Ad-Hoc Coordination and Query Execution Plan

Hillary Caituiro-Monge *(hcaituiro@ieee.org)*, Manuel Rodríguez-Martinez *(manuel@ece.uprm.edu)*
Department of Electrical and Computer Engineering
University of Puerto Rico at Mayagüez

Abstract

Next-generation Information Systems will integrate large amounts of heterogeneous data sources located on distributed networks like the Internet. We present a peer-to-peer autonomic environment in which any participating government site can request or serve data, and must engage in a collaborative effort aimed at satisfying the requests for data and services associated with the queries posed by interested end-users. We call this type of scheme autonomic web services collaboration, orchestration, and choreography in electronic government information systems. The main feature of our new approach is the elimination of a central coordination site running the queries and the autonomic query execution. Instead, Web services become lightweight coordination services that can be run from a client or data source site. Moreover, control information is embedded with the request for data, and also with the partial results. We present the architecture of this framework, the ideas behind it, and a performance study to validate it. The results of this study demonstrate that this framework provides a more flexible, scalable and efficient framework for web services collaboration in electronic government applications.

1. Introduction

The emergence of wide-area networks such as the Internet has provided users with access to vast amounts of rich data sets that are located on data sources distributed over these networks. Database Middleware Systems were developed to integrate heterogeneous data sources distributed over a computer network. Web services play a critical role in the integration of distributed systems over wide-are networks. They need to collaborate in order to execute complex requests. To do so, web services are statically linked for purposes of fulfilling some goals such as performance, scalability and reliability. Also, the execution of requests triggers dynamic relationships among them. Referring to web services the set of static relationship among them is called orchestration. [6] The process to build dynamic relationship among web services is called choreography. **[2]** [3]

Next-generation **I**nformation **S**ystems (NIS) of organizations will incorporate large amounts of heterogeneous distributed information sources accessible through web services. Each entity (e.g. government agency) will have its own set of information systems and policies to access them. The information sources might be located on servers, desktop computers, mobile devices, or embedded computer systems. Orchestration and choreography between these information sources will be a critical requirement to collect the vast amounts of valuable information.

The main feature of our new approach is the decentralized coordination through lightweight web services that can be run from a client or information source site. Notice that a central coordination site simply cannot serve well to all the different kinds of applications and information sources that will be found in a NIS. Additionaly, control information should be embedded with the request for data, and also with the partial results. This feature is necessary to avoid imposing a connection-oriented type that this would be unfeasible for mobile devices. Instead, our approach is to include control information in an XML control document that is attached with a request for a service, or with partial results. This XML document indicates the next service to be requested, the possible target site(s) and how should it process any partial results.

The remainder of this paper is organized as follows. Section 2 motivates the need of Net Traveler. Section 3 presents the related work. Section 4 describes the architecture of Net Traveler. Performance evaluation is presented in section 5. Finally, the paper conclusions and future work is presented in section 6.

2. Motivation and example scenario

NIS in organizations (e.g. Government) poses a great number of challenges to computer community that is working to build transparent, scalable, and reliable Information Systems (IS) to allow end-users to execute queries that need access to heterogeneous and geographically distributed databases. End-users usually request information that includes several entities. ISs are independently designed and implemented, so it is so difficult to get them together to work. In this context, it is necessary to gather the information from several ISs among different entities. Thus, the orchestration and choreography of the ISs will allow an efficient and effective result. Moreover, ISs need to be accesses through a standard and universal middleware. Web services fulfill such requirements, since they are broadly used in wide-area networks (e.g Internet).

An example to use through this paper is an E-Government process for auctions. It consists of many government entities and government officials which work following a sequence of steps to complete the auction process. An official starts the process making a request of some products (e.g. computers). Such requirement falls under the hands of an official in charge of coordinating the complete auction process. This last builds an auction plan and sends pertinent communications and instructions to several non-government and government entities. The non-government entities which receive the communication are the providers which have the requested product. They gather information from its own registers and send a proposal to an evaluation office in charge of collect and evaluate the proposals. Finally, an entity that is in charge of making orders and notifying about the result sends a final message to the winner and to the original requester.

3. Related Work

The Web Service Choreography Interface (WSCI) **[1]** is an XML-based interface description language. It describes the flow of messages exchanged describes the flow of messages exchanged by a Web Service participating in choreographed interactions with other services. WSCI describes the observable behavior of a Web Service. There are three differences with this standard. The first difference is that it covers only choreography and does not address orchestration. The second is that this does not implement a framework. And, the third difference is that it does not give support for autonomic choreography.

Pahl et al. **[2]** addresses web service choreography proposing a formal ontology framework for web services that supports the description, matching, and composition through logic reasoning techniques. Although this work covers choreography, it does not address autonomic orchestration.

Peltz [3] published "Web Services Orchestration and Choreography" that is about combining Web services to create higher level, cross-organizational business processes. Peltz's work is almost the description of a model for orchestration and choreography. We gave a strategy, implement it, and conclude based on a performance evaluation.

4. System Architecture

Figure 1 depicts the different elements contained in the architecture of the proposed framework. The information sources are represented by computers with cylinders, while each rectangle represents a local execution environment.

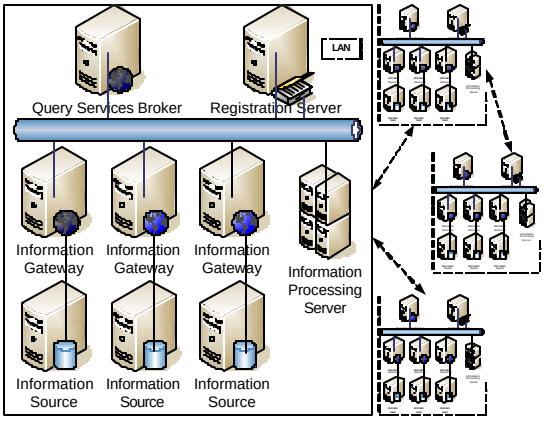

Figure 1: Architecture

The local execution environment represents the confines of a government agency. The purpose of each of the components is as follows:

- **Information Gateway (IG)**. Provides access to other members of the system to the data sets maintained by a given source site.
- **Query Services Broker (QSB)**. Works on behalf of a client site (which can be a data source site that become a client) to find the data and query processing capabilities necessary to solve a given query. This element has responsibilities such as query plan generation, query coordination, and acquisition of final results. The Data Broker can access data sources that lay outside its local execution environment.
- **Information Processing Server (IPS)**. Provides a commodity service for query execution. This is an optional element that will be most likely used in environments that require parallel processing capabilities, or which have many low powered devices.
- **Registration Server (RS)**. This server is used to coordinate access to a federation and to the resources that it provides.

4.1. Federation model

The fundamental organizational unit in the system is the ad-hoc federation of peer government sites, which is a collection of computing devices that have agreed (or are configured) to work together as a group to exchange data, and share computational resources to complete a related task. A federation can span one ore more local execution environments. Federations are ad-hoc because they can be formed or dissolved over time based on the decisions taken by the member data sources. As a special case, a permanent federation is one that is never dissolved and corresponds to the type of federations assumed by existing middleware solutions. Permanent federations

(PF) are composed for such NIS whose domain and relationships are highly coupled and are frequently used. Ad-hoc federations (AF) are composed for such NIS whose domain and relationships are loosely coupled and unpredictable. Our system exhibits Peer-to-Peer behavior in the sense that a given data source site can be serving data to one or more sites, while it retrieves data from others (not necessarily at the same time).

4.2. Ad-hoc coordination and query execution plan model

In ad-hoc coordination (AC), knowledge about sites might be partially known or simply not known at all. Thus, AC starts with incomplete information that is subsequently completed by other web services. Under these circumstances of uncertainty, centralized coordination does not work neither effectively nor efficiently, since it will need to gather information from far located peers several times in order to complete an execution plan. Even worse, it could not complete the plan due to the lack of information in its directory. Decentralized coordination requires that peers collaborate among them for purposes of **completing an execution plan on the road**.

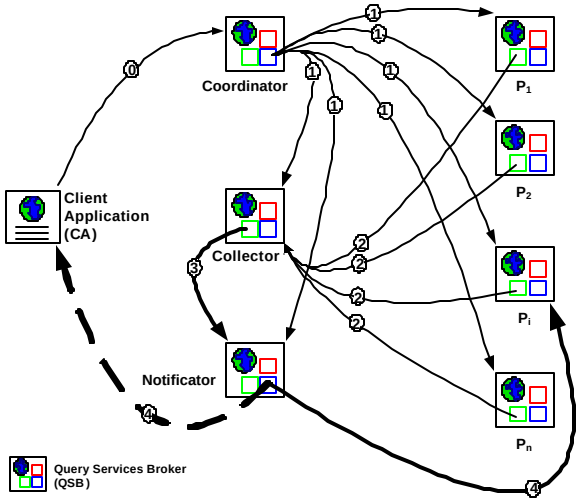

Figure 2: Ad-hoc environment

Ad-hoc coordination in Net Traveler begins when an end-user sends a request to a QSB trough a CA or another interface (e.g. arrow labeled with 0 in Figure 2). As an immediate consequence of such action, this QSB become in a initial coordinator (e.g. QSB labeled with "Coordinator" in Figure 2). At this steep, Q_1 build an initial plan A_1 based on its knowledge and send it to its nearest QSBs (e.g. QSBs labeled as "Collector", "Notificator", "P_1", …, "P_n" in Figure 2). A_1 is stored in the XML format, containing control information about the sequence of web services to contact and data that can be partial results. This plan is sent as a message to the other web services, which can re-send it to others web services or respond with the information requested. For example in Figure 2 the "Coordinator" sends a message to "P_1", "P_2", …, "P_n" including information in such manner that they will send the responses toward the "Collector".

4.3. Orchestration and choreography model

Orchestration refers to an executable process and choreography refers to tracking the message sequences among peers. [3] Referring to web services, orchestration is the static and/or fixed relationship among them. Usually it is determined in the logic of the source code, or some times defined by a system administrator. Generally, such relations occur between different web services classes and/o between web services from different architectural layers.

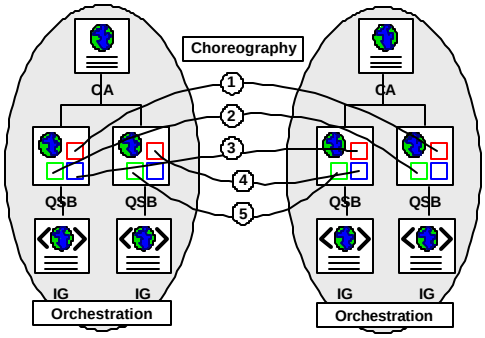

Figure 3: Orchestration and choreography

Choreography, is the dynamic relationship among web services, typically it occurs as a result of the execution of a given temporary plan to complete a task. Choreographs or plans define temporary relationships which leads to conform ad-hoc federations. Figure 3 shows a sketch of a choreography through the arrows labeled as 1, 2, …, 5, where each arrow represents an interaction between two QSBs and numbers represent the sequence of such interactions.

4.4. Message passing model

Net Traveler follows an asynchronous message passing model where web services do not have to wait blocked to obtain responses. For example CAs send a message and continue executing its own tasks. The response can be obtained as a result of a second inquiry to the NIS or by means of a notification from the NIS. This scheme is followed by the web services at each step of the query

execution. Our asynchronous approach allows redirect messages through different paths. Figure 2 shows the sequence of execution represented as a directed graph. A difference of the synchronous model, pairs of edges between vertexes are not necessary. Instead responses can follow a different path in order to be more efficient and in this way execution paths are real world like.

5. Performance Evaluation

For the performance evaluation was done using forty one computers. Thirteen computers were Pentium IV of 2.40GHz with 512MB of RAM (**lab 1**). Seventeen computers were Pentium III of 500Mhz with 128MB of RAM (**lab 2**). And, the last one was a Pentium III of 1GHz with 512MB of RAM (**computer 0**). All of them were networked by a 100 Mbps Ethernet and were running Linux RedHat and the Java Web Services Developer Pack.

To describe the web services distribution we will use the schema of Figure 2. The QSBs Coordinator, Collector, Notificator and P1, P2, …, P10 were placed in the lab 2 using one computer for each. Several CAs were placed in the lab 1, from one to twenty seven. Finally, the Test Application (TA) was stored in computer 0. The query used was a request for an auction: buy some equipment whose cost was less than a constant X, This is because we were interested in measure the response time of the synchronous and asynchronous message passing rather than the performance of database access or the resolution of complex queries. In Figure 4, the workload is the number of messages occurred during the execution of the plan. And, the response time is the time that takes to interchange a given number of messages.

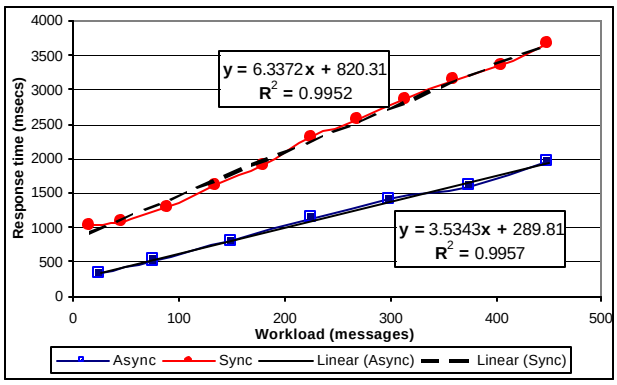

Figure 4: Response time for an increasing workload in a synchronous and asynchronous message passing model

The trend line of the response time curves fits a linear curve. This line grows at the same time that the workload increases. The workload was produced by the execution of one to twenty seven Client Applications. The results show that our asynchronous scheme for choreography is superior to a synchronous approach. Notice that has the number of messages as the number of messages increase, the response time of synchronous choreography is twice as large as that of our approach. This evidence shows the promise of Net Traveler as a system for E-Government collaboration.

6. Summary and Conclusions

In this paper we presented a new middleware architecture called Net Traveler that is a decentralized, peer-to-peer framework for Web Service Orchestration and Choreography. This framework was based on a model for composition of ad-hoc peer-to-peer web services, where one site performs a given task and ships its results, plus some control information, to another site that will continue with the computational process.

The preliminary results form the prototype shows that this model is feasible for the depicted research problem. However, there are many outstanding challenges, which are the following steps to follow in this research. For example, there it is critical to improve the scalability and fault tolerance of the model. Also, the design of a more complex and accurate performance evaluation model is needed, in order to predict its behavior in wide area environments. We also need to develop an algorithm to find plans for queries.

References

[1] A. Arkin, S. Askary, S. Fordin, W. Jekeli, K. Kawaguchi, D. Orchard, S. Pogliani, K. Riemer, S. Struble, P. Takacsi-Nagy, I. Trickovic, S. Zimek, "Web Service Choreography Interface (WSCI) 1.0", W3C, 2002.

[2] C. Pahl, M. Casey, "Ontology support for web service processes", ACM SIGSOFT Software Engineering Notes, Proceedings of the 9th European software engineering conference held jointly with 10th ACM SIGSOFT international symposium on Foundations of software engineering, Volume 28 Issue 5, September 2003.

[3] C. Peltz, "Web services orchestration and choreography", Computer, Volume: 36, Issue: 10, Oct. 2003, pp. 46 – 52.

[4] M. Rodriguez-Martinez, N. Roussopoulos, "MOCHA: A Self-Extensible Database Middleware System for Distributed Data Sources", In Proceeding of the ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA, May 2000, pp. 213-224.

[5] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications", to Appear in IEEE/ACM Transactions on Networking.

[6] J. Udell, "Orchestrate services", InfoWorld, July, 02.