An Intelligent Control Architecture for Adaptive Service-based Software Systems with Workflow Patterns

Chang-Hai Jiang[1]*, Hai Hu[1]*, Kai-Yuan Cai[1]*, Dazhi Huang[2]**, and Stephen S. Yau[2]**
*Beijing University of Aeronautics and Astronautics, Beijing 100083, China
{huhai, changhai.jiang, kycai}@buaa.edu.cn
**Arizona State University, Tempe, AZ 85287-8809, USA
{dazhi.huang, yau}@asu.edu

Abstract

Service-oriented architecture (SOA) for distributed computing has become increasingly popular due to the great advantage that distributed applications can be rapidly synthesized with the needed services provided by various service providers through broadband networks. Systems based on SOA are called Service-based Systems (SBS). An important and difficult issue is how to develop SBS adaptable to constantly changing user requirements, environments and resource constraints. In this paper, an intelligent control architecture is presented to address this issue. This architecture has three layers based on the intelligent control theory for developing and deploying SBS. An example to illustrate the intelligent control architecture for adaptive SBS is given and preliminary experimental data is presented to demonstrate the feasibility of our approach.

1. Introduction

Recently, the emergence of service-oriented architecture (SOA) has enabled great advances in distributed applications, such as web-service based applications, enterprise computing infrastructures, and Global Information Grid (GIG) [13][14]. Systems based on SOA, called *service-based systems (SBS)*, can be rapidly composed with services provided by various service providers through heterogeneous networks. However, SBS need to be adaptable to constantly changing user requirements, environments, and resource constraints [5][10]. To achieve this goal, the following two major aspects in rapid development of SBS need to be considered:

- How to design a SBS adaptable to changing user requirements, performance specifications and resource constraints?
- How to detect or predict violations of timing and resource constraints for certain workflows in runtime, and adapt the system to overcome such violations?

In this paper, we will present the concept of a three-layer architecture based on intelligent control theory for designing and deploying SBS with changing time and resource constraints. Multiple and dynamic QoS requirements and resource constraints are taken into consideration The principles of feedback control [6] have been applied for adaptive Web server architecture to guarantee connection delay of services. Our architecture will adopt the principles of feedback control and make SBS based on our architecture adaptable to changing user requirements and network environment (including available bandwidth and security levels). An example will be given to illustrate the feasibility of using our architecture for SBS.

2. Adaptive Service-based System

Before introducing our adaptive SBS, we first need to identify the basic entities in an SBS and show how workflows are executed to satisfy user requirements. In [12], various entities in an SBS, including *agent, service, host, domain, user, service provider*, and *domain administrator*, as well as the relations among them were defined. We will summarize them here for sake of completeness:

- Agent, a software component owned by a user and acting on behalf of the user to use various services.
- Service, the capability provided by a service provider and provisioned by a host.
- Host, a computer in the network, where agents and services can be deployed.
- Domain, a group of hosts administrated as a unit with common rules. Domains and hosts are managed by domain administrators of various organizations with different resource usage policies.

[1] The work reported here was supported by the National Science Foundation of China and Microsoft Research Asia Grant No. 60633010 and the 863 Programme of China Grant Nos. 2006AA01Z174 and 2006AA01Z157.
[2] The work reported here was partially supported by National Science Foundation under Grant No. CCF-0725340.

0730-3157/08 $25.00 © 2008 IEEE
DOI 10.1109/COMPSAC.2008.184

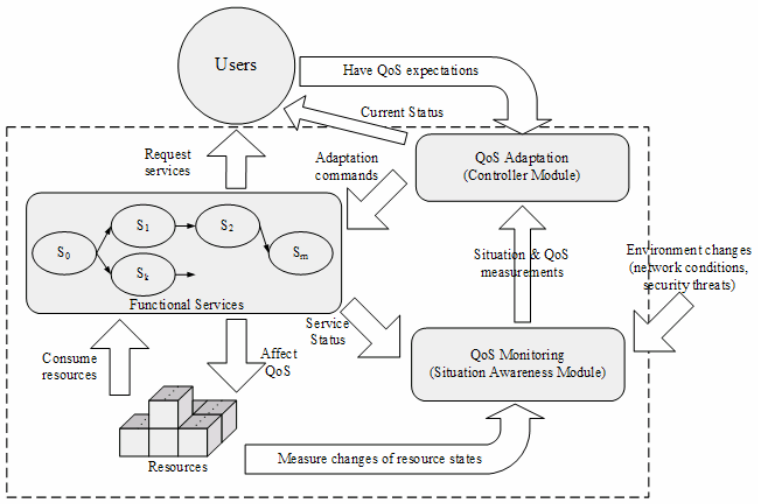

Figure 1. The conceptual view of our adaptive SBS

A workflow is a routine to define how services are integrated and executed to complete a specific user task. The goal of a workflow is to satisfy both the functional and non-functional requirements of a user task. In this paper, we will focus on the latter, and more specifically on QoS requirements since substantial research has been done on workflow planning and web service composition to construct and execute a workflow to satisfy users' functional requirements. For a specific user task, there are usually a number of workflow patterns satisfying the goal. Workflow patterns address business requirements in an imperative workflow style expression, but are removed from specific workflow language [8].

Given the above definitions, the problem discussed in this paper can be stated as follows: Given the user requirements for an SBS, a set of services and their providers, and various workflow patterns that can satisfy the user requirements, how to design and synthesize an SBS that is adaptive, reliable and optimized for multiple QoS.

The above problem can be viewed as a control problem, i.e., how to design a control system and the corresponding controllers and observers to meet certain performance requirements. Observers monitor system status and the actual output of the control system, whereas controllers make decisions for system adaptations and take action to ensure that the actual output satisfies the reference input reflecting the functional and non-functional user requirements. Hence, the problem can be addressed from a software cybernetics perspective, where basic principles of control theory are applied to control the development process as well as the execution of software systems [1]. In this paper, we will discuss our idea of applying the principles of intelligent control theory to design the desired adaptive SBS. Figure 1 is the conceptual view of our adaptive SBS with the monitor and controller modules.

It is difficult to develop the centralized monitor and controller modules for an entire SBS in real software system due to the very large number of possible situations and the size of business process. An alternative approach is to use the divide and conquer strategy: divide the system hierarchically into several layers and design the specific monitors and controllers for each layer to satisfy a certain part of the requirements. When control engineers design complex control systems like the flight control system for airplanes, they usually adopt layered architecture where controllers for the rudders control system, flight course control system, and mission control system are separated from each other in a hierarchical manner. We adopt a similar strategy to design adaptive SBS.

To synthesize controller modules in an adaptive SBS, a Performance Index (PI) is defined for assessing the system performance. Note that the adaptive SBS needs to consider not only the QoS requirements, but also many other factors, such as resource constraints and security issues. The goal of the controller modules for an adaptive SBS is to optimize the overall PI rather than just QoS performance. In this paper, we define the PI as a function of service response time and security level.

3. Our Three-layer Architecture

The *Three-layer Intelligent Control Structure* [4][9] is a widely adopted system architecture for hierarchical intelligent control systems. It integrates control mechanisms in three different layers of the controlled system: organization, coordination and execution layers. Figure 2 depicts the structure of a typical three-layer intelligent control system. We will adopt the principles of three-layer intelligent control structure to design adaptive SBS. Figure 3 depicts the architecture of an adaptive SBS based on three-layer intelligent control structure, in which the three layers are mapped to the following three management/control processes:

User management

In the organization layer, the controller determines whether a user request should or should not be accepted based on the feedback, such as service and resource status. When a user request is accepted, it creates a task for the workflow management layer to handle the request. Some requests might be shredded to ensure the QoS performance of requests being

handled. It calculates the PI for all previous user requests, and identifies "PI not satisfied" events.

Workflow management

In this layer, the controller will handle the task allocated by the User management layer by selecting and executing one or more workflows to complete the task. It should adjust the priority of each request in the task queue based on the feedback of current QoS and the estimation of future environment variables, such as network condition and security level. A prioritization algorithm needs to be developed in this layer to select the best workflow pattern to provide the optimal QoS performance according to current resource condition. Here, we assume that *a priori* knowledge on multiple candidate workflow patterns that complete the same user task is available. Such knowledge can be either provided by domain experts or generated using workflow composition techniques [2].

Service management

In this layer the system selects the most suitable services based on the selected workflow pattern and the feedback of service status and environment variables, such as service provider status, network condition, and security level. For example, in an undesirable network condition where only very limited bandwidth is

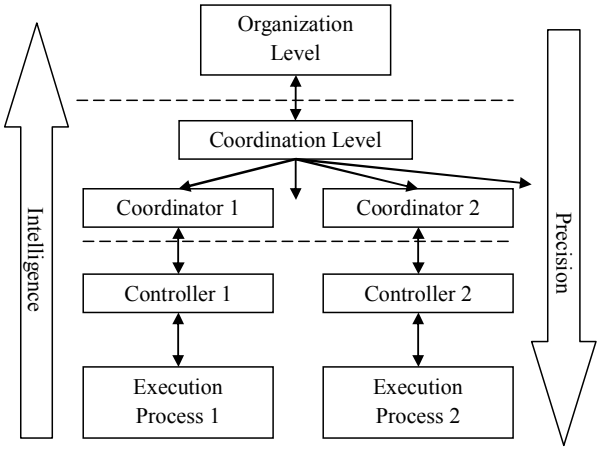

Figure 2. A hierarchical intelligent control system

available, it is preferable to use an encoding service to provide higher compression ratio to reduce the transmission time for the encrypted content. If a service provider disconnects frequently, it is preferable to use another one with more stable connection. In general, the controller in this layer finds and synthesizes the services to provide the optimal QoS performance under the current situation.

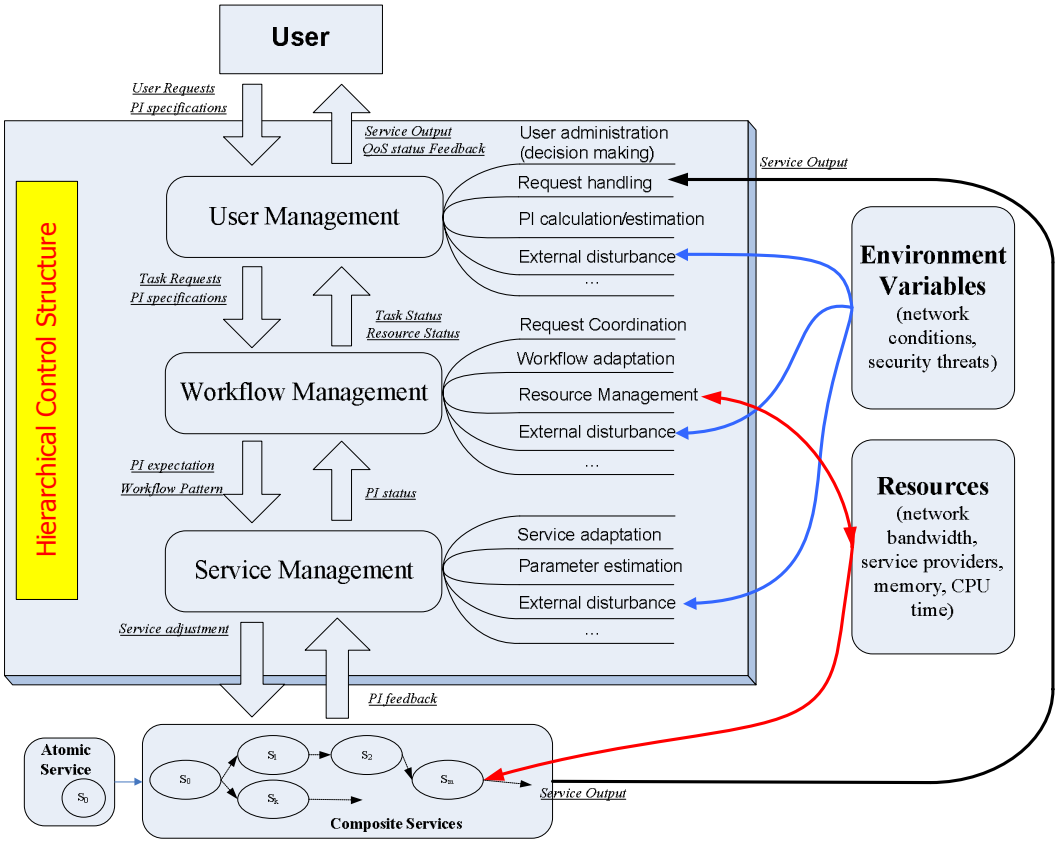

Figure 3. Our adaptive SBS based on the three-layer intelligent control structure

4. A Simple Example of Adaptive SBS

4.1. Basic Scenario

In order to demonstrate the feasibility of our adaptive SBS, a simple example of adaptive SBS is developed. A multi-media web server providing media content compression, encryption and downloading services is implemented based on our adaptive server system [3] to show the deployment of our example adaptive SBS.

There are two major QoS requirements in the multi-media web server: response time and security. The server is expected to provide media content with acceptable delay and a certain level of security. Environment factors, such as network congestion and security threats, will affect the QoS. The example adaptive SBS is expected to handle such incidents properly.

A sequential workflow pattern for a typical request handling process in the example adaptive SBS is depicted in Figure 4. When a user initiates a download request, the host extracts, compresses and encodes the request media content before sending it over the internet to the user who later uncompress' and decodes it.

4.2. Performance Index

In order to assess the performance of the example adaptive SBS, the PI should be defined as a function of the two QoS requirements.described in Section 4.1. We can devise performance indexes, such as a weighted combination of the two factors shown below:

$$p = w_1 \times k_{time} \times \frac{1}{T_{all}} + w_2 \times k_{\text{security}} \times S$$

where p denotes the PI, w_1 is the weighting factor for response time, T_{all} is the total response time for a single request handling process, w_2 is the weighting factor for security, S is the security level provided by a specific encryption service. k_{time} and $k_{security}$ are constants that regulate T_{all} and S to help measuring the PI. It follows from the above definition that higher PI reflects shorter response time and higher security level, which indicates better QoS.

4.3. Control System Structure

In this subsection, we will present the design of each layer of our three-layer architecture in the example adaptive SBS.

User Management

For this example, we have used the FCFS (First Come First Serve) policy for handling user requests. The overall PI will be calculated and updated online when a request is completed or resource status has changed. If the estimated PI does not satisfy user specifications, an event will be sent to the workflow management layer for workflow adjustment. Here a situation prediction control policy is devised and implemented as a supervisory controller to determine the proper actions for workflow adjustment. If the workflow management layer fails to complete the adjustment to achieve the expected PI, controller actions, such as temporarily reject new user requests, will be taken to ensure the QoS requirements will be satisfied as soon as possible. For example, when too many download requests arrive at the same time on the host side, it will reject any new requests until all the current tasks are completed with satisfactory PI. In a multiple host scenario, the user management control automatically migrates user sessions to less loaded hosts when needed.

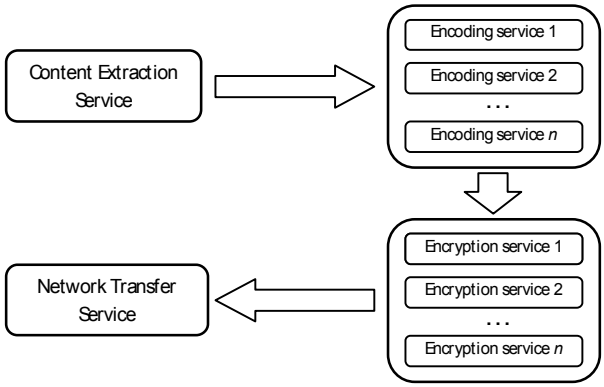

Figure 4. The sequence workflow pattern of our example adaptive SBS

Workflow Management

When the controller in the workflow management layer receives a task from the user management layer, the workflow management selects a ***workflow pattern*** from a set of workflow patterns [8] according to the expected PI and current resource status. If no workflow pattern can satisfy the expected PI, the workflow management layer will select the pattern that provides the best PI. According to the resource status and restriction, the controller will select the best pattern in the predefined workflow patterns to achieve the best PI. For example, when a user requests several different contents to be processed simultaneously, it might be better to select a parallel workflow pattern rather than a sequence pattern. Artificial intelligence techniques, such as machine learning and genetic algorithms, can be applied to the workflow pattern selection.

Services Management

The controller in the services management layer monitors the execution of the selected workflow pattern and its atomic services. When the status of service providers and environment variables changes, the controller initiates service adjustments to ensure that the PI expectation can be met. Adjusting control actions includes service provider selection and

service parameter adjustment. For example, when a compression service is temporarily unreachable, the controller needs to find a substitute with the second best PI expectation. Since parameter estimation plays an important role in this process, an adaptive controller with quantitative parameter estimation [7] module may be a good candidate here.

5. Runtime Data Analysis

In order to examine the feasibility and performance of our three-layer architecture, a series of experiments has been conducted to examine the performance and stability of the example adaptive SBS under various situations, such as facing an unstable network connection or continuous attack attempts at runtime. The example SBS is deployed on a distributed server group consisting of three servers: a host and two services providers. Two users from different PCs send service requests to the host at random time using different connections: 56K dial-up and ADSL. The requested content is also randomly selected from web pages, pictures and music files. There is also a simulated attack that initiates attack attempts to the host from time to time. We assume that all these attacks are blocked, but will raise the alert level of the host, which determines the minimum required security level for encryption services. There are three different encoding services on the encoding service provider and four encrypting services provided by the encryption service provider. All encryption services can use encrypting key length ranging from 1 to 255 to provide various security levels for communications. The response time and the security level over a period of time were recorded.

Figure 5 is a snapshot of the running system. The workload of each user is shown in Table 1. The workloads were arbitrarily generated to simulate user behavior. The current setup is enough to examine the basic functions of the adaptive SBS. More complicated scenarios will be examined in the future.

Table 1. Workloads of users

Workload	Number of download requests	Average file size (Kb)	Average expected security level (0-255)
User 1	16	115	94
User 2	18	67	95

Figure 6 and 7 depict the total response time and security level (encryption key length) of the two user workloads at runtime. Figure 6 shows that the response time of the adaptive SBS stabilized to a satisfactory level at runtime, and no download request took more than 5 seconds to complete. Figure 7 shows that the actual security level of each request was constantly higher than expected. When the attack

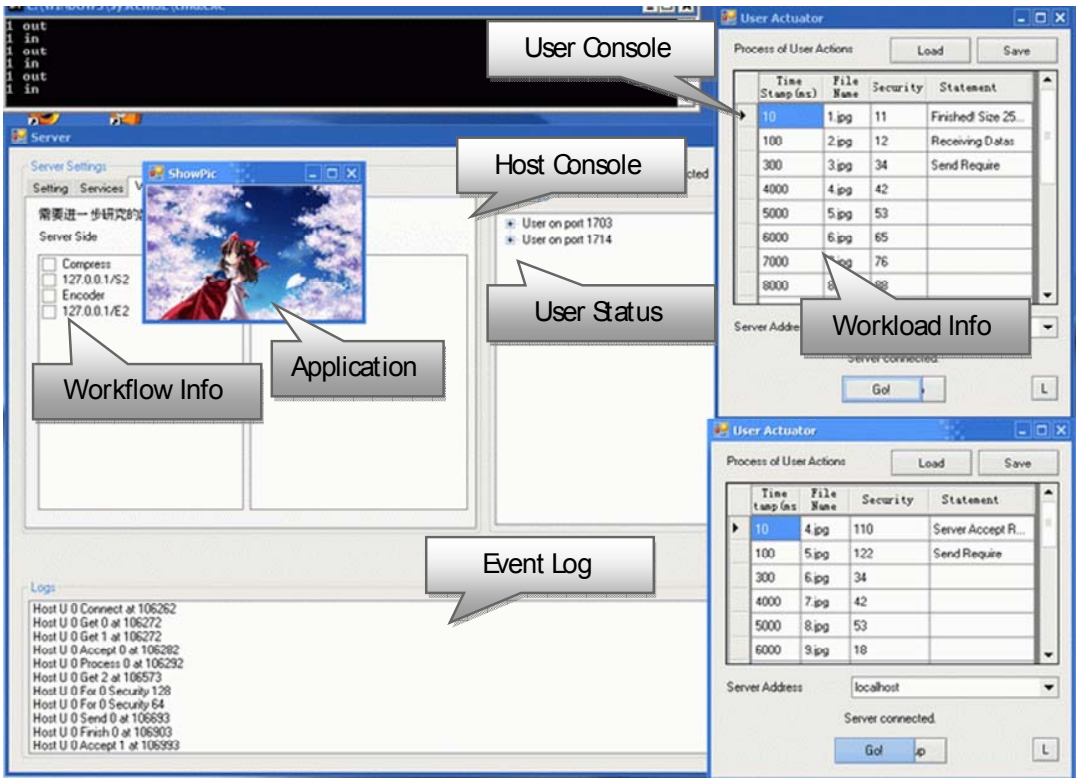

Figure 5. The execution environment and components of our adaptive SBS example

attempt was detected at the time when the 6th request is received, the adaptive SBS automatically adjusted the workflow and services to guarantee that the minimum security level is 128. The data shows that most of the requests were completed without violating the minimum security constraint.

6. Conclusions

How to design service based systems that are adaptable to dynamic user requirements, performance specifications and resource constraints is an important issue in distributed computing area. The major contribution of this paper lies in two parts: a three-layer intelligent control architecture for adaptive SBS is developed, and an example of the adaptive SBS which adopts the architecture has been simulated to show how the adaptive SBS can satisfy various user requirement and environment variations.

Future research will include: (1) examine and select better control algorithms and performance index, (2) explore the QoS characteristics of various workflow patterns, (3) automatically generate or synthesize the controller on each layer of the adaptive SBS, (4) balance the trade-off between efficiency and overhead of applying advanced control techniques, and (5) apply our approach on more complicated applications and services.

References

[1] K.Y. Cai, J.W. Cangussu, R.A. Decarlo, and A.P. Mathur, "An Overview of Software Cybernetics", *Proc. 11th Annual Int'l Workshop on Software Technology and Engineering Practice*, 2004, pp 77-86.

[2] M. Carman, L. Serafini, and P. Traverso, "Web Service Compositions as Planning," *Proc. Int'l Conf. on Automated Planning and Scheduling,* June 2003. Available at: http://sra.itc.it/people/carman/papers /icaps03_workshop.pdf.

[3] H. Hu, C.H. Jiang, K.Y. Cai, and W.E. Wong, "A Control-Theoretic Approach to QoS Adaptation in Data Stream Management Systems Design", *Proc. 4th IEEE Int'l Workshop on Software Cybernetics (IWSC)*, July 2007, 237-248.

[4] J.E. Lentz and L. Acar, "A Hierarchical Intelligent Controller for A Three Link Robot Arm", *Proc. 1995 IEEE Int'l Symp. on Intelligent Control*, August 1995, pp. 209-214.

[5] Y. Li, K. Sun, J. Qiu and Y. Chen, "Self-Reconfiguration of Service-Based Systems: A Case Study for Service Level Agreements and Resource Optimization", *Proc. IEEE Int'l Conf. on Web Services 2005 (ICWS)*, July 2005, pp. 266-273.

[6] C. Y. Lu, Y. Lu, T. F. Abdelzaher, J. A. Stankovic, S. H. Son, "Feedback Control Architecture and Design Methodology for Service Delay Guarantees in Web Servers," *IEEE Trans. on Parallel and Distributed Systems,* vol.17(9), Sept. 2006, pp. 1014-1027.

[7] L. Ljung, *System Identification: Theory for the User (Second Edition)*, Prentice Hall, 1998.

[8] N. Russell, A.H.M. ter Hofstede, W.M.P. van der Aalst, and N. Mulyar, "Workflow Control-Flow Patterns : A Revised View", *BPM Center Report BPM-06-22*, BPMcenter.org, 2006.

[9] T. Shibata, T. Fukuda, "Hierarchical Intelligent Control for Robotic Motion", *IEEE Trans. on Neural Networks*, vol. 5(5), 1994, pp. 823-832.

[10] J. Wang, H. Chen, Y. Xu and S. Liu, "An Architecture of Agent-based Intelligent Control Systems", *Proc. 3rd World Congress on Intelligent Control and Automation*, July 2000, pp. 404-407.

[11] S. S. Yau, D. Huang, L. Zhu and K.Y. Cai, "A Software Cybernetic Approach to Deploying and Scheduling Workflow Applications in Service-based Systems", *Proc. 11th Int'l* Workshop *on Future Trends of Distributed Computing Systems (FTDCS),* March, 2007, pp. 149-156.

[12] S .S. Yau, D. Huang and L. Zhu, "An Approach to Adaptive Distributed Execution Monitoring for Workflows in Service-based Systems", *Proc. 4th IEEE Int'l Workshop on Software Cybernetics (IWSC)*, July, 2007, pp. 211-216.

[13] S. S. Yau, H. Davulcu, S. Mukhopadhyay, D. Huang, Y. Yao and H. Gong, "Adaptable Situation-Aware Secure Service-Based (AS3) Systems", *Information Security Research,* C. Wang and S. King (eds.), 2007, Chapter 5, pp. 585-596.

[14] W3C Web Services Architecture. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.

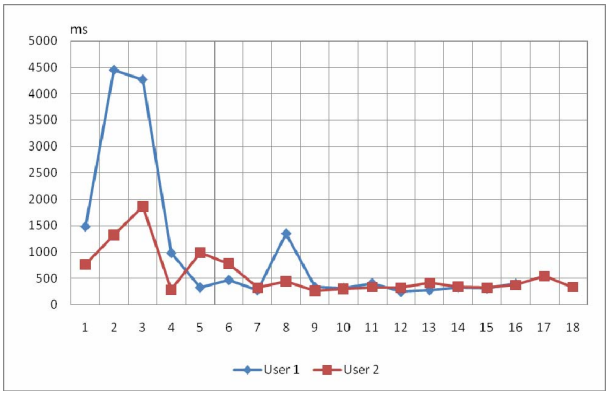

Figure 6. The response time performance of our adaptive SBS example

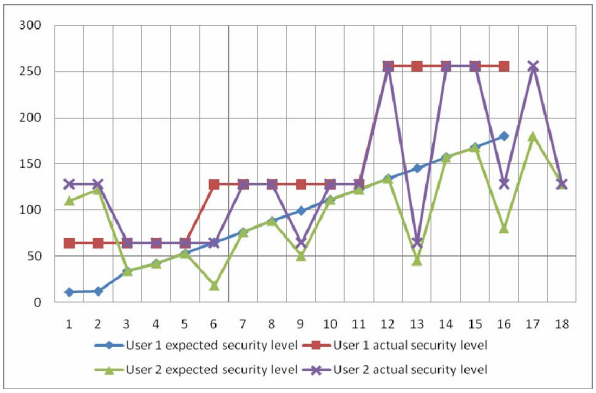

Figure 7. The security level performance of our adaptive SBS example