Investigations on implementation of e-ATM Web Services based on .NET technique

Subhash Medhi, National Hydro Electric Power Corporation Ltd., Loktak Power Station, Manipur, India

Tulshi Bezboruah, Department of Electronics and Communication Technology, Gauhati University, Guwahati, Assam, India

ABSTRACT

Web service is accepted by all communities of services due to its interoperability, flexibility of design and architecture. Improving performance is the key research point for developers and researchers. We propose to design, develop and implement a prototype research electronic automated teller machine web service to study the loads, performance and scalability of the service. The service is based on Visual Studio .NET framework. It is developed by using C# as programming language, Internet Information Service (Version: 2005) as web server and, Microsoft Structured Query Language as database server. To study its different aspects the service has been tested by deploying it on Mercury LoadRunner (Version 8.1). In this paper we will present the architecture, testing procedure, results of performance testing as well as results of statistical analysis on recorded data of the service.

Keywords:  C#, e-ATM, IIS, Mercury LoadRunner, MS SQL, SOAP, Web Service, XML

1. INTRODUCTION

Web Service (WS) is a service designed to operate within machines to communicate over diverse networks (Catalina M. Llado, Ramon Puigjaner, Connie U. Smith, 2005). It is a server application that acts on server side and execute job when requested by an application. In the article “Web Services Architecture retrieved from web site: http://www.w3.org/TR/2002/WD-ws-arch-20021114” says that the Uniform Resource Identifier (URI) recognises WS, whose objects and bindings are self defined, described and exposed as Extensible Mark up Language (XML) object. It is a heterogeneous computing technology that offers interaction and collaboration among business community and customers (John B. Oladosu, Funmilola A.
Ajala and Olukunle O. Popoola, 2005). The WS eases Business to Business (B2B) collaboration by incorporating compatible services to a large extent (Bora A., Bhuyan M.K and Bezboruah T., 2013). Application of WS reduces the time, cost constraints for discovering, discussing and running e-business communications (Chen Minder, Chen Andrew N.K. and Shao Benjamin B.M, 2003). The paradigm of WS could be the tactical model for success of next generation of Internet computation (Jia Zhang & Liang-Jie Zhang, 2005). AWS supports direct communication with other software applications using XML based messages via internet protocols only. It is based on three common protocols exclusively XML which includes: (a) the Simple Object Access Protocol (SOAP) for making communication between applications and information exchanges, (b) Web Service Description Language (WSDL), a format of XML which describes detail laid down procedures and access format of data and (c) Universal Description Discovery and Integration (UDDI), which is the central directory where available services are published.

1.1. Related Works

Catalina M. Llado, Ramon Puigjaner and Connie U. Smith (2005) illustrated the use of WS for the selection of performance modeling tools in a plug and play manner to determine the best tools to analyze the performances of software. To establish the viability of a performance WS, they developed a prototype where the modeling tool used the WS is Software Performance Engineering tool SPE.ED and the tool that solves the model is Quality Network Appliance Provider (Qnap). They suggest that performance model WS may be used in Software Performance Engineering (SPE) to automate the estimation of software architecture and design.

Hou Zhai-wei, ZHAI Hai-xia, and Gao Guo-hong (2010) studied the performance of SOAP processing on Java platform and suggested that WS performance can be improved by optimizing the SOAP messages. The XML parsing, de-serializing and serializing are the most time consuming phases during SOAP messages exchanges and that effect the WS performance in a large scale. By adopting some optimizing methods the WS performance can be improved such as a) SOAP compression and b) Use of Cache in client-side caching, server-side caching and multilevel cache.

Henrique Jorge A. Holanda, José Merseguer Hernández, Giovanni Cordeiro Barroso and A. B. Serra (2010) proposed a model to transform the WS Business Process Execution Language extension for people (WS-BPEL4People) into Generalized Stochastic Petri Nets (GSPN) and compared the performance of the generated model GSPN with WS-BPEL4People in terms of response time and found that it is not differ by more than 6.7% with the existing WS-BPEL4People. And justified the validity and usability of the generated model to evaluate the performance of the WS-BPEL4People.

Tripathi Sandesh and Abbas S Q (2010) performed the study of WS performances under simulation and hosted condition. It was Identified that serialization and deserialization is the time consuming phases of WS messages. They presented the comparison of WS performance parameters like response time(s) and throughput (byte/s) under load of 300 users and observed the similar performance of the WS under simulation and hosted condition.

Douglas Rodrigues et al. (2011) presented a study on WS performance with or without security standards. The WS is crucial while exchanging data. 5 concurrent users while given load on the WS, the average response time of the system without security is 283.68ms, with digital signature, the time is 1547.76ms, with encryption, the time is 2861.31ms and the WS with digital signature followed by encryption is 3094.20 ms and they concluded that the average response time of WS without security is lowest and the time taken in encryption is more than the digital signature.

Bora A, Bhuyan M.K and Bezboruah T (2013), the authors presented a performance analysis on hierarchical prototype WS with 5000 data size for medical advices of doctors for various diseases. The WS has been tested.
Ranking Algorithm for Semantic Document Annotations
www.igi-global.com/article/ranking-algorithm-semantic-document-annotations/72703?camid=4v1a