Review of Computer Engineering Research
2016 Vol.3, No.1, pp.13-24
ISSN(e): 2410-9142
ISSN(p): 2412-4281
DOI: 10.18488/journal.76/2016.3.1/76.1.13.24

WEB SERVICE COMPOSITION FOR E-COMMERCE WEB APPLICATION

Anis ISMAIL[1] † --- Abd El Salam AL HAJJAR[2] --- Abdel Karim KASSEM[3]
[1,2]Communication and Computer Network department, University Institute of Technology, Lebanese University, Saida, Lebanon
[3]Computer Science and Communication department, Arts, Sciences and Technology University, Beirut, Lebanon

ABSTRACT

Web Services are based on distributed technology and provide standard means of interoperating between different software applications across and within organizational boundaries with the use of XML. Web Services technologies allow interaction between applications. Sometimes a single service given alone does not meet user's needs. In this case, it is necessary to compose several services in order to achieve the user's goal. Web service composition is one of the most challenging problems of recent years. The number of service providers is increasing, and along with that for a request they offer multiple services with the same functionality, so it makes the problem of composition quite complex. In this paper, we compose multi web services for E-Commerce. We implement a new web application that use two or many web services to make some transactions and to choose the best deal.

Keywords: Web services, Distributed technology, Web services composition, XML, E-Commerce, Transaction.

Received: 6 December 2015/ **Revised:** 6 January 2016/ **Accepted:** 12 January 2016/ **Published:** 16 January 2016

Contribution/ Originality

The paper's primary contribution is finding that the composition of web services is the process of combining Web services in order to offer value-added services. Web Services composition consists in combining several Web Services into a Composite one, in order to satisfy complex users queries.

1. INTRODUCTION

A web service is a software system designed to support interoperable machine-to-machine interaction over a network. The Web service has an interface described in a machine-processable format. Other systems interact with the Web service in an away prescribed by its description using the Simple Object Access Protocol (SOAP) messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards. This definition was

† Corresponding author

published by W3C [1]. Various techniques for providing product configuration as a web Service were disclosed by Varmaraja, et al. [2].

The appearance of web service has created more opportunities for organizations to establish more agile and flexible collaborations with other organizations. As individual Web services are restricted in their capability that is why we need to compose existing services to create new functionality to meet the requirements of business processes. Web service composition is the process of combining Web services in order to offer value-added services. Composite services in turn are defined as an aggregation of elementary and composite services. The Web services composition process should satisfy both functional, non-functional requirements and guarantee the correctness of the result. Web service composition is currently an active area of research, with many languages being proposed by academic and industrial research groups due to its complexity. However, the flexibility of composition comes at the penalty of increased system engineering complexity.

In this paper, we present a new web application that uses the composition of the web services Technique to make best deal in E-transactions. For that, we implement a new web site that use the Amazon web service and ebay web service to request such product and compare the prices to choose best deal

2. RELATED WORK

Web service composition lets developers create applications on top of service-oriented computing native description, discovery, and communication capabilities. Such applications are rapidly deployable and offer developers reuse possibilities and users seamless access to a variety of complex services. There are many existing approaches to service composition, ranging from abstract methods to those aiming to be industry standards. In this section, we describe some research proposed for the Web service for composition.

Recently, Raj, et al. [3] propose an approach on identifying the most appropriate service based on the user's preferences of the requested WS. The given WS description may contain the parameters, which may have relations with the requested WS of the specific domain in different aspects like name, parameters and types. The domain specific WS classification can be done using Naive Bayes classification algorithm. But this method has some limitations of not considering functionality based classification. The coupling and cohesion properties are not considered in the composed WSs. Said, et al. [4] proposes a new SOA architecture called "Generic SOA" that allows dealing with legacy systems problem and enhancing SOA elasticity.

The author [5] proposes a novel approach of semantics-based matchmaking, which is named process-context aware matchmaking. This process locates the suitable service during web service composite modeling. During matchmaking, the approach utilizes not only the semantics of technical process but also those of business process of a registered service, thus further improving

the precision of matchmaking. The process-context aware matchmaking was integrated with business-process-driven web service composition in a cohesive development environment based on Eclipse. The work describes a way to match web services for composition but doesn't integrate the composition of web services.

To improve the exactitude of a Web service search, Ye and Zhang [6] proposed a method that explicitly specifies the functional semantics of services. They specified a service and a user requirement using object, action and constraints as well as input and output parameters. Utilizing this information, they found a service to satisfy the user requirement. However, they did not consider how the popularity of web services can be applied to service composition.

The authors [7] tried to improve the accuracy of automatic matchmaking of Web services by taking into account the knowledge of past matchmaking experiences for the requested task. In their method, service execution experiences are modeled using case based reasoning. This method can be helpful for improving the exactitude of composite service, but it's still packed of complex problems related to the composition process.

The University of Georgia implements an extension [8] of Graph Plan [9] an AI planning algorithm, to automatically generate the control flow of a Web process. This extension is does not cover the preconditions and effects of the operations, we also take into consideration in the planning algorithm the structure and semantics of the input and output messages. The approach was presented to solve both the process heterogeneity and data heterogeneity problems. And the system generates outputs, an executable BPEL file which correctly solves non-trivial real-world process specifications. The authors described in parts of their paper the project proposed by BPEL [10] to automate the composition, but neither one of those works propose the automatic selection of web service using the behavior experience named popularity.

In the some context, Shin, et al. [11] proposes a composition method that explicitly specifies the uses of functional semantics of web services. Specifically, the proposed method is based on a graph model, which represents the functional semantics of Web services. In this approach, the service functionality of a service is represented by a pair of its action and the object of the action. The information about services is organized and stored in a proposed two-layer graph model. Given a user request, they search for composition paths in the graph model and construct a composite service from the paths discovered. However, the web services selection is not taking in consideration the notoriety to get link in the schema composition.

Liu, et al. [12] propose a Web service model in which inputs and outputs of services are expressed using RDF graph pattern, as well as domain ontology. They improve the exactitude of composite services without preconditions and effects using semantic propagation based on graph substitution and also they don't take a request user when selecting web services.

3. WEB SERVICES

Web services are defined as self-contained, modular units of application logic which provide business functionality to other applications via an Internet connection. Web services support the interaction of business partners and their processes by providing a stateless model of "atomic" synchronous or asynchronous message exchanges. These "atomic" message exchanges can be composed into longer business interactions by providing message exchange protocols that show the mutually visible message exchange behavior of each of the partners involved. The issue of how web services are to be described can be resolved in various ways.

The Web Services Definition language (WSDL) [13] is an XML-based language, which specifies a Web service by defining messages that provide an abstract definition of the data being transmitted and operations that a Web service provides to transmit the messages. Four types of communication are defined involving a service's operation (endpoint): the endpoint receives a message (one-way), sends a message (notification), the endpoint receives a message and sends a correlated message (request-response), and it sends a message and receives a correlated message (solicit-response).

Operations are grouped into port types, which describe abstract end points of a Web service such as a logical address under which an operation can be invoked. A WSDL message element defines the data elements of an operation. XML Schema syntax is used to define platform-independent data types which messages can use. Each message can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language. Concrete protocol bindings and physical address port specifications complete a Web service specification.

Typically, the Binding and Service segments do not change from one WSDL release to the next. In the definitions segment, only the WSDL version changes. For that reason, when you read a WSDL, you will typically spend most of your time reading the Request and Response Definitions segments. We show a sample WSDL fragment for Amazon web service. The Definitions section of the WSDL defines the namespaces used throughout the WSDL, and the name of the service, as shown in the following snippet of the Product Advertising API WSDL.

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://webservices.amazon.com/
AWSECommerceService/2013-08-01"
targetNamespace="http://webservices.amazon.com/
AWSECommerceService/2013-08-01">
```

This example shows that the:

- Default namespace is xmlns="http://schemas.xmlsoap.org/wsdl/"
- SOAP namespace used is xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
- Schema used is xmlns:xs="http://www.w3.org/2001/XMLSchema"
- Product Advertising API WSDL namespace is "http://webservices.amazon.com/AWSECommerceService/2013-08-01"

  The date at the end is the version number. It is the date the WSDL became public.
- TargetNamespace is "http://webservices.amazon.com/AWSECommerceService/2013-08-01"

  The TargetNamespace is an XML schema convention that enables the WSDL to refer to itself (as the target). The TargetNamespace value is the Product Advertising API WSDL namespace

Namespaces are collections of parameters and operations in which their names are unique. The advantage of using namespaces is that the WSDL can define terms, like string, just by referring it its namespace, xs. Also, prepending the namespace to a parameter ensures that there is no danger of name collisions.

Each namespace declaration starts with "xmlns:" (XML namespace:) and is followed by the abbreviation for the namespace. For example, in the following namespace declaration, xs becomes the abbreviation for the URL of the schema.

xmlns:xs="http://www.w3.org/2001/XMLSchema"

Throughout the remainder of the WSDL you will see parameters defined in terms of namespace abbreviations, for example:

type="xs:string"

ref="tns:HTTPHeaders"

These abbreviations provide the namespace in which the parameters are defined.

Product Advertising API enables you to specify the version of the WSDL you want to use. This functionality ensures that future enhancements and changes to Product Advertising API WSDLs will not be intrusive to your applications. For example, when Product Advertising API adds new elements to its WSDL, applications that validate against an older Product Advertising API WSDLs will not be affected.

Product Advertising API WSDL version names are based on the date that they become active. The version of the WSDL is specified in the Product Advertising API WSDL namespace declaration. In the preceding example, the version of the WSDL is 2013-08-01.

xmlns:tns="http://webservices.amazon.com/

AWSECommerceService/2013-08-01"

In reality, the date, here, is the WSDL's file name.

The AWSE Commerce Service directory contains all of the Product Advertising API WSDL versions. You use the Version parameter in REST requests to specify the version of the WSDL you want to use. The default version is 2011-08-01. If you want to use a different WSDL version, including the latest, you must specify it in each request, for example, in REST.

http://webservices.amazon.com/onca/xml?
Service=AWSE Commerce Service
&AWS Access Key Id= [AWS Access Key ID]
&Operation=Item Search &
Search Index=Books&
Author=Steve%20Davenport&
Version=2013-08-01
&Timestamp=[YYYY-MM-DDThh:mm:ssZ]
&Signature=[Request Signature]

SOAP requests always specify a namespace, which includes the WSDL version. To avoid problems due to future WSDL changes, be sure to specify a WSDL version in your SOAP application.

Other example is the eBay that has emerged as a platform where people can buy and sell practically anything. With more than 135 million users worldwide, eBay is a thriving online marketplace. Best of all, eBay isn't constrained by HTML. By using eBay's web services APIs, members of the eBay Developers Program can hook into the eBay platform using XML to integrate eBay into their own applications.

Like many web services, the eBay API comes in multiple flavors. There's the original XML-over-HTTPS POST interface, a format that's quite similar to REST. There's also a newer, more fashionable SOAP interface, which uses the Doc/Literal format.

Both have their own benefits and weaknesses. Pure XML is easy to produce and consume on any platform, something that's vital given the shaky state of PHP's SOAP support. However, SOAP has the advantage of eliminating the trouble of manually parsing XML into usable data structures.

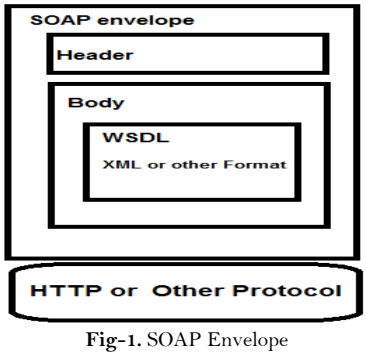

Fig-1. SOAP Envelope

PHP programmers have a third alternative, which combines the best parts of both interfaces: Services_Ebay. Written primarily by Stephan Schmidt, Services_Ebay is a PEAR package that wraps around the XML API to provide an object-oriented interface to eBay. Additionally, it takes advantage of several new PHP 5 features to create powerful code that's simple to use. You can't run this code under PHP 4 because PHP 5 plays such a key role.

Now that you've installed Services_Ebay, you need to configure it with your authentication credentials. This is developer-specific information that allows eBay to identify you and your application.

```
require_once 'Services/Ebay.php';
// load authentication data from config file
$config = parse_ini_file('ebay.ini');
// pass authentication data to Services_Ebay
$session = Services_Ebay::getSession($config['devId'],
                                     $config['appId'],
                                     $config['certId']);
$session->setToken($config['authToken']);
$session->setUrl(Services_Ebay_Session::URL_PRODUCTION);
```

After including the Services_Ebay package, the code loads in a configuration file, ebay.ini, usingparse_ini_file(). This file contains the credentials you must provide when making an eBay services API call. Specifically, these are your:

- developer ID
- application ID
- certification

Auth&Auth Token

The first three authentication credentials are collectively developer keys, and they identify the developer that's contacting eBay.

The fourth, Auth&Auth Token, identifies the user making the request without exposing her username and password. This person is frequently distinct from the application's developer, and the feature allows programmers to write applications for others to use without compromising security.

To acquire your own credentials, sign up for the eBay Developers Program.

Now that you have the data loaded, the next step is to create an eBay web services session. This session is an object that holds your credential data. Create a session by calling Services_Ebay::getSession() and passing your keys as arguments. In this case, the keys are in the $config array created byparse_ini_file().

The next two steps modify the session to set the Auth&Auth Token and the URL of the web service by using the setToken() and setUrl() methods. By default, the session points itself at

eBay's testing server (known as the Sandbox). To override this value and direct the server at the live Production site, pass the Services_Ebay_Session::URL_PRODUCTION constant.

4. WEB APPLICATION ARCHITECTURE

Web Service composition is one of the important activities in service oriented architecture. Our goal of this paper is providing one set of single web services with high combining ability in order to meet the complex needs of users. We have studied several methods for web service composition that still there are problems such as reduction of accuracy, increase of response time and unselect optimal composition. We proposed a method to resolve these problems and according to the response time, accuracy and composition optimality parameters, we show proposed procedure to the better results is achieved.

The web browser represents the main entry point of our system, it allows retrieving, presenting and traversing information resources on the World Wide Web and information resource is identified by URI/URL (Uniform Resource Identifier). After accessing to the system URL, the web browser presents it as a graphical user interface on a web page, that allows the user to search an collect the pertinent information by entry the specific key words. The system search feature powered by eBay and Amazon, because its architecture based on these two servers services (Figure 2). Firstly, Ebay provide a web service; can access it by using eBay's web services APIs (application program interface). Members of the eBay Developers Program can hook into the eBay platform using XML (EX tensible Markup Language) to integrate eBay into their own applications. The eBay Web Service Provide an API which is an extensive system that allows us to call data directly from the eBay servers. There is an extensive API test tool to explore the API results using any and all available parameters for XML request. Afterward, we receive the URI which is the web URL of the API you are using with the HTTP Headers (Hypertext Transfer Protocol Header), these will be automatically populated by the choices we has made by setting up our API call, Once we have set up the tool with the API call, we can access the XML Requested from the tool in order to access the data that will be posted to the API and determine what to return. The search parameters are contained within XML tags, once the call has been made, the Key words will be applying from the tool in order to export the eBay documents result.

On other hand, we access the Amazon Relational Database Service using the SOAP (Simple Object Access Protocol) web services messaging protocol. This interface is described by a WSDL (Web Services Description Language) document that defines the operations, parameters, requests, and responses used in web service interactions and we can think of a WSDL as the contract that defines the language and grammar used by web service clients and servers The WSDL references an XML

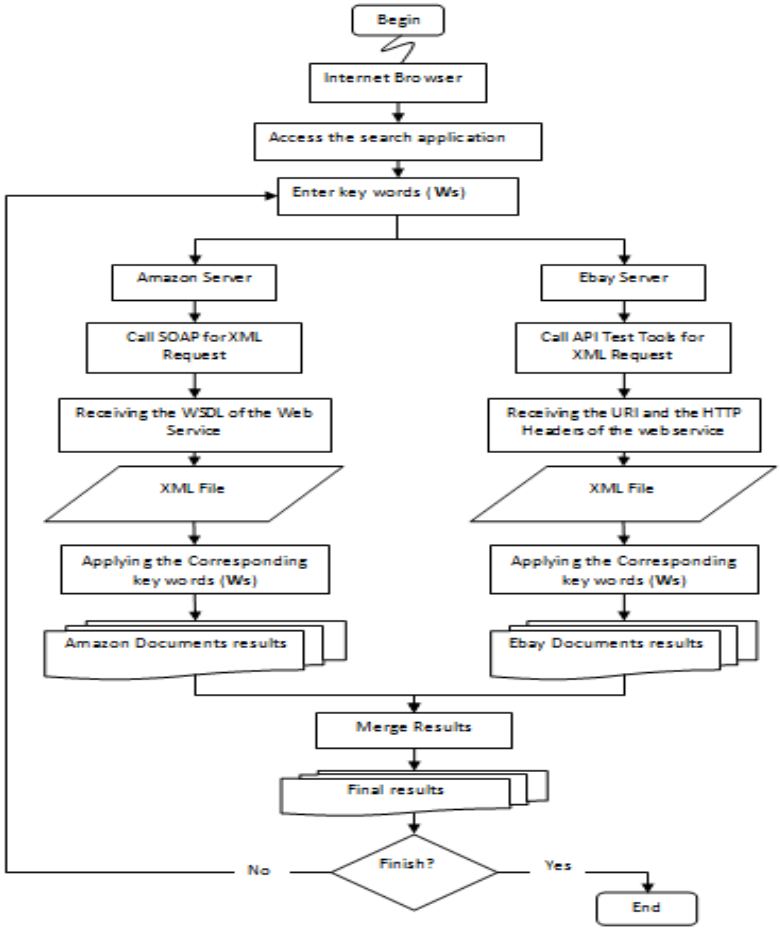

Fig-2. Proposed system architecture

Schema document, which strictly defines the data types that might appear in SOAP requests and responses, after that the key words will be applying from it to export the Amazon documents results that will be merged with the Ebay Documents results in order to export the final Results of the key word looking for, after that if the results was failed we have to search for another key words else the target will be founded with successful end.

5. SIMULATION AND VALIDATION

In this paragraph, we present the simulation and validation of our system. The system simulation is a set of techniques that use computers to pretend the operations of numerous processes. Then, it is a tool used by a user (e.g. researcher, engineer etc...) to treat and explorer the results of system actions (e.g. information extraction, data presentation in a website, etc). The validation of simulations is a process to ensure that the simulation and the implemented models cover all the points and the business logical requirement already proposed in our system architecture.

Simulation refers to an instance at which you execute a model. A computer model means algorithms, equations and graphical user interface used to capture the behavior of the system being modeled. In this paper, the system is presented by a website developed using PHP as main programming language and connected to a MYSQL database.

The interface (Figure 3) of our engine supported by Amazon and Ebay is a basic one that seeks to capture all necessary input from the user in an elegant and simple form.

Fig-3. Main Interface

Below we can see screenshots of when we lunched the application and one of the application displaying results (Figure 4) from both Amazon and Ebay.

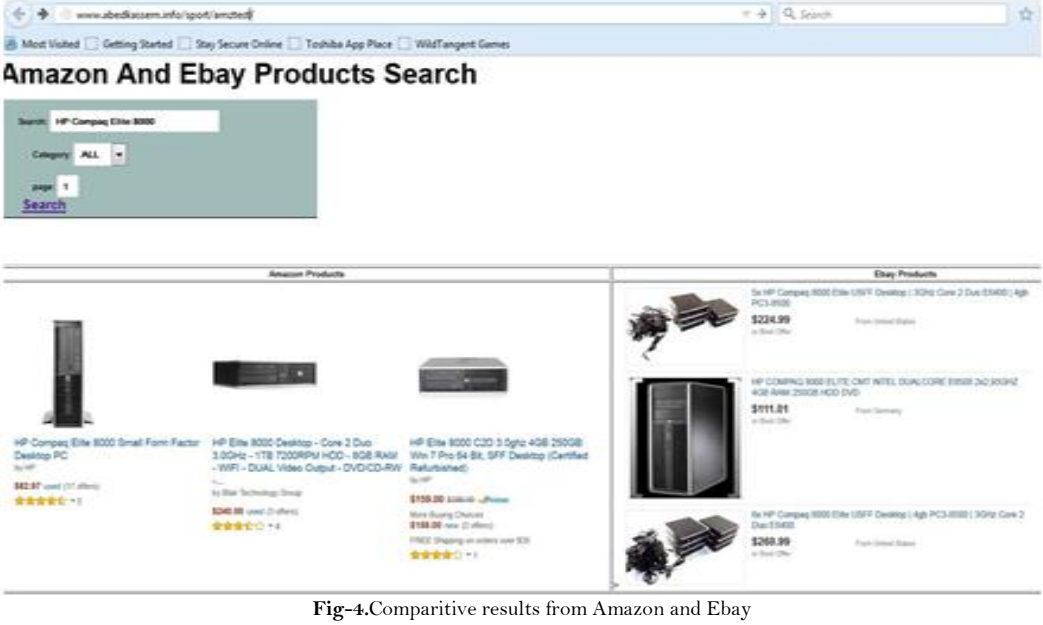

Fig-4. Comparitive results from Amazon and Ebay

Below we can see screenshots of when we lunched the application and one of the application displaying fusion results (Figure 5) from both Amazon and Ebay

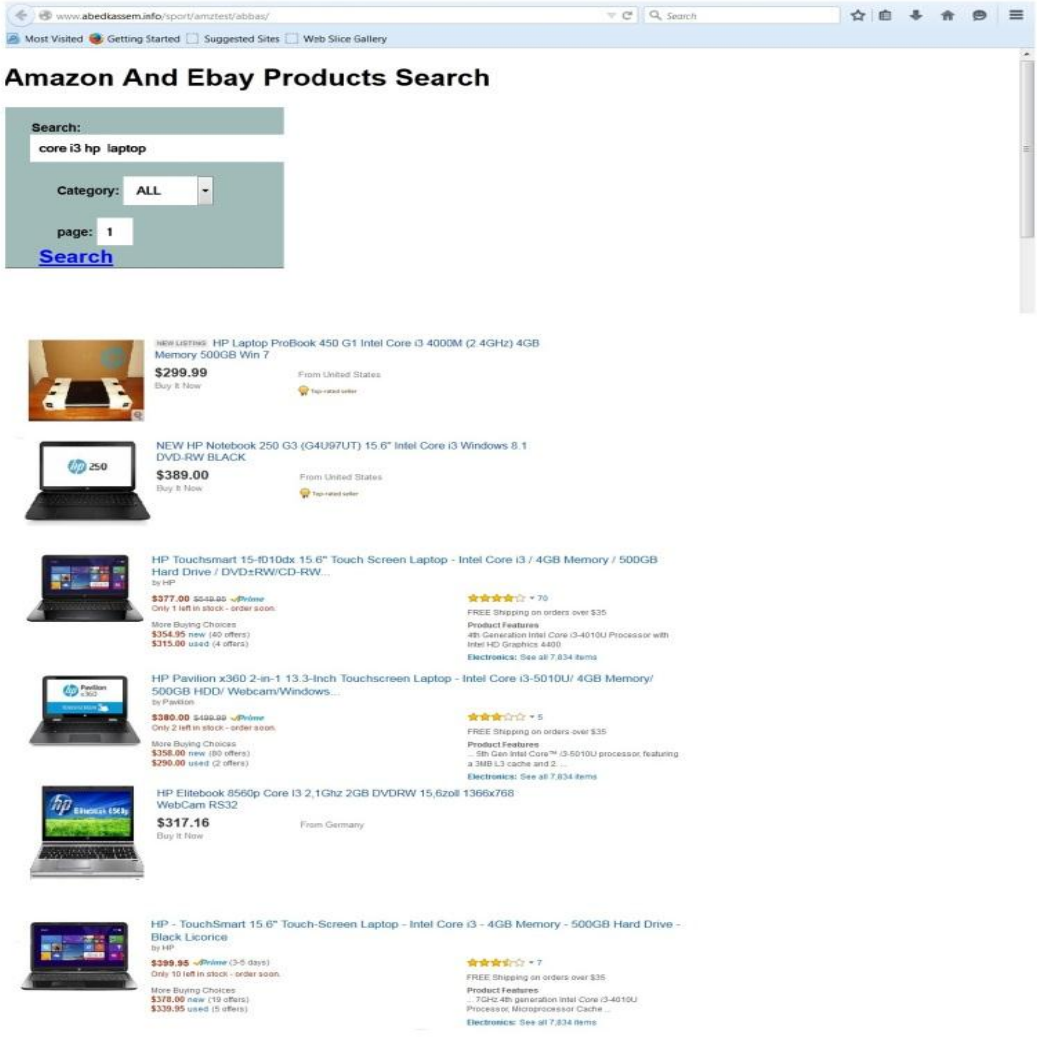

Fig-5. Fusion Results.

6. CONCLUSION

Web services are software components that are described, published, found, used, and composed into new services. Web service composition is considered as the cornerstone for Web service development. Appropriate descriptions enable Web services to be discovered and reused for creation of new services. In order to compose a requested service the component services have to be discovered using matchmaking techniques. The component services have to be selected and orchestrated/choreographed suitably to achieve the desired functionalities and behaviors as expected from the composite service.

We presented the design and implementation details of a Web application supporting E-commerce Transactions. Thanks to the dynamism of the Web, "crawling" constitute the backbone of applications that facilitate the search for Web information. Its purpose is to retrieve this information, to filter, index, put in databases, and thus exploit. The web interface is very impressive in its content, but a little too abundant for a simple search.

Funding: This study received no specific financial support.

Competing Interests: The authors declare that they have no competing interests.

Contributors/Acknowledgement: This work has been done as a part of the project "Extraction et fusion de l'information à partir du Web par agrégation des services et de l'analyse du contenu" by Lebanese University.

REFERENCES

[1] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, and C. Ferris, "Web services architecture," W3C Technical Reports and Publications, 2004.

[2] M. Varmaraja, H. Lu, A. Singhal, and E. Chikovani, "Web service architecture for product configuration." Available: http://www.google.com/patents/US9021064, 2009.

[3] T. Raj, T. F. Michael, K. Ravichandran, and K. Rajesh, "Domain specific web service composition by parameter classification using naïve bayes algorithm," *World Applied Sciences Journal*, vol. 29, pp. 99-105, 2014.

[4] M. Said, M. Hazman, and H. Hassan, "GenericSOA: A proposed framework for dynamic service composition," *International Journal of Computer Science Issues*, vol. 11, pp. 94-99, 2014.

[5] W. Han, X. Shi, and R. Chen, "Process-context aware matchmaking for web service composition," *Journal of Network and Computer Applications*, vol. 31, pp. 559–576, 2008.

[6] R. Akkiraju, A. Ivan, R. Goodwin, B. Srivastava, and T. Syeda-Mahmood, "Semantic matching to achieve web service discovery and composition," in *Proceedings of CEC/EEE'06, IEEE Computer Society, Washington, DC*, 2006.

[7] D. Thakker, T. Osman, and D. Al-Dabass, "Knowledge-intensive semanticweb services composition," in *Proceedings ofUKSIM'08, IEEEComputer Society, Washington, DC*, 2008.

[8] Z. Wu, A. Ranabahu, K. Gomadam, A. P. Sheth, and J. A. Miller, "Automatic composition of semantic web services using process and data mediation," Technical Report, LSDIS lab, University of Georgia, 2007.

[9] S. Russell and P. Norvig, *Artificial intelligence: A modern approach*. New Jersey: Pearson Education, 2010.

[10] M. Pistore, P. Traverso, P. Bertoli, and A. Marconi, "Automated synthesis of composite BPEL4WS web services," presented at the IEEE Intl Conference on Web Services ICWS'05, 2005.

[11] D. H. Shin, K. H. Lee, and T. Suda, "Automated generation of composite web services based on functional semantics," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, pp. 332–343, 2009.

[12] Z. Liu, A. Ranganathan, and A. Riabov, "Modeling web services using semantic graph transformations to aid automatic composition," in *Proceedings of ICWS'07, IEEE Computer Society, Washington, DC*, 2007.

[13] E. Christensen, "The web services description language WSDL." Available: http://www-4.ibm.com/software/solutions/webservices/resources.htm, 2001.