal"/> </wsdl:output> </wsdl:operation> </wsdl:binding> <wsdl:service name="Service"> <wsdl:port name="ServiceSoap" binding="tns:ServiceSoap"> <soap:address location="http://localhost:1031/ComputationalMathWebServices/Service.asmx"/> </wsdl:port> </wsdl:service> </wsdl:definitions> </pre>
--	--

TABLE 3: Percentage analysis.

Questions	Excellent		Very Good		Satisfactory		Very Poor		Unacceptable	
	MATWS	AEPLT	MATWS	AEPLT	MATWS	AEPLT	MATWS	AEPLT	MATWS	AEPLT
Easy to use										
Number	7	3	8	5	5	9	0	2	0	1
%	35	15	40	25	25	45	0	10	0	5
Buttons access										
Number	10	1	9	2	1	10	0	5	0	2
%	50	5	45	10	5	50	0	25	0	10
Efficiency										
Number	7	4	9	8	4	5	0	2	0	1
%	35	20	45	40	20	25	0	10	0	5
Visual appearance										
Number	11	2	7	2	2	10	0	4	0	2
%	55	10	35	10	10	50	0	20	0	10
Easy recovery from errors										
Number	10	2	7	4	3	9	0	3	0	2
%	50	10	35	20	15	45	0	15	0	10

6. Design of the User Interface

A user interface is the system by which service requesters interact with a machine. User interfaces exist for a wide range of systems and provide

- (i) input, allowing the users to manipulate a system
- (ii) output, allowing the system to indicate the effects of the users' manipulation.

The user interface is one of the most important parts of any program because it determines how easily you can make the program do what you want. A good program with a badly designed user interface has little value. The aim of this paper is to produce a user interface which makes it easy, efficient, and enjoyable for the service requester to use. The user interface has a virtual keyboard and some controls that can be created dynamically as and when needed.

6.1. Design of the Virtual Keyboard. A virtual keyboard is provided, which is an arrangement of buttons imitating a physical keyboard. Virtual keyboard has alphabets and mathematical symbols like $\lfloor, \wedge, \vee, \uparrow, \oplus$, and so forth; Pressed Key values are displayed in a label which uses session variable to hold its value even postback occurs. All keys are designed to use their click event to set its value in the session variable. It is believed that the virtual keyboard is a very powerful input component for mathematical expressions, which will prove useful even in the presence of a physical keyboard, where the buttons react to mouse clicks.

7. Results and Discussion

The user interface is evaluated by conducting a case study and the web services are evaluated by using web services performance metrics.

7.1. Evaluating Performance of the User Interface. To evaluate the performance of user interface, twenty-nine end-users are selected randomly to participate in the experiment and they do not have any preknowledge on web services. Nine end-users are not interested with the invocation of web services. Out of the remaining twenty end-users, ten are from mathematical background and are very familiar with Internet and ten have only the basic knowledge about Internet.

To evaluate the user satisfaction of the user interface, a user study [12] has been conducted. The twenty end-users are asked to invoke the web services using AEPLT and MATWS user interface.

After the end-users completed the invocation, a user feedback is obtained by asking them to choose options for the given questions to assess their usability with the user interface. The rational of each question is indicated in parentheses. The specific research questions posed were [13] the following.

- (1) Is it easy to operate? (Easy to use)
- (2) Are buttons of the right size and can be located easily? (Buttons access)
- (3) Is it efficient to use? (Efficiency)
- (4) Is it visually pleasing? (Visual appearance)
- (5) Does it provide easy recovery from errors? (Easy recovery from errors).

The survey is based on a five-point Likert scale [14] which provides five choices for each question to measure the degree of the answer, such as Excellent, Very Good, Satisfactory, Very Poor, and Unacceptable. Those answers are mapped to the score of 1 to 5. Five represents the most positive answer and 1 represents the most negative answer. Therefore quantitative data is obtained, so that the data can be analyzed with relative ease. Table 3 shows the percentage analysis of the obtained data. The percentage analysis in Figure 4 shows that MATWS

FIGURE 4: Percentage analysis.

TABLE 4: Descriptive statistical analysis.

Questions	Median		Mode		Range		Standard deviation	
	MATWS	AEPLT	MATWS	AEPLT	MATWS	AEPLT	MATWS	AEPLT
Easy to use	4	3	4	3	2	4	0.79	1.04
Buttons access	4.5	3	5	3	2	4	0.6	0.97
Efficiency	4	4	4	4	2	4	0.75	1.1
Visual appearance	5	3	5	3	2	4	0.69	1.07
Easy recovery from errors	4.5	3	5	3	2	4	0.75	1.1

FIGURE 5: Median.

FIGURE 6: Mode.

has highest percentage in Excellent and Very Good category compared to AEPLT. There is no Unacceptable and Very Poor feedback in MATWS. AEPLT is dominant in the Satisfactory range only.

Table 4 summarizes the quantitative information of the feedback received from the service requesters. The graph in Figure 5 shows that the middle value in MATWS is 4 and above which denotes that more number of feedbacks are in the Very Good and Excellent category. In AEPLT, it is 3 which

denotes that feedbacks are in both the upper and lower level of the five point Likert scale equally.

The repetition of the values 4 and 5 most often in MATWS indicates that feedbacks are in the upper level of five point scale as shown in Figure 6. In AEPLT, it is 3 which shows that interface has been rated Satisfactory by most users.

The range 2 in MATWS indicates that the lower level is 3 which means that feedbacks are not in the Very Poor and Unacceptable category. But in AEPLT as shown in Figure 7, it

FIGURE 7: Range.

FIGURE 8: Standard deviation.

is 4 which indicates that the lower level is 1 which shows the users have rated in the lower level of the Likert scale also.

The mean for MATWS is 4 and above. Hence a standard deviation less than one as shown in Figure 8 indicates that the feedback is not in the Very Poor and Unacceptable range. For AEPLT, the mean is from 2 to 4 and standard deviation greater than 1 indicates that the feedback is scattered in all levels.

7.2. Performance of Web Services. Service providers need to know the resource demands of the web services at different workloads to compete with other service providers offering the same or similar service. Clients need to know the response time, throughput of the web services. Hence stress testing is performed on the web services by allowing more number of agents to access the services simultaneously. Web services have been deployed in the local host with Intel Core 2 Duo Processor and 2GB RAM. The Pylot Web service performance tool is used to analyse the performance of web services.

The following web service performance metrics are considered for evaluating the web services.

(i) *Response Time.* Response time is defined as the interval from when a user initiates a request to the instant at which the response is received by the application. To calculate the response time, stress testing is performed on the server side by allowing different number of agents 10, 20, 30, 40, and 50 to access services simultaneously. The resultant graph in

FIGURE 9: Response time-MATWS versus AEPLT.

FIGURE 10: Throughput-MATWS versus AEPLT.

Figure 9 shows that the response time increases linearly and MATWS survives more load than AEPLT.

(ii) *Throughput.* Throughput is calculated as the total number of invocations for a given period of time. The graph in Figure 10 shows that, as time increases, more number of requests is handled by MATWS compared to AEPLT.

8. Conclusion

In the past few years, the web communities have undergone a vast change to the user-centric environment and we argue that MATWS for propositional logic and set theory with user interface can bring more and more benefits to the service requester. The results indicate that the user interface using virtual keyboard help service requester for easy participation and rich user interaction. Also it has been proved that the created web services are efficient by conducting stress testing on the server side and evaluating the web service performance metrics.

Future work involves creation of more number of web services in other areas of discrete mathematics such as algebra and number theory. The existing discovery techniques are

more suitable for e-commerce based applications and hence it has been planned to create an efficient framework for the discovery of mathematical web services.

Conflict of Interests

The authors Adlin Sheeba and Chandrasekar Arumugam declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] R. C. Boyatt and M. S. Joy, "Developing an ontology of mathematical logic," *Scientific Journal of the Technical University of Civil Engineering Bucharest*, vol. 6, pp. 6–11, 2010.
- [2] R. Thomas Gruber and G. R. Olsen, "An ontology for engineering mathematics," in *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, 1994.
- [3] J. Mbale, "Automated Education Propositional Logic Tool (AEPLT): used for computation in discrete mathematics," *International Journal of Logic and Computation*, vol. 3, no. 1, pp. 27–33, 2012.
- [4] O. Gasquet, F. Schwarzentruher, and M. Strecker, "Satoulouse: the computational power of propositional logic shown to beginners," in *Tools for Teaching Logic*, vol. 6680 of *Lecture Notes in Computer Science*, pp. 77–84, 2011.
- [5] Lehmann and Heiko, XploRe Quantlet Client: Web Service for Mathematical and Statistical Computing, Discussion papers of interdisciplinary research project 373, no. 2003, 23, <http://nbn-resolving.de/urn:nbn:de:kobv:11-10050233>.
- [6] <http://www.mathworks.in/products/matlab/>.
- [7] <http://www.wolfram.com/mathematica/>.
- [8] <http://www.gap-system.org/>.
- [9] <http://www.reduce-algebra.com/>.
- [10] <http://www-01.ibm.com/software/analytics/spss/>.
- [11] Wikipedians, B. Data Structures. PediaPress, <http://books.google.co.in/books?id=XhYrw6-4OVgC>.
- [12] H. Xiao, Y. Zou, R. Tang, J. Ng, and L. Nigul, "Ontology-driven service composition for end-users," *Service Oriented Computing and Applications*, vol. 5, no. 3, pp. 159–181, 2011.
- [13] A. Bahrami, *Object Oriented Analysis and Design*, McGraw-Hill, International Editions, 2005.
- [14] S. A. McLeod, "Likert Scale—Simply Psychology," 2008, <http://www.simplypsychology.org/likert-scale.html>.