Optimized Querying of E-Government Services

Mourad Ouzzani Athman Bouguettaya Brahim Medjahed
Department of Computer Science
Virginia Tech
{mourad,athman,brahim}@vt.edu

Abstract

Web services are deemed as the natural choice for deploying e-government applications. Their use would open the door for e-government to fully get advantage of the envisioned *Semantic Web*. We are investigating the building of a comprehensive *Web Service Management System* that treats Web services as *first class objects*. In this paper, we focus on optimized querying of e-government services. Our approach is based on quality of service is adjusted through a *dynamic rating scheme* and *multilevel matching*.

1 Introduction

Digital government applications are becoming commonplace. A major propelling technology for e-government is the emerging concept of *Web services* [1]. The nexus between the two is becoming very strong as Web services provide the platform of choice for deploying the different functionalities offered by governments. A *Web service* is a set of related functionalities that can be programmaticaly accessed and manipulated through the Web [2]. Examples of e-government services include checking the eligibility for a health insurance, applying for food stamps, renewing a driver license, etc. The powerful concept of Web service is taking root because of the convergence of government and business efforts to make the Web the place of choice for all types of human activities. The ability to efficiently access and share e-government services is a critical step towards the full deployment of digital government. This requires the development of techniques to address various challenging issues. Required techniques include service description, discovery, querying, composition, monitoring, security, and privacy. All these techniques would be part of a comprehensive *middleware* for managing *autonomous* and *heterogeneous* Web services. For that purpose, we are investigating the architectural components of a *Web Services Management System* (*WSMS*). The aim of a *WSMS* is to do for Web services what DBMSs have done for data.

In this paper, we focus on a new *query scheme* that offers *database-like* query facilities over Web services. Users submit queries that are answered through a combined access to various Web services. The challenge is then to devise the "best" alternative of Web services combinations with respect to the delivered quality. We propose an *optimization model* based on *Quality of Service* (*QoS*) that would capture users' requirements for efficiency. *QoS* encompasses several quantitative and qualitative measurements of the behavior of Web services. Query optimization is based on aggregating *QoS* parameters of different Web services and selecting the best combination thereof. Quality of service is adjusted to the behavior of the different Web services expressed through a *dynamic rating scheme* and *multilevel matching*.

2 E-Government Services Querying

In the new *query scheme*, queries are resolved by combining invocations of Web service operations. In this section, we elaborate on the different underlying concepts.

2.1 Query Paradigm

A fundamental challenge in enabling Web service queries is how to obtain the combination of actual operations from the declarative expression of a query. Thus, we propose a multi-level query paradigm

where queries go through several transformations that lead to the *service execution plan*. First, we need to present users with "objects" to express and submit queries. The next step is to link those objects to the actual Web services. We define a set of Web service-like operations that are typical to a given application domain. These are *virtual* operations that links objects used to express queries and the actual Web services. More precisely, we define a three-level query paradigm as follows:

- *Query Level* – Contains a set of relations defined through mapping rules over the virtual operations. These relations are used to allow users to submit database-like queries.
- *Virtual Level* – Contains Web service-like operations typically offered in a particular application domain. They are obtained based on expertise from that domain.
- *Concrete Level* – Represents the space of e-government services offered on the Web. These are potential candidates to answer queries. They are matched with operations from the virtual level.

2.2 Mapping Rules

Each relation defined at the query level is mapped to one or a set of virtual operations. Relations are represented as conjunctive queries over virtual operations. More precisely, let $\mathcal{R}$ be the set of relations defined at the query level and $\mathcal{VOP}$ the set of virtual operations. For any relation $R_i \in \mathcal{R}$, $R_i(x_1,\ x_2,\ ...,x_n)\ :-\ \bigwedge_j\ Vop_j(y_{j_1},\ ...,\ y_{j_m})$ where x_i are the attributes of R_i, $Vop_j\ \in\ \mathcal{VOP}$, and y_{j_i} are input and output variables of the corresponding operation. This definition means that to get R_i's tuples, we need to invoke the different operations Vop_j.

2.3 Virtual Operations Representation

For each operation, we need to describe attributes necessary for the Web service discovery, operation matching, and operation invocation. In particular, the discovery and matching processes depend tightly on semantic attributes. Thus, we assume that government agencies across federal, state and county levels agree on a *shared ontology*. That ontology would be used to describe semantic attributes of virtual and concrete operations.

Each virtual operation is formally represented by a quintuple $Vop\ =\ (In,\ Out,\ Domains,\ Category,\ Function)$ where In is the set of input variables, Out the set of output variables, $Domains$ a set of pair $(x, range)$ where x is an enumerable variable and $range$ is the set of all possible values for x, $Category$ describes the domain of interest of the operation (e.g., insurance, housing, school), and $Function$ describes the function offered by the operation (e.g., eligibility, application, interview). $Category$ and $Function$ capture the semantics of that operation. They are part of the *ontology* defined and agreed upon by government agencies.

3 E-government Services Query Processing

In this section, we present different techniques necessary to process queries according to the multi-level query paradigm. We propose a multi-level matching approach that allows finding similar or partial answers to a query. We also outlines the fundamental premises of our optimization model based on QoS adjustment.

3.1 Virtual Operations Matching

It is not always possible to find an exact match for a given virtual operation. In addition, the same functionality may be offered in various ways by different Web services. Users may be inclined to accept similar or close answers to their queries. This is especially true in the context of social services where the objective is to get whatever social benefits are available for a needy citizen. Consequently, we consider different matching levels for virtual operations.

Let Vop and Cop be two virtual and concrete operations respectively. Vop is matched to Cop using one of the following alternatives:

- *Exact match* – The concrete operation matches the virtual operation in every aspect. Two operations *match exactly* if they have the same sets of input and output variables, and they have the same $Category$ and $Type$ attributes.
- *Overlap match* – Looks for operations that offer *close* functionalities to that of the virtual operation. Two operations Vop and Cop *match by overlap* if they have the same sets of input and output variables, and their $Category$ and $Type$ attributes overlap (but they cannot be identical in the same time).
- *Partial match* – The input and output attributes of the two operations do not necessarily coincide. Two operations Vop and Cop *match partially* if $Out(Cop)$ is a subset of $Out(Vop)$ or $In(Cop)$ is a subset of $In(Vop)$, and their $Category$ and $Type$ attributes are the same or overlap. An example of the first subset relationship is simply an operation that returns less information than what is expected by the query. An example for the second subset relationship is an operation that does not take into account all provided input variables and may subsequently miss some answers.

We assign to each level a *matching degree* that quantifies how exact the matching is. This would help the citizen and the case manager in assessing the results of their queries. The *matching degree* has a direct impact on the quality of the query results and subsequently on the optimization process.

3.2 Optimization Model

Given a query, an important challenge for the system is to find the "best" service execution plan with respect to an objective function. Quality of service (*QoS*) is playing a crucial role in assessing the added-value of competing Web services [3]. The *WSMS* optimally solves queries based on *QoS*. In our approach, Web services are selected and combined based on the *QoS* they offer adjusted through a *dynamic rating scheme* and *multilevel matching*. Each time a Web service is selected in solving a query, it is rated by comparing its advertised *QoS* with its actual *QoS*. In addition, different *levels of matching* have been considered in matching virtual and concrete operations. Each level has a *matching degree* that is also used to adjust the objective function.

QoS is defined through a number of parameters supplied by the Web service providers. The objective of the optimization process is to maximize or minimize each value. In the proposed system, we consider the following *QoS* parameters:

- *Latency* – Latency represents the average time it takes for an operation to return results after its invocation.
- *Availability* – Availability defines whether the Web service is present and ready to be invoked. It represents the probability that a service is available. Large values mean high availability. Small values indicate unpredictability of whether the service is available.
- *Security* – The ability to provide confidentiality and non-repudiation of exchanged information. This is crucial for e-government that manipulates large amounts of sensitive information.

Due to the different fluctuations that may occur with a Web service, the *QoS* advertised by that Web service may not be always fulfilled. Furthermore, the Web service may change some of its *QoS* parameter values over time. To ensure that *QoS* parameters are used in a way that represents the *actual* quality of a Web service, we propose to adjust the advertised values of those parameters. The idea is to *rate* Web services by monitoring them. The *promised QoS* (*pQoS*) is the value of the *QoS* parameters advertised by the service provider. The *delivered QoS* (*dQoS*) is the value of the *QoS* parameters obtained by monitoring the Web service. The difference between *pQoS* and *dQoS* represents the *QoS distance* of the Web service.

Any new Web service receives initially the highest rating. Rates range over a `[0, 100]` scale where `100` is the highest value. Web services with a negative *QoS distance* above a certain negative threshold will have their rating lowered. On the contrary, if the *QoS distance* has a positive value greater than a certain positive threshold, the Web service rating is increased if it does not have already the highest value.

3.3 Service-based Query Processing and Optimization

Each Web service operation involved in resolving a query is characterized by three parameters: promised *QoS*, rating, and matching degree. Each of these parameters has an impact on the quality of the query execution plan. Users submit queries using relations from the query level. A query is subject to several transformations leading to a *service execution plan*. This plan consists of iterations over expanded variables, invocations of Web service operations, and data flow control to route data and collect results.

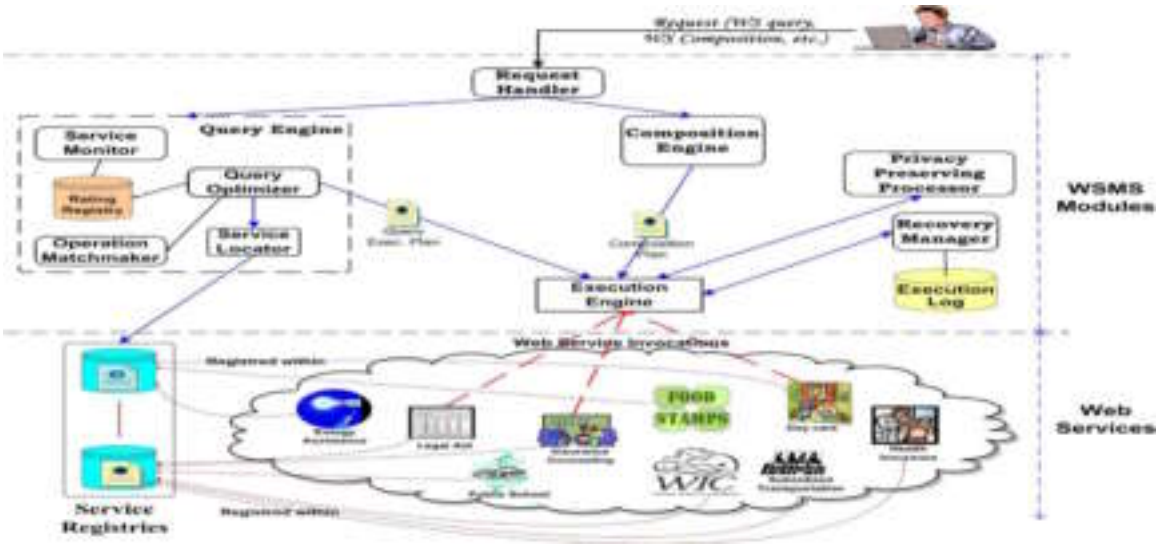

Figure 1: A Web Service Management System for E-Government Services

In the following, we outline how the *query engine* (Figure 1) in processing and optimizing queries.

- *Query unfolding and variables expansion* – Each relation is unfolded to virtual operations using the corresponding mapping rule. We obtain a new query consisting of virtual operations and conditions on their attributes. For conditions with inequalities, the corresponding input variables are expanded into all their possible values based on their ranges.
- *Operations matching* – For each virtual operation, the system looks for relevant Web services through available service registries. For each returned Web service, its description is searched for operations that match the virtual operation based on the different levels that we have defined.
- *Service execution plan generation and optimization* – Located operations from the matching phase are selected and combined together based on the defined objective function and feasibility. *Feasibility* means that the obtained plan can be actually executed. The query engine mainly checks that input variables required by any operation to be invoked are bound.

Acknowledgment

This research is supported by the National Science Foundation under grant 9983249-EIA.

References

[1] B. Medjahed, A. Rezgui, A. Bouguettaya, and M. Ouzzani. Infrastructure for E-Government Web Services. *IEEE Internet Computing*, 7(1), January/February 2003.

[2] S. Tsur, S. Abiteboul, R. Agrawal, U. Dayal, J. Klein, and G. Weikum. Are Web Services the Next Revolution in e-Commerce? (Panel). In *Proceedings of 27th International Conference on Very Large Data Bases*, Rome, Italy, September 2001. Morgan Kaufmann.

[3] S. Vinoski. Service Discovery 101. *IEEE Internet Computing*, 7(1), Jan/Feb 2003.