Dynamically Composing Web Services from On-line Sources

Snehal Thakkar, Craig A. Knoblock, Jose Luis Ambite, Cyrus Shahabi

University of Southern California
Information Sciences Institute
4676 Admiralty Way
Marina del Rey, California 90292
{thakkar, knoblock, ambite}@isi.edu, shahabi@usc.edu

Abstract

The power of web services can only be realized when web services are utilized as building blocks to dynamically compose new web services. The Building Finder application is an example application that integrates information from several web services by modeling the web services as information sources in a mediator-based architecture. The paper also describes a mediator framework to dynamically compose new web services, similar to the Building Finder, by generating an integration plan to integrate information from the existing web services.

1. Introduction

The introduction of web services to the World Wide Web has made it possible to build new applications that integrate information from different web services and web sources. Recently, several tools have been introduced to rapidly develop and deploy web services. The XML-based standards for information interchange and web-based exchange protocols, such as SOAP, address syntactical issues involved in integrating information from different web services. Even with the emergence of web services and different information exchange protocols, integrating heterogeneous information sources and web services still remains a challenging problem. In part because information sources often use different data formats, different access methods and have different accuracy. In addition, not all information sources allow accessing their data using web services.

Three major challenges are involved in integrating information from the web sources using mediator based techniques: (1) some form of access method such as web service or wrapper must exist to access different sources, (2) web sources including web services must be modelled as information sources, and (3) algorithms to dynamically compose new web services using the source models must be developed. In this paper, we describe an application that integrates geo-spatial information from web services, such as Microsoft's Terraservice, with various web sources, such as white page directories on the web, to demonstrate how we can overcome the above-mentioned challenges to create a useful application. Furthermore, we describe a mediator-based framework to dynamically compose web services, such as the Building Finder, by integrating information from several web services.

The remainder of this paper is organized as follows. Section 2 describes our previous work on mediator-based approach to integrate information from web sources and machine learning techniques to build wrappers. Section 3, describes an application termed the Building Finder, which integrates information from web services and web sources. Section 4 describes a mediator framework to dynamically compose new web services from existing web services. Section 5 concludes the paper by re-capping the techniques described in the paper and open issues.

2. Previous Work

In previous projects we have developed an information agent architecture termed Ariadne to enable users to easily create their own specialized Web-based information agents [Knoblock et al. 2001]. The research focus in Ariadne is on technology for rapidly constructing information agents to extract, query, and integrate data from web sites as well as databases. The resulting system includes tools for constructing wrappers that access and extract information from web sites and query planning technology for dynamically and efficiently answering queries using these sources.

A simple example (see Figure 1) illustrates how Ariadne provides integrated access to multiple information sources. Numerous web sites provide restaurant reviews, such as Zagat, Fodors, and CuisineNet, but none are comprehensive, and checking every site can be time consuming. In addition, information from other sources can be useful in selecting a restaurant. For example, the LA County Health Department publishes the health rating of all restaurants in the county, and the U.S. Census bureau provides a map server that shows selected locations on a map. Using Ariadne, we can integrate these web sources relatively easily to create a virtual application where the system creates a map showing the restaurants that meet the user's requirements regarding cuisine, location, health rating, and more.

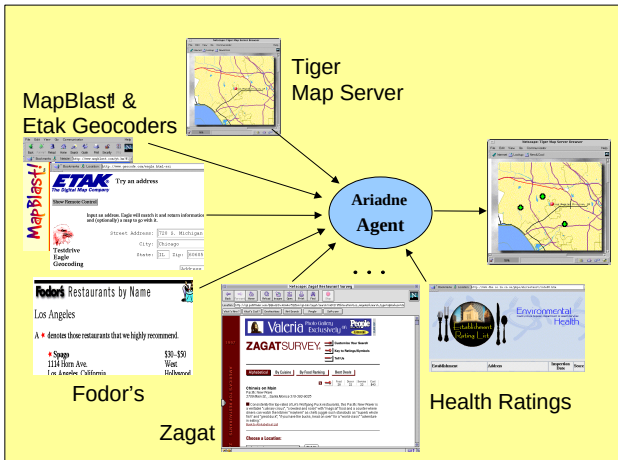

Figure 1 Ariadne Mediator: Example Application

An essential component of Ariadne is a wrapper around each individual data source, which provides uniform access to the information sources so that they can be queried as if they were relational databases. We have developed machine learning techniques to build wrappers around websites with relatively minimal user assistance [Knoblock et al. 2001b]. Using our wrapper building tools, we have developed several wrappers around popular web sources of different categories, such as the CNN website for news information and Yahoo Weather for weather related information.

In Ariadne the mediator designer defines a domain model, which is an ontology of the application domain that integrates the information in the sources and provides a single terminology over which the user poses queries. The domain model is represented using the Loom knowledge representation system [MacGregor 1988]. Each information source is defined described using the terms in the domain model [Arens et al. 1996; Knoblock et al. 2001]. The source descriptions are utilized to reformulate user queries into source queries.

We have also developed Theseus, an efficient execution system for information agents [Barish et al. 2000]. More specifically, Theseus empowers users by (a) simplifying the process of specifying agents that can manage and monitor dynamic data sources, such as web sites, and (b) providing a flexible, optimised platform for agent execution. Theseus allows users to specify a query execution plan to build a new information agent by using existing information agents.

3. The Building Finder Application

The Building Finder application allows a user to input an area of interest and provides a satellite image of the area with the houses and streets superimposed on the satellite image. Furthermore, the Building Finder also provides additional information such as name of the person, address, property value, and phone number for the houses in the area of interest. Section 3.1 describes detail information about the web sources integrated by the Building Finder application. Section 3.2 describes our approach to integrate the information from these sources.

3.1. Web Sources and Services

The building block services for the Building Finder application consist of following:

- Microsoft TerraService: TerraService provides access to a large repository of satellite images and topographic maps covering most of United States and Canada [Barclay et al. 1999]. The maps and satellite images provided by TerraService are from the United States Geological Survey (USGS). More information on TerraService can be found at http://terraservice.net.
- USGS Landmark web service: The USGS Landmark Service is also a part of Microsoft TerraService and provides access to landmark point across continental United States. The landmark points provided by this service are also provided by USGS.
- Tigerline files: The U.S. Census Bureau provides a comprehensive list of streets, zip codes and cities in form of the Tigerline files. The Tigerline files can be downloaded from U.S. Census Bureau's web site. We have downloaded the Tigerline files to our database.
- The SwitchBoard white pages: SwitchBoard white pages are located at http://www.switchboard.com. The SwitchBoard white pages provide listing of all residential phone numbers and other information from United States.
- Geocoder: Geocoder provides geographic coordinates for a given address. Several geocoders such as Etak are available on the web. However, most of the geocoders are off by as much as half block and do not provide any bounds on accuracy. We built a geocoder that provides more accurate geographic coordinates by integrating information from various web sources.
- Real Estate Property tax sites: The assessor's department in various counties and cities provide on-line information about different properties in the county or city. The information provided by the property tax sites include assessed value for the property, estimated property tax, and dimensions of the property. Various property tax sites have different coverage and provide different data. For example. USPDR located at http://www.uspdr.com provides property information for the state of New York, while the Los Angeles county assessor's department website located at http://assessor.co.la.ca.us/DataMaps/pais.asp provides information about properties in Los Angeles county. The Building Finder application finds property values from several property tax web sites. Figure 2 shows the organization of different property tax sites and coverage for each property tax site.

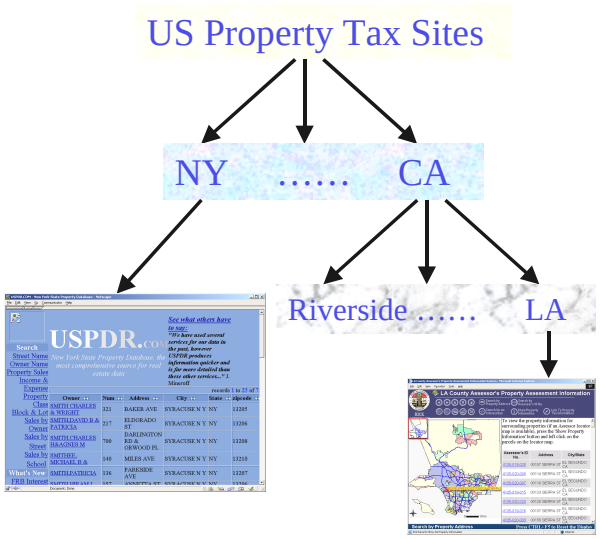

Figure 2 Property Tax Sites

The Building Finder utilizes the following access methods to obtain information from different data sources: (1) wrappers to extract structured information from the SwitchBoard white page web source and the various property tax sites, (2) SOAP and XML to access satellite imagery and points of interest from Microsoft TerraService and Landmark web services, respectively, and (3) a database query to access information from Tigerlines database.

The SwitchBoard white pages site and various property tax sites were wrapped using our wrapper building tools. Once a wrapper is built, the system can treat each web source as a database, i.e., the system can send queries to the SwitchBoard white page wrapper such as, "find all people in the city of El Segundo whose last name starts with the letter A". The wrapper returns an XML document that contains the results of the query.

XML web services are web applications that support standard protocols such as SOAP and XML. Terraservice provides access to a large repository of satellite images and topographic maps using SOAP and XML protocols [Barclay et al. 1999]. The Landmark service allows easy access to the USGS gazetteer, which provides important points such as hospitals, schools, and churches in the United States. The Building Finder provides area of interest to Microsoft TerraService web service through SOAP messages and receives a satellite image from the web service through a SOAP message. Similarly, the Building Finder also queries the USGS landmark points from the Landmark web service. The issue however is how to integrate the imagery obtained from the Terraservice, the address information obtained from the white page data, and information from various property tax sites.

3.2. Integrating Web Source and Services

The Building Finder integrates information from the sources described in Section 3.1. Figure 3 shows the architecture for the Building Finder application. The Building Finder receives the region of interest as input from the user and queries the streets from a Tigerlines database containing street network information to find all streets in the region of interest. The result of the query is a set of tuples consisting of street name, city, state and zip code, which are used to query the Switchboard white page wrapper to find all the addresses on the streets. The result

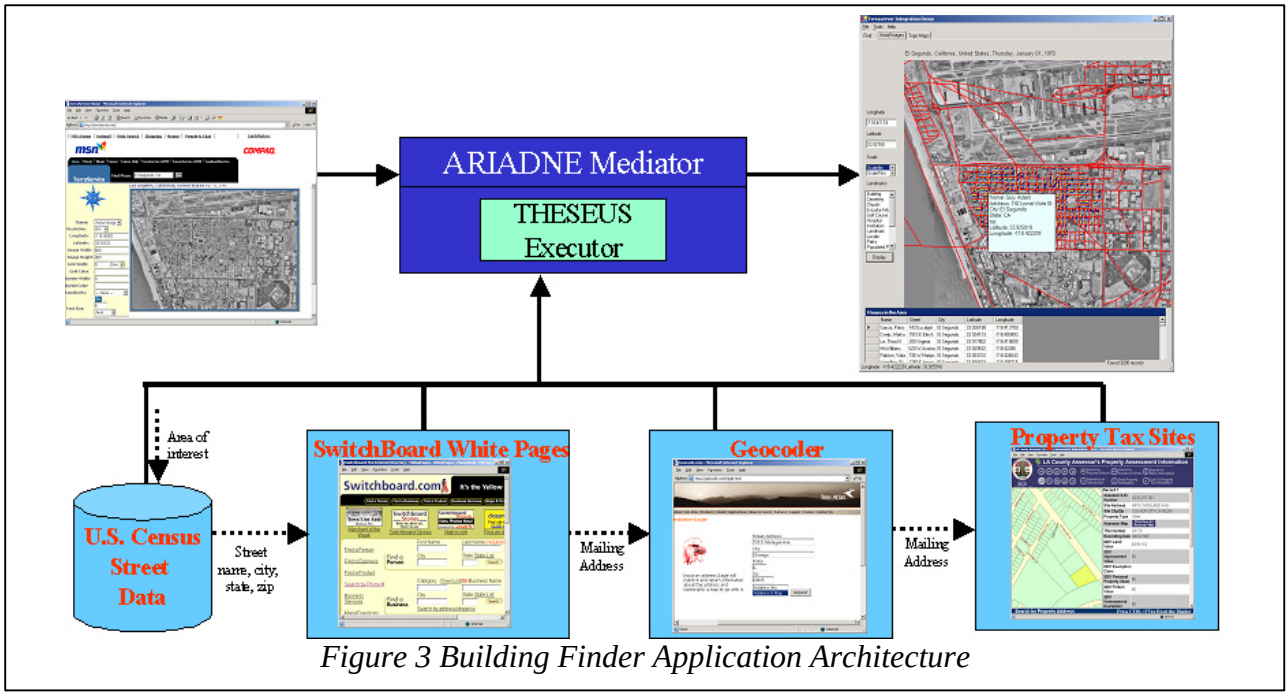

Figure 3 Building Finder Application Architecture

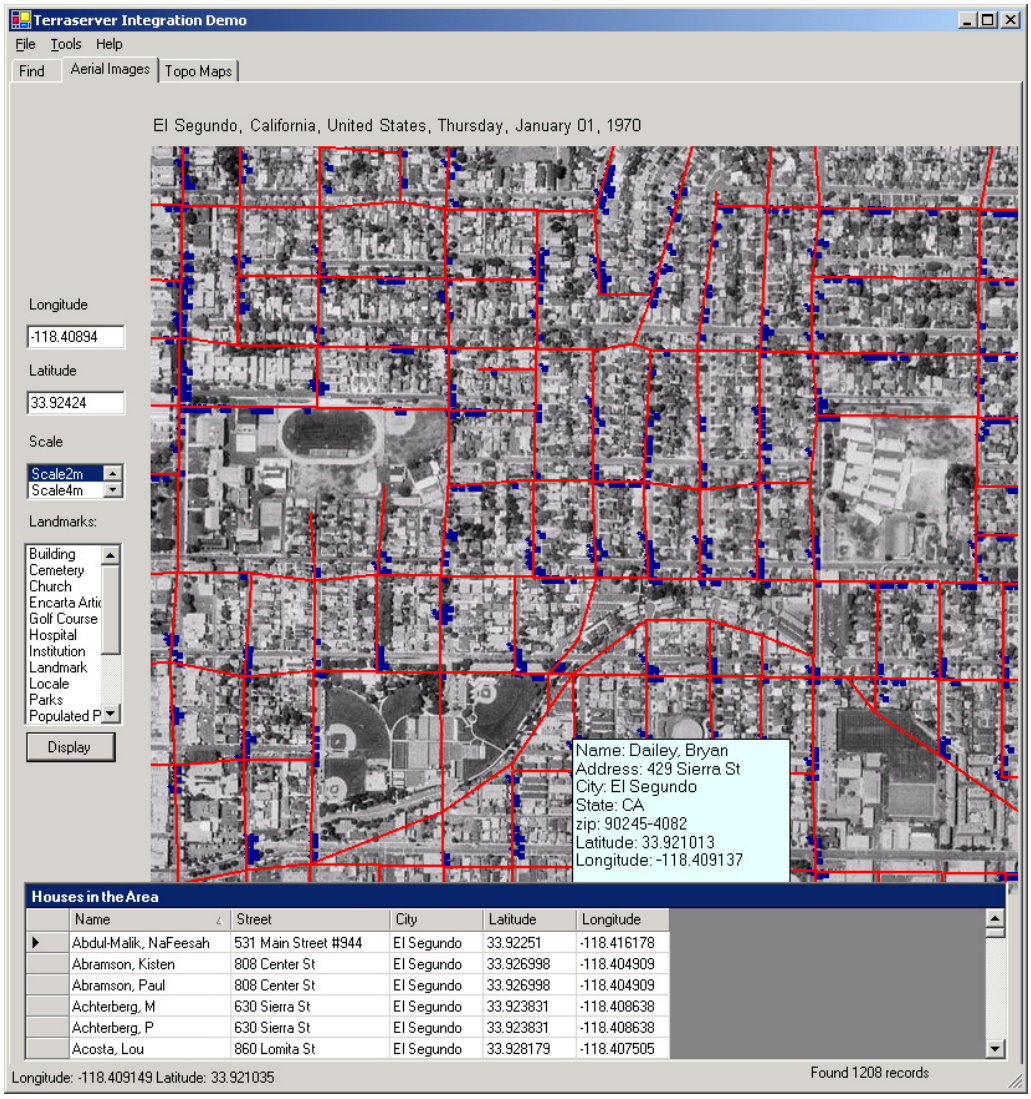

Figure 4 Screenshot of the Building Finder in Action

of the Switchboard white pages website is then provided to the geocoder agent, which provides the geographic coordinates for the addresses. The region of interest is also used to query Microsoft TerraService for a satellite image.

The Building Finder application finds the property values for all addresses found from the SwitchBoard White pages by querying different property tax sites. A naïve approach would be to send all addresses to all property tax web sites. However, this naïve approach requires a large number of queries to various property tax sources. The Building Finder application first finds the county for each address by providing city and state information from each address to the National Association of Counties (NACO) web site that provides the county information for each city. The county information is used to determine which property tax site should be queried for each address. The results from various property tax sites are combined to produce property value information for each address. Finally, the latitude and longitude points representing different addresses and information about the addresses are superimposed on the satellite imagery. Figure 4 shows a screen shot of the Building Finder application in action.

The Building Finder application provides the above-mentioned plan to the Theseus execution system, which efficiently executes the plan and obtains the results from all the web sources. The Theseus execution system utilizes a dataflow architecture to execute the independent steps in the integration plan in parallel. For example, Theseus can find satellite image from Microsoft TerraService in parallel with querying streets from the Tigerlines database. Furthermore, the Theseus execution system allows a wide range of access methods to different types of data sources [Barish et al. 2000].

4. Mediator-based Approach to Web Service Integration

The Building Finder application successfully integrates information from several web services and sources. However, the integration plan for the Building Finder application is pre-defined and very limited. A mediator-based approach can be utilized to dynamically compose new web sources similar to the Building Finder application. The Information integration literature includes a wide variety of mediator systems [Wiederhold 1996], such as the Information Manifold [Levy et al. 1996], InfoMaster [Genesereth et al. 1997], InfoSleuth [Bayardo Jr. et al. 1997], and Ariadne [Knoblock et al. 2001]. All of the above-mentioned systems can answer specific user queries by integrating information from various information sources, for example, find property values for all properties in Los Angeles, California where 'John smith' resides. The goal of the mediator framework described here is not only to answer the user queries, but also to generate a new web service that can answer similar user queries by integrating information from several web sources. So, the mediator framework needs to generate an integration plan for a template query, e.g. find property values in a given city where a given person resides. The mediator framework generates a new web service that can execute the integration plan for template query to answer similar user queries. The mediator framework utilizes the Theseus [Barish et al. 2000] execution system to execute the integration plans.

Section 4.1 describes how the information sources are modelled in the mediator's domain model. Section 4.2 describes the algorithm to reformulate user queries into source queries.

4.1. Modeling Web Sources

At the heart of any mediator system is the language utilized to describe the source models. The ability to define complex source models provides a good backbone for the mediator system. The mediator system built here utilizes source models similar to source models utilized by the Ariadne mediator system [Knoblock et al. 2001]. The sources are described by a set of binding patterns for the source, a set of inputs, a set of outputs provided by the source, and a set of constraints that describe the source. For example, a geocoder accepts a street address, a city, a state and a zip code as input parameters and provides latitude and longitude for the given address. In addition to the binding constraints a geocoder web source may have other constraints related to the data that the geocoder provides. For example, the geocoder may be able to only geocode addresses in the state of California. As a part of the source description the mediator models the constraints on the sources as well. Figure 5 shows the model of a geocoder web source that can geocode addresses in the state of California.

The mediator system built here utilizes an attribute-level ontology to relate different source models with the domain model. The attribute-level ontology provides a relationship between the attributes in the mediator's domain model and the attributes provided and accepted by the sources. The attribute level ontology provides a simple way to determine how the sources relate to the mediator's domain model and how the sources are related to each other. The attributes for each source also determine the class of the source, for example, the name attribute for a white page source that describes a person name, must be classified differently than the name attribute for a yellow page source that describes a company name.

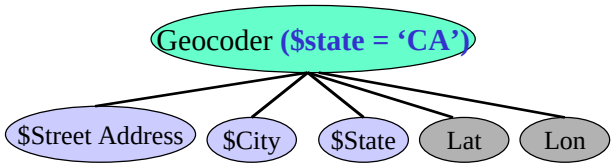

Figure 5 Source Model for Geocoder

4.2. Reformulating User Queries

The mediator can utilize the source models to reformulate user queries into the source queries. The mediator utilizes a forward chaining algorithm to construct an integration plan consisting of a set of source queries. Next, the mediator optimises the integration plan using a data flow analysis algorithm. Finally, the mediator utilizes the source constraints to filter the data at the tuple level.

4.2.1. Forward Chaining Algorithm

The mediator system receives a list of inputs provided by the user and a list of outputs requested by the user. The system initialises set of sources to empty set. First, the mediator finds all sources whose binding patterns can be satisfied with the inputs from the list of inputs, add those sources to the list of sources. Next, the mediator adds the outputs of the sources from previous step to the list of available outputs and list of inputs. Next, the mediator checks to see if all requested outputs are present in the list of available outputs. If all requested outputs are not present in list of available outputs the mediator repeats the process with a new list of inputs till all requested outputs are present in the list of available outputs. Figure 6 shows the forward chaining algorithm.

4.2.2. Dataflow Analysis Algorithm for Optimisation

In order to reduce the number of queries sent to the source, the mediator system must optimise the plan generated by the forward chaining algorithm. The mediator performs dataflow analysis to remove unnecessary source queries from the generated plan. The algorithm begins with set of outputs and finds all the sources that generate at least one output in the list of outputs. Next, the inputs to the selected sources are added to the list of outputs and the process is repeated till no more sources can be found in step 1. The

```
UI = Set of User Inputs
UO = Set of Requested Outputs
AI = Set of Available Inputs
AS = Set of Available Sources
AO = Set of Available Outputs
ForwardChaining(UI, UO)
1. AI = Φ, AO = Φ, AS = Φ
2. AI = UI
3. {∀S(I,O,C) | ( I n  AI = Φ)} Repeat
     4.      AI += O
     5.      AO += O
     6.      AS += S
7.          If (AO  n  UO != Φ) Go to 3
```

Figure 6 Forward Chaining Algorithm

resulting list of sources is the minimal set of sources that answers the user query.

4.2.3. Tuple Level Filtering

In addition to the dataflow analysis, the mediator also performs tuple level optimisation. The mediator starts the tuple level optimisation with a list of sources and source constraints for each source. The mediator inserts a filtering condition before each source query to filter out the rows that do not meet the source constraints. For example, in the case of the geocoder that can only geocode addresses in California, the mediator inserts a filtering condition that filters out all addresses that are not in the state of California. This feature can be very useful when the several sources provide similar data, but cover different regions. In the case of the integration plan for the Building Finder application tuple level filtering is very useful. If the user query were to find properties for all people with name 'John Doe', the traditional mediators would generate an integration plan where all property tax sources will be queried for all records found from the SwitchBoard White Pages.

Our mediator framework, analyses the source models for different property tax sources. The source models for the property tax sources have constraints based on the county, for example, the Los Angeles County property tax source has a constraint that the input address must be in Los Angeles County. The mediator determines that the NACO web source can provide county information for each address and queries the NACO counties site to find the county information for each address. The mediator introduces a filter that routes the addresses from the SwitchBoard white pages to different property tax sites based on the county information, similar to the discrimination matrix described in the [Ashish et al. 1997]. Figure 7 shows a fragment of the integration plan where the mediator framework generates a plan that queries the Los Angeles property tax source for all records found from the SwitchBoard that are in the county of Los Angeles, queries the San Francisco property tax source for all records found

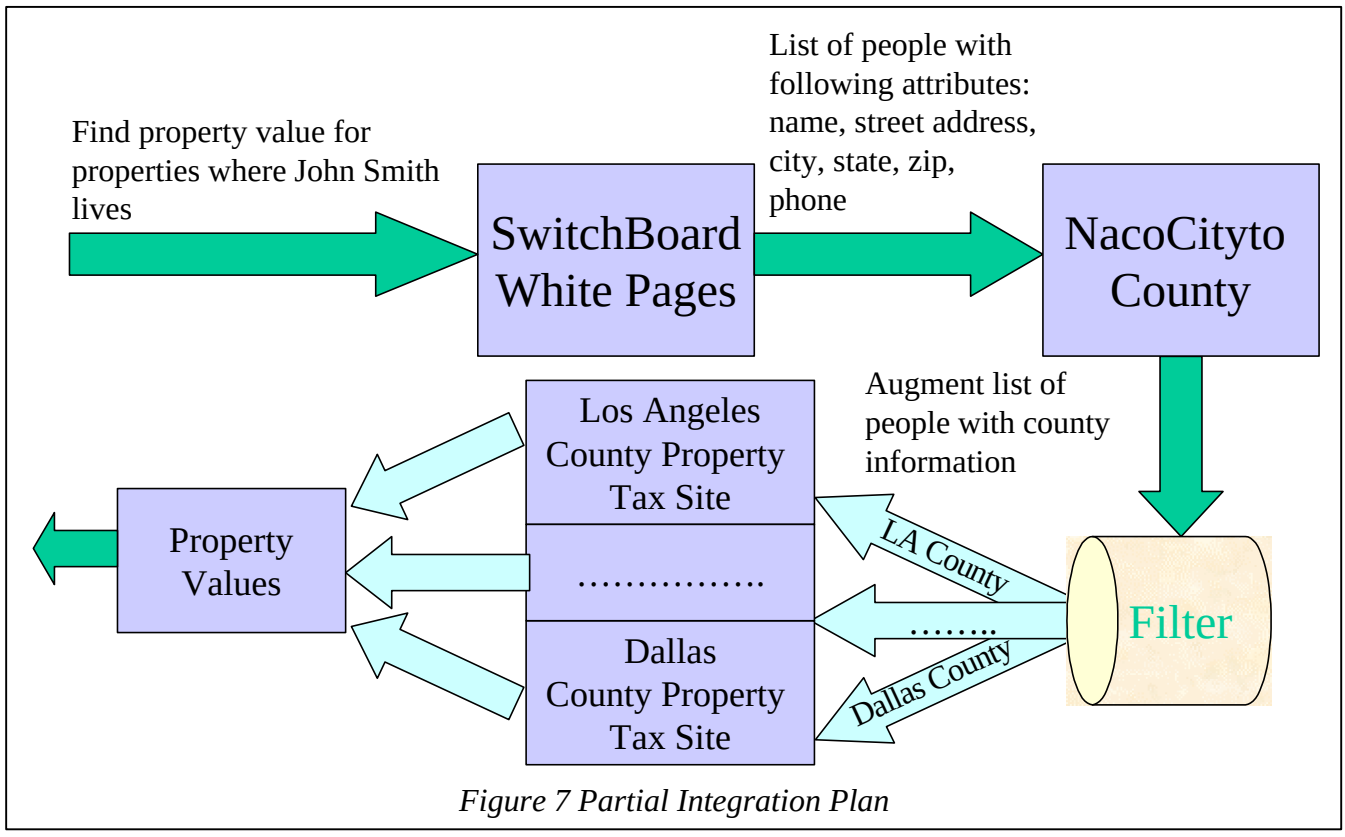

Figure 7 Partial Integration Plan

from the SwitchBoard that are in the county of San Francisco and so on.

The plan generated by the mediator is a universal integration plan, similar to the universal plans described in the [Shoppers 1987], that can answer not only the user query, but also all queries with similar structure. For example, the integration plan shown in Figure 7, can answer the query find property values for properties where the given person lives. Once the universal integration plan has been generated it can be used to generate a new web service for all possible values of the input parameters.

5. Discussion

In this paper, we described the Building Finder application, which integrates information from several web services to find buildings in satellite imagery and related information for the buildings. We described a mediator framework that models the web services as information sources and dynamically composes new web services, such as the Building Finder by generating a universal integration plan to answer the user queries.

Acknowledgement

This research was supported by an un-restricted cash gift from Microsoft Research.

References

[Arens et al. 1996] Arens, Y., Knoblock, C. A. and Shen, W. M. 1996. Query Reformulation for Dynamic Information Integration. *Journal of Intelligent Information systems, Special Issue on Intelligent Information Integration* **6**(2/3): 99-130

[Ashish et al. 1997] Ashish, N., Knoblock, C. A. and Levy, A. 1997. Information Gathering Plans with Sensing Actions. *European Conference on Planning, ECP-97*, Toulouse, France

[Barclay et al. 1999] Barclay, T., Gray, J. and Stuz, D. 1999. Microsoft Terraserver: A Spatial Data Warehouse, Microsoft Corporation

[Barish et al. 2000] Barish, G., DiPasquo, D., Knoblock, C. A. and Minton, S. 2000. A Dataflow Approach to Agent-Based Information Management. *In Proceedings of the 2000 International Conference of on Artificial Intelligence*, Las Vegas, NV

[Bayardo Jr. et al. 1997] Bayardo Jr., R. J., Bohrer, W., Brice, R., Cichocki, A., Flower, J., Helal, A., Kashyap, V., Ksiezyk, T., Martin, G., Nodine, M., Rashid, M., Rusinkiewicz, M., Shea, R., Unnikrishnan, C., Unruh, A. and Woelk, D. 1997. Infosleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments. *In Proceedings of ACM SIGMOD-97*

[Genesereth et al. 1997] Genesereth, M. R., Keller, A. M. and Duschka, O. M. 1997. Infomaster: An Information Integration System. *In Proceedings of ACM SIGMOD-97*

[Knoblock et al. 2001] Knoblock, C., Minton, S., Ambite, J.L., Ashish, N., Muslea, I., Philpot, A. and Tejada, S. 2001. The Ariadne Approach to Web-Based Information Integration. *International Journal on Intelligent Cooperative Information Systems (IJCIS)* **10**(1-2): 145-169

[Knoblock et al. 2001b] Knoblock, C. A., Lerman, K., Minton, S. and Muslea, I. 2001b. Accurately and Reliably Extracting Data from the Web:A Machine Learning Approach. *IEEE Data Engineering Bulletin* **23**(4): 33-41

[Levy et al. 1996] Levy, A. Y., Rajaraman, A. and Ordille, J. J. 1996. Query-Answering Algorithms for Information Agents. *In Proceedings of AAAI-96*

[MacGregor 1988] MacGregor, R. 1988. A Deductive Pattern Matcher. *In the Proceedings of the Seventh Nation Conference on Artificial Intelligence*, Saint Paul, Minnesota

[Shoppers 1987] Shoppers, M. 1987. Universal Plans for Reactive Robots in Unpredictable Environments. *Proceedings of the InternationalConference on Artificial Intelligence, IJCAI-87*

[Wiederhold 1996] Wiederhold, G. 1996. Intelligent Integration of Information, Kluwer