Security Issues of Service-Oriented Middleware

Jameela Al-Jaroodi, Alyaziyah Al-Dhaheri
Faculty of Information Technology, UAEU, Al Ain, UAE

Abstract
Security is a major challenge for all distributed applications; however, it imposes additional concerns and requirements when Service-Oriented Computing (SOC) is used. Therefore, addressing the security issues is critical for the success of any SOC based project. Thus when utilizing service-oriented middleware (SOM), security could be included as part of the SOM architecture. SOM is a concept that has been widely adopted in the past few years, and has been regarded as a very feasible approach to tackle the security issues in the SOC paradigm. In this paper, we cover different SOC-based projects that have adopted SOM as an integral part of the security enforcement in their projects. We then move on to identify common security requirements especially inherent to SOC and which have been implemented in the set of projects we have covered. Finally, a general security model for SOM is proposed consisting of a set of decoupled services that compose the required SOM security functions.

Keywords:
Middleware, security, service-oriented computing, service-oriented middleware.

I. INTRODUCTION

Service-oriented computing (SOC) is the computing paradigm that utilizes services as the fundamental elements for developing applications. The service-oriented architecture (SOA) is the core principle in building the service model for SOC, where SOA presents an approach for building distributed systems that deliver application functionality as services to either end-user applications or other services [1]. This is made possible as SOC provides patterns for the design, development, deployment and management of a loosely coupled business application infrastructure. In this framework, business functionality is published, discovered, and consumed as part of a business ecosystem of network-aware and reusable technical and business services [2]. Developing service-oriented applications/services is not dependent on a certain technology, but is mostly implemented using SOAP, CORBA, RPC and Web Services. SOC basically involves three main players: service provider, service broker and service requestor. The service provider designs and develops a service. The service broker makes this service available to the rest of the world through public registries such as Universal Description Discovery and Integration (UDDI) for web services. The service requestor locates the entries in the public registry using some discovery method and then binds with the service provider to invoke the required services. There are many people (researchers and practitioners) with many different levels of understandings and uses of SOC [3]. Hence, SOC is not a technology but a mean to organize the information technology (IT) infrastructure and business functionality on which it will reflect the components tendency toward autonomy and heterogeneity. SOC is gaining wide popularity as business processes are more likely to be involved with other participants, leaving SOC to oversee the integration needs when multiple applications running on varied technologies have to communicate. Moreover, it has been seen as an evolutionary step in the software architecture world since it promotes software reuse, eases integration, and reduces development time therefore it could significantly reduce the cost of a project. Not to mention the future aspect as with its loosely coupled nature, it easily allows to plug-in new services and upgrades.

Middleware is a software layer intended to support distributed applications by masking the complexities related to the heterogeneity and distribution of the execution and network environments. It also spares the developers the need to worry about the non-functional requirements for the application which includes interoperability, reusability, dynamic reconfiguration, scalability, quality of service (QoS) and security. This shows that the middleware layer has similar functionalities as the SOC approach, where both hide the heterogeneity of the underlying environments and offer integration capabilities and on demand software reuse. As a result, service-oriented middleware (SOM) has been introduced where middleware has been sought to help with the design and development of services within the SOC. In SOC-based environments, the SOM is in charge of enabling the deployment of services and the coordination among the elements of the SOC, which are the service registry, service provider and service consumer. That is, SOM provides runtime support for service providers to deploy services on service hosts and further advertise their presence to the registry, and for service consumers to look up, discover and use the advertised services. SOM hides the heterogeneity of the underlying environment, by introducing languages for rigorous service description and protocols for service discovery and access [4]. According to [3], a general SOM should provide the following functionalities:

- Runtime support for service providers to deploy and advertise services.
- Support for service consumers to discover and use registered services.

Manuscript received January 5, 2011
Manuscript revised January 20, 2011

- Abstractions to hide the heterogeneity of the underlying environments by introducing supportive languages and protocols.
- Service transparency to client applications.
- Interoperability with a variety of devices and systems.
- Efficient handling of large volumes of data and high communication loads.
- Integrated support for security.

The remainder of the paper is structured as follows. Section II provides coverage on the security issues in SOC environments and SOM, as well as the benefits that SOM brings to the SOC-based projects. In Section III, we cover the latest approaches researchers proposed to achieve better security through SOM. In Section IV, we provide short summaries of SOC-based projects that have used SOM to introduce or enhance security features. In Section V, we identify the main security requirements and their significance to the approaches discussed in Sections III and IV and to SOM in general. In Section VI we introduce a framework to provide a security-supporting SOM based on a set of decoupled security services. Finally, Section VII concludes the paper.

II. BACKGROUND

With the wide adoption of SOC, many security issues have risen due to its openness and loosely coupled nature. Security is an important issue especially when dealing with transactions that require the exchange of information between distributed systems and applications. This becomes a major concern as the data traveling could be at risk of interception or even modification. Therefore, the security goal is to set security policies, enforce them, detect violations, prevent/minimize attacks and try to recover if attacks succeed. The main security issues that applications nowadays face are related to data encryption, data access and policy management. According to [5] some of the common vulnerabilities of web services which are the core of SOC include insecure communication, information leakage, insecure configuration and insufficient authentication. Due to the absence of standardized security guidelines for application development, customers and vendors have tried to come up with solutions which succeed when working individually but fail miserably in the case of integration. Therefore, the move towards adopting SOM can improve the security vastly as the middleware, independent of the target applications, will support many security features, yet these features are highly independent. From these features we have identity management where the identities of the accessing parties are verified, message protection from interception, access control and audit trails. Identity management and authentication involve the management of user IDs and passwords to identify and authenticate users as well as confirm their authorizations. This requires full coordination between the numerous user directories across the enterprise services. Next is the access control which is granted to parties after validating their request using the Identity management, this ensures that the parties are given access to the appropriate resources with the proper authorization. Audit trails are concerned with monitoring and recording the steps performed in operations and, in case of problems, identifying the causes. This is made possible with the use of audit logs which are analyzed to ensure system integrity through verification and to take necessary actions if violations were detected. Message protection is another important issue that needs to be addressed in distributed systems. Messages traverse multiple domains; therefore, they need to be protected beyond the span of control of the service and systems which happens using digital signatures and cryptographic keys [6]. It is also crucial to have a centralized administration that oversees the whole security aspects; this will ensure that the security features are well coordinated together. However, in a largely distributed environment centralization may become an additional problem to solve.

III. SOM SECURITY APPROACHES

Security became an important issue because applications are no longer contained within closed, tightly controlled environments and currently most transactions and operations occur online and require data transmission. The applications and data involved need to be protected from malicious and unintentional attacks as well as from any possible risks of exposure. Well defined access polices, encryption mechanisms and authentication models can help in providing security. In this section we will cover the latest approaches proposed to achieve better security through SOM. Our coverage will include analysis of the proposed security mechanisms and the technologies used. The list of projects discussed is representative of the latest research published in this field, were we have chosen ones that have explicitly described the security mechanisms they included.

A. Requirements for pervasive SOM

In pervasive environments middleware is used to promote interoperability but some argue that this is only true if all the devices use the same middleware [7]. Thus, the use of SOC is suggested to minimize the dependency on a single middleware system and reach the desired level of interoperability. The authors also offer an insight on the requirements and challenges needed to build a service oriented middleware for pervasive and ubiquitous environments. Security is classified as one of its primary requirements which is divided into three dimensions; the first is the Protection against third-party code which is further divided into Malware Protection and Service Level Agreement (SLA). The former guarantees protection against device contamination, while the latter ensures that the mutual-agreed contracts between the trusting entities are met. The second dimension defines Reputation, which is calculated based on previous interactions, and devices with higher reputation are considered more reliable. The third dimension is Privacy, which is maintained through Authentication, Authorization and Non-Repudiation. As for the security challenges highlighted, the first is related to the

message exchange that occurs between the distributed processes making the messages exchanged prone to unauthorized access. Therefore, the use of robust content authorization and validation mechanisms along with the message exchange is required. Another security challenge is mobility support, where autonomous negotiations and verifications of SLAs could be implemented to secure code mobility. As for device mobility, autonomous and adaptable mechanisms are used to identify security violations. Finally the third challenge is service coordination, where malicious neighbors could vandalize the cooperation information for the applications.

B. Security Requirements for Semantic SOC

The requirements needed to build a security infrastructure in a semantic service-oriented architecture are covered in [8]. The authors base their assumptions on the European research project Access-eGov. This project aims to facilitate the registration of eGovernment services and the discovery of web services for clients across Europe. Therefore, the paper emphasizes four major security concerns which are communication, trust, privacy and access control. Communication security in Access-eGov covers both the physical and application level security. The former is guaranteed through the use of encryption and digital signatures on the XML-messages, while the latter is secured with the use of XML-firewalls to check each XML-encoded message for verification. Trust is established with the use of authentication between the communicating sides, whether it was for inquiring about and requesting services or when new nodes are added to the system. Privacy is required to protect the data in both the client and service provider. In addition, the ability of privacy-tuning makes the discovery and execution of web services more efficient. Access-control is achieved through the employment of attribute-based access control as defined by XACML. The security mechanisms are to be distributed between the different Access-eGov components which are the middleware platform, the personal assistant and the back-office. The middleware will handle the access control to the personal assistant and will contain the service security semantics. The personal assistant will handle the client side tuning of the privacy settings as well as the communication encryption. As for the back-office security it will impose the XML-firewalls and the traffic control proxy servers.

C. Pattern-Based design of SOM for remote object federation

In [11] a software pattern design is proposed for an SOM based on a federated model of remote objects. A federation is a group of peers offering different services to other peers. The security side of this system is handled by the management of the federations where the federation controls all of its peers and those who can access them. Only certified objects can join a federation and only exported methods can be invoked. Moreover, peers can only be invoked by their federation or by peers from their federation. Peers are protected from interfering with the execution of other peers by confining their execution to their own threads which has its own interpreter as a thread-specific storage. Invocation interceptors are also used to further reinforce the security aspect, where the client-side invocation inceptor adds security information to the invocation before it sends it to the server.

D. Single Sign-On Integration in a Distributed Enterprise Service Bus

A solution to the limitation of the current federated identity management approaches, as they only operate under static and predefined conditions is offered in [12]. The paper introduces dynamic federated identity management and advanced authorization mechanism that is built using the highly distributed SOM, PEtALS Enterprise Service Bus (ESB). The security architecture is divided into two layers the semantic layer and the service layer. The semantic layer oversees the Quality of Protection (QoP) agreements by matching the user QoP requirements to the authorization policy expressed by the service provider; it also separates the authentication process from the authorization process. The service layer handles the security mediation which includes the federated identity manager, message security and secure transport. Furthermore, this layer is also responsible for securing the ESB through authentication which is achieved using a single sign on mechanism that propagates identity along service chains, as well as the protection of the information exchanged through the bus using SSL encryption. Access control is achieved with the aid of three components, first the user registry which stores end-users identities, second the service level containing authentication functionalities requirements and identity federation mechanisms, thirdly the semantic level which matches between QoP requirements and the related authorization policies.

E. A Multi-Layer Security (MLS) Enabled QoS Management Architecture

The authors in [15] present QoS-MLS architecture which is aimed to solve the problem of QoS and security in SOA environments. QoS-MLS expands the QoS management scope to include quality of security service at the middleware layer in order to achieve the intended MLS.Therefore, security QoS parameters will be added at the user and service provider level. At the user level, the security attribute will be in the form of a classification (red, yellow, orange, green) that represents the required quality and level of security. As for the service provider level, the security parameters will include information concerning authentication, authorization, encryption, certification, logging/auditing, intrusion detection, transport/network security. The Establishment Service and the Security Configuration Service are the two main components that the architecture relies on to achieve MLS. The Establishment Service is responsible for admitting/rejecting a quality of security service contract (QoS-MLS contract) which the user/application needs to access the service requested. It relies on the policy manger to verify if the user has the

authorization to access the requested classification, then retrieves the security resource requirements for the host and compares it to the actions allowed on each object/service using the user's security classification. If this succeeds, then the security resource requirements are passed to the Security Configuration Service to establish a quality of security service contract. The Security Configuration Service consists of configuration validation and setup services for three components which are the host, network infrastructure and security infrastructure. If the security resources configuration requirements of the three components are satisfied, then the main Security Configuration Service notifies the Establishment Service that the quality was achieved. As a result, a signal is sent to the Establishment Service to go on with the contract and begin serving the users requests.

IV. SOM SECURITY EXAMPLES

In this section we cover a few examples of SOM with security functionalities included in their architecture. This will include the problem description, the proposed solution, and technologies used. This will provide an insight on how SOM is used to enhance security in diverse SOC-based projects and allow us to identify the important aspects currently covered and those yet to be covered.

A. Cumulus

Cumulus [9] is a service-oriented middleware for web services, designed to support runtime web services interoperability. The Cumulus middleware has an on-demand nature for its services; one of them is the security service. Thus service consumers may opt to include these services as they see suitable for their applications. The security service adopts the WS-security located at the network firewall, which handles message encryption and distribution of tokens and signatures.

B. CROWN

Crown [10] is a grid SOM system established to provide a Grid-enabled research environment for resource sharing and collaboration. With the emphasis on the security aspect in Grid computing, CROWN features a strong security architecture that is put in place to maintain safe interactions. The security solution provided includes mechanisms to handle distributed access control (authentication and authorization), secure communication (message level security such as encryption/decryption and signing/ verification), trust management (automated trust negotiation) and trust federation (Identity mapping and credential conversion service). Furthermore, one of the main issues in service oriented grid systems is the management of resources in open environments. The issues related to the encapsulation and deployment of these resources is resolved with the use of ROST (Remote and Hot Deploy with Trust) technique which dynamically updates runtime environment configuration without the need to restart the system during deployment. It also uses the ATN (autonomic trust negation) technique, that protects sensitive information while conducting the trust negotiations.

C. wsBus

wsBus [13] is a lightweight QoS-aware SOM for dependable web services interaction using broker pattern. wsBus provides enhanced QoS attributes for SOA-based applications, which include reliability of web service messaging, scalability of its service provisioning, and security of communications. wsBus provides several QoS features including security, ahich is achieved through two main principles, Authentication and Authorization. Authentication is handled by checking the identification information in the SOAP message or by authenticating the digital certificate of the server that sent the request. If the authentication succeeds then the message is routed otherwise a SOAP fault is returned to the requestor. As for the authorization aspect, it is handled during message routing where the destination point contains an access control list, which the wsBus refers to in order to verify the requestors' credentials to exchange messages with the endpoint.

D. s-AMM

s-AMM [14] is a service-oriented asynchronous messaging middleware introduced to solve the existing problems in Message Oriented Middleware (MOM) when expanded into large-scale network environments. s-AMM enhances the scalability and flexibility of MOM and is responsible for adapting SOAP messages to services, asynchronous transmission and supporting the enterprise business integration. Security is implemented in the three components of the s-AMM architecture. The components are: (i) Message Processing Handler, is responsible for handling the message security, (ii) Common Service, this component provides the handler with services to apply the message security, (iii) Private Service, provides s-AMM with encryption and decryption services. The security service that is provided by s-AMM is based on a message security framework that adds a SOAP message processing layer at both sender and receiver. On the sender's side, the layer processes the SOAP message before passing it to the message processing handler. The message on the sender's side passes through a set of processors: a security attribution handler (adds time stamp, period of validity and sender to SOAP message), a digital signature processor, an encryption processor (XKMS specs) and an authentication processor (X.509 certification, user/password, SAML ticket). As for the receiver side the message goes through a security attribution processor, an authentication, a decryption processor, a signature validation processor and finally is handed to access control.

V. SECURITY SOM REQUIREMENTS

Based on our study of the various approaches of SOM and the particular ones that address security, we identified several security requirements that are essential to SOC. From these requirements, we can mention communication

security, identity management, message protection, access control and trust management. However, it was clear that each one of the projects that included security functionalities, has designed/implemented them in different ways that suit their particular needs. This results in problems in interoperability and reuse. Following we discuss some of the different ways security was integrated within the SOM frameworks we studied.

Access control is one of the requirements included in almost all the projects we covered. For example, in [12] a role-based access control has been setup to maintain a single-sign on approach where identity is propagated across the service chain. Another example is CROWN [10], where a distributed access control mechanism has been implemented to facilitate resource sharing and collaboration in loosely coupled environments. On the other hand, [8] suggests using an attribute-based access control within eGovernment with security policy specified by XACML.

Message protection is another aspect that several projects focused on when implementing their security requirements in the SOM. This kind of protection covers message encryption/decryption, as well as issuing and verifying digital signatures. s-AMM [14] for example uses a message security framework to provide message protection based on serialization followed by the addition of security attributes to protect the serialized messages. Others also enforced message level security including [8], [9], [10] and [13]. After the message level security, transport security is also required to secure the information in the messages exchanged between the communicating nodes. For example transport security is implemented in [12] using SSL encryption to secure the data during transmission.

Trust management is another important security issue to tackle. CROWN [10] addresses resource management where one of the issues involved is trust. The proposed approach includes having trust federation and trust negotiation services to achieve secure resource sharing and collaboration in loosely coupled environments like the Grid. Furthermore, projects such as those described in [7] and [8] also adopted mechanisms to enforce some levels of trust. For example, [7] suggests the use of a reputation-based mechanism to decide whether an entity can be trusted. In [8] trust is established through authentication between communicating entities for all types of exchanges.

QoS security is one of the components that should be addressed as part of the QoS support at the SOM level. The QoS-MLS architecture [15] is an example where the QoS support is extended to include QoS security that incorporates MLS for service-oriented architectures. QoS-MLS relies on system management tools, multi-level security policies and security attributes related to the targeted service to meet the desired level of security. wsBus [13] is also an example which emphasizes security as a key attribute of QoS.

VI. Proposed Security SOM Framework

The extensive study we did on SOM in general [3] and Security SOM in particular, revealed that security is not adequately covered or supported in SOM. However, the exact nature of SOC applications requires extensive support for security to ensure the safety of the interactions among the distributed services. Furthermore, our research in security middleware [19], resulted in identifying a general set of security requirements that apply to any type of middleware including SOM.

Here we put together several of these requirements and present them as a collection of modules (services) that can be integrated into an SOC application as part of the middleware services used (See Figure 1). This can be viewed as a generic security SOM framework that can be adopted easily by SOM-based projects to enhance their security features.

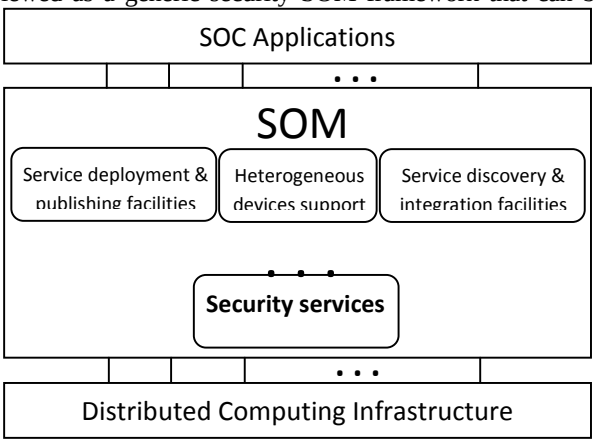


Figure 1. Overview of the SOM architectural model.

The proposed framework includes the security services component which can offer a variety of independent security services. Thus, instead of being forced to incorporate a monolithic middleware solution with complete and fully integrated security functions, which would result in a heavy weight, large footprint layer; we can selectively use the required smaller services. The different projects we studied confirm the need for a more modular design since each one of these project uses a different set of security features and has its own distinct operational conditions to take into account.

Security services in SOM could be developed as a set of independent services that compose the original security middleware functions, where these services can be requested by the client applications without having to worry about the underlying complexities of handling all the security features. The decoupled services can be implemented using any technology or language therefore eliminating interdependencies. The security services that we propose consist of: Identity Management Service (Authentication, Authorization, and Access Control), Communication Security Service, Message Protection

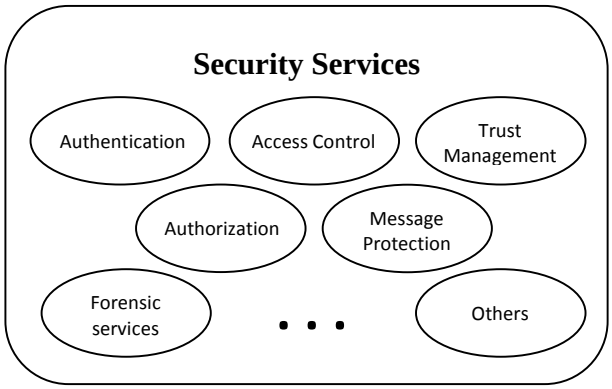
Service, Trust Management Service and Network Security Forensics Service (audit trails, traffic monitoring, activities and transactions logs) (See figure 2). Within each of these main services, several service levels and properties are supported from light weight minimal functions (suitable for low-end devices) to fully fledged high-end features and controls (suitable for servers and high-end devices). This will allow SOC applications developers to mix and match their security needs with the suitable service components without having to include any unnecessary features.

Figure 2. Services in the security services SOM module.

In addition, the SOM should be able to provide the application developer with an expressive model to specify the security needs and available resources. Based on the specification the SOM could look up and bind the suitable services that will satisfy the security requirements. Following we will discuss briefly the services proposed and how they can be achieved.

Identity Management Service: This service governs the critical processes of authenticating and authorizing clients/devices/services. Using local or external identity providers, the authentication will provide the assurances needed regarding the identity of the interacting clients to facilitate usage and access control. Security assertion markup language (SAML) and extensible access control markup language (XACML) are two possible approaches to be used to support authentication [16]. In this part, several configurable versions of the processes may be made available such that SOC applications developer can choose the suitable ones to integrate into their applications. For example, a centralized authentication algorithm may be implemented as one service, while a distributed trust-based model can be implemented as another. Therefore, an application designed for a traditional client/server setup can use the centralized service, while an ad hoc network application can rely on the distributed version.

Message Protection Service: This service helps prevent unauthorized reading and/or modification of exchanged messages. This service intercepts the incoming/outgoing messages to incorporate confidentiality (through encryption/decryption), integrity (through digital signatures), identification and authentication (through tokens). One possible technology to use to achieve this purpose is WS-Security. This standard describes SOAP messaging to provide the necessary protection for messages using a variety of security models and encryption techniques [17]. Here as well several services based on different encryption algorithms can be provided to allow the user to select the suitable one. In addition, the different features should be separated to allow for selective use depending on the available resources. For example, a service to be used on a mobile device that may not have enough power to support full encryption, thus it may only use simple digital signatures or verification methods.



Trust Management Service: This service allows various clients and service provider to manage and control trust levels, which is essential to facilitate identity management. One method to support this is to construct trust relations between certified public keys that are used to mediate security critical actions. This is done by relating abilities to public key authorization certificates. This module will allow us to separate the formulation and management of security policies and credentials from the applications, making it easier to manage trust policies across several applications and services [18]. Again a careful review of trust management models and techniques reveals a vast set of possible implementation methods. Therefore, it becomes essential to have a generic framework where several different implementations are made available for application developers to choose from.

Communication Security Service: This service is responsible for securing the exchanged messages while on transit between different nodes. The technologies used include encryption, decryption and digital signatures. In addition, non-repudiation mechanisms could be included. In some domains such protection is provided at the lower layers of the network protocol stack, which relieves application developers from this responsibility. However, some applications require specialized techniques to be used to satisfy the clients' or organizations' requirements. In this case, making several techniques available will help in the development process.

Network Security Forensics Service: This service allows developers to include monitoring and control functions to help identify sources of problems and sequences of events when problems occur. Services to activate audit trails, traffic monitoring agents, activities and transactions logs can be incorporated into the applications for this purpose. Several techniques may be used including secure logs and audits performed and stored periodically. This can be defined by the developers when selecting and integrating the services into their applications. Independent techniques may be made available as services within the SOM framework allowing application developers to selectively integrate the suitable forensic and monitoring components.

VII. Conclusion and Future Work

In this paper we studied some example research projects that advocate for including security functionalities within the context of service-oriented middleware (SOM). This allows applications developers to integrate security functions as middleware services without having to deal with all the details. However, several SOM approaches do not consider full security solutions, but incorporate specific functions that meet their main requirements within the addressed application domain. As a result we proposed a generic security SOM framework that supports a wide variety of security functionalities with flexible configuration and selection criteria that allow SOC applications developers to select any subset of these functions to incorporate within their applications. This improves the availability and efficiency of security services and enhances the security of the developed applications, while allowing for a flexible method to include only the necessary functions. As a result, when addressing light-weight applications for low-end devices, it is possible to strip down security requirements to a bare minimum and choose the least resource consuming security services. Yet, a high-end service provider may include each and every function available to fortify its applications security. In addition, it becomes possible to include a subset of the available services as needed rather than having to integrate a complete solution where only some features will be used.

The proposed framework is basic and provides a top view of possible approaches and technologies that can be adopted. We intend to use this as our starting point to carefully study the possible approaches for adding security features in SOM and use those to design a full model for the framework and use it to develop prototypes that can be experimented with and evaluated. This will give us a more precise coverage and more practical solutions.

VIII. References

[1] Papazoglou, M.P., "Service-oriented computing: concepts, characteristics and directions," in proc. Web Information Systems Engineering (WISE) 4th International Conference, pp. 3- 12, 10-12 Dec. 2003

[2] "Service Oriented Architecture", Skyway Software. Retrieved Feb. 2010, http://www.skywaysoftware.com/resources/terminology/#Service_Oriented_Architecture

[3] Al-Jaroodi, J., N. Mohamed, and J. Aziz, "Service Oriented Middleware: Trends and Challenges," in proc. 7th International Conference on Information Technology: New Generations (ITNG), IEEE CPS, Las Vegas, USA, April 2010

[4] Issarny, V., M. Caporuscio, N. Georgantas "A Perspective on the Future of Middleware-based Software Engineering," in proc. Future of Software Engineering, pp:244-258, May 2007.

[5] "Service Oriented Architecture Security Vulnerabilities – Web Services," Systems and Network Analysis Center Information Assurance Directorate. 2008. Retrieved Feb. 2010, http://www.nsa.gov/ia/files/factsheets/SOA_security_vulnerabilities_web.pdf

[6] Gunnar, P. "Service Oriented Security Architecture," in Information Security Bulletin, November 2005, Vol. 10, pp:325-330.

[7] Marcio E.F. Maia, Lincoln S. Rocha, Rossana M.C. Andrade, "Requirements and challenges for building service-oriented pervasive middleware," in proc. international conference on Pervasive services, p. 93-102, 2009.

[8] Durbeck S., R. Schillinger, J. Kolter, "Security Requirements for a Semantic Service-oriented Architecture," in proc. 2nd International Conference on Availability, Reliability and Security, p.366-373, April 10-13, 2007.

[9] Wohlstadter E., S. Tai, T. Mikalsen, J. Diament, I. Rouvellou, "A Service-oriented Middleware for Runtime Web Services Interoperability," in proc. IEEE International Conference on Web Services (ICWS'06), p.393-400, September 2006.

[10] Huai, J., C. Hu, T. Wo, J. Li, "CROWN: A Service-Oriented Grid Middleware System: Experience and Applications," in proc. 11th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2008), 5-7 May 2008, Orlando, Florida, USA 2008.

[11] Zdun, U. "Pattern-based design of a service-oriented middleware for remote object federations," in ACM Transactions on Internet Technology (TOIT), vol. 8 no. 3, pp: 1-38, May 2008.

[12] Sliman L., Y. Badr, F. Biennier, N. Salatge, Z. Nakao, "Single Sign-On Integration in a Distributed Enterprise Service Bus," in proc. International Conference on Network and Service Security, pp.1-5, 24-26 June 2009.

[13] Erradi A., P. Maheshwari "wsBus: QoS-aware middleware for reliable Web services interactions," in proc. IEEE International conference on e-Technology, e-Commerce and e-Service, pp. 634-639, March/April 2005.

[14] Hu,J., Z. Zeng, G. Zhao, C. Long, F. Luo, "s-AMM: A Service-oriented Asynchronous Messaging Middleware," in ISECS, vol. 2, pp.312-316, 2nd International Symposium on Electronic Commerce and Security, 2009.

[15] Mohammad, M., A. Chen, G. Wang, C. Wang, R. Santiago, "A Multi-Layer Security Enabled Quality of Service (QoS) Management Architecture," 11th IEEE International Enterprise Distributed Object Computing Conference, 2007.

[16] Yu, W.D. "An intelligent access control for Web services based on service oriented architecture platform," in proc. Software Technologies for Future Embedded and Ubiquitous Systems, 2nd Int'l Workshop on Collaborative Computing, Integration, and Assurance, April 2006.

[17] Patterns: service-oriented architecture and web services, IBM Corp., Riverton, NJ, 2004.

[18] Foley S.N., T.B. Quillinan, M. O'Connor, B.P. Mulcahy, J.P. Morrison "A framework for heterogeneous middleware security," in proc. 18th International Parallel and Distributed Processing Symposium, 26-30 April 2004.

[19] Al-Jaroodi, J., I. Jawhar, A. Al-Dhaheri, F. Al-Abdouli and N. Mohamed, "Security Middleware Approaches and Issues for Ubiquitous Applications," in special issue of Computers & Mathematics with Applications, Elsevier, 2010.

Jameela Al-Jaroodi received her Doctorate of Philosophy degree in Computer Science from The University of Nebraska-Lincoln, USA in 2004. Since August 2006, she has been with the Faculty of Information Technology, at The United Arab Emirates University, UAE as an Assistant Professor. Prior to joining UAEU, Dr. Al-Jaroodi was a research assistant professor at Stevens Institute of Technology in New Jersey, USA. Currently, her research interests involve middleware, distributed collaborative systems, security, information systems, and mobile and pervasive computing. Her research generated over 70 refereed articles in international Journals and conferences such as Journal of Network and Computer Applications (JNCA), Concurrency and Computation: Practice and Experience, IEEE Transactions on Distributed Systems, IEEE International Conference on High Performance Computing and Communications and IEEE International Conference on Cluster Computing. While at Stevens, Dr. Al-Jaroodi received the Research Excellence Grant from Sun Microsystems, Inc. In addition, several areas of her research were also supported by the United States National Science Foundation (NSF), Nebraska Foundation, the National Center for Information Technology in Education (NCITE) and UAEU research grants.

Alyaziyah Al-Dhaheri obtained her Bachelor Degree in Information Technology with specialization in Information Systems from the Faculty of Information Technology (FIT) at The United Arab Emirates University (UAEU), UAE in June 2009. She was a research assistant at FIT and contributed positively in research efforts in the areas of security middleware and service oriented computing. She also published refereed international articles in her research field. Ms. Al-Dhaheri received several awards such as second place in the Scientific Innovations Competition at the 6th GCC Cultural and Scientific Week, 2007 and a certificate of recognition for participating in helping to further innovation and excellence in education throughout the UAE, 2008. Alyaziyah is currently pursuing a master's degree in business administration at the Faculty of Business and Economics at UAEU, UAE.