AWS-Net Traveler: Autonomic Web Services Framework for Autonomic Business Processes

Hillary Caituiro-Monge, Manuel Rodriguez-Martinez
Computing and Information Sciences and Engineering, University of Puerto Rico at Mayagüez
{hcaituiro, manuel}@ece.uprm.edu

Abstract

Today's information technologies, such as autonomic computing, offer companies new ways to do businesses. Those that fail to embrace and use such advantages in the next years are in danger to disappear or become zombie ones. In contrast, those pioneering these next-generation technology-business strategies will gain strategic advantage. This paper is about AWS-Net Traveler, which is an Autonomic Web Services Framework for purposes to perform Business Processes in an Autonomous mode. The main feature of this proposal is a decentralized architecture that heavily relies on Peer-to-Peer Web Service Brokers to coordinate, plan and perform Web Service choreographies. We include several use cases scenarios and how to address them using the proposed framework. Also, here is discussed the business model for Autonomic Business Processes model based on Autonomic Choreography of Web Services. Finally, we introduce the Architecture of AWS-Net Traveler and the description of their main components.

Keywords
Web Services, Composition of Web Services, Autonomic computing, Autonomic business

1. Introduction

Today's information technologies, such as autonomic computing [2], offers companies new ways to do businesses in an innovative and profitable ways. Autonomic computing has self-managing characteristics as fundamental property which includes self-configuration, self-optimization, self-healing, and self-protection as core concepts [3]. Due to such properties, the most suitable business operations to get benefits from this technology are business processes, which, in this new context they are named as "Autonomic Business Processes". The ultimate goal of autonomic computing and autonomic business process is perform without human intervention to accomplish their tasks or sub processes. Indeed they have self-managing attributes, they allow human intervention when necessary or desired [11]. A classic example to understand what autonomic behavior is, many authors cite our nervous system [3]. It autonomously control our system, freeing us of manage every part of our body and organs. But at any moment we can consciously take control of some activity like our breathing.

To achieve autonomic behavior on business processes they have to interact among them to cooperate and complete their duty. In that sense is desirable that enterprises publish their business process to be accessible at least in the domain of their business needs. Nowadays, organizations in general are putting some part of their business process in the Internet through Web services. It makes Web services the most suitable information technology to support business-to-business (B2B) interactions [4] [5] [8]. Figure 1 depicts a basic interaction diagram between a client and Web services. This client makes a request to a Web services and it contact other Web services to complete the request.

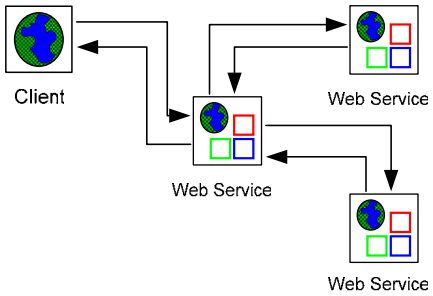

Figure 1 Basic interaction between a client and Web services

Autonomic Business Processes (ABP) rely on B2B and Enterprise Application Integration (EAI) architectures for handle their interactions. The new generation of these architectures is using Web services as their core foundations [8]. Over them, high level business process languages are being developed in this effort. For example the Business Process Execution Language for Web Services (BPEL4WS) [6] is being developed by IBM. Microsoft and BEA and pushed to become a standard. The technical work is only part of the overall effort that is being pursued to achieve ABP. There, also are a lot of cultural issues since this wave of world-scale business bring new elements to the table such as languages, business traditions, and so on. Despite such barriers, the information technologies solutions, standards should become available. Once technologies become available then organizations, users, enterprises will figure out how to use them as many of them are doing these days.

As result of such early adoption, even without having a complete set of standards, every day new Web services emerge and others disappear as businesses promote or demote their Web based services. Also, organizations that rely on third party Web Services to accomplish some business or non business tasks have to sign strategic deals with Web services-owners to ensure availability and fixed prices. Even, companies currently owning Web services are increasing their offered services and using third party Web services in order to offer new services as part of their stack of available services. As a direct outcome the pool of Web services is increasing in such manner that it is becoming very difficult to get a list of Web services that offers some service. Furthermore, there is no way to accurately select the most convenient Web service to carry certain task.

Fortunately such huge pool of Web services generates business and strategic opportunities. In this context profit and non-profit organizations have more alternatives from where to pick up Web services of their interest. With this kind of availability of services are less needed traditional pre-arranged and fixed service contracts with partners. The most part of past and current global range business models across countries rely on human-made business networks where organizations have to invest a lot of money and time. This practice requires that companies have to send people to find organizations offering interesting services for the organization. Those representatives have travel to several regions and make alliances with local partners after reaching some agreement which make happy to the parties. In this process are taken in account local business laws and practices also the culture of the society. A business model driven by a huge pool of Web services tends to eliminate the boundaries among countries. Also cultural differences are expected to be overcome as result of the use of Web services standards and languages. The ultimate objective in this line is that, at some time some company in any location in the world will be able to create a business model involving operations around the world, without need to send a representative to start a partnership.

Most of the prevalent business models cover pre-negotiated contracts with partners. If there were involved Web services, they have to specify the conditions and parameters governing the transactions to be used for a business process. For example Amazon has a business partnership network with FedEx, UPS, USPS carriers and several third party Book sellers. Some of them have Web services to provide their core business services. However, in a world-scale pool of Web services this business model become neither efficient nor effective, since fixed relationships inhibit them for dynamically discovering new service providers. Organization that can assemble a network composed by the best service provider will be in advantage over those having fixed business networks.

Current Web services technologies and standards allow that these fixed-type business partnerships happen without to much difficulty. It is done by simply configuring the pre-selected business partners and putting the sub-processes of the business process following some order. For example Figure 2 shows an abstraction of how a fixed business process could be specified. It is a simplified view of the business process specification for an online books seller should include the prioritized list of books providers, the list of mail carriers and the list of charging services. As Figure 2 illustrates, for creating the business process, there is necessary to know the partners in advance as a requisite.

In autonomic business processes such kind of specific relationships with partners should be avoided but not prohibited. Reducing the specific reference of services gives more freedom of selection to the infrastructure. Here, infrastructure refers to the software components in charge of performing autonomic businesses. Thus, the infrastructure, which is capable of selecting among thousand of thousand of Web services will be able to find the best service providers combination. Task that is not able to be done with human intervention, since it will introduce a considerable delay that is not desired in online transactions where clients considers one second as a fair response time. Figure 3 depicts a simplified

version of an autonomic business process definition in which are included references to the required kind of services, but not to the specific ones as in Figure 2.

```
<BusinessProcess>
      <BooksProviders>
            <sequence> Amazon, Abbooks </sequence>
      </BooksProviders>
      <MailCarriers>
            <set>  USPS, UPS, FedEx </set>
      </MailCarriers>
      <ChargingServices>
            <set>  VISA, MASTERCARD, PAYPAL </set>
      </ChargingService>
</ BusinessProcess >
```

Figure 2 Sketch of a script for a human-service-selected Business Process

We propose a framework to perform transactions which are based on such kind of business processes specification that appears in Figure 3. This framework to which we will refer also as the infrastructure provides peer-to-peer Web services brokers to perform several tasks. One of these tasks is the autonomic selection of Web services from the universal abstract pool where are contained all Web services. Those brokers can be deployed across organizations on the side of the Web services and also they can be deployed independently as a service offered for some organization. Optionally user can hold their own broker to represent them in business processes as their representing agents.

```
<AutonomicBusinessProcess>
            <step1>BookSelling</step1>
            <step2> MailCarrierSelection </step2>
            <step3> PaymentProcessing </step3>
</AutonomicBusinessProcess>
```

Figure 3 Sketch of a script for an Autonomic Business Process

This framework has to solve several challenging issues in order to accomplish its ultimate goal that is offer a mechanism for autonomic business processes using autonomic Web services. In this paper we address the solution from the business side and from the information technology side. We present a new model to perform business processes in autonomous manner. Also, we present the software architecture of our framework that gives support to autonomic Web services.

1.1. Contributions

The main contribution of this work is a Autonomic Web Services Framework for Autonomic Business Processes. That a difference of current approaches which require that Web services or business actors to be known ahead, it create such business partnership “on-the-fly” by means of using P2P discovery mechanisms from which to select the most suitable partners. Also, another very meaningful contribution is the software architecture that leads to the conformation of ad-hoc federations [1] to perform and autonomic completion of a plan in an autonomous way.

1.2. Roadmap

This paper follows an inductive strategy to present the research contribution. In the second section we introduce example scenarios to illustrate the problem and the opportunities that they bring. The third section is about the approach of how to address the issues introduced in the first section. It is done by means of a business-oriented solution. The fourth section presents a software-side solution. It is done through software architecture. The main components are described and also the relationship among them. The fifth section is a generalization of the framework through an abstract model. Finally, the related work, conclusions, future work and references are covered.

2. Example scenarios

2.1. Government attends claims of citizens

2.1.1. Scenario and problem. When a citizen have a claim (e g. a hole in a road, a damaged bridge, a blackout, etc.) on its hands, she or he has to file a claim in the respective govern agency or sometimes in a private agency. Citizens most of the time do not have idea of where to file their claims. Or some of those claims require the assistance of several private and govern entities.

2.1.2. Solution. Have a mechanism (e.g. Web Portals, Web Services, etc) to handle citizen related claims. For example, when doing a claim, a citizen can use the nearest computer with access to internet and file its claim thought a Web Portal. She or he can file a simple claim or a composite claim composed of several simple claims. Behind the scenes the Web Portal can send the petition to a Web Service with brokering capabilities or to a Software Agent. This Broker Web Service starts the negotiations that will lead to the process to complete the request. Agencies which are involved in the solution will be discovered

"on-the-fly". As a result, the citizen will have status information about its claim. This notification can be triggered by a user request, or a time constrains or to be event based.

2.2. Online books seller

2.2.1. Scenario. Certain online books seller provides services around the world. They have to ship books to almost every country. Also, it has several providers ranging from third party books sellers to users which shell new or used books. Current book on-line sellers relies in hardwire contracts which are strategic partnerships with providers. For example Amazon relies on USPS, FedEx, UPS, etc and also it relies in several mid-size book providers. Those providers increase in number across the globe. And it is expected that they will increase even more in the near future.

2.2.2. Problem. Let's imagine a person living in the jungle of Manu which is located in Cusco, Peru. People residing there have access to the Internet through satellite channels. So, they are able to place online orders at any time, if the weather allows them to do it. After selecting several books in an USA based online store, they try to buy those by doing checkout. But, unfortunately the online books seller's web portal shows a message telling them that they do not deliver to Manu. The problem is that no one of the mail carriers with who the online book has contracts is able to deliver packages to Manu. Also, if so, it could be very expensive for them arrange a special delivery.

2.2.3. Solution. A desirable solution is an autonomic mechanism to dynamically select the most convenient mail carriers to deliver the package. Those qualified are both cheap and fast at the same time. This mechanism could pick one carrier to transport the books from somewhere in USA to Lima and another carrier to transport those books from Lima to Manu. The carrier selection process could be done by software brokers without human intervention.

2.3. Planning a trip

2.3.1. Scenario. Nowadays, common people is using Web sites on the Internet to plan their trips to anywhere in the world. There are few companies providing these services through Internet. These online services include purchasing fly-tickets, renting-a-car, reserving a taxi, reserving a habitation in a hotel and others.

2.3.2. Problem. The problem is that both vendors and users have to participate in the business process. Vendors have to choose the third party companies which will participate in the business process. It is that they sign strategic agreements with them to have a guaranteed service for a period of time. On the other side, users have to submit queries several times to get what they want. Also, if they want to have a rent-a-car and a room in the hotel, they have to repeat the process for each one.

2.3.3. Solution. This problem can be addressed using autonomic business process through a web-based solution which allows completing the business process without human intervention. It starts working when users select some parameters and submit their request to the online web-based trip planner. This service gets the best third party services and sends the request to them. In this way the search is not limited by the service providers that are pre-selected. Thus, cheaper prices can be found.

3. The business side: autonomic business processes (ABP)

Companies implement business processes as a basis of their operations. A business process is the sequence of steps required to complete a business transaction (e.g. online selling of a book). Autonomic business is about doing business processes without human intervention. It does not mean that a human intervention is not desired at all. It means that human intervention is not required to perform a business process. However, it is always desirable that someone can interfere in the business process at any time. This autonomic interaction can be achieved through the use of a set information technologies.

A company having autonomic business processes integrated in their operations is going to have a strategic position respect with those ones that no implement it. In fact, they will be in position to reduce the cost involving the business process. It is achieved as direct benefit from the autonomic selection of third party service providers. Since those providers will put their business operations through Web services within an universal pool known as the Internet. Thus, enabling the autonomic creation of federations of Web services [1] in such manner that the cost will be minimized and the quality of service will be maximized. Since those federations are created with members of the Web services pool, then

there can be chosen those sub set of Web services that represent the best option to complete the business process.

3.1. Model for autonomic business processes

A traditional business process is a systematic process where its parts are glued together in a sequential and parallel with interdependency among them. Before the business process become operative there is a setup stage where human intervention is always necessary. Business processes include providers and consumers. In that sense managers have to sign strategic contracts with providers to be able to offer a stable product to their consumers. Also, they have to identify and focus in a segment of the universe of consumers. The model for autonomic business processes eliminates the traditional setup stage for a business process. Instead, the selection of providers or partners is done for each business transaction, considering the best options at that time. This strategy is also called autonomic business processes or "on-the-fly" business processes.

An autonomic business process should be first specified through a high level language. This specification includes the workflow to accomplish the desired task. Actors in this workflow are almost Web services descriptions unless human intervention is absolutely necessary. In that case it falls into a hybrid business process category. At each step of the workflow the infrastructure select the best Web service [10] qualified to perform the sub-task. When some Web service fails then another component is selected instead. Also, a sub-task can be divided among several Web services for load balancing.

3.2. Challenges for autonomic business processes

There are several challenges for ABPs which start from technology issues, following security issues, trusting issues and cultural issues. The discovery of business processes can be an extension of Web services discovery. Also, security concerns have a high priority, since business processes mostly involve money transactions. A gig problem is trusting third party business processes, even more when they have to be autonomously selected. In addition to those mentioned, the most challenging problem is the cultural one. But it can be overcome when the technology to do this will be widely available. Furthermore, the spreading of autonomic business processes is most likely to eliminate such cultural barriers that today's business has in hand.

4. The software side: autonomic web services framework (AWSF)

Web Services are being deployed everywhere and offering a wide spectrum of services. Also, organizations are publishing those Web Services as an open infrastructure. For example, Amazon had published ECS (Amazon E-Commerce Service) enabling developers interact with it through standard protocols. The building blocks of our Framework are Web services. By composing them, can be built complex service-oriented applications (SOA).

4.1. AWS-Net Traveler architecture

Net Traveler's architecture to support autonomic business processes is a non-centralized one. In this section we show Figure 4 shaping the architecture and also we describe each component.

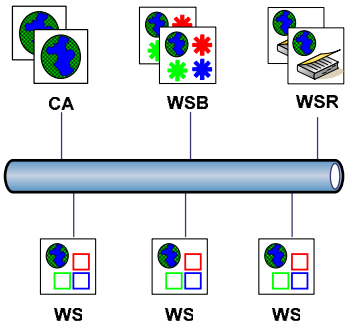

Figure 4 WS-Net Traveler Architecture

4.1.1. Web Service (WS). In our architectural view, Figure 4, WS represents a Web service. It can be described as a software component made accessible over the Internet. A WS is intended to be machine-understandable and to achieve this purpose it uses URI / UDDI / WSDL / SOAP and other specifications. By means of these standards WSs can be adequately identified, discovered and defined, also they are able to interchange data.

4.1.2. Web Service Broker (WSB). WSBs, in general, are responsible for the completion of a cross Web Services Business Processes. Using P2P techniques they deal with discovery and negotiation. They can be plugged with software components to expand their functionality. These extra capabilities include autonomic management, fault tolerance, transaction management and load balancing. WSBs are deployed by organizations owning WSs or also

by organizations which their business is to offer Web Services brokering services. WSBs can be grouped in permanent and ad-hoc federations. [1] They have decentralized coordination.

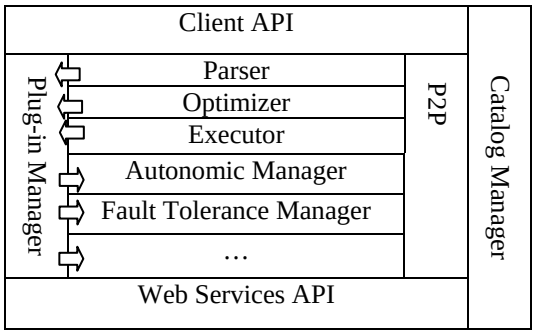

Figure 5 Web Service Broker Architecture

WSB architecture (see Figure 5) includes several core components which are in charge of basic functions and pluggable components which can be installed, upgraded and removed. The first component is the *Client API* which is in charge of the interaction with clients that can be WSB, WS or other applications. The second component is the Parser, which is in charge of interpreting and organizing the request in a execution plan. In third place is the Optimizer that optimizes the plan in accordance with their local information and the information of the neighbors WSBs. If there is something that needs to be executed whiting the context of the QSB the Executor component does it.

A QSB interacts with the Catalog Manager to find other QSB. And, the Catalog Manager, if necessary, triggers discovery mechanisms or it gets Web services definitions from the Web service registration server or another component holding WS descriptions. The P2P communication component is in charge of P2P interactions among WSBs. Finally the Plug-in Manager is in charge of install, update and remove new component in the WSB. Some important pluggable pre-installed components are the Autonomic Manager, the Fault Tolerance Manager and the Transactional Manager.

4.1.3. Web Service Registration (WSR). Additionally, this architecture presents a component in which WSs can be registered. It is an alternative way to make WSs visible to others for use. Also, for collecting WSs can be used P2P techniques for discovery.

4.1.4. Web Service Client Access (CA). CAs, in this architecture, represent front-end web -based or non web-based GUIs which capture users' requests. They are optional, because WSs and WSBs do not rely on them to perform their activities.

4.2. Fault-tolerant and scalable Web service broker architecture

Fault tolerance and availability among Web services brokers is addresses through a bi-dimensional architecture for fault tolerance (see Figure 6). The vertical dimension is for fault tolerance and the horizontal dimension is for scalability through load balancing. Web services brokers are replicated for fault tolerance with the purpose of replace faulty WSBs. In this case can be used active or semi-active replication, which is more suitable for this kind of component, because it will have constant activity. When WSBs are overloaded, the load could be distributed among other WSBs. Thus a task can be divided and each part can be performed by an WSB, instead that all the load will be executed by only ine WSB.

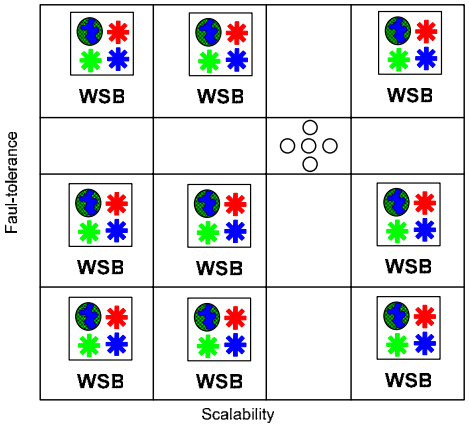

Figure 6 Fault-Tolerant and Scalable Web Service Broker Architecture

4.3. AWS-Net Traveler global architecture

Figure 7 depicts the AWS-Net Traveler Global Architecture. This shows how the components of the AWS-Net Traveler Architecture are deployed whiting a wide-area environment like the Internet. The big circle is a Local Group (LG) where components are grouped usually by functionality. A LG can be composed by several WSs, WSBs, CA, WSRs. Also, a LG includes the orchestration of these components as defined in Caituiro et al. work [1].

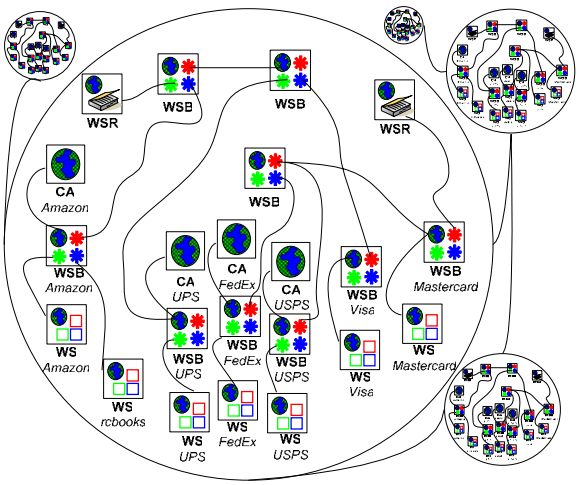

Figure 7 AWS-Net Traveler Global Architecture

Figure 7 also shows small circles which are other Local Groups (LG). Different LG can be linked by mean of the cross relationships among their members.

4.4. Useful definitions

A *federation of Web Services (FeWS)* is a collection of WSs grouped with some purpose. This usually is for functionality or to solve users' requests. For example a federation is created to allow potential vacationers to plan a vacation or regular student to matriculate in a new season. For a vacation planning the federation members are the airline company Web Service, the rent-a-car company Web Service, and the hotel company Web Service.

A *family of Web Services (FaWS)* is a collection of Web Services having an equivalent functionality but they are not necessarily identical. For example Expedia Travel and Travelocity offer equivalent travel services.

A *choreograph* is the execution plan, which is created for each user request. A difference of an *orchestration* that is fixed, a *choreograph* changes for every user request. [1]

A Web Service is *equivalent* to another if it is capable de provide an equivalent service. An equivalent service is one that for example if we want to travel from San Juan, PR to Lima, Peru allows us to do that with one Web service and their equivalent.

5. Model

Web Services can be modeled using graphs. Where vertices are the Web Services and edges are the communication links among them.

5.1. Definition: universe of Web services

The *Universe of Web Services and their connections* includes those running on the Internet and intranets. It can be defined, using graph theory, as a directed graph $U = (W, L)$ where W is the set of all Web Services and L is the set of the connections among Web Services. An element of L is of the form of the pair (w_i, w_j), which means that there is an active connection going from w_i to w_j.

6. Related work

Maximilien et al. [7] in their work introduce a method for service selection considering the services users' preferences and the trustworthiness of providers which is self-adjusted. This approach attaches agents to each Web service, creating a layer between consumers and services in order to include major sophistication in the interaction among consumers and services. Our framework does not use agents, instead introduce a Web service broker that is in charge of service selection and several other basic operations plus those plugged operations.

Birman et al. [9] in the paper "Adding High Availability and Autonomic Behavior to Web Services" present an approach extending the Web Services Architecture with additional components for Fault Detection, Enhanced Communication, and High Assurance. Although, service consumers must remain unmodified for the basic form of fault tolerance, services have to be modified with additional components. Furthermore, for and advanced form of fault tolerance both service consumers and services have to be modified in order to implement the features introduced in this solution. Net Traveler, our approach, includes a Web service broker, a middleware component among web services, does not require service consumers or services to be modified. Indeed extending Web service stack architecture is an optimal solution, currently, there are a large amount of published Web services which are already deployed. We intend to include those installed Web services as part of our solution without requiring any modification on them. Additionally, we believe that separating Web services which represent

some service from Web services which represent management functions is a desirable outcome in an service oriented architecture with loosely coupled components.

Day et al. [10] as their main approach in their work "Selecting the Best Web Service" use dynamic Web service selection. In fact, Web service consumers or clients decide which service to use for selection. This task is accomplished by means of collaboration among them, where Web service clients act as peers on a P2P network. Unlike our approach, this solution requires Web services clients to handle the task of selection, requiring them to include such complexity. Although this approach brings a solution for the problem of selecting the best Web service, it to be used require to redesign and redeploy existing Web services clients.

7. Conclusion and Future Work

In this paper we presented an Autonomic Web Services Framework for Autonomic Business Processes. We addressed it from two points of view: the first was from the business point of view and the second from the software point of view. From the business point of view we described how the business processes should be changed to become autonomic. And, from the software point of view we presented an architecture for Autonomic Web Services where the main component is a Web services broker. Also, was described the Web service broker component architecture and a fault-tolerant and scalable architecture for Web services brokers was introduced.

Currently, we are implementing a prototype and a simulation to validate our framework and we will present it when will be done. Also, there are still several detail to be specified and described, including the model.

8. References

[1] H. Caituiro-Monge, M. Rodríguez-Martinez, "Net Traveler: A Framework for Autonomic Web Services Collaboration, Orchestration and Choreography in E-Government Information Systems", ICWS, IEEE, San Diego, California, 2004, pp. 2-10.

[2] S.R. White, J.E. Hanson, I. Whalley, J.O. Kephart "An architectural approach to autonomic computing", International Conference on Autonomic Computing, 2004, 17-18 May 2004, pp. 2-9.

[3] J.O. Kephart, D.M. Chess, "The vision of autonomic computing", Computer, January 2003, Volume: 36 , Issue: 1, pp. 41-50

[4] K. Hogg and P. Chilcott and M. Nolan and B. Srinivasan, "An evaluation of Web services in the design of a B2B application", CRPIT '26: Proceedings of the 27th conference on Australasian computer science, Australian Computer Society, Inc., Dunedin, New Zealand , 2004, pp. 331-340.

[5] B. Medjahed and B. Benatallah and A. Bouguettaya and A. H. H. Ngu and A. K. Elmagarmid, "Business-to-business interactions: issues and enabling technologies", The VLDB Journal, Springer-Verlag New York, Inc., 2003, pp. 59-85.

[6] BEA Systems, IBM, Microsoft, SAP AG and Siebel Systems, "Business Process Execution Language for Web Services version 1.1", http://www-128.ibm.com/developerworks/library/specification/ws-bpel/, May 2003.

[7] E. M. Maximilien, M. P. Singh, "Toward Autonomic Web Services Trust and Selection", In Proceedings of 2nd International Conference on Service Oriented Computing (ICSOC 2004), New York, November 2004,

[8] H. Kreger, "Fulfilling the Web services promise", Communications ACM, AC M Press, volume: 46, number: 6, 2003, pp. 29-ff.

[9] K. Birman, R. v. Renesse, W. Vogels, "Adding High Availability and Autonomic Behavior to Web Services", ICSE '04: Proceedings of the 26th International Conference on Software Engineering, IEEE Computer Society, 2004, pp. 17-26.

[10] J. Day, R. Deters, "Selecting the best web service", CASCON '04: Proceedings of the 2004 conference of the Centre for Advanced Studies on Collaborative research, IBM Press, Markham, Ontario, Canada, 2004, pp. 293-307.

[11] J. Shaw, "The Autonomic Enterprise", Quest Magazine, December 2003.