A Control-Based Approach to Balance Services Performance and Security for Adaptive Service Based Systems (ASBS)

Chang Liu, Changhai Jiang, Hai Hu, Kai-Yuan Cai[1]*
*Beijing University of Aeronautics and Astronautics,
Beijing 100191, China
kycai@buaa.edu.cn

Dazhi Huang, and Stephen S. Yau**
**Arizona State University,
Tempe, AZ 85287-8809, USA
yau@asu.edu

Abstract— *A major advantage of service-based computing technology is the ability to enable rapid formation of distributed computing systems by composing massively available services over various types of networks. User confidentiality and privacy should be well protected under various situations. This paper proposed a control-based approach to balance the trade-off between security and performance of Adaptive Service-Based Systems (ASBS). The relationship between encryption/decryption delay and the content size for cryptographic algorithms providing different security levels are measured and modeled. An example application based on the ASBS prototype is implemented to demonstrate the effectiveness of the proposed approach.*

Keywords-Service-based System, Security, Software Cybernetics

I. INTRODUCTION

Service-based systems (SBS) [2-4] are becoming more popular in large-scale distributed systems because their capabilities can be independently developed, shared and managed by various providers as services. However, this also imposes several challenges for SBS. The security techniques for these systems should be flexible, scalable and adaptable to the changing environments and user requirements, and provide security support for dynamic service discovery and execution.

Cryptographic algorithms are widely adopted to ensure the security of network transactions. For example, the secure socket layer (SSL) protocols are extensively used by online services providers and commercial web sites. The core encryption algorithm, namely the RSA public-key cryptography is used to protect transactions between the client and the server. However the RSA algorithm applies asymmetric encryption which costs intensive computational overhead when encrypting large files such as pictures, online video and other media files. Thus, in order to satisfy different security requirements in SBS, multiple cryptographic algorithms are used for different contents and situations.

Some research has been done on the tradeoff between strength of cryptographic algorithms (security) and the overhead (mainly affecting performance and resource consumption) [5]. However, existing research did not incorporate resource management into the solutions, i.e. no dynamic resource allocations are considered to improve performance. This is strongly required in practical applications though.

In this paper, we intend to address the above problem in a control-based approach within the framework of the three-layer intelligent control architecture for ASBS [6]. Figure 1 depicts the overall structure of the three-layer intelligent control framework of the ASBS.

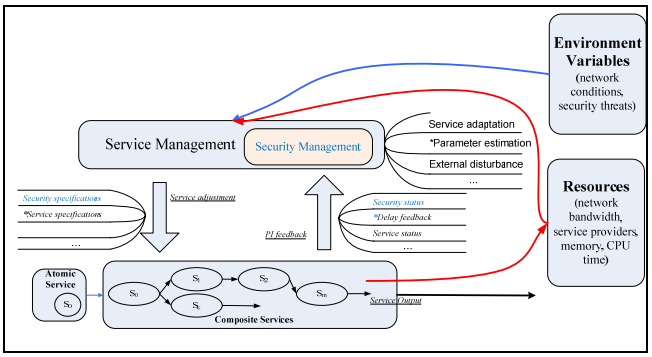

Figure 1. Three-layer intelligent control framework for ASBS

First, we need to analyze the tradeoff between security and performance under various resource constraints, and then develop controllers to adapt security configurations and resource allocations. Secondly, the relation between security level and computational overhead for various cryptographic algorithms (including different key-lengths) are investigated by conducting a controlled experiment. We also have to consider the overhead of the adaptation process. A simple example is the rekeying process when a controller decides to use a shorter or longer key. Depending on the operating environment (e.g. Signal-to-noise ratio in communication channels), the rekeying process can be quite costly. Hence, such overhead needs to be taken into consideration when designing controllers to handle the tradeoff between security and performance.

The rest of the article is organized as follows. In Section 2, following this introduction, we introduce the fundamental theories and techniques used for ASBS, and the security level among various cryptographic algorithms with different

[1] Cai is supported by the National Science Foundation of China and Microsoft Research Asia (Grant No. 60633010).

0730-3157/09 $25.00 © 2009 IEEE
DOI 10.1109/COMPSAC.2009.178

key length. In section 3, we propose a control-based approach to balance the trade-off between security and performance of ASBS. In section 4, we demonstrate our methodology, including the model of our problem, the optimization function and the rekeying process. Section 5 models the encryption/decryption process. Section 6 gives an example of the security-adaptive process for a service-based system, and then we conclude the article by section 7.

II. Related Studies

The topic studied in this paper is related to several research areas in computer science society. First it is related to studies on ASBS. Yau et al. proposed a software cybernetic approach to deploy and schedule workflow applications in SBS in [4]. Jiang et al. [6] proposed an intelligent control architecture for adaptive service-based software systems to design and develop SBS that are adaptable to constantly changing user requirements, environments, and resource constraints. In recent years, more effort has been attracted to the research on how to design and adapt systems to satisfy various QoS requirements. QoS-aware service composition [7-9] aims at finding an optimal or sub-optimal service composition satisfying various QoS constraints, such as cost and deadline, within a reasonable amount of time. Various techniques have been proposed for QoS-aware service composition, such as service routing and genetic algorithms [10, 11]. However, the QoS models considered in existing QoS-aware service composition methods are usually very simple. Furthermore, runtime adaptation of service composition cannot be efficiently handled by most QoS-aware service composition methods.

Compared to above systems, our proposed system addresses the security problem of SOA in the QoS category; design a Security Management Module （ SMM ） and detailed adaptive security policy for operation. We confine to balance the trade-off between security level and delay caused by the rekeying process and the increased/decreased delay of substituting new security policy, as well as the transaction. Moreover, we justify an approach of choosing a set of optimized parameters. According to these, it is feasible to run our system to seek an optimized solution to satisfy security QoS problems.

Some researchers address the QoS of delay and security issues in wireless networks. He et al. provide an integrated solution to end-to-end delay and security, where the middleware adaptation provides tunable delay and security support according to network condition in [5]. In security adaption, SBS need a framework to integrate the delay and security support for services. Ong et al. presents a framework that provides differential security levels for different devices, users and application security requirements in [12]. Lenstra et al. presents a framework that provides differential security levels for different key length in varieties of cryptosystems in [13].

III. Overall Approach

The problem to be addressed in this paper is that how to design SBS whose security policy is flexible, scalable and adaptable to the changing environments and user requirements, and provide security support for dynamic service discovery and execution. According to the three-layer intelligent control framework for ASBS, this problem can be viewed as a control problem in the Service-Management Layer. In order to provide scalable and adaptable security to the transactions between various services provided over various types of networks, a SMM is required to monitor and control the security policy for each transaction in the ASBS. In this paper, security policy is confined to the combination of different encryption algorithms and key lengths. Figure 2 gives a pictorial view of the security management module. The SMM is a component in the SML, which handles all security-related aspects in the information transmission process within the Composite Services. It is designed to automatically model and estimate security aspects related parameters in the Composite Services using the feedbacks from atomic services on their service and security status. A controller is developed to determine the optimal security policy for each transaction. The actual output is compared with user requirements to decide whether adaptation actions such as increasing the resource or change the security policy are needed. The processes involved in changing the security policy are generally named as the rekeying process. A possible cost in terms of extra delay is incurred when rekeying occurs. More details on the rekeying process can be found in Section 4.3.

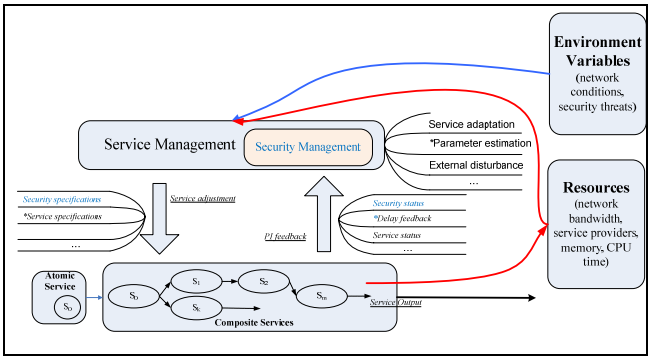

Figure 2. The SMM within the ASBS

More specifically, the adaptation of security policies involves the following steps:

Step 1: Model the trade-off problem between security and delay in ASBS. Define an appropriate Performance Index (PI) that balances the trade-off of service security and delay.

Step 2: Model the relationship between security level, content size and encryption/decryption delay for a set of cryptographic algorithms with different key lengths. More specifically, find the function/algorithm f that gives:

$$Delay = f(keylength, FileSize, Algorithm)$$

Step 3: Design a controller that selects the optimal combination of cryptographic algorithm and key length that optimizes the overall PI.
Step 4: When a new transaction needs to be established, use the controller to determine the security policy to be adopted.
Step 5: Compare the actual PI outputs and the requirements specified by the user to decide whether an adaptation action is required. If not, complete the transaction with the current security policy.
Step 6: If the current security policy can't meet the user requirements, take adaptation actions such as increasing the computational resource and try to satisfy user requirements.
Step 7: If the user requirements are satisfied complete the transaction, if not go to *Step 4* to select another security policy according to the latest user requirements and resource constraints. When the new security policy is determined, first evaluate the possible benefit of applying the new policy and compare it to the possible delay caused by the rekeying process to see if the rekeying process is worthwhile. If so, start the rekeying process to change security policy for the unsatisfied transactions. If not, go to *Step 4* to select an alternative security setup and conduct the comparison again. If the result is still negative, remain the current security setup unchanged and go to *Step 8*.
Step 8: After rekeying, if the user requirements are satisfied, completed the transaction. Otherwise relax QoS requirements from the user and complete the transaction.

The following section gives more details on how each step is conducted.

IV. METHODOLOGY

4.1 Modeling the problem.

In this section, we model the trade-off problem between security and delay by designing a PI to incorporate them into a single metric which can be later monitored, estimated and controlled.

According to the intelligent control architecture proposed in our earlier works [6], the workflow is composed of atomic services and a workflow pattern indicating how they compose an application. The selection of atomic services and the workflow pattern is determined by the Workflow Management Layer which is not concerned in this study, so we assume that the services and workflow remains the same throughout the runtime. For each service in the workflow, there are three security related specifications to be considered:

1. S_{in_min}, the minimal input security level, that is, what is the minimal acceptable encryption level of the input contents to the service provider.
2. S_{out_min} , the minimal output security level, which is the minimal acceptable encryption level of the output of the atomic service.
3. S_{app_min} , the minimal security requirement of the application, which is defined by the application and may change according to the security situation.

For each transaction between two atomic services, the minimal required security level can be then denoted by S_{min}, which is defined as:

$$S_{min} = Max(S_{in\_min\_SourceService}, S_{out\_min\_DestinationService}, S_{app\_min})$$

The overall delay of the transaction, denoted as T_{all}, is composed of three parts: the encryption delay, the transmission delay and the decryption delay, i.e.,

$$T_{all}=T_{enc}+T_{trans}+T_{dec}$$

The goal of adaptation here for each transaction includes the following requirements:

1. The deadline requirement from the user perspective.
2. The security requirement from the user perspective, which defines the minimal acceptable security level to the user.
3. When security situation changes, such as a new password hacking technique has been identified on the network, the security policy is still strong enough to protect the transactions against the new threat without stopping and rekeying all the transactions. More specifically, the security policy needs to have certain margin in security level to avoid the rekeying process which incurs significantly more delay.
4. When network transfer bandwidth drops, there is still a margin to complete the transactions in time.

Considering the above requirements, we define the PI for the SMM as follows:

$$PI = K\frac{S_{\max}-S_{\min}}{S-S_{\min}}+\frac{T_{\max}-T_{\min}}{T_{\max}-T}$$

Whereas S_{max} denotes the maximum security level that can be achieved when $T = T_{max}$, that is, the maximum security level that can be achieved within the maximum allowed delay from the user perspective; S_{min} is the minimal required security level for the transaction determined by the service specifications and user requirements; S denotes the security level to be used for the transaction which needs to be determined; T_{max} is the maximum delay allowed by the user for the transaction to be completed; T_{min} is the minimum delay that can be achieved using the minimum security setups, i.e., the delay achieved when $S = S_{min}$. T denotes the delay corresponding to S which can be estimated when S is determined. K is a weight factor that can be tuned to emphasize security or delay, if there is no preference, $K = 1$.

The problem is then transformed into an optimization problem that tries to find the optimal security setup S which gives the minimum PI for the transaction.

4.2 The Optimization Function.

In order to satisfy both the minimum security and maximum delay requirements, we have $S_{min}<S<S_{max}$ and $T_{min}<T<T_{max}$. Moreover, when the security level is determined and the size of the content is known, the total delay T can be estimated using the following formula:

$$T = T_{all} = T_{Enc}(S, Size) + T_{Dec}(S, Size) + T_{trans}(Size, Band)$$

The encryption delay and decryption delay are functions of the size of the content and the security setup S. More

specifically, when the encryption algorithm and the size of the content to be encrypted are determined, the encryption time consumption can be estimated by a function of them. Similar approach applies to the estimation of the decryption delay. The transmission delay is determined by the size of the content and the network bandwidth which is straight forward. To sum up, T is determined by a function $T = f(S, Size)$.

Assume that function f is continuously differentiable, then we can obtain the optimal security setup S* and the corresponding delay T* that gives the minimum PI when the derivatives of PI equals to zero. More specifically, we have

$$\begin{cases} \dfrac{\partial PI}{\partial S} = 0 = -K\dfrac{S_{\max}-S_{\min}}{(S^*-S_{\min})^2} + \dfrac{(T_{\max}-T_{\min})(-1)^2}{(T_{\max}-T^*)^2}\dfrac{\partial T}{\partial S} \\ \dfrac{\partial PI}{\partial Size} = 0 = 0 + \dfrac{(T_{\max}-T_{\min})(-1)^2}{(T_{\max}-T^*)^2}\dfrac{\partial T}{\partial Size} \end{cases}$$

$$\rightarrow \begin{cases} \dfrac{\partial T}{\partial S} = K\dfrac{S_{\max}-S_{\min}}{T_{\max}-T_{\min}}\left(\dfrac{T_{\max}-T^*}{S^*-S_{\min}}\right)^2 \\ \dfrac{\partial T}{\partial Size} = 0 \end{cases}$$

Note that $T^*=f(S^*, Size)$, the equation to obtain S* is

$$\frac{\partial f(S, Size)}{\partial S} = K\frac{S_{\max}-S_{\min}}{T_{\max}-T_{\min}}\left(\frac{T_{\max}-f(S^*, Size)}{S^*-S_{\min}}\right)^2 \quad (1)$$

By now we have developed an algorithm to select a security setup for each transaction to optimize the overall PI that balances security and delay. However in order to calculate S*, the function f needs to be identified for each security setup.

4.3 The Rekeying Process.

Although the SMM is designed to select security setups that are adaptable to the change of user requirements and security situation, there are still chances that the current security setup fails to meet the minimum security requirement or meet the changed deadline of transaction. In this situation, the security management module will need to update its optimization function with the latest requirements (S_{min} and T_{max}) and obtain a new security setup to optimize the overall performance index in the new scenario. Then a rekeying process is required for all on-going and pending transactions. For each transaction, when the rekeying occurs, it may be in one of the following four stages:

1. The encryption process is on-going.
2. The encryption process is finished and the transmission process is on-going.
3. The encryption process has not yet started.
4. The encryption and transmission are performed in parallel in applications with streaming data(such as audio/video streams).

For case 1, the encryption process is terminated immediately and the transmission process is blocked. Then we restart the encryption process with the new security setups and resume the transmission process. For case 2, the transmission process is terminated immediately and encrypted content is discarded. Then restart the encryption process with the new security setups and the transmission process. For case 3, the encryption process and transmission process is blocked immediately, wait until the new security setup is ready. Then resume the encryption and transmission process. For case 4, both encryption process and transmission process are blocked, waiting for the new security setup. Then resume the encryption and transmission process. Figure 3 gives an example of the rekeying process in a workflow.

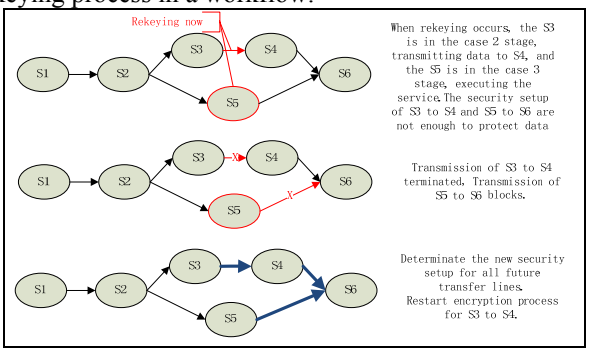

Figure 3. An example of the rekeying process

Due to the fact that the rekeying process involves blocking and restarting the encryption and transmission process, it is considered to cost significant delay of the whole workflow. Thus it is vital to ensure that the workflow have ample margin in both security level and delay to avoid security and deadline violations caused by changes in requirements and situation. The performance index defined in the above subsection aims to leave abundant margin for both requirements thus the hazard of possible rekeying is minimized.

The delay caused by the rekeying process also needs to be considered in the adaptation process when deciding whether a rekey is needed. A typical scenario is when the security requirement is lowered from S2 to S1, and the optimization function shows that by changing the security setup from higher security level to lower level the encryption/decryption delay can be reduced by Td. On the other hand, the security management module estimates that the delay caused by the rekeying process is Tr. If Td < Tr, then the overall delay will increase after taking the rekeying action to adjust the security setting. So for the Step 7 of the overall approach in Section 3, before deciding whether a rekeying process is needed, the estimated delay of the rekeying process (based on the runtime situation) is compared with the possible benefit of taking the adaptation action to see whether it worth the effort. Moreover, if the result is negative, then the optimization function needs to generate another security policy that can generate more possible benefit and make the rekeying process worthwhile. If it fails to do so, then the system better stay unchanged and go to the next step.

In a particular situation, for example, where the request after the next request is for the same security level but with

a much larger content, let's denote its processing time saved by the rekeying is Td'. It is possible that Td<Tr but Td'>Tr, however, we only regards to the parameters and requirements of the next request. More specifically, the Security Management Module only compares the Td and Tr to decide whether a rekeying process is needed. Then, in the next operation, we consider the larger file, first consider for the moment if the minimum security requirement and changed deadline of transaction are satisfied. If not, the delay caused by the new rekeying process is Tr', justify whether Td'>Tr' as before. If so, the SMM will take the rekeying process, since the rekeying process does save resource.

V. Modeling the Overhead

In order to determine the relationship between encryption/decryption delay and the content size for different combinations of cryptographic algorithms and key lengths, a series of controlled experiments are conducted to collect the runtime delay for encrypting/decrypting contents sizing from 2^3bits to 2^{21}bits using a total of 13 combinations of four cryptographic algorithms with various key lengths. According to the analysis on cryptographic algorithms in [13], the level of security of the 13 combinations are categorized into seven levels, there are about two combinations providing similar security at each level. In such series of experiments, we focus on the transmission of encrypted files with different cryptosystems, so we assumed that the key has been known in the symmetric algorithm, such as DES, triple DES, and AES. Moreover, a pair of private key and public key in RSA has been computed in advance, as well as the public key has already been published. The delay cost by key exchanging in symmetric algorithm, as well as the computation of a pair of private key and public key are considered in the preparation time of the cryptosystems. Virtually, such preparation time is trivial compared to the delay cost by the transmission time which is justified by our experiment data. These experiments were run using C# and .Net on a computer with an Intel Xeon 3.06 GHz CPU and 2.0 GB RAM, running Microsoft Visual Studio 2008. Table 1 tabulates the combinations examined in this experiment.

TABLE 1. SECURITY LEVEL OF DIFFERENT COMBINATIONS

Security Level	DES	RSA	SHA-2	AES
1	56bits			
2		816bits	128bits	
3	112bits	1776bits		
4	168bits	2432bits		
5		3072bits		128bits
6		7689bits		192bits
7		15360bits		256bits

We sampled 21 different scales of the content: from 2bits to 2^{21}bits and use linear interpolation to obtain the delay-size charts. Figure 4 gives a pictorial view of the relationship between delay (nano-seconds) and content size for the combinations at the fifth security level.

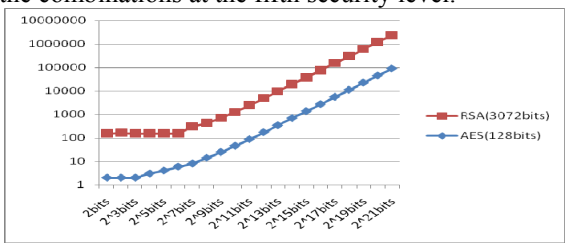

Figure 4. The delay (ns) of security level 5

Apparently the AES algorithm with key length of 128bits is superior to the RSA with 3072bits keys in terms of the computational overhead. Since they provide similar level of security, the AES algorithm should always be selected when the security level is set to level 5.

VI. A Running Example

In order to demonstrate the feasibility of the proposed approach, an example application based on the ASBS prototype is implemented. The application adopts a sequential workflow pattern and takes user data as input, compresses it with a compression service and decompresses it with a decompression service.

There are two transactions involved in this application, whose security setups are determined by the SMM. All the cryptographic combinations mentioned in Section 5 are implemented in the SMM and the delay functions are pre-calculated. 20 user requests with different delay and security requirements are simulated and the actual delay and security level outputs are recorded. A situation change that raises the minimal security level (from 128bit to 256bit) is also simulated in the experiment (at the fourth request) to test whether the security policy is robust enough to avoid rekeying. Figure 5 and 6 give the actual delay and security output of the example application over 20 requests. Data shows that the proposed approach can really balance security and delay to avoid violations of user requirements. Moreover, it's robust against changes of security situation. Apparently the actual security level of *User 1* is above the minimum require security level except for two points at the 13th and 18th requests, this because the SMM cannot always follow up the change of user requirement.

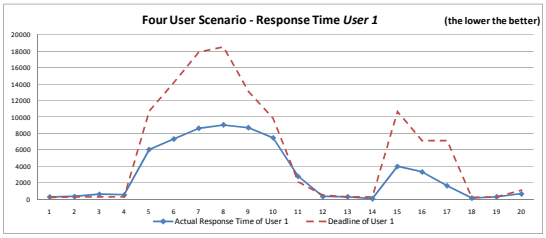

Figure 5. The actual response time of *User 1* in the four user scenario

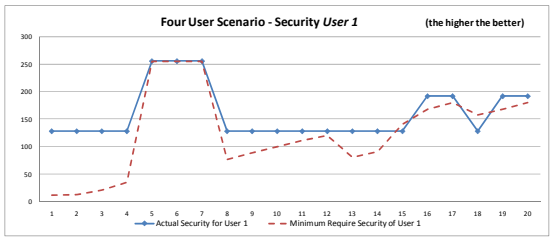

Figure 6. The actual security level of *User 1* in the four user scenario

VII. CONCLUSIONS

This paper proposed a control-based approach to the security adaptation problem in adaptive service-based systems. A security management module is introduced into the original three-layer intelligent control architecture to address the trade-off between security levels and delay of transactions within the workflow. A performance index that incorporates security requirements and delay deadlines is proposed to transform the problem into an optimization problem. By solving the optimization equation, a controller that selects the optimal security policy for each transaction is designed. Moreover, the relationship between encryption/decryption delay and the content size for different combinations of cryptographic algorithms and key lengths is measured and modeled through a series of controlled experiments. An example application using the proposed security technique is implemented to demonstrate the feasibility of the approach and experimental data shows that the system provides desirable balance between security and delay requirements. Moreover, the security policy is robust against changes in the security situation of the system.

REFERENCES

[1] Kai-Yuan Cai, "Optimal software testing and adaptive software testing in the context of software cybernetics", *Information & Software Technology*, 2002, 44(14), pp. 841-855.

[2] 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 International Workshop on Software Cybernetics (IWSC)* , July 2007, pp. 237-248.

[3] M.W. Jang and G. Agha, "Dynamic Agent Allocation for Large-Scale Multi-Agent Applications", *Proc. International Workshop on Massively Multi-Agent Systems*, December 2004, pp. 19-33.

[4] 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* International *Workshop on Future Trends of Distributed Computing Systems (FTDCS)* , March, 2007, pp. 149-156.

[5] Wenbo He, Klara Nahrstedt, "An integrated solution to delay and security support in wireless networks", *Wireless Communications and Nerworking Conference(WCNC)* , vol. 4, April 2006, pp. 2211-2215.

[6] Chang-Hai Jiang, Hai Hu, Kai-Yuan Cai, Dazhi Huang, and Stephen S. Yau, "An Intelligent Control Architecture for Adaptive Service-based Software Systems with Workflow Patterns", *32nd Annual IEEE International Computer Software and Applications Conference*, 2008, pp. 824-829.

[7] E. Sirin, J.A. Hendler, and B. Parsia, "Semi-automatic Composition of Web Services Using Semantic Descriptions", *Proc. the Web Services: Modeling, Architecture and Infrastructure (WSMAI) Workshop in conjunction with the 5th International Conference on Enterprise Information Systems (ICEIS 2003)*, April, 2003, pp. 17-24.

[8] S.J. Woodman, D.J. Palmer, S.K. Shrivastava, and S.M. Wheater, "Notations for the Specification and Verification of Composite Web Services", *Proc. the 8th IEEE International Enterprise Distributed Object Computing Conference (EDOC'04)*, September 2004.

[9] J. Jin, and K. Nahrstedt, "On Exploring Performance Optimization in Web Service Composition", *Proc. ACM/IFIP/USENIX International Middleware Conference,* October 2004, pp. 115-134.

[10] G. Canfora, M. Di Penta, R. Esposito, and M.L. Villani. An Approach for QoS-Aware Service Composition based on Genetic Algorithms. *Proc. 2005 Conference on Genetic and Evolutionary Computation*, June 2005, pp. 1069-1075.

[11] C. Guo, M. Cai, and H. Chen, "QoS-Aware Service Composition based on Tree-Coded Genetic Algorithm", *Proc. 31st Annual International Computer Software and Applications Conference (COMPSAC 2007)*, July 2007, pp. 361-367.

[12] C. S. Ong, K. Nahrstedt, and W. Yuan, "Quality of protection for mobile multimedia applications", *IEEE International Conference on Multimedia and Expo(ICME2003)*, Baltimore, MD, July 2004.

[13] Arjen K. Lenstra, Eric R. Verheul, "Selecting cryptographic key sizes", *Journal of Cryptology*, vol. 8, , 2001, pp. 255-293.