CHECKING THE PERFORMANCE OF DYNAMIC COMPOSITE VIDEO-ON-DEMAND WEB SERVICE

G. Maria Kalavathy
Research Scholar
Sathyabama University, Chennai
Tamil Nadu, India
Email: maria_kalavathy@yahoo.co.in

P. Seethalakshmi
Director Center for Academic Excellence
Anna University Trichy
Tamil Nadu, India
Email: auropansee@yahoo.co.in

ABSTRACT

The main objective of this paper is to design and implement Dynamic Composite Video-on-Demand Web Service (DCVWS) and check its performance through Parallel Performance Monitoring Service (PPMS) during run-time. The media web services such as Video-on-Demand (VoD), Music-on-Demand (MoD) and News-on-Demand (NoD) are composed at design time using BPEL (Business Process Execution Language) designer and composition can be changed at run-time depends on the user preferences. In addition to existing BPEL exception handling logic, there is a need to check the performance of dynamic composition of service oriented applications at application and system level. This paper proposes Parallel Performance Monitoring Service (PPMS) as a monitoring web service executed in parallel with each requested Media Web Service using multi threading technology and it monitors the execution of Dynamic Composite Video-on-Demand Web Service (DCVWS) which are represented as a BPEL process. The system QoS parameters such as finding response time, delay and external errors, calculating percentage of successful completion of individual media web services, and checking the time at which response time starts to increase are used to measure the run-time performance of the media web services. The application QoS parameters measure run-time performance of media web services that include media characteristics such as frame rate and frame resolution. These parameters are monitored at run-time to get a clear view of how VoD web services perform within their operational environments. The effectiveness of PPMS has been evaluated for Video-on-demand composite web service and its results indicate an improvement in the performance of run time monitoring of media web services.

Keywords: Media-on-Demand, Media Service Composition, Multi threading, PPMS.

1. INTRODUCTION

Emerging advances in distributed media services, such as video conferencing, media-on-demand and ubiquitous multimedia streaming, demands a scalable, robust and adaptive media service infrastructure. The media service composition concepts are prominent approaches to advance construction of large scale distributed media services in a scalable, easy-programmable and efficient manner [1]. The media service compositions are done dynamically; therefore the composed service is named as Dynamic Composite Video-on-Demand Web Service (DCVWS). The VoD web service composition is a process where multiple media services such as media retrieval, transcoding and display services are connected via functional and data dependencies over heterogeneous and distributed network infrastructures. The DCVWS and its performance monitoring at run time are necessary to provide adaptive media web services. The dynamic service composition allows the media services to be composed dynamically from components of distributed web services according to the requirements from different users. The components of distributed media services were developed and hosted in heterogeneous environments. This Composite Media web service provides streaming of real-time media that allows the user to choose the media services according to their choice in a user friendly way. Though the above provides flexibility in choosing the kinds of information, ensuring the high performance of multiple media services is still a challenge. The demanded services are to be readily available to the end users; else it will create negative effects on the reputation of service provider or result in loss of business opportunities. Hence the performance of the VoD web services is to be checked during run-time and its results are informed to the service provider to take corrective action. The performance of the web services are influenced by many factors such as network traffics, host workloads and host running environments. It is very important to monitor the run time performance of dynamically composed media web services during the process of their invocation in client side. There are few reported researches on performance monitoring of media web services especially video-on-demand services after it is deployed.

The DCVWS are designed and implemented as BPEL processes and the proposed Parallel Performance Monitoring Service (PPMS) checks the performance of each individual media web services that are requested using multithreading technology. The runtime performance characteristics such as response time, timeout, external errors, percentage of successful completion of individual web services, occurrence of fault and media quality in terms of frame rate and frame resolution are monitored at system level and application level. The results of this run time performance monitoring are used by service providers to get a clear view of how media web services perform within their operational environments and to perform control actions to modify and adjust their behavior accordingly at run-time.

The specific monitoring challenges of media web services are finding the percentage of successful completion of media web service, measuring the media quality in terms of frame rate and frame resolution and checking when the response time starts to degrade by checking the load of the media web servers. These performance monitoring challenges for Video-on-Demand web services are addressed in this paper effectively. The rest of this paper proceeds as follows: Section 2 explains the related work on dynamic media web service composition and monitoring methods of web services. Section 3 illustrates the designing of DCVWS based on user preferences in terms of media quality and media format. The various performance parameters that are monitored at run-time using PPMS are explained in Section 4. In Section 5 the performance monitoring of DCVWS is evaluated and results are explained. Section 5 concludes this paper with its merits.

2. RELATED WORK

The media service composition concepts are main approaches to advance construction of large scale distributed media services in a scalable, easy-programmable and efficient manner. According to user preferences in terms of QoS parameters the media service composition can be made flexible. It will create high dynamic effect and also gains the user satisfaction. The QoS-based web service selection and composition in service-oriented applications has gained more attention of many researchers [2][3]. Performance is one of the most important Non-Functional Requirements (NFR) of service based media applications. Because service-based software development for media applications is emerging technology, there have been no reported performance assurance studies on media web service applications. Most of the performance assurance testing is performed before the deployment of the web services.

The MSU video quality measuring tool [4] measures video quality using metrics such as peak-to-peak signal-to-noise ratio (PSNR), Delta, MSAD (mean absolute difference of the color components) , MSE SSIM Index (measuring of three components luminance similarity, contrast similarity and structural similarity), VQM (uses DCT to correspond to human perception), MSU Blurring and MSU Blocking. These metrics are not in terms of service oriented computing concepts. In our paper, video quality is measured in terms of frame rate and frame resolution and the percentage of successful completion of video for each video services requested by the user. Compile-time analysis techniques to perform the white-box testing of exception handlers in Java web services are analyzed by Fu et al[5]. Huang et al. [6] have designed a software tool to assess web application security which is based on software testing techniques such as dynamic analysis, black-box testing, fault injection and behavior monitoring. Offutt and Xu [7] proposed an approach to test web services based on data perturbation and interaction perturbation, which uses two types of communication mechanism such as RPC communication and data communication. Liu et al [8] proposed a web test model, which considers each web application component as an object and generates test cases based on data flow between those objects. Gorton and Liu [9] designed a middleware infrastructure, the transaction and directory services and the load balancing to compare the performance of six different J2EE-based distributed applications. Avritzer et al. [10] compared the performance of different Object Request Broker (ORB) implementations that are related to the CORBA Component model. Liu.Y et al. [11] evaluated the suitability of light-weight test cases on distributed applications.

Carolyn McGregor and Josef Schiefer [12] described a framework which uses process definition information to define web service to the solution manager service. They also introduced the concept of the Event Processing Container providing a robust, scalable and high-performance event processing environment able to handle a large number of process events in near real-time. Luciano Baresi et al. [13] proposed an approach to monitor timeouts, runtime errors and violations of functional contracts of service compositions defined by BPEL processes using assertions. Liguo Yu [14] proposed software wrapping technique that is used at client side. The clients interact with the service through the wrapper which customize the messages exchanged between client and service and monitors the performance of the service by calculating the response time only. But this paper is an attempt to calculate response time using software wrapping technique from the service provider point of view to take immediate action if the performance is poor. Khaled Mahbub et al. [15] described the framework to monitor behavioral properties and assumptions at run time using event calculus. William N. Robinson [16] proposed REQMON monitoring system that raises only an alert by sending a failure message to the global monitor. It can't recover web service from the point at which the failure occurs. Arne Koschel and Irina Astrova [17] designed a configurable event monitoring web service which is useful in the context of Event Driven Architectures (EDA) and Complex Event Processing (CEP). Onyeka Ezenwoye and S.Masoud Sadjadi [18] presented an approach to transparently adapting BPEL processes to tolerate runtime and unexpected faults and to improve the performance of overly loaded web services. They presented an another approach in which when one or more partner services do not provide satisfactory service the request for service is redirected to one of these static, dynamic and generic proxies, where the failed or slow services are replaced by substitute services [19]. But this approach is not used for recovering the web service from the point at which the fault is occurred.

There is no reported research on performance monitoring of media web services especially calculating the percentage of successful completion of media web service. Hence it has been chosen to design and implement DCVWS and check its run time performance through PPMS. It is very much useful in media web applications to recover from the point at which the fault has occurred.

3. DESIGN AND IMPLEMENTATION OF DYNAMIC COMPOSITE MEDIA WEB SERVICE (DCMWS)

The Dynamic Composite Video-on-Demand Web Service (DCVWS) is designed as a business process using BPEL Designer at design time. According to the user preferences such as media quality, media type, software availability and type of device used, the composition can be changed at run-time. The BPEL process includes the sub processes such as Client authentication process, search and SLA process and different types of video service with different quality. The web services such as authentication service, new user registration service are available in the first sub process client authentication process. The second sub process Search and SLA includes Search service, SLA service, Selection service and PPMS. The search and SLA service is implemented with NOT-Shift-AND search algorithm [20] to perform parallel search on the database called Web Service Map (WSM) to find the web service host that contains the requested service with required parameters.

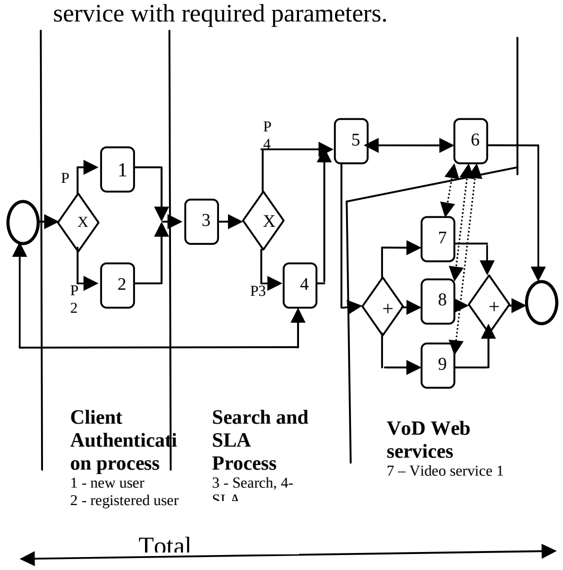

Fig. 1. BPMN diagram for Orchestrating DCVWS

If the requested service with required parameters is not available then Service Level Agreement (SLA) is made on the parameters that are conflicting with service provider and service user. The third sub process VoD web services includes various VoD web services with different media quality and media formats that are deployed on multiple web service hosts. The BPMN (Business Process Modeling Notation) diagram is used to explain the composition of these web services as shown in fig 1.

Sample DCVWS:

Case 1:

For instance at first time user requesting for video with parameters such as cost range, frame rate, frame resolution, device type and software used, assuming that the same is available as service s8:

DCVWS will be: s1→s3→s5→s6→s8

Case 2:

If the requested service is not available, service level agreement (SLA) is drafted between the service provider and service user by invoking SLA service S4. Based on the agreement, the available service s9 is invoked. Now the DCVWS will be: s1→s3→s4→s5→s6→s9

Case 3:

When the service is interrupted in between and the same user logs in next time, the information about the video service already viewed has to be provided. This time the DCVWS will be: s1→s9

4. PARALLEL PERFORMANCE MONITORING SERVICE (PPMS)

The PPMS is the web service and one of the components of BPEL process of DCVWS. The BPEL engine incorporates fault handling logic to handle only software exceptions and that is not enough to monitor the run-time performance of web services. The PPMS monitors the run time performance of DCVWS using three mechanisms such as software wrapping in server side, methods to monitor the media quality of individual media web services and polling technique to monitor how much percentage of video is delivered correctly. These three mechanisms correspond to five classes of performance characteristics such as response time, timeouts, external errors, media quality and percentage of successful completion of the media web service.

For each media service requested, PPMS thread is created in server side that checks the above mentioned performance characteristics. When such undesirable condition is detected by the PPMS, it returns the status of monitoring such as Service Unavailable Fault, Timeout fault and execution fault, loss in media quality during VoD services is delivered to service providers to take necessary recovery actions.

A. *Monitoring Timeouts and Response time*

The timeout is the interrupt signal that is generated when the client does not get the response with in the specified time limit. ActiveBPEL allows the designer to set the timeout for scopes and all the operations in the scope must finish their execution within the time limit set by timeout. But PPMS set the time limit and exception handling procedures for each and every individual web services and monitors this time limit at runtime. The time limit is chosen as 75 seconds using the default value of the parameter tcp_ip_abort_cinterval which is the second threshold timer used during connection establishment.

The response time is the delay between a request and the completion of an operation which is monitored by software wrapping technique in the server side. Software wrapping refers to a reengineering technique that surrounds a software component or system with a new software layer to hide the internal code and the logic of the component. The server receives the client request through wrapper. Here the wrapper provides the customization of messages exchanged between the client and the service to monitor the performance of the service i.e., calculating response time.

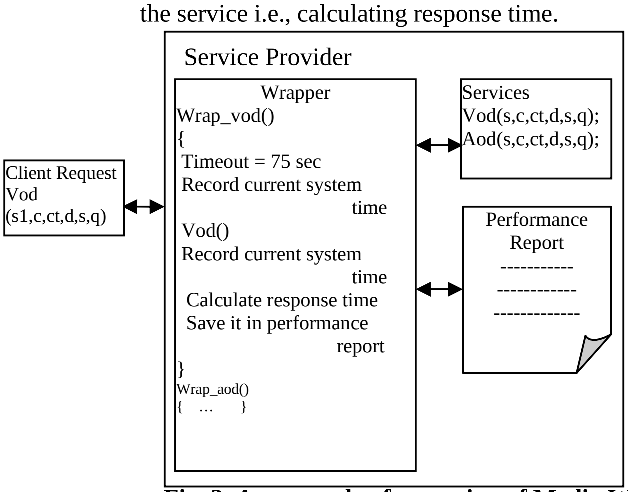

Fig. 2. An example of wrapping of Media Web services.

Fig. 2 shows an example of a wrapper program which is available as a part of PPMS in server side. The client request for VoD web service comprises of service name, cost-range, completion time, device type, software available and quality (vod(s1,c,ct,d,s,q)) which is given to server through wrapper. For each VoD request a new thread of PPMS is created which in turn executes the wrapper program that calculates the response time of the video-on-demand service, checks the timeout and records the results in the performance report i.e., log file. In this approach software wrapping is implemented at the service provider side, because the PPMS is used to monitor the performance of the media web service from the server point of view to take recovery action if the response time starts increasing. This involves less overhead to execute the small wrapper code before starting the execution of requested VoD web service and it does not affect the currently running VoD service.

B. *Monitoring External Errors*

When the individual web service in the dynamic composition fails due to unforeseen internal bug and external errors, the whole composition i.e., process is failed. To recover from this situation, PPMS defines faultHandler that can take care of the failure of the invoked service using a catchall clause. For each type of exceptions, the handling method is specified that is displaying the error type to the service provider and terminate the execution. A part of sample code in PPMS is shown below which is used to handle the faults occurred in individual web service in the composition.

```
<faultHandlers>
<catchall>
<sequence>
 <!--handle exceptions by communicating the
error to the service provider to take necessary steps
and  terminating-- >
   <error>
      <errortype> Software Exception </errortype>
      <action>Terminate </action>
   </error>
   ----
</sequence>
</catchall>
</faultHandlers>
```

C. *Monitoring the Percentage of Successful Completion*

When the Video-on-Demand (VoD) service is requested, a monitor thread is created for PPMS to monitor the performance of VoD web service and synchronization is achieved between them to exchange the data. Parallelization of web services with communication construct are provided by Multithreading technology with synchronization. The parallelism with communication construct is used to execute concurrent web services and to synchronize or exchange certain data between them during execution. Fig. 3 explains the parallel execution of VoD web service and PPMS with communication among the services. The special processing element facilitates data sending and retrieval and formalizes it as process related instances.

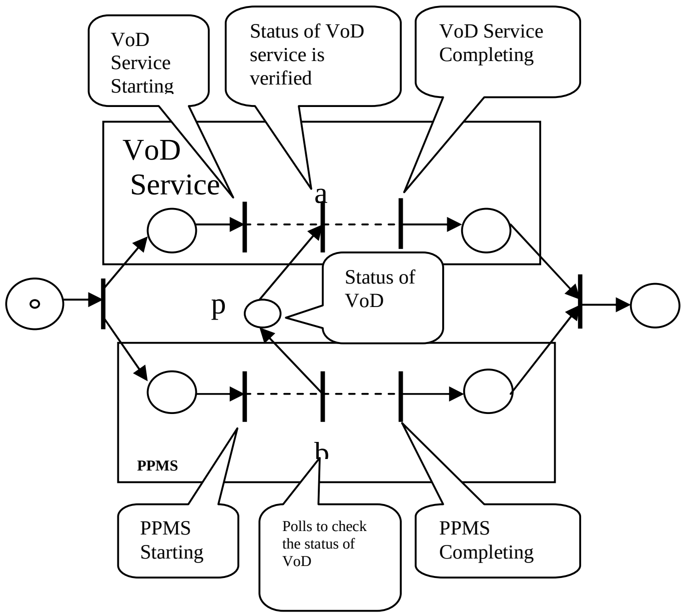

Fig. 3. Parallel execution of VoD Web service and PPMS.

The fig. 3 shows the simultaneous operation of PPMS and VoD web service. The information about the status of the VoD service is monitored by PPMS at different times using polling technique and stores the status such as polling time, correctness of the service, polling time interval in the polling table. The correctness of the service is indicated by the flag variable called response. The sample polling table is shown in table 1. At run time the percentage of completion is calculated by the following equation using the information given by PPMS:

$$\text{Percentage of successful completion (P)} = \frac{n \times Pt}{N} * 100 \quad (1)$$

In the equation 1, n represents number of polls, Pt is polling time interval and N is the total media time.

For instance let the polling time interval (Pt) be 2 minutes for a video clip running for 10 minutes. Then PPMS polls for five times. If the third poll happens to be non-responsive, then it is intimated immediately to take necessary action. The action to be taken is out of scope of this paper and will be done in future using corrective adaptation techniques.

TABLE I
POLLING TABLE

	Polling time (Pt) in seconds	**Response**
Poll 1	120	YES
Poll 2	120	YES
Poll 3	120	NO

D. Calculating Frame rate

The important factor that determines the quality of video is the frame rate that varies according to the bandwidth of the network. Initially, for the known start and end time of the video, the frame rate (fps-frame per second) for the requested video is calculated using algorithm Calc_Frame_Rate (v1,Start_Time, end_Time) as mentioned below.

Algorithm to calculate Frame Rate:

```
Calc_Frame_Rate(Video v1, Time Start_Time, Time End_Time)
video_splitter(v1) // splits the video in to frames and stores them in to a folder named 'frames'
int Frame_Count = 0
while not end of frames
Frame_Count ++
Start_Seconds = 3600 * s_hh + 60 * s_mm + s_ss
End_Seconds =  3600 * e_hh + 60 * e_mm + e_ss
Total_Seconds = End_Seconds – Start_Seconds
Frame rate = Frame_Count / Total_Seconds
```

The frame rate is calculated by dividing the frame count by total media time. The calculated initial frame rate is subjected to change according to the bandwidth of the network. Since the network bandwidth is known at the time of collecting user preferences, the frame rate can be adjusted effectively during run-time.

5. EVALUATION

To evaluate the proposed PPMS, a series of experiments on dynamically composed media web services such VoD service executions were carried out. In this experiment BPEL process was created for media web services which include the individual web services such as authentication service, new user service, Search service, SLA service, Selection service, PPMS, VoD service with different quality and type.

The objective of this experiment was to measure a) response time for multiple clients b) number of timeouts detected at run time c) percentage of successful completion of VoD service d) number of exceptions raised due to external errors and e) media quality in terms of frame rate and frame resolution. The performance parameter response time of the requested media web service is the sum of the response time of the individual services in the composition described in the figure 3. The response time (RT) of the Video-on-Demand web service is calculated by software wrapping technique. For each request of VoD, new thread of PPMS is created which executes software wrapping code to calculate the response time. The sample output of log file which is created at run time is shown in Fig. 4. The parameters that are monitored and stored in this log file are client ID, service name, wait interval, process ID, media file name, response time, media quality (frame rate and frame resolution) and errors if occurred.

```
Media Service History
Client IP: 196.168.1.73
Service Name: SelectionVideo
Wait Interval: 3sec
Process Id: java.lang.ProcessImpl@140cc82
File name: Advaitha.wmv
Error: Timeout expires

Media Service History
Client IP: 196.168.1.75
Service Name: SelectionVideo
Wait Interval: 3sec
Process Id: java.lang.ProcessImpl@1852f24
File name: Advaitha.wmv
Response Time: 3sec
Exit value: 0
PlayDuration: 8sec
Frame rate: 60fps
Frame resolution: 720x576
Termination: Normal Exit
```

Fig. 4: Sample output of PPMS

The graph for dynamic requests is shown in Fig. 5 which describes the variation in response time based on the number of simultaneous requests handled by the service provider. As shown in the graph, the response time increases almost linearly until about 100 concurrent requests. After 100 requests, the performance in terms of response time degrades considerably.

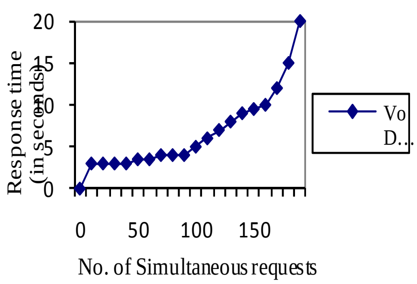

Fig. 5. No. of Simultaneous requests vs Response time

The next performance parameter is the timeout which is the interrupt signal that is generated when the client does not get the response within the specified time limit. The network traffic conditions and server workloads lead to timeout of the service that is it cannot respond within the specified time interval. ActiveBPEL allows the designer to set the timeout for scopes and all the operations in the scope must finish their execution within the time limit set by timeout. But PPMS sets the time limit as 60 seconds and exception handling procedures for each and every individual web services and monitors this time limit at runtime.

The percentage of successful completion of the media service was found using polling technique. The PPMS thread parallely monitors the media web service by sending polling message and checks the response. If the response is received, it is entered in the polling table. Otherwise the service provider checks the polling table and calculates the percentage of successful completion of media service using the Eq. (1).

The external errors are monitored by PPMS using passive testing and the corresponding exception handler is invoked to give the information about the external errors to the service providers. The passive testing is the testing or observing the interaction between a service provider and clients. The external errors are simulated that is restart the client or pause the operation in between. All these errors are monitored and the fig. 6 shows the failure rate of VoD service due to external errors. It is measured as the ratio of the number of failure requests over total number of requests over a period of time.

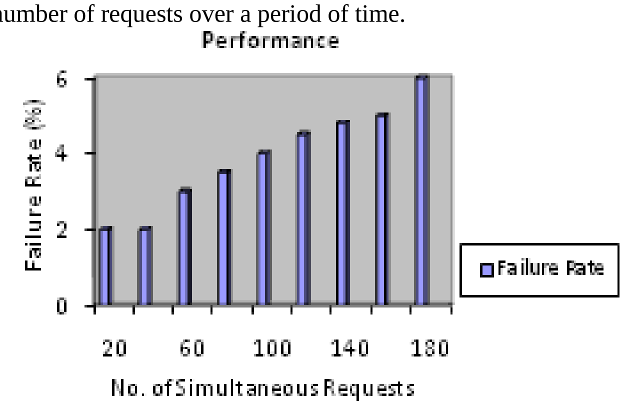

Fig. 6. No. of Simultaneous requests Vs Failure rate

6. CONCLUSION

The Dynamic Composite Video-on-Demand Web Service (DCVWS) was designed, implemented and through Parallel Performance Monitoring Service (PPMS) its runtime performance was checked. Software performance assurance of web services has not been thoroughly investigated because of its dynamic binding and independent service implementation. But the PPMS provides four mechanisms such as software wrapping in server side, monitoring timeouts, polling technique to monitor the successful completion of media web service compositions defined by BPEL and method to calculate frame rate. These four mechanisms were implemented to monitor the performance characteristics such as response time, timeouts, external errors, percentage of successful completion of the media web service and media quality. This paper focused on monitoring the portion of uninterrupted video delivery which proved to be the most significant part of the study. This result will be more useful for the service providers to take necessary recovery actions. The importance of checking the performance of dynamically composed VoD web services were realized and also showed the improvements on run-time performance monitoring of media web services.

REFERENCE

[1] Wolf-Tilo Balke and Klara Nahrstedt, “Multimedia Service Composition: A Brave New Topic”, ACM Multimedia Conference, 2004, pp.1-2.

[2] Liu Y, Ngu, A.H.H, Zeng L, “QoS Composition and Policing in Dynamic Web Service Selection”, WWW, 2004, pp.66-73.

[3] Zeng. L, Benatallah. B, Dumas. M, Kalagnanam. J and Sheng. Q.Z, “Quality driven web services composition”, WWW, 2003, pp. 411-421.

[4] MSU Video Quality Measurement Tool, Available: http://compression.ru/ video / quality_measure / video _measurement_tool_en.html

[5] Fu C Ryder, B G Milanova and Wonnacott D, “Testing of java web services for robustness”, 2004 ACM SIGSOFT international symposium on Software testing and Analysis, 2004, pp.23-24.

[6] Huang, Y Huang S, Lin T and Tsai C, “Web application security assessment by fault injection and behavior monitoring”, 12th International World Wide Web Conference, 2003, pp.148-159.

[7] Offutt J and Xu W, “Generating test cases for web services using data perturbation”, ACM SIGSOFT Software Engineering Notes, 29(5), 2004, pp.1-10.

[8] Liu C, Kung D, Hsia P, “Structural testing of web applications”, 11th International Symposium on Software Reliability Engineering, 2000, pp.84-96.

[9] Gorton I and Liu A, “Software component quality assessment in practice: successes and practical impediments”, 24th International Conference on Software Engineering, 2002, pp.555-558.

[10] Lin C, Avritzer A, Weyuker E and Sai-Lai L, “Issues in interoperability and performance verification in a multi-ORB telecommunications environment”, International Conference on Dependable Systems and Networks, 2000, pp.567-575.

[11] Liu Y, Gorton I, Liu A, Jiang N and Chen S, “Designing a test suite for empirically-based middleware performance prediction”, 14th International Conference on Tools Pacific: Objects for Internet, Mobile and Embedded Applications, 2002, pp.123-130.

[12] Carolyn McGregor and Josef Schiefer, “A Framework for Analyzing and Measuring Business Performance with Web Services”, IEEE International Conference on E-Commerce (CEC’03), 2003, pp.405-412.

[13] Luciano Baresi, Carlo Ghezzi and Sam Guinea, “Smart Monitors for Composed Services”, 2nd International Conference on Service Oriented Computing, ICSOC’04, 2004, pp.193-202.

[14] Liguo Yu, “Applying Software wrapping on Performance Monitoring of Web Services”, Journal of Computer Science, 2007, Vol. 6, pp. 1-6.

[15] Khaled Mahbub and George Spanoudakis, “Run-time Monitoring of Requirements for Systems Composed of Web-Services: Initial Implementation and Evaluation Experience”, IEEE International Conference on Web Service (ICWS’05), 2005, pp. 257-265.

[16] William N Robinson, “Monitoring Web Service Requirements”, 11th IEEE International Requirements Engineering Conference, 2003, pp.65-74.

[17] Arne Koschel and Irina Astrova, “Event Monitoring Web Services for Heterogeneous Information Systems”, World Academy of Science, Engineering and technology, 2008, Vol. 33, pp.50-52.

[18] Onyeka Ezenwoye and S Masoud Sadjadi, “A Proxy-Based Approach to Enhancing the Autonomic Behavior in Composite Services”, Journal of Networks, Vol. 3, No.5, May 2008, pp. 42-53.

[19] Onyeka Ezenwoye and S Masoud Sadjadi, “Robust-BPEL: Transparent Autonomization in Aggregate Web Services Using Dynamic Proxies”, Technical Report: FIU-SCIS-2006-06-01.

[20] G.Maria Kalavathy, P.Seethalakshmi, “Enhancing the Availability of Composite Real-Time Media Web Service”, 10th IEEE Conference on High Performance Computing and Communications, Sep 25-26, 2008, pp.1001-1006.

Biography

G.Maria kalavathy has received B.E degree from Madurai Kamaraj University in Computer Science & Engineering in 1993, and M.E degree from Sathyabama University in 2002. She has 15 years of teaching experience and currently she is working as a Assistant Professor, Department of Computer Science and Engg. in St.Joseph’s college of Engineering, Chennai, India. Her area of interest includes Service Oriented Architecture, Multimedia Streaming and Distributed Computing. Currently she is pursuing Ph.D degree from Sathyabama University, India.

Dr. P. Seethalakshmi has received her B.E. degree in Electronics and Communication Engineering in 1991 and M.E. degree in Applied Electronics in 1995 from Bharathiar University, India. She obtained her doctoral degree from Anna University Chennai, India in the year 2004. She has 15 years of teaching experience and currently she is occupying the position as Director, Center for Academic Excellence, Anna University, Trichy, India. Her area of research includes Multimedia Streaming, Distributed Computing, Wireless Networks, Network Processors and Web Services.