Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 4, Issue, 06, pp.077-084, June, 2012

INTERNATIONAL JOURNAL
OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

AN EFFICIENT APPROACH FOR DYNAMIC COMPOSITION OF SERVICE, QUERY OPTIMIZATION AND PRIVACY PRESERVATION USING WSMS

Mr. D. Palanikkumar Durai and Thirunavukkarasu M.E (Phd), M. Lakshmi and G. Nathiya

Department Of Computer Science and Engineering, Anna University Of Technology, Coimbatore

ARTICLE INFO

Article History:
Received 14th March, 2012
Received in revised form
11th April, 2012
Accepted 17th May, 2012
Published online 30th June, 2012

Key words:
Web service management system;
Service composition;
Service optimization; Privacy preservation.

ABSTRACT

A customized service to citizens can be provided using service-oriented architecture. Web Service Management System (WSMS) is a comprehensive framework that provides an integrated view on the management of web service. It helps in providing a service-centric framework to deliver services to citizens. The proposed WSMS enable citizens to select their needed services. It focuses on the following key components: service composition, service optimization and service privacy preservation. The service composition provides support for combining various services. It also enables the customized delivery of services. The service optimization selects the best needed service to fulfill citizens' service requirements. The privacy preservation provides security support.

INTRODUCTION

Service-Oriented Computing (SOC) a new paradigm, which moves the field of computing from a data-centric to a service-centric view. Service-oriented computing is an emerging cross-disciplinary paradigm for distributed computing, which is changing the way software applications are designed, delivered and consumed. At the heart of service-oriented computing are services that provide autonomous, platform-independent, computational elements that can be described, published, discovered, orchestrated and programmed using standard protocols to build networks of collaborating applications distributed within and across organizational boundaries. Service-oriented architecture (SOA), which is a logical way of designing a software system to provide services to either end-user applications or to other services distributed over a network, via published and discoverable interfaces.The flagship technology and embodiment of service-oriented computing is web services technology. The introduction of web services has been key to the adoption of service-oriented computing. A web service can be defined as a functionality that can be programmatically accessible via the web. A fundamental objective of web services is to enable interoperability among different software and data applications running on a variety of platforms [3]. Web services can be adapted to access data and applications across the web. Web services also provide an efficient vehicle for users to access the functionalities available on the web. The development of web services has so far mostly been the result of standardization bodies usually operating on a consensus basis and driven by market considerations [2]. In this context, innovation and long-term deployment issues are not usually of primary concern. Because of the global nature of the web, the standardization process has so far been very fragmented, leading to competing and potentially incompatible web service standards. Governments and commercial organization have meanwhile invested very heavily in web services technologies. These investments have resulted in a fast-growing number of web services being made available. The prevalent business model will, in all likelihood, include a autonomous and competing communities of web service providers vying for consumers attentions. It is, however, important that these investments produce the expected results and soon for the area to really make an impact. One-key impediment in web service technology has been the lack of systematic methodology for managing the entire life cycle of web services that would include delivering, selecting, optimizing and composing services [7]. This needs to take place within a secure, trustworthy, and privacy protecting environment.

In summary, a WSMS is a comprehensive framework that provides an integrated view on the management of web services including automatic service composition, service query optimization, and service privacy preservation. Service composition deals with the automatic selection and integration of individual web services to provide value-added services. Service optimizations are concerned with the ability to provide an optimization framework suited for web services [9]. Privacy preservation ensures that interactions with web services can be conducted secure fashion. The remaining sections can be organized as follows: Section II presents related work. Section III specifies problem definition and overview of proposed WSMS. Section IV, describes the implementation of WSMS. Section V, results and discussion. Section VI conclusion.

***Corresponding author:** palanikkumard@gmail.com, banu175@gmail.com, smartnathiya1@gmail.com

RELATED WORK

Francisco Curbera *et al* [2] proposed BPEL4WS.It can be described in two ways as *Executable* processes, which represents actual behavior of a participant and abstract processes, as message exchange behavior of each of the parties involved in the protocol. BPEL4WS includes an unresolved issue as security, transaction handling and service reliability. Bouguettaya *et al* [3] developed an E-Government service. The main objective of e-government is to improve government-citizen interactions resulting in a significant reduction in the use of paper, mailing, shipping activities and consequently, improving the services provided to citizens. Casati *et al* [5] introduces an *Eflow* process. *EFlow* is a system that supports the specification, enactment, and management of *composite* e-services. The service processes should be able to adapt changes in the environment and needs of different customers with minimal or no user intervention. Chun *et al* [7] developed a dynamic workflows composition for e-government service delivery. The dynamic composition represents how dynamically composing the inter-agency service as well as the related information, by identifying, integrating and coordinating individual components. It also achieves a customized generation of workflows using Condition-Action Rule Base. Dan. Wu *et al* [8] developed an automating Daml-S. The main purpose of DAML-S is to support effective automation of various Web services related activities such as service discovery, composition, execution, and monitoring. SHOP2 is an Hierarchical Task Network (HTN) planner for Process Model. It implemented a system which completely plans over set of DAML-S descriptions using a SHOP2 planner, and then executes the resulting plans over the Web. Hellerstein *et al* [10] introduces an Optimizing Queries with Expensive Predicates. The expensive predicates in a query plan should be moved to reduce the total cost of the plan. The newly enhanced query optimizer specifies an correctly optimizing queries with expensive predicates, which produces plans that are orders of magnitude and faster than plans generated by a traditional query optimizer. Francisco Curbera *et al* [11] introduces Web Services Flow Language (WSFL). WSFL involves two types of Web Services compositions as flow models, represent the description of a business process. Global models, as description of the overall partner interactions. The interaction can be hierarchical and peer-to-peer interactions. Liu *et al* [12] proposed SOE.Service Oriented Enterprise (SOE) provides an flexible platform for managing top-down changes. SOE combines multiple Web services together to provide a value-added service. SOE's schema proposed a change model as a guide to react to changes. Yang *et al* [16] developed a scheme in Organizing and Accessing Web Services on Air. Mobile commerce (m-commerce) refers to the process of conducting business using wireless devices and communications. A multichannel model is defined to carry information about m-services. B+ tree, signature indexing, and hashing techniques used to enable efficient access to wireless channels. Srivastava *et al* [20] proposed an Query Optimization Over Web Services. The query optimization over web services tackles WSMS problem in optimizing the query for Select-Project-Join queries spanning over multiple web services. It results in minimizing the query's total running time. It also provide an algorithm for determining the optimal granularity of data "chunks" to be used for each web service call.

PROBLEM DEFINITION

The existing system focus on dealing with changes of services.It doesn't focus on relationships between web services or customized composite web service generation.One-key impediment has been the lack of systematic methodology for managing the entire life cycle of web service. There is no composition of service in traditional techniques because the combined web services may not be compatible with each other in terms of functionalities.There is no maintenance of survey about the user. The existing system does not deals with the privacy preservation of user's data. There is no value-added service in the existing technique. It does not specify whether a requested service is provided to appropriate user.

Web service standards

The key standards of web service providing related functionalities that can be adopted to dealing with the changes of services are as follows:

WSDL (Web service Description Language)

WSDL is an XML format language used to describe network services. It can be defined as a set of endpoints that operates on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then it can be bound as concrete network protocol and message format to define an endpoint. The related concrete endpoints are combined into abstract endpoints (services). WSDL is an extensible, which allows description of endpoints and their messages to communicate regardless of what message formats or network protocols are used.

SOAP (Simple Object Access Protocol)

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol consists of three parts as: first part as an envelope that defines a framework for describing what is in a message and how to process it, second as a set of encoding rules for expressing instances of application-defined data types, and third part as a convention for representing remote procedure calls and responses.

UDDI (Universal Description, Discovery, and Integration)

UDDI is a platform-independent framework. It can be defined as describing services, discovering businesses, and integrating business services by the use of Internet. It serves as a directory for storing information about the web services. UDDI has two functions: It is a SOAP-based protocol that defines how clients communicate with UDDI registries. It is a particular set of global replicated registries.

BPEL (Business Process Execution Language)

BPEL is a notation for describing business process behavior based on Web Services. It can be defined in two ways as *Executable* processes, which represent actual behavior of a participant and *abstract* processes, as message exchange behavior of each of the parties involved in the protocol. BPEL provides an interoperable integration model by facilitating the expansion of automated process integration in terms of both the intra-corporate and the business-to-business spaces.

OVERVIEW OF WSMS

The problem in existing system represent that it does not specify whether requested service is delivered to the appropriate user. To overcome this drawback, proposed system maintains a WSMS is a comprehensive framework which provides an integrated view on the management of web service.The architecture of WSMS is shown in fig 1.With many enterprises moving towards a service-oriented architecture, web services are becoming a most popular and standard method of sharing data and functionality among loosely coupled systems. The proposed general-purpose Web Service Management System (WSMS) prototype enables querying an multiple web services in a unified and transparent fashion through a SQL-like interface.

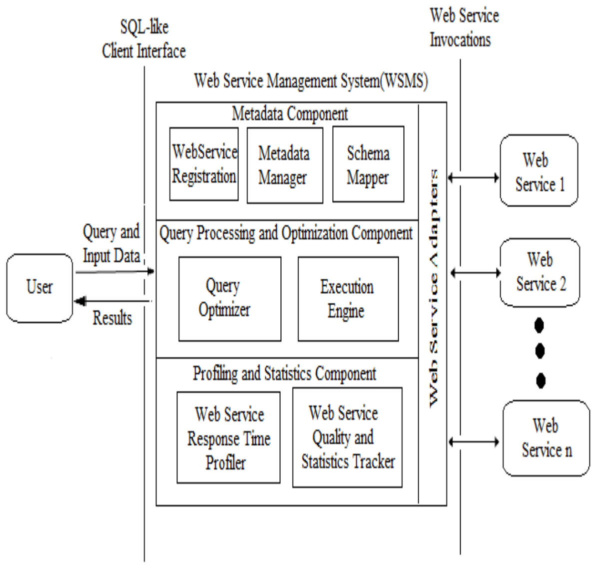

Fig 1.The architecture of WSMS

WSMS consists of three major components as:

Metadata Component: The job of this component is to: deals with the registration of new web services, manages the metadata informations and mapping their schemas to the global view that are exported in the client interface by using schema mapper.

Query Processing and Optimization Component: This component has two functions: The first function takes a user-input query and finds the optimal query plan. The query plan defines the most efficient way to process the query by issuing calls to the relevant web services. The second function i.e. execution engine then executes this query plan by making actual web service calls and also by efficiently managing the data exchange between the WSMS and the various web services.

Profiling and Statistics Component: This component profiles the web services for their response time characteristics. It also maintains relevant statistics over the web service data. The query optimizer makes use of these statistics for making an optimized decisions.

WSMS FOR SERVICE DELIVERY

Dynamic Service Composition: The service composition component enables customized delivery of services to citizens. Since it is impossible to predict all possible service requests, WSMS needs to compose web services in a dynamic fashion on demand. This can be done with the help of dynamic service composition mechanism.

Composability Model for Web Services: A major issue when defining a composite service is whether its component services are composable. A set of composability rules is generated to compare syntactic properties of web services. These rules include:

Mode composability: That compares operation modes. Consider two web services are communicating through operations that are mode composable, ensure that at least one of the protocols expected by a web service must be supported by the other. For web service interactions to take place, operations at client and server sides must have "dual" modes.

Binding composability: That compares the binding protocols of interacting services. For example, it would be difficult for service to receive a message in MIME protocol which format as HTTP protocol message. It checks at least one of the protocols expected by a web service must be supported by the other.

Message composability: That compares the number of message parameters, their data types, business roles, and units. The idea is to check that each input of an operation is data type compatible with the output of the other operation. The input's unit and business role should be the same as the output's unit and business role, respectively. This means that the parameters of each input message map to all or some of the parameters contained in the output message of the other operation.

Composition Soundness

Composition soundness tests whether composite services are sound. By sound, we mean that the way component services are composed provides an added value. We introduce the notion of composition templates [7].A composition template is associated with each composite service and gives the general structure of that service.

Composition templates: Are used to compare the values added by different compositions.

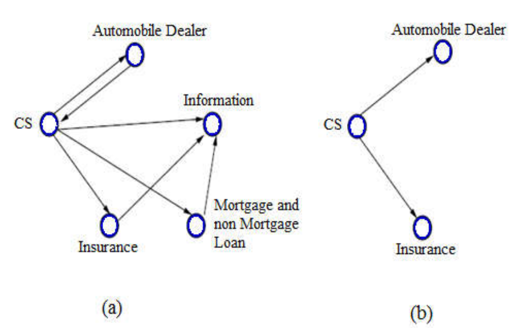

Fig 2. Example for Composition Template

It is modelled by a directed graph (V, E) where V is a set of service category names and E is a set of edges. A special vertex corresponds to the composite service and has the special value "CS".An edge (vi, vj) $\in$E means that a service of category name v*i* precedes a service of category name v*j*. For example, consider the template depicted in Fig.2b. This template is a subgraph of the template depicted in Fig.2a . This means that the second composite service would provide a subset of the functionalities offered by the first one. For example, it does not provide "financing" operations since it does not outsource from a "mortgage and nonmortgage loan" service.

Automatic Composition of Web Services

A matchmaking algorithm is proposed for the automatic composition of web services (Table I). The algorithm uses the composability rules to generate composition plans. By composition plan, we refer to the list of component services and their interactions with each other to form the composite service. Since the number of generated plans may be large, composers have the possibility to control the number of generated plans through the nb_requested_plans input.

For every pair of operations (opik, opjl), the algorithm checks mode composability (line 10) and message composability (line 12). The matchmaking algorithm uses the following functions to check composability: *purpose compatible()*,*category compatible()*, *quality composable()*, *message composable()*, and *sound()*.The functions *purpose compatible()* and *category compatible()* return *true* or *false* depending on whether a composite service operation has a purpose or category compatible with the purpose or category of a component service operation. *quality composable ()* returns *true* if a composite service operation is qualitatively composable with a component service operation. The message_compatible () function returns true or false depending on whether a message Mi is message compatible with Mj. The sound () function checks the soundness of the generated plan. Once a template has been computed for the generated plan, it is compared with stored templates. Note that as stated previously, composition plans are returned to composers even if they are not sound. Each iteration of the while statement generates a composition plan. The algorithm stops until the user-specified number of plans are generated.

Table I. Matchmaking Algorithm

(01) Input: *WSi, repository, nb requested plans {*
(02) *nb generated plans = 0*
(03) *matched = ∅*
(04) while *nb generated plans ≤ nb requested plans* do
(05) *{ plan = ∅*
(06) for each operation *opik* $\in$*Oi* do
(07) *{ found = false*
(08) for each service *WSj* from repository | *category compatible (Cik, Cj)*
(09) and *purpose compatible (Pik, Pj)* and *(Bi ∩ Bj = ∅)* do
(10) *{for* each operation *opjl* $\in$*Oj* | *(modeik and modejl are dual)* and *(opjl* $\in$*matched)*
(11) if *purpose compatible (Pik, Pjl)* and *category compatible (Cik, Cjl)* and *quality composable (opik, opjl)*
(12) and *message composable (inik, outjl)* and *message composable (injl, outik)*
(13) then *{found = true*
(14) *plan = plan* $\cup$*{(opik, opjl)}*
(15) *matched = matched* $\cup$*{opjl}*
(16) break *}*
(17) if *found* then break
(18) *}* /* *for* in line (08) */
(19) if $\neg$*found*
(20) then *{output ("no matchmaking for", opik)*
(21) break}
(22) *}* /* *for* in line (06) */
(23) if $\neg$*found* then break
(24) else if *sound (plan)*
(25) then output *(plan, ST)* /* ST is a Stored Template */
(26) else if *relevant (plan, τ relevance)* and *complete (plan, τ completness)*
(27) then output *(plan, ST, τ relevance, τ completness)* /* Test for QoC parameters */
(28) else output *(plan, "not sound", τ relevance, τ completness)*
(29) *nb generated plans = nb generated plans + 1*
(30) } /* *while* in line (04) */}

Service Optimization in WSMS

The purpose of the service optimization component is to select the best service(s) or their compositions to fulfil citizens' service requirements. We adopt a two- phase optimization strategy. In the first phase, the query optimizer transforms an algebraic expression into the most efficient one. It then performs QOWS optimization in the second phase to select the service execution plan with the best quality.

QoWS for SEPs (Service Execution Plan)

A Service Execution Plan (SEP) consists of a set of service operations from the retrieved service instances. It specifies the order to execute the service operations.

Table II. QoWS for a Service Execution Plan

QoWS Parameter	Aggregation Function
Latency	$\sum_{i=1}^{n} lat(op_i)$
Reliability	$\prod_{i=1}^{n} rel(op_i)$
Availability	$\prod_{i=1}^{n} av(op_i)$
Fee	$\sum_{i=1}^{n} fee(op_i)$
Reputation	$\frac{1}{n}\sum_{i=1}^{n} rep(op_i)$

Now, we need to compute the *QoWS* parameters for the entire service execution plan that may contain multiple service operations. Based on the *QoWS*, we define a set of aggregation functions to compute *QoWS* of service execution plans, as shown in Table II. Thus the quality of a SEP can be characterized as a vector of QoWS, *Quality* (*SEPi*) = (*lat*(*SEPi*), *rel*(*SEPi*), *av*(*SEPi*), *fee*(*SEPi*), *rep*(*SEPi*)).

lat (latency) and *fee* (usage fee) take scalar values (R^+). *av* (availability) and *rel* (reliability) represent probability values (a real value between 0 and 1). Finally, *rep* (reputation) ranges over the interval [0,5].

Score Function

We define a score function to compute a scalar value out of the QoWS vector of the SEPs. This can facilitate the comparison of the quality of the SEPs. Since users may have preferences over how their queries are answered, they may specify the relative importance of *QoWS* parameters. We assign *weights*, ranging from 0 to 1, to each *QoWS* parameter to reflect the level of importance. We use the following score function S*F* to evaluate the quality of the service execution plans. By using the score function, the QoWS optimization is to find the execution plan with the maximum score.

$$SF = \left(\sum_{Q_i \in neg} WG_i \frac{Q_i^{max} - Q_i}{Q_i^{max} - Q_i^{min}} + \sum_{Q_i \in pos} WG_i \frac{Q_i - Q_i^{min}}{Q_i^{max} - Q_i^{min}} \right) \quad (1)$$

where *neg* and *pos* are the sets of negative and positive *QoWS* respectively. In negative (resp. positive) parameters, the higher (resp. lower) the value, the worse is the quality. *WGi* are weights assigned by users to each parameter. *Qi* is the value of the *i*th *QoWS* of the service execution plan obtained through the aggregate functions from Table II. Qi^{max} is the maximum value for the *i*th *QoWS* parameter for all potential service execution plans and Qi^{min} is the minimum. These two values can be computed by considering the operations from service instances with the highest and lowest values for the *i*th *QoWS*.

Best plan Composition

We present two approaches for finding the best plan: exhaustive search and greedy search [21]. The exhaustive search enumerates the entire space of composition plans. Suppose that a service request needs to access M web services (e.g., transportation, congregate meals, etc.). The composition plans are generated by composing M corresponding services. We assume that there are N service providers that compete to offer each service. Therefore, the complexity of the exhaustive search is O (N^M), which is exponential. The greedy search achieves the polynomial complexity by using a divide-and-conquer strategy. It generates an optimal sub plan from each service through local search. It then combines these sub plans to form the final plan. In order to apply the divide-and-conquer strategy, we take logarithms on the aggregation functions for reliability and availability. Specifically,

$$\text{Reliability} = \sum_{i=1}^{n} \log\left(rel(op_i)\right) \quad (2)$$

$$\text{Availability} = \sum_{i=1}^{n} \log\left(av(op_i)\right) \quad (3)$$

Thus, the greedy search has a complexity of O (N×M).

Privacy Preservation

While providing integrated services to citizens, redundancy must be eliminated and management effort and cost should be reduced. The agencies may have different (and possibly conflicting) privacy policies. Moreover, citizens typically have different preferences with respect to their privacy. For example, citizen may typically do not want to release their income or health-related information. The challenge is to achieve seamless interoperability between different agencies without violating citizens' privacy.

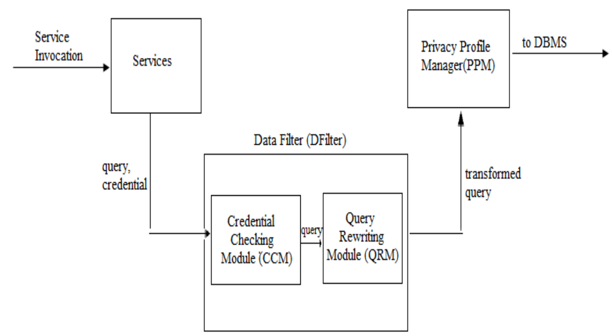

Fig. 3. Data filtering mechanism for privacy preservation

In WSMS, users submit their service requests along with their Digital Privacy Credentials (DPCs).When a service receives a request from a given user (see Fig 3), it first checks that user has the necessary credentials to access the requested operation according to its privacy policy. The DFilter is composed of two modules: the Credential Checking Module (CCM) and the Query Rewriting Module (QRM). The CCM uses the credential received with the query to determine whether the service requester is authorized to access the requested information. If the credential authorizes access to only part of the requested information, the QRM redacts the query so that all the privacy constraints are enforced. The PPM is responsible for enforcing privacy at a finer granularity than that enforced by the CCM. The PPM is a translation of the content based privacy model in that it implements the privacy preferences of individual citizens. It maintains a repository of privacy profiles that store individual privacy preferences. Requests made by citizens to update their privacy profiles are also handled by the PPM.

IMPLEMENTATION METHODOLOGY

The proposed system uses the Web Service Management System (WSMS).The WSMS specify how efficiently delivering government services to citizens with service centric framework. WSMS is designed to automatically deliver services that are customized for citizens. It mainly focus on three components (as shown in fig 4) as: service composition, service optimization and privacy preservation. It also uses matchmaking algorithm for the automatic composition of web service.The proposed system also deals with service trust and change management.

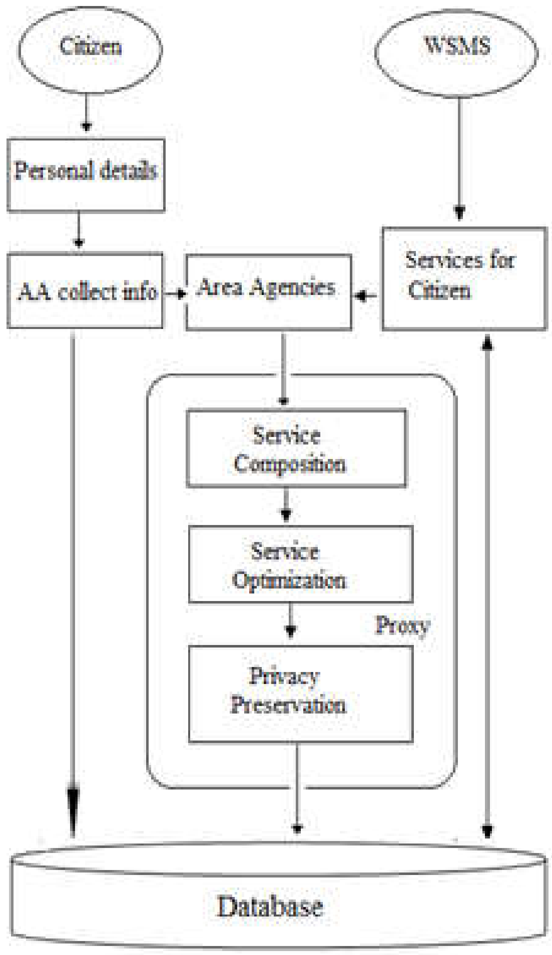

Fig. 4. System Architecture

We organize WSMS into five modules as: user interaction management, service management, area agency, proxy as a server and reports. These modules can be implemented using asp.net. The user interaction management is the users who are going to retrieve the benefits from the WSMS database. In before they have to enter the personal information to the data server to get balanced service. The service management module specifies the central storage and authority for the process. The user details will be stored by this module. Citizen service details are also stored in the WSMS and the proxy will transfer the data to the particular user. The area agency module is responsible for identifying what the citizen need by the web service. The proxy will act as the server to reduce the server work load. Finally, reports will maintain the entire process activities. Note that the system can easily be extended by adding other databases. New services can also be included with minimum overhead by registering themselves with the UDDI service registry.

RESULTS AND DISCUSSION

In Web Service Management, each individual user can identify and get their needed service from the different service providers. The three techniques such as service composition, service optimization and privacy preservation can be used as key components in web service management system. Service composition is used in such a way that it should combine various services. It checks whether combined services provides a value added service by composition soundness. The service composition also performs mode, binding and message composibility for dynamic composition. An automatic service composition can also be done with matchmaking algorithm. Service optimization is used to select the best service to fulfill citizens requirement. The service optimization uses techniques called exhaustive and greedy searching techniques for finding the best composition plan. Privacy preservation provides security by making use of digital privacy credential technique. The administrator will maintain the entire process and manages the user and service management interactions. The reports are maintained and updated in the database. For performance evaluation, we first evaluate the time for generating composition plans (Fig.5). We consider three execution times. The first execution time includes mode, binding, and operation semantics composability. The second execution time corresponds to message composability. The last execution time corresponds to composition soundness. The results show that most of the time is spent on checking message composability (Fig.5).

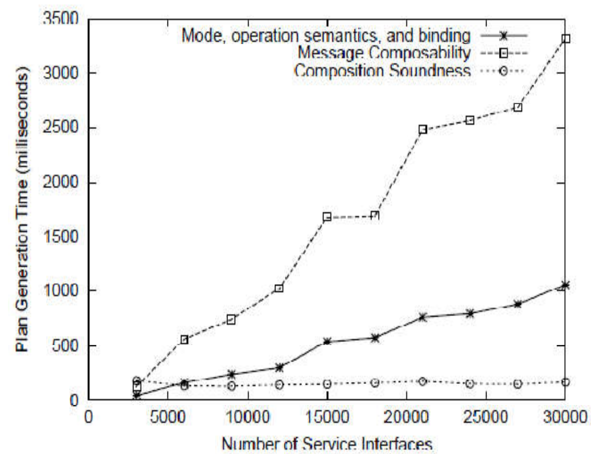

Fig. 5. Plan generation time

Indeed, the second time requires comparing the parameters of each composite service operation with the parameters of each operation of a component service. In contrast, the first time includes comparing operation modes, categories, purposes, and binding protocols that are less CPU-intensive. Composition soundness is the property that consumes the least generation time. Indeed, syntactic and operation semantics composability compare composite services with all service interfaces in the business registry. Composition soundness also compares generated plans with stored templates whose number is much smaller than the number of service interfaces (100–500 templates vs.3,000–30,000 interfaces). This also explains the relative stability of the composition soundness time. Note that the plan generation time shown in Fig.5 does not consider access time to UDDI business registry and stored templates repository. We also assess the impact of QoC parameters on the number of generated plans (Fig.6). We particularly consider the composition completeness (CC) ratio. We conducted experiments for CC = 33% and CC= 66%.The results show that the number of generated plans is higher for CC = 33% (Fig.6). Indeed, for CC = 33%, plans are generated if at least 33% of composer operations are composable with component operations.

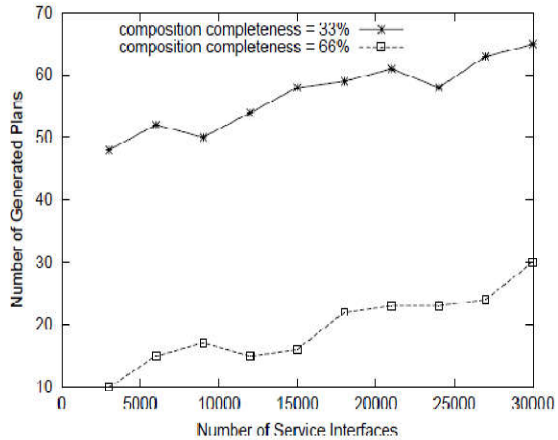

Fig. 6. Number of generated plans

However, for CC = 66%, plans are generated if at least 66% of composer operations are composable with component operations. The results also show that the number of generated plans for CC = 33% is, on average, greater than 50 (i.e., minimum number of requested plans).This means that plans are generated for almost every specified composite service. However, for CC = 66%, the number of generated plans is at most equal to 30 (i.e., less than the minimum number of requested plans).This means that for some composite services, no plan has been generated. This confirms our expectation about the impact of CC on the number of generated plans; a relatively low value of CC generates more plans, each plan containing a small number of composable operations. Thus a high value of this ratio generates a smaller number of plans, each plan having more composable operations.

M-Services

The significant advances in wireless technologies open a promising application area for web services. As a key extension of web services, mobile services (m-services) cater for the increasing population of mobile users.In the mobile web services domain, the resource constrained mobile devices are used as both web service clients and providers, still preserving the basic web services architecture in the wireless environments. Specifically, m-services are a special set of web services that can be accessible by mobile hosts over wireless networks [16], [17]. They work together with mobile devices to offer anytime/ anywhere accessible services. In contrast to web services with wired infrastructures, m-services are more suitable for time and location critical tasks [22]. For instance, a stock quote service can help users make quick response by providing timely quote prices. However, users may prefer desktops to cell phones or personal digital assistants (PDAs) when carefully preparing a travel package for vacations.

The mobile environment poses great challenges for providing and consuming m-services. Mobile devices have low CPU and memory capacities, limited power supply, small screen size, and restricted input mechanisms. Wireless networks are limited by their small bandwidth. They also suffer from link outages, which result in temporary unavailability. These limitations hinder existing web service technologies from directly working with m-services. For instance, the limited bandwidth may not be enough to convey SOAP messages [17]. In addition, SOAP is expensive for mobile hosts in terms of both power consumption and waiting time. In mobile environment, users' context, such as location and activity, may change rapidly. M-services need to track these changes and provide context-aware functionalities. However, service description techniques, such as WSDL, have not provided support to model context. UDDI enables service discovery in the web service framework. However, the multiple costly round trips required by UDDI lookup are troublesome for m-service discovery [17]. The frequent unavailability of wireless network may cause failures in service discovery processes.

CONCLUSION

A service-centric framework, WSMS, is used to provide services to citizens. WSMS manages the life cycle of social services. A set of key service components in WSMS, consisting of service composition, optimization, and privacy preservation achieves seamless cooperation to provide prompt and customized service to citizen's.. In particular, WSMS provides mechanisms that dynamically compose services, select services based on their quality attributes, and ensure the privacy of citizens when requesting and receiving services. The proposed framework also helps in identifying individual needs and to fulfill the requirements of the user. The performance of service composability can be evaluated as time for generating the plan and number of generated plans. One of our ongoing work is to consider XML Schema's user-defined data types and define data type compatibility among message parameters in terms of XML Schema inference. An important next step is to extend algorithms to allow different input tuples to follow different plans, leading to even higher overall performance. Extending the query optimization algorithms to incorporate caching to significantly speedup query processing is an another important direction for future work.

REFERENCES

[1] J.L. Ambite et al., 2002. "Data Integration and Access," Advances in Digital Government: Technology, Human Factors, and Policy, A.K. Elmagarmid and W.J. McIver, eds., pp. 85-106, Kluwer Academic Publishers.

[2] Francisco Curbera, Tony Andrews et al., 2003. Business Process Execution Language for Web Services (BPEL4WS), http://xml.coverpages.org/bpel4ws.html.

[3] A. Bouguettaya, A. Rezgui, B. Medjahed, and M. Ouzzani, 2004. "Internet Computing Support for Digital Government," Practical Handbook of Internet Computing, Chapman & Hall/CRC.

[4] G. Marchionini, H. Samet, and L. Brandt, 2003. "Introduction to Special Issue of Digital Government," Comm. ACM, vol. 46, no. 1, pp. 24- 27.

[5] F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, and M.-C. Shan, June 2000. "Adaptive and Dynamic Service Composition in eFlow," Proc. Int'l Conf. Advanced Information Systems Eng. (CAiSE).

[6] B. Medjahed and A. Bouguettaya, July 2005. "A Multilevel Composability Model for Semantic Web Services," IEEE Trans. Knowledge and Data Eng., vol. 17, no. 7, pp. 954-968.

[7] S.A. Chun, V. Atluri, and N.R. Adam., 2002. "Dynamic Composition of Work-Flows for Customized

Egovernment Service Delivery," Proc. Third NSF Conf. Digital Govt. Research, pp. 383-389.

[8] Dan. Wu et al., 2003. "Automating DAML-S Web Service Composition Using Shop2," Proc. Int'l Semantic Web Conf., pp. 195-210.

[9] S.D. Gribble et al., 2000. "Scalable, Distributed Data Structures for Internet Service Construction," Proc. Fourth Symp. Operating Systems Design and Implementation, pp. 319-332.

[10] J.M. Hellerstein and M. Stonebraker, 1993. "Predicate Migration: Optimizing Queries with Expensive Predicates," Proc. SIGMOD, pp. 267-276.

[11] Francisco Curbera, Frank Leymann, Dieter Roller, and Marc-Thomas Schmidt, 2003. Web Services Flow Language (WSFL), http://xml.coverpages. org/wsfl. html.

[12] X. Liu and A. Bouguettaya, 2007. "Managing Top-Down Changes in Service-Oriented Enterprises," Proc. IEEE Int'l Conf. Web Services (ICWS), pp. 1072-1079.

[13] Microsoft, 2003. Web Services for Business Process Design (XLANG), http://xml.coverpages.org/xlang. html.

[14] OASIS, 2006. Web Services Evening (WS-Evening), http://www.w3.org/Submission/WS-Eventing.

[15] M.P. Papazoglou and D. Georgakopoulos, 2003. "Introduction to Special Issue of Service-Oriented Computing," Comm. ACM, vol. 46, no. 10, pp. 24-28.

[16] X. Yang, A. Bouguettaya, B. Medjahed, H. Long, and W. He, Nov 2003. "Organizing and Accessing Web Services on Air," IEEE Trans. Systems, Man, and Cybernetics—Part A: Systems and Humans, vol. 33, no. 6, pp. 742-757.

[17] T. Pilioura, A. Tsalgatidou, and S. Hadjiefthymiades, 2003. "Scenarios of Using Web Services in M-Commerce," SIGecom Exchanges, vol. 3, no. 4, pp. 28-36.

[18] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Sheng, 2003. "Quality-Driven Web Service Composition," Proc. Int'l Conf. World Wide Web (WWW).

[19] A. Rezgui, M. Ouzzani, A. Bouguettaya, and B. Medjahed, Nov 2002. "Preserving Privacy in Web Services," Proc. Fourth ACM Workshop Information and Data Management (WIDM '02), pp. 56-62.

[20] U. Srivastava, J. Widom, K. Munagala, and R. Motwani, 2006. "Query Optimization over Web Services," Proc. Int'l Conf. Very Large Data Bases (VLDB).

[21] Q. Yu and A. Bouguettaya, 2008. "Framework for Web Service Query Algebra and Optimization," ACM Trans. Web, vol. 2, no. 1.

[22] V. Venkatesh, V. Ramesh, and A.P. Massey, 2003. "Understanding Usability in Mobile Commerce," Comm. ACM, vol. 46, no. 12, pp. 53-56.

****