

BIOGRAPHICAL INFORMATION

Andreas Matheus
Assistant Lecturer
Technische Universität München
Munich, Germany

Specific Responsibilities

2001 – today: Assistant lecturer at the Technische Universität München – research in the field of geospatial information systems; especially Geo Web Services and OpenGIS.

Past Experience

1995 – 1997: Siemens, Germany - software development and engineering for client server software in the field of production and logistics

1997 – 1998: Siemens, US – software development and proposal work for facility automation in the field of production and logistics

1998 – 2000: Siemens, Germany – project lead for a research program with team members in Germany and the US

Educational Information

Dipl.-Ing.: Study of Electrical Engineering at Gerhard-Mercator-Universität, Germany

Professional Memberships

OpenGIS Consortium



SEMI-AUTOMATIC ORCHESTRATION OF GEO WEB SERVICES

Andreas Matheus
Technische Universität München
Fakultät für Informatik
Boltzmannstr. 3
D-85747 Garching, Germany
Email: matheus@in.tum.de

ABSTRACT

Web Services are prayed to become the new building blocks of tomorrow's internet. Web Service (re)use, which is based on the register-find-bind paradigm, is the basic technique here. Unfortunately, developing complex applications by orchestrating Web Services requires more meta-information than provided by WSDL and UDDI today. This paper introduces a Web Service Orchestration Engine (WSOE) that supports the developer by building applications from orchestrating Geo Web Services. Different to other approaches, the WSOE does not rely on classifying/describing the service functionality. The foundation of the WSOE is a knowledge base that contains concepts, rules and facts for the geographic problem domain. They are used by service developers to describe the meaning of input and output messages and by the application developers, trying to build an application from orchestrating the existing Geo Web Services. The application developer has to specify the problem that describes the functionality of the application using the constructs from the knowledge base. The WSOE takes this problem description and solves it by creating an orchestration of sub-problems until been solved by a single Web Service. If that orchestration of services is a solution to the specified problem, it can be used for developing the application.

This paper introduces research in progress.

INTRODUCTION

One statement about Web Services is that "Web services are self-contained, modular business applications that have open, Internet-oriented, standards-based interfaces. Web services communicate directly with other Web services via standards-based technologies." (OASIS, 2001). A Geo Web Service can be characterized as a Web Service for processing or accessing of geospatial data. "Geospatial data" means information that identifies the geographic location and characteristics of natural or constructed features and boundaries on the earth. This information may be derived from, among other things, remote sensing, mapping, and surveying technologies..." (FGDC, 1994). Geo Web Services support, as regular Web Services do, the register-find-bind paradigm. When it comes to build more complex services on the base of the registered Geo Web Services, the user is facing the challenge to understand the

functionality of the service (what it does) and how to use it (how to invoke it). For a human user, this can be achieved by any description, registered with the service as supported by the Universal Discovery, Description and Integration (UDDI). For additional information, the potential consumer of the service can contact the producer of the service. But, the potential consumer can only understand the service, if the producer and the consumer share a common (domain specific) knowledge and speak (understand) the same language. The problem of sharing the same meaning is known as the rhetorical theory “triangle of meaning”, as introduced by C.K. Ogden and I.A. Richards (Richards, 1979). Therefore, the development of more complex services require not only an explicit description of the registered services and the resources been processed but also a common, domain specific knowledge that can be shared and understood between humans.

STANDARDS FOR (RE)USING WEB SERVICES

For the approach to have a machine, assist a human user with orchestrating services to more complex applications, the description and the shared knowledge must not only be machine readable but also machine understandable. In the field of service discovery, description and integration the most important standards are the Universal Description, Discovery and Integration (UDDI) (OASIS, 2001) and the Web Services Description Language (WSDL) (W3C, 2001). "WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information" (W3C, 2001). This message based view of a Web Service focuses on 'how' a Web Service can be executed and which messages are exchanged with the service. "A common analogy used for UDDI is a “phone book for Web services.” It has business names, business mailing addresses, contact names, contact phone numbers, Web services offered by businesses, addresses of Web services, metadata describing the “interfaces” of Web services, etc." (OASIS, 2001). However, none of these standards address to define the meaning of the service’s functionality, the input parameters and the output. Furthermore, the standards do not allow to define the coordination of sequencing of services to build a complex solution. Two major standards, addressing this issue are the Web Services Flow Language (WSFL) (IBM, 2001) and XLANG. But, both standards focus on the sequencing of services according to a known process flow, where the service bindings and functionalities are known a priori. Generally, this is not the case if it comes to build a solution to a user specific problem by orchestrating services, even they belong to one problem domain.

Recent activities in the field of defining meaning to resources, are known under the term Semantic Web. "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." (Lee, 2001). The formalization of meaning can be expressed using an ontology, which is a part of the Semantic Web. Within the Artificial Intelligence (AI) context one can describe "the ontology of a program by defining a set of representational terms. In such an ontology, definitions associate the names of entities in the universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these terms." (Gruber, 1993).

The semantic markup for internet resources, especially for Web Services is important for the approach of semi-automatic orchestration. The use of an ontology allows to explicitly express the meaning of Web Services and the data been processed. Technically, an ontology can be described by using the Resource Description Framework (RDF) and the Defense Advanced Research Projects Agency (DARPA) Markup Language (DAML). Extension to DAML which uses expressions from the Ontology Inference Language (OIL) result in DAML+OIL. "The World Wide Web was originally built for human consumption, and although everything on it is machine-readable, this data is not machine-understandable. It is very hard to automate anything on the Web, and because of the volume of information the Web contains, it is not possible to manage it manually. The solution proposed here is to use meta-data to describe the data contained on the Web." (W3C, 1999). The metadata it's underlying data model is structured in triples (subject, predicate and object) stating facts and relationships by defining objects and properties of the objects and relationships between the objects. "The RDF data model, however, provides no mechanisms for declaring these properties, nor does it provide any mechanisms for defining the relationships between these properties and other resources. That is the role of RDF Schema" (W3C, 1999). The data model is represented using the Extensible Markup Language (XML), which makes the specification of data semantics available in a standardized, interoperable manner. The DARPA Markup Language (DAML) is an extension to XML and RDF, providing "a rich set of constructs with which to create ontologies and to markup information so that it is machine readable and understandable." (DARPA). "The Ontology Inference Layer OIL is a proposal for a web-based representation and inference layer for ontologies, which combines the widely used modeling primitives from frame-based languages with the formal semantics and reasoning services provided by description logic. It is compatible with RDF Schema (RDFS), and includes a precise semantics for describing term meanings (and thus also for describing implied information)." (On-To-Knowledge). "DAML-S supplies Web service providers with a core set of markup language constructs for describing the properties and capabilities of their Web services in unambiguous, computer-interpretable form. DAML-S markup of Web services will facilitate the automation of Web service tasks including automated Web service discovery, execution, interoperation, composition and execution monitoring." (DAML-S).

The approach of semi-automatic orchestrating of Web Services requires the formalization of the semantics of Web Services functionality and the data been processed in a machine-interpretable way. It is important to "describe what is being sent across the wires and why, not just how it is being sent." (Anupriya et al., 2002). Therefore, semantic service descriptions and domain-specific knowledge are the basis for a decision support system, as introduced in this paper.

SEMI-AUTOMATIC ORCHESTRATION OF GEO WEB SERVICES

Software developing activities often focus on implementing a solution to a given problem, using existing modules. The functionality and interfaces of these modules are typically described by manuals, which are available to the programmer. These

modules usually provide fine grained functionality, which can be used to solve almost any problem. Web Services usually provide a higher functionality, which is more suitable for solving business logic problems. Therefore, the orchestration of Web Services will not be able to solve any problem at hand. The orchestration of registered Geo Web Services focuses on the solving of high level (business) problems such as image map requests or geocoding of spatial objects. For this field of use, the finding of a service workflow is domain specific and problem depended. Because the finding of an appropriate service for a particular position in the workspace may depend on the result of the execution of the previous service, the workflow can not be modeled stationary. The finding of a solution to a given problem can only be found by using domain knowledge and semantic information for the service and the processed data.

A motivating example

The user of the system likes to know the current temperature for the City of Munich. Even this seems to be a simple problem, the solution can probably not be found by software developing using a common programming language. The reason for this is that the temperature sensor for the location Munich must be accessible. Therefore, the use of Web Services (and the Internet) is a prime example how to solve the problem. There might exist different registered services that provide different temperature kinds (e.g. current temperature, average temperature, etc.) or return the temperature in different units of measure (e.g. Celsius or Fahrenheit) for just one location, an area or for multiple locations.

In order to find a solution to the problem, the user can define his input knowledge: Location is Munich; and define the expected meaning of the output: Current temperature in Celsius. Based on this simple description, a solution would be all services and service orchestrations, which input means location is Munich and the output is temperature in Celsius.

High-level description of the system

For the approach of semi-automatic orchestration of Geo Web Services, each service is described by the semantics of the input and output data and the restrictions on the input and output data using concepts of the domain knowledge base. The user specifies the problem by using concepts from the knowledge base, identifying the input and output of the problem. Also, the user may place restrictions on the input and/or output, using available restrictions for the used concepts. The Web Service Orchestration Engine (WSOE) supports the user in the process of finding a solution to the specified problem by recommending services, which may be used in the solution-workflow. Each recommendation comprises of a matching factor that describes the fitness of a suggested service and a link to the service and I/O-data description. From this information, the user can obtain information, if more than one service is recommended and the user is asked to select the most appropriate service. The problems that can be solved with this approach are limited to the concepts, relationships and rules within the knowledge base for the geospatial problem domain.

The knowledge-base

The knowledge base contains of concepts that describe the domain of discourse, facts that express restrictions on concepts and rules that regulate which constraints must apply when matching two concepts in the process of solution finding.

The concepts of the knowledge base represent semantics of things that belong into the domain of discourse. In the geospatial problem domain, this can be features, which may have geospatial characteristics that represent entities of the real world. Each concept comprises of a machine-readable serialization, and a human-readable description. The concepts can have relationships to other concepts. These relationships can be of different kind. In the geospatial problem domain, concepts and relationships can be of spatial kind. This allows two potential perspectives: Concepts and relationships that describe a feature or the relationships between them are based on a non-spatial model and concepts, which represent different spatial models. However, all these models must use standard relationships to express generalization and specialization, using the *isA* relationship. In order to represent spatial relationships, domain specific spatial relationships must be defined. Egenhofer et al. (Egenhofer, 1998) introduces different spatial operators that must be used in order to allow appropriate expression of spatial problems.

Facts allow to express restrictions to formalize the problem at hand and the service capabilities. An example for using facts is that the problem exists to find a service that returns the current temperature for a particular city, e.g. Munich. Because no solution is acceptable that finds the temperature for all cities but for Munich, the solution requirement and the service capability is evaluated in order to find a match. Using a problem solving algorithm, the fact of the solution requirement must be provable using the facts that express the capabilities of services.

The requirements for matching-rules enable the expressing of constraints that must be taken under consideration when matching two concepts. For example, the chaining of two services is based on the concept Location. Location specifies a 0-dimensional position relative to the earth. But, the chaining does only make sense, if both service use the same encoding. This requirement can be expressed using a matching rule.

FINDING SERVICE RECOMMENDATIONS

Each service is described by it's input/output data semantics, describing concepts and facts from the existing knowledgebase. The use of the concepts support the sharing of common knowledge. The facts allow the formalization of restrictions on the concepts. E.g., the semantics of 'the current temperature of the city of Munich' can be described by the concepts *CurrentTemperature* and *City* with the restriction *Munich* on the concept *City*. In the knowledge base, different concepts can and facts can represent the city of Munich. Therefore, inference logic exists that can resolve different representations of the city of Munich. In the geospatial problem domain, a different representation can exist using the geospatial decomposition of space. In one model, Munich can be presented as a point feature, in other models it can be represented by 2-, 3- or even 4-dimensional features. However, all concepts have relationships

among each other which can be followed in order to examine the fitness of a matching.

Before the system starts the match-making, which results in service recommendations, the following preconditions must be true:

1. The system can access the problem specific knowledge base that contains concepts, relationships between the concepts, rules and facts about the problem domain and registered services.
2. Each registered service uses a set of concepts and facts from the knowledge base to specify the meaning of the input and output of the service.
3. The input that the system requires is the formalization of the problem, done by the user. The formalization specifies the concepts and restrictions for the initial and goal state of the problem. For the motivating example, the user likes to receive the current temperature of the city of Munich. The initial state can be described using a concept *City* and place the restriction *Munich* on it. This results in the fact *City(Munich)*. The goal state is represented by the concept *CurrentTemperature*.

As the first step, the system takes the problem initial state and tries to prove it. Therefore, the knowledge base is queried for all services, which input concept build a path to the initial problem concept. For each matching service, the facts for that service are evaluated, which describe the capabilities of the service. For the example, all services are marked that have a path to the concept *City*. But, only these services are recommended to the user, which capabilities meet the condition *City(Munich)*.

Next, the system tries to prove the problem goal state. In order to do so, all services are marked, which output description builds a path to the concept *Temperature*.

A solution to the given problem exists, if one service is found that allows to prove the problem initial and goal state.

If only services exist that allow to prove either the problem initial or goal state, the system starts a second iteration. In order to do so, a set of new initial and goal states is build. The set of new initial states depend on the found services that allow to prove the initial problem state and the set of goal states depend on the found services that allow to prove the problem goal state. For each possible combination of a new intermediate initial and goal state, the system repeats the first step; trying to prove the intermediate states. Because the variation of state combinations gets very big, the user must select the services from the first matching that are to be processed first. Because the number of intermediate states depend on these services, the variation can be kept small.

Solution kind one: Perfect solution using one service

For this solution, let's assume that one service exists that allows to prove the problem initial and goal state. For the example, this would be the case if the service input provides the fact *City(Munich)* and the output is linked to the concept *CurrentTemperature*. This solution would be recommended to the user with a 100/100% match.

Solution kind two: No perfect solution

For this solution, let's assume a service exists that allows to prove the problem initial state, but the output is linked to the concept *Temperature*. Now, it depends on the relationship between the concept *Temperature* and the concept *CurrentTemperature*. If the knowledge base contains an *isA* relationship between the concepts *Temperature* and *CurrentTemperature* (*CurrentTemperature isA Temperature*), the match on the conceptual level is 50% because the distance between the concepts is one. This solution is recommended to the user with a 100/50% match.

Solution kind three: Service chaining and matching rules

For this solution, let's assume a service exists that allows to prove the problem initial state, but the output is linked to the concept *Location* which has no relationship to the concept *Temperature*. Also, another service exists that allows to prove the problem goal state and the input meaning is linked to the same *Location* concept that the other service is linked to. The knowledge base contains a matching rule for the concept *Location*. It requires that both services use the same encoding and provide service capabilities in respect to the service area. In order to determine the fitness of both services, the system must prove that (i) the location for the city of Munich, provided by the first service uses the same encoding and (ii) that the location of Munich is within the service area of the second service. For the encoding rule, this can be proven by checking the facts for both services, describing each possible encoding. The system uses the spatial operator *within*, in order to prove that the location of Munich, encoded by the first service is accepted by the second service.

MODELLING THE KNOWLEDGEBASE

One requirement of the system is to match concepts. For this matching process, only concepts from the knowledgebase exist, which specify the domain of discourse. Each concept may have relationships with other concepts.

Matching rules

The system supports four different types of matches:

1. Two concepts are identical. In this case a 100% match exist.
2. A path between the two concepts exist that follows the *isA* hierarchy in the generalization direction. In this case, the distance between the concepts can be taken to determine the appropriateness of the match. E.g. the concepts *CurrentTemperature* and *AverageTemperature* specialize the concept *Temperature*. If a service output's semantic is described by the *Temperature* concept but the problem output concept is the *CurrentTemperature* concept, the matching correctness is 50%.
3. A path that exists between the concepts that follows the *isA* hierarchy in the specialization direction is not allowed and results in a 0% match. This can be explained by the hierarchy of the concepts. The topmost concept is *Thing*. Everything is a thing and therefore fits everywhere. But, all things that make

things more special may assume (implicit or explicit) restrictions. In the above example, there is no semantically match between the concepts *AverageTemperature* and *CurrentTemperature* because the path follows generalization and specialization.

4. A path between two concepts exist that do not follow the *isA* relationship. In this case, the matching depends on the role of the relationship. E.g., the role *synonym* is treated as an identical match.

Spatial relationships and spatial concepts

From these thoughts, the following guidelines for modeling a geospatial knowledge base can be derived:

1. The knowledge base must differentiate between spatial and non spatial concepts. The spatial concepts can be categorized into 0-, 1-, 2-, or 3-dimensional concepts.
2. Based on these concepts, more detailed concepts can be derived that describe the domain of discourse.
3. A spatial operator can apply to instances (individuals) of spatial concepts, stating a relationship between two instances or a restriction on the instance. As an example, two instances of the concept *LandParcel* (*landparcelA* and *landparcelB*) can be adjacent to each other. If *AdjacentTo* is a transitive spatial relationship, this relationship can be defined by the relationship *AdjacentTo* (*landparcelA*, *landparcelB*).
4. The knowledge base must contain matching rules for concepts. These matching rules define restrictions that must be met when match-making is done with the associated concepts.

CONCLUSION AND OUTLOOK

This paper introduced a simple Web Service Orchestration Engine that may find a solution to a given problem by orchestrating Web Services. The introduced system is not limited to solve problems of the geospatial problem domain; the solving power depends on the existing knowledge base and the registered services. The approach uses the input and output state description for services and the initial and goal state description for the problem. In the process of finding a solution, the system tries to proof the initial and goal state of the problem. The functionality of the system was explained by a simple example.

Because the current approach finds a solution -if existing- based on the semantic description of the processed data only; the syntactical issues of interoperability of the services (data structures, protocol binding, etc.) are not taken into account. However, these issues become relevant when invoking the solution services. Therefore, including these issues in the solution finding process is a relevant topic for future work.

Another topic of research includes the comparison of the current problem description with existing, saved problem solutions, found by other users. The system can propose a solution to the current user, before starting the process of trying to create a new solution.

BIBLIOGRAPHIC

- Ankolekar, Anupriya et al. *Anupriya Ankolekar, Mark Burstein, Jerry R. Hobbs, Ora Lassila, David Martin, Drew McDermott, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terry Payne, Katia Sycara, DAML-S: Web Service Description for the Semantic Web*
- DAML *Defense Advanced Research Projects Agency, About the DAML Language, <http://www.daml.org/about.html>*
- DAML+OIL *Defense Advanced Research Projects Agency, <http://www.daml.org>*
- DAML-S *DAML Services, <http://www.daml-s.org/>*
- Dublin Core *Dublin Core Metadata Element Set, Version 1.1: Reference Description, 1999-07-02*
- Egenhofer, Max et al. *Max Egenhofer, Reginald G. Golledge, Spatial and temporal reasoning in geographic information systems, Oxford Univ. Press, 1998*
- FGDC *Executive Order 12906, Published in the April 13, 1994, edition of the Federal Register, Volume 59, Number 71, pp. 17671-17674*
- Fensel, Dieter et al. *Dieter Fensel, V. Richard Benjamins, Enrico Motta, Bob Wielinga, UPML: A framework for knowledge system reuse*
- Gruber, Thomas R. *Toward Principles for the Design of Ontologies Used for Knowledge Sharing, Revision: August 23, 1993*
- Helman Paul et al. *Helman Paul, Robert Veroff, Walls and Mirrors, Intermediate Problem Solving and Data Structures, Modula-2 Edition, The Benjamin Cummings Publishing Company, INC., 1988*
- IBM *Frank Leymann, Web Services Flow Language (WSFL 1.0), May 2001*
- Lee et al. *Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001*
- Luger, George F. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 4th edition, Pearson Educated Limited 2002*
- Nilsson, Nils J. *Problem-solving methods in artificial intelligence, McGraw-Hill Book Company, 1971*
- Nilsson, Nils J. *Principles of Artificial Intelligence, Springer-Verlag, 1982*
- OASIS *UDDI Executive White Paper, November 14, 2001*
- OIL *Welcome to OIL, <http://www.ontoknowledge.org/oil>*
- OWL *Web Ontology Language 1.0, <http://w3c.org/TR/owl-ref>*
- Pearl, Judea *Heuristics, Intelligent Search Strategies for Computer Problem Solving, Addison-Wesley Publishing Company, 1984*
- W3C *Web Services Description Language (WSDL) 1.1, 15 March 2001*
- W3C *Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation 22 February 1999*
- W3C *Resource Description Framework (RDF) Schema Specification, W3C Proposed Recommendation 03 March 1999*
- W3C *Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000*
- W3C *Annotated DAML+OIL Ontology Markup, W3C Note 18 December 2001*
- Richards, I.A. *I. A. Richards, B. 1893-D. 1979, <http://bradley.bradley.edu/~ell/iarichar.html>*