



Infrastructure for E-Government Web Services

WebDG is a comprehensive infrastructure for providing customized government services over the Web while maintaining citizens' privacy.

**Brahim Medjahed,
Abdelmounaam Rezgui,
Athman Bouguettaya,
and Mourad Ouzzani**
Virginia Tech

In efforts to use information and communication technologies for the civil and political conduct of government, many countries have begun supporting e-government initiatives. The ultimate goal is to improve government-citizen interactions through an infrastructure built around the "life experience" of citizens. This spans several governmental sectors, including social services. In our e-government project, researchers at Virginia Tech and Purdue University have teamed up with Indiana's Family and Social Services Administration (FSSA), which provides welfare programs to assist low-income citizens, strengthen families' and children's well-being, and help elderly and disabled people.

The current process for collecting social benefits within FSSA is time-consuming and frustrating, as citizens must often visit multiple offices located within and outside their hometown. Case officers must use myriad applications, manually determin-

ing which will satisfy citizens' individual needs, deciding how to access each form, and combining the results returned by different applications – all without divulging citizens' information to unauthorized entities. This difficulty in obtaining much-needed help hinders many people's ability to become independent, with consequences for health and well-being.

To facilitate the use of welfare applications and expeditiously satisfy citizens' needs, we wrapped these applications in modular Web services.¹ Adopting Web services in e-government enables government agencies to provide value-added services by defining a new service that outsources from other e-government services; to uniformly handle privacy issues; and to standardize the description, discovery, and invocation of social programs.

The core of our research is to develop techniques to efficiently access e-government services while preserving citizens' privacy. To that end we designed and

implemented an infrastructure called Web Digital Government. WebDG's main contributions revolve around two features:

- *Composing e-government services.* The framework for automatically composing e-government services is based on a set of rules that check composability of services.
- *Preserving privacy.* To protect privacy, requests for services contain users' privacy credentials, which filtering mechanisms use to ensure that only authorized entities can access sensitive information.

The current implementation is based on emerging standards for describing, discovering, and invoking services. Readers can see a demo version of WebDG online (www.nvc.cs.vt.edu/~dgv).

Scenario: Collecting Benefits

To illustrate the drawbacks of the current system and how WebDG can help, we can examine a typical scenario. A pregnant teen – let's call her Mary – goes to an FSSA office to collect social benefits. Mary needs a government-funded health insurance program. She would also like to receive nutritional advice. Because Mary will not be able to take care of the future newborn, she is also interested in finding a foster family.

Fulfilling all of Mary's needs requires access to services scattered among various agencies. The case officer – let's call him John – must first manually look up which social programs offer health insurance and food assistance for Mary. Medicaid and WIC (a federally funded food program for Women, Infants, and Children) would be the best candidates. Assuming Medicaid (a healthcare program for low-income citizens) is locally accessible, John has to connect to the corresponding application and interact with it. Because WIC is a federal program, however, John has no direct access to the corresponding application. That means Mary must visit another agency, perhaps in a different town, to apply for the benefit.

Still more difficulties arise when John tries to find a foster family service. Using local resources, he finds no matching program, although Teen Outreach Pregnancy (TOP), an FSSA partner, does offer such services. To complicate things further, each time John connects to an application, he has to make sure that it abides by privacy rules related to the access to and use of sensitive information such as Mary's social security number (SSN).

This process of manually researching and access-

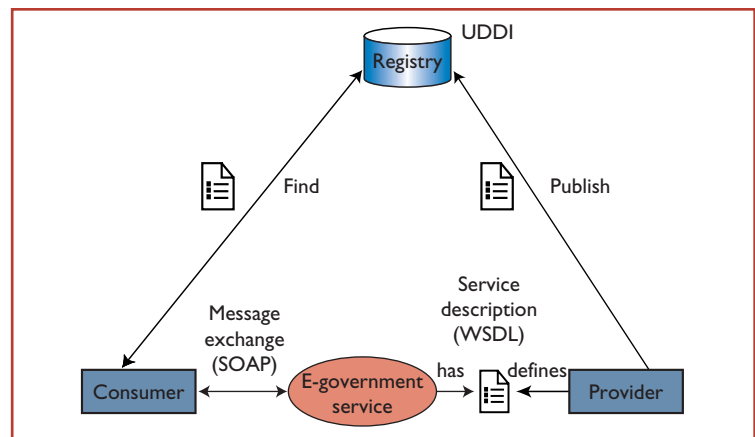


Figure 1. Interactions among WebDG services. Consumers access a registry to discover the services (such as WIC) that government agencies (such as the Bureau of Family Resources) offer.

ing individual services is clearly time-consuming. It would be more efficient if John could specify Mary's needs once and address them all together. He could then seamlessly access all related services through a single access point² – perhaps a Pregnancy Benefits service that outsources from WIC, Medicaid, and TOP – regardless of the programs' locations or providers. Such an infrastructure could also protect the privacy of Mary's sensitive information across the entire system of social services. That is exactly what WebDG aims to do.

Organizing WebDG Services

As Figure 1 shows, interactions among WebDG services involve three types of participants: provider, registry, and consumer. Providers include FSSA bureaus or divisions (for example, the Bureau of Family Resources) as well as external agencies such as the U.S. Department of Health and Human Services. The providers publish descriptions of their services, such as operations and network locations, in a registry. Consumers, including citizens and case officers, access the registry to locate services of interest. The registry returns relevant descriptions, which consumers use to invoke corresponding Web services.

Providers describe the operational features of WebDG services in the Web Services Description Language (WSDL).³ Each operation has one of four possible modes:

- *one-way*, in which the service receives a message
- *notification*, in which the service sends a message
- *request-response*, in which the service receives a message and sends a correlated message
- *solicit-response*, in which the service sends a message and receives a correlated message

Related Work

Web services can work with technologies such as software components to enable interorganizational interactions.¹ Such components are suitable for building robust and secure applications within an organization, and Web services can wrap them to provide connectivity between autonomous and heterogeneous interorganization applications. For example, we can use IONA's Orbix E2A Web Services Integration Platform to create Web services from existing Corba objects.

Standardization efforts (such as BPEL4WS²) and commercial platforms (.Net and Web-Sphere,³ for example) are increasingly targeting Web services. The assumption, however, is that users should find the Web services to be outsourced. Users might be required to have full tech-

nical knowledge of the outsourced services' details. In e-government applications, however, features such as service composition must be transparent to users. The WebDG approach focuses on a select type of users (citizens and case officers) to provide customized e-government services.

To our knowledge, this work is one of the first to approach privacy in Web services from a purely technical viewpoint. For example, W3C's Platform for Privacy Preferences Project (P3P) enables users and developers to specify privacy for Web sites, but not Web services. WebDG preserves privacy when Web services are accessed by other applications (not necessarily via a Web browser). Additionally, P3P provides no mechanisms that guarantee that Web sites actually implement their privacy policies. There is a need

to develop techniques and standards that let e-government users specify their privacy preferences and service providers specify their privacy policies. Further, database managers must be able to specify data privacy profiles. WebDG is a first step toward addressing privacy in e-government Web services from a multilevel standpoint, on the basis of user, service, and data.

References

1. S. Vinoski, "Where is Middleware?" *IEEE Internet Computing*, vol. 6, no. 2, Mar. 2002, pp. 83-85.
2. *Business Process Execution Language for Web Services (BPEL4WS)*, V. 1, BEA, IBM, and Microsoft, 2002; www-106.ibm.com/developerworks/webservices/library/ws-bpel/.
3. S.J. Vaughan-Nichols, "Web Services: Beyond the Hype," *Computer*, vol. 35, no. 2, Feb. 2002, pp. 18-21.

For example, the WIC service offers a request-response operation called `checkEligibility`. This operation receives a message that includes a citizen's income and family size and returns a message indicating whether the citizen is eligible for WIC.

WebDG stores WSDL descriptions in a registry based on Universal Description, Discovery, and Integration (UDDI).⁴ The registration of the Medicaid service, for example, includes the URL for communicating with this service and a pointer to its WSDL description.

WebDG services communicate via SOAP messages.⁵ Because SOAP uses XML-based messaging over well-established protocols like HTTP and SMTP, it is platform-independent, but it has a few drawbacks. For one thing, SOAP does not yet meet all the scalability requirements of Web applications. Unlike communication middleware such as Corba and Java RMI, SOAP encoding rules make it mandatory to include typing information in all SOAP messages.

Additionally, SOAP defines only simple data types such as `String` and `Int`. Hence, using complex data types may require the XML parser to get the corresponding XML schema definitions from remote locations, which might add processing overhead. However, the data types used in e-government are generally agreed upon a priori, which reduces the associated overhead.

Composing WebDG Services

A major issue when defining composite services is

checking whether the Web services can actually work together, a process called *composability*.⁶ For example, it would be difficult to invoke an operation if no mapping existed between the parameters requested by that operation and those transmitted by the client service. To deal with this issue, WebDG defines a set of rules that check composability for e-government services by comparing *syntactic* (such as operation modes) and *semantic* (such as domain of interest) features. The service composer, often a case officer, provides a high-level specification of the desired composition that simply contains the list of operations to be performed, without referring to existing services. Based on composability rules, WebDG then generates a composition plan that gives the list of outsourced services and how they interact with each other — through plugging operations, mapping messages, and so forth — to achieve the desired composition.

Semantic Composability

The semantics underlying Web services is crucial to checking composability. Currently, WSDL provides no support for semantic service descriptions, but some researchers are now using ontologies to help capture Web services' semantic features. This could pave the way for the envisioned Semantic Web and open the door to automatic service composition.⁶

In a nutshell, an ontology defines taxonomies based on the semantic proximity of terms. We define two ontologies for service operations: Cat-

egory and Type. We assume that government social agencies would agree on the ontologies ahead of time. Each operation includes two elements, one each from the Category and Type ontologies. Service providers identify the category and type relevant to each operation and initialize the corresponding elements accordingly.

Each element of the Category ontology contains three attributes: name, synonyms, and specialization. The *name* gives the current operation's domain of interest; examples include `health care`, `adoption`, `nutrition`, and `disability`. The *synonyms* attribute contains a set of alternative names for the domain of interest. `Medical` is a synonym for `health care`, for example. *Specialization* defines a set of characteristics for the current category. For example, `insurance` and `children` are specializations of the `health care` category.

The Type ontology also includes name, synonyms, and specialization attributes for each element. The name describes the business functionality that the current operation offers, such as `eligibility`, `counseling`, and `mentoring`. Synonyms and specialization attributes work just as they do in the Category ontology.

When checking the composability of interacting operations, it is important to ensure compatibility between their types and categories. For example, it would be semantically incorrect to connect the `counseling` and `eligibility` operations because these operations offer different functionalities. To define compatibility for operation categories (and operation types), let us consider two operations, op_i and op_j . We say that $\text{Category}(op_i)$ is compatible with $\text{Category}(op_j)$ if:

- $(\text{Name}_i = \text{Name}_j)$ or $(\text{Name}_i \in \text{Synonyms}_j)$ or $(\text{Name}_j \in \text{Synonyms}_i)$ and
- $\text{Specialization}_i \subseteq \text{Specialization}_j$.

The first condition checks that the domains of interest for op_i and op_j are similar (first disjunct) or synonyms (second and third disjuncts). The second condition ensures that op_j provides all the characteristics of op_i 's category. Assume that op_i 's category deals with health insurance for children ($\text{Name}_i = \text{health care}$ and $\text{Specialization}_i = \{\text{children}, \text{insurance}\}$). The operation op_j should not only deal with health care, but also provide insurance for children. Based on compatibility between categories and types, we define the semantic composability rule: op_i is semantically composable with op_j if $\text{Category}(op_i)$ is compatible with $\text{Category}(op_j)$ and $\text{Type}(op_i)$ is compatible with $\text{Type}(op_j)$.

Composition Soundness

Another important aspect to consider is whether combining Web services in a specific way is worthwhile. The idea is to define a rule, called *composition soundness*, to check whether composition plans are sound. (By sound, we mean that the way services are combined provides an added value.) For that purpose, we introduce the notion of a *composition template*.

For each composition plan WebDG generates, it builds a composition template that gives the general structure of that plan. A directed graph (V, E) , where V is a set of service category names and E is a set of edges, models the template. WebDG uses the Category ontology that defines service operations to describe service categories. An edge $v_i \rightarrow v_j$ specifies that a service of category name v_i precedes a service of category name v_j . As Figure 2a (next page) shows, a service WS_i precedes another service WS_j if WS_i invokes an operation of WS_j .

Figure 2b depicts the template of a plan WebDG generated for the Pregnancy Benefits (PB) composite service. The edge `Pregnancy` \rightarrow `Health care` specifies that a solicit-response operation of PB, namely `searchPCP`, is connected to a request-response operation of the Medicaid service. (`searchPCP` returns the list of available primary care providers.)

In the WebDG system, government agencies define a subclass of templates called *stored templates* ahead of time. For example, social services agencies would agree that a composite service providing pregnancy benefits should combine nutrition, health care, adoption, housing, employment, and legal services (see Figure 2c). Because stored templates inherently provide added values, they can prove or disprove the soundness of composition plans. A plan is sound if its template is a subgraph of a stored template, which means that the plan provides part or all of the interactions modeled by a stored template. The template in Figure 2b, for example, is a subgraph of the stored template in Figure 2c. This means that the corresponding plan offers a subset of the functionalities defined in the stored template, and thus, that the plan is sound.

Preserving Privacy

Privacy is a major issue that e-government needs to address.⁷ Citizens generally must divulge sensitive information, such as their SSN or salary, to access e-government services. Two characteristics add to the complexity of the privacy problem in e-government: sharing of citizens' information

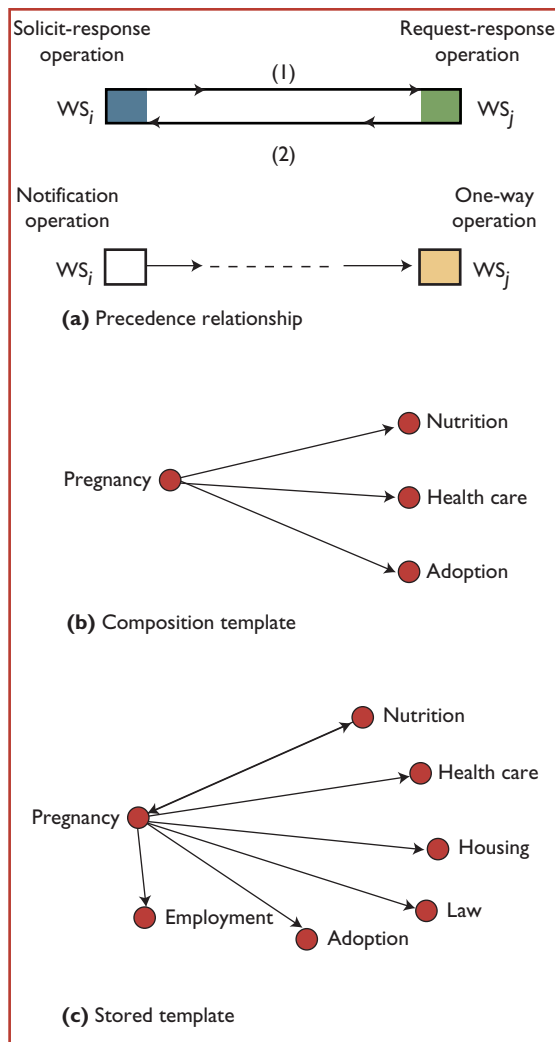


Figure 2. Composition soundness of a plan for pregnancy benefits. (a) WebDG uses the precedence relationship to build composition templates. (b) The composition template lays out a plan to fulfill a citizen's needs. (c) Overlap with the stored template proves the plan is sound.

among government agencies and citizens' differing privacy requirements.

Privacy is generally misperceived as an issue whose natural solution consists of good security mechanisms. Security and privacy are tightly inter-related issues, but secure e-government infrastructures do not necessarily ensure privacy. For example, a government agency's business rules might forbid a case officer from accessing some specific sensitive information about a given group of citizens. Although this agency might use security mechanisms such as authentication or encryption to prevent unauthorized access, the case officer could find a way to access the sensitive information. The officer might be reassigned to a new func-

tion, and the old manager could forget to revoke the officer's existing privileges, thus allowing access to private information about citizens. Our system focuses on privacy enforcement; we assume that appropriate security mechanisms, such as secure communication channels, already exist within the e-government environment.

Privacy Model

A typical citizen-government interaction involves three participants: users, services, and databases. This naturally defines a three-layered model for privacy.

User privacy. In typical situations, users interacting with an e-government service provide a significant amount of sensitive information. Users will expect or require different levels of privacy according to their perception of the information's sensitivity. For example, a user might have tighter privacy requirements regarding medical records than employment history. The user's perception of privacy needs also depends on who receives the information and the purposes for which the information is used.⁷

Ideally, users should have some degree of control over who can access their information and for what purposes. A user who provides a home address to the WIC service, for example, might wish to limit access to that information to only that service. In WebDG, users specify their privacy preferences through privacy profiles, which they must have the right to create, edit, and update. The government agencies' internal business rules could, however, override users' privacy profiles. A new regulation might state that a service must share a specific piece of information with some government department regardless of citizens' privacy preferences.

Based on privacy profiles and agency regulations, WebDG assigns a user accessing an e-government service a privacy credential that determines access rights (such as read, update, and save) to data objects. A data object could be a health record for a given citizen, a specific portion of such a record, or a set of health records for a group of citizens. Formally, let D be the set of data objects that exist in the e-government infrastructure. Let U be the set of users and R be a set of access rights. The privacy credential is a function f that maps pairs of users and data objects to sets of access rights such that $f : U \times D \rightarrow P(R)$, and $f(u, d) = r$ if the user u has all the access rights in r on the data object d . $P(R)$ is the set of partitions of R .

Based on credentials, we introduce the notion

of privacy scope. A privacy scope PS determines what and how data objects in D can be accessed by a user u : $PS(u) = \{d, f(u, d) \mid d \in D\}$.

Service privacy. Each e-government service has its own privacy policy that specifies a set of rules applicable to all users. Service privacy specifies three types of policy: usage, storage, and disclosure. The *usage policy* states the purposes for which a service can use the information collected. Medicaid's usage policy states that citizens' information will not be used for purposes other than those directly related to providing health services. The *storage policy* specifies whether and for how long the service can store the information. Medicaid keeps information about citizens for one year after they leave the welfare program. The *disclosure policy* specifies how, and to whom, a service can reveal the information. Medicaid may prohibit external users from accessing statistical information, such as average income and racial distribution, that reveals general characteristics about the applicants.

Data privacy. Data objects in a government database can be privacy-neutral or privacy-sensitive. Statistical health information about the people living in a given state is privacy-neutral, for instance, and thus available to the public. Information that relates to specific individuals is privacy-sensitive. Moreover, different Web services may need different information from the same data object. An IRS officer can access a record in the U.S. National Database for New Hires to check the accuracy of an employee's tax form, for instance, whereas an officer at a child-support agency can access the same data object to check whether a parent is complying with child-support obligations. Data objects must therefore be able to expose different views to different Web services. To capture the access rights that different entities can exert over a data object, we define data privacy profiles. The privacy profile PP for a given data object d specifies the access views that it exposes to each user: $PP(d) = \{(u, f(u, d)) \mid u \in U\}$.

Privacy Enforcement

An important premise in our privacy enforcement approach is that users access databases through e-government services. When a service receives a request from a given user, it first checks that the user has the necessary credentials to access the requested operation according to its privacy policy. Let's look at the Medicaid service, which states that the only person who can update a citizen's pri-

vacancy profile is that citizen. If the request is valid, the service translates the user's request into an equivalent data query and submits it to the appropriate government database management system.

Before the Medicaid service submits a query to the DBMS, it sends the query through a privacy-preserving data filter (DFilter). This 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 to enforce all the privacy constraints. For example, the QRM deletes the salary field from a service request that translates into the SQL query `select name, age, salary from Medicaid.enrollees` before submitting it to the DBMS if the credential of the user who made the request does not allow access to enrollees' salaries.

The *privacy profile manager* (PPM) enforces privacy at a finer granularity than the CCM does. The local CCM might decide that an organization can access local information regarding health records for a group of citizens, but some of those citizens might explicitly request that parts of their records not be made available to third-party entities. In this case, the local PPM would discard those parts from the generated result. The PPM is a translation of the consent-based privacy model in that it implements individual citizens' privacy preferences. It maintains a repository of privacy profiles that stores individual privacy preferences. The PPM also handles citizens' requests for updating their privacy profiles.

Implementation

Figure 3 (next page) shows the WebDG system as implemented across a network of Solaris workstations. Citizens and case officers access the system via a graphical user interface implemented in HTML and Java servlets. WebDG currently includes seven FSSA applications implemented in Java (JDK 1.3). The Axis Java2WSDL utility in IBM's Web Services Toolkit automatically generates WSDL descriptions from Java class files, which WebDG publishes into a UDDI registry.

WebDG uses the service management client within Apache SOAP 2.2 to deploy e-government services. Apache SOAP provides a server-side infrastructure for deploying and managing services, and a client-side API for invoking those services. Each service has a deployment descriptor that includes

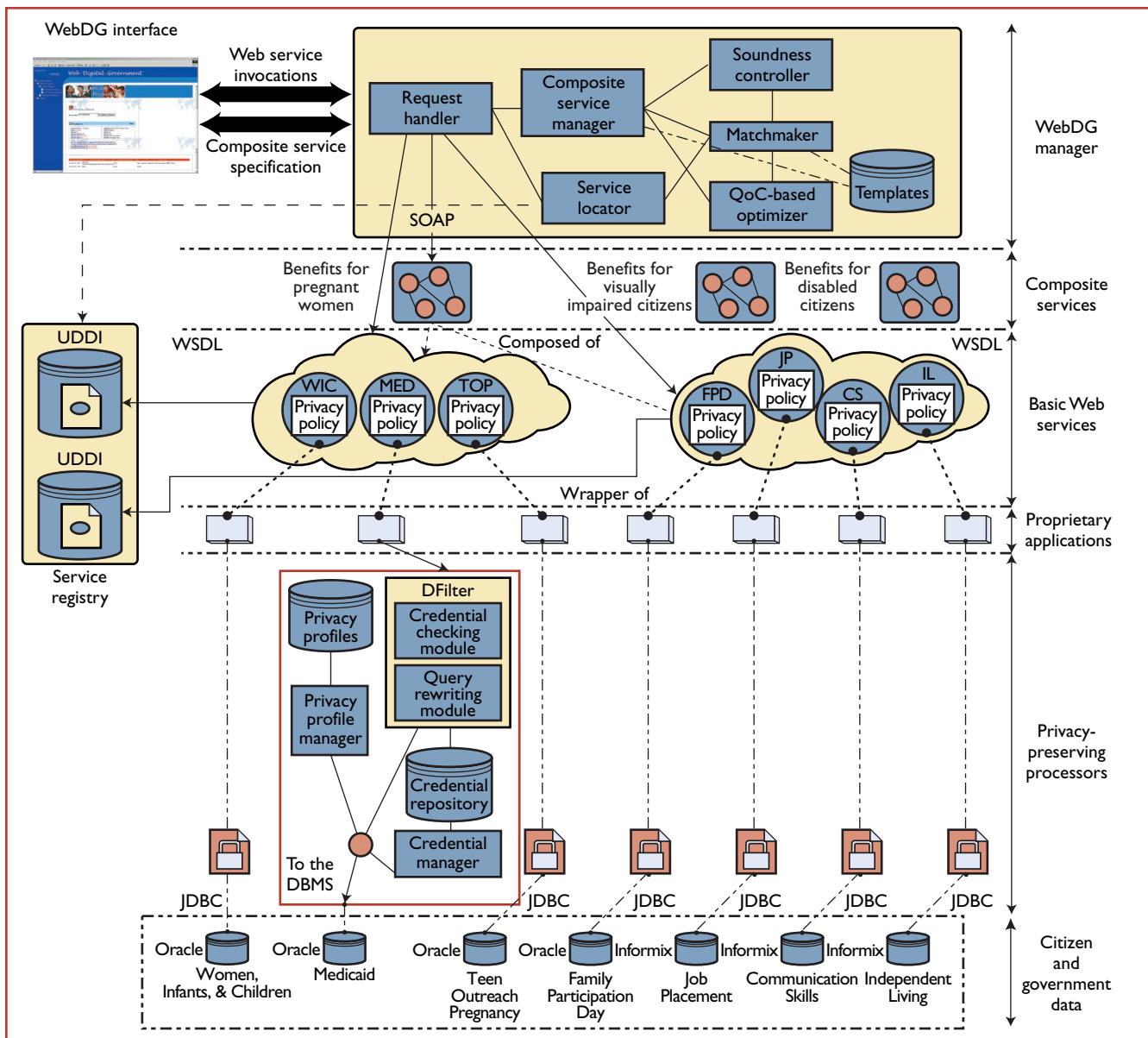


Figure 3. The WebDG architecture, as implemented for social government services. The WebDG manager and privacy-preserving processors compose and preserve privacy in e-government Web services.

the unique identifier of the Java class to be invoked, the session scope of the class, and operations in the class available for the clients. WebDG deploys each service using its descriptor and the URL of the Apache SOAP servlet `rpcrouter` as input arguments.

The *WebDG manager* is at the system's core. The *service locator* (SL) looks up WSDL descriptions in the registry. Once the *request handler* discovers a service, it invokes the service's operations through a SOAP binding stub.

When service operations attempt to access the FSSA databases (implemented in Oracle 8.0.5 and Informix 7.0), a privacy-preserving processor intercepts the operation invocations and allows or

disallows access based on privacy profiles, privacy credentials, and data filters. Figure 4 shows TOP's Search Family Adoption operation, which returns information about foster families in a given state, in this case Virginia. The value "no right" for the attribute `Race` means that the current user does not have system permission to access information about family F1's race. The value "not accessible" for the `Household Income` attribute indicates that family F1 does not want to disclose that sensitive information.

The *composite service manager* (CSM) uses the Java API for XML Processing (JAXP) to parse the XML-based composite service specifications. It then returns them to the *matchmaker*, which

checks composability rules. The matchmaker then sends each composite service operation's category to the SL. The SL parses each located service's WSDL description and returns it to the matchmaker. The SL retrieves only services whose category is compatible with the operation's category. After checking composability, the matchmaker generates composition plans and sends them to the *optimizer*, which selects plans based on quality of composition (QoC) parameters. Users define thresholds for such parameters as time, cost, and the plan's relevance to the user's specification. The optimizer returns plans to the matchmaker if the QoC parameter values are greater than the user's thresholds. The matchmaker forwards the selected plans to the *soundness controller* (SC) to check that the way they are combined provides added value. The SC then returns plans, along with their compatible stored templates (if any), to the CSM.

Conclusion

We continue to develop and extend the WebDG infrastructure for organizing, composing, and preserving privacy in e-government services. Future work will include a querying scheme based on Web services. That raises several challenging issues because Web services are a priori unknown and their number is very large. Moreover, they do not have underlying schema as databases do, and they are usually autonomous, heterogeneous, and highly volatile. Our plan is to resolve users' queries by combining access to diverse Web services. The first step is to find the best alternatives for combining Web services to resolve the query. □

Acknowledgments

This research is supported by the U.S. National Science Foundation under grant 9983249-EIA and by a grant from the Commonwealth Information Security Center (CISC).

References

1. S. Vinoski, "Where is Middleware?" *IEEE Internet Computing*, vol. 6, no. 2, Mar. 2002, pp. 83-85.
2. M.P. Singh, "Physics of Service Composition," *IEEE Internet Computing*, vol. 5, no. 3, May 2001, pp. 6-7.
3. *Web Services Description Language (WSDL) 1.1*, World Wide Web Consortium, 2001; www.w3.org/TR/wsdl.
4. *Universal Description, Discovery and Integration, Version 3*, OASIS, Billerica, Mass., 2000; www.uddi.org.
5. *Simple Object Access Protocol (SOAP) 1.1*, World Wide Web Consortium, 2000; www.w3.org/TR/soap.
6. T. Berners-Lee, "Services and Semantics: Web Architecture," white paper, World Wide Web Consortium, 2001; www.w3.org/2001/04/30-tbl.

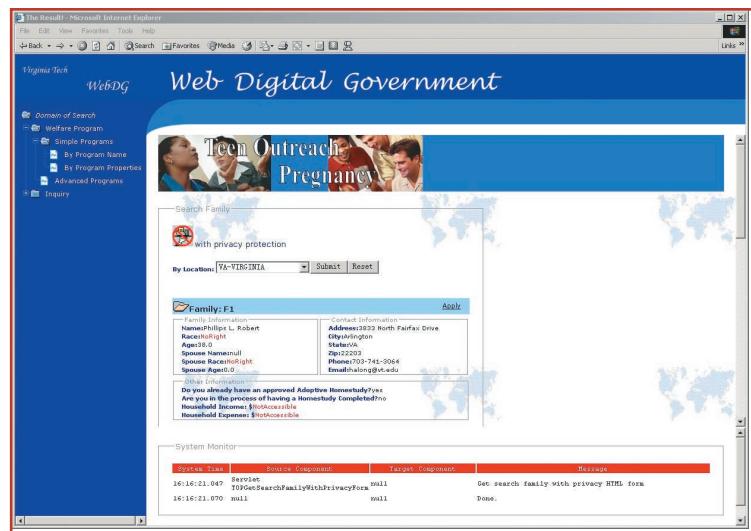


Figure 4. Preserving privacy in WebDG. Access privileges and privacy controls combine to hide sensitive information from view.

7. A. Adams, "User's Perception of Privacy in Multimedia Environment," PhD thesis, School of Psychology, Univ. College London, 2001.

Brahim Medjahed is a PhD candidate in the Computer Science department at Virginia Tech. His research interests include Web services, Web databases, e-commerce, and digital government. Medjahed received a BS and an MS in computer science from USTHB University, Algeria. He is member of the IEEE, the IEEE Computer Society, and the ACM. Contact him at brahim@vt.edu.

Abdelmounaam Rezgui is a PhD candidate in the Computer Science department at Virginia Tech. His current research interests include security and privacy in digital government, Internet databases, and Web services. Rezgui received an MS in computer science from Purdue University. He is member of the IEEE and the ACM. Contact him at rezgui@vt.edu.

Athman Bouguettaya is the program director of Computer Science and the director of the E-Commerce and E-Government Research Lab at Virginia Tech. His research interests are in Web databases and Web services. He is on the editorial board of the *Distributed and Parallel Databases Journal*, and he was a guest editor for a special issue of *IEEE Internet Computing* on database technology on the Web. Contact him at athman@vt.edu.

Mourad Ouzzani is a PhD candidate in the Computer Science department at Virginia Tech. His research interests include query optimization over Web services, digital government, and Web-based databases. Ouzzani received a BSc and an MSc in computer science from USTHB University, Algeria. He is a member of the IEEE, the IEEE Computer Society, and the ACM. Contact him at mourad@vt.edu.