International Journal on New Computer Architectures and Their Applications (IJNCAA) 1(3): 565-580
The Society of Digital Information and Wireless Communications, 2011 (ISSN: 2220-9085)

Model-based Security Engineering of SOA System Using Security Intent DSL

Muhammad Qaiser Saleem[1], Jafreezal Jaafar[1], Mohd Fadzil Hassan[1]

[1] Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 31750 Tronoh, Perak Darul Ridzuan, Malaysia.

qaiser_saleem73@hotmail.com, jafreez@petronas.com.my, mfadzil_hassan@petronas.com.my

Abstract. Currently most of the enterprises are using SOA and web services technologies to build their web information system. They are using MDA principles for design and development of WIS and using UML as a modelling language for business process modelling. Along with the increased connectivity in SOA environment, security risks rise exponentially. Security is not defined during the early phases of development and left onto developer. Properly configuring security requirements in SOA applications is quite difficult for developers because they are not security experts. Furthermore SOA security is cross-domain and all required information are not available at downstream phases. Furthermore, business process expert; who is the actual stakeholder of the business process model is unable to specify security objectives due to lake of security modelling elements in a general purpose modelling languages like UML. As a result, business process expert either ignore the security intents in their model or indicate them in textual way. A security intents DSL is presented as a UML profile where security intents can be modelled as stereotypes on UML modelling elements during the business process modelling. Aim is to facilitate the business process expert in modelling the security requirements along the business process modelling. This security annotated business process model will facilitate the architectural team in specifying the concrete security implementation. As a proof of work we apply our approach to a typical on-line flight booking system business process.

Keywords: Service Oriented Architecture, Model Driven Security, Unified Modeling Language, Business Process Modeling, Security Intents

1 INTRODUCTION

1 Introduction

IT-infrastructure have been evolved into an enterprise landscape which is basically a distributed and loosely coupled, Service Oriented Architecture (SOA) environment [1], which offer the Information Technology (IT) agility demanded by the business [2, 3]. In SOA systems; software applications are deployed over the Internet as a service. To support a business ventures; these services are integrated within and across organizations to form Internet-based Web Information System (WIS) and perform cross application transactions [4]. In the new business scene, where companies are using intensive use of Information and Communications Technologies (ICT), they are also increasing their vulnerabilities. With the increase in number of attacks on the system, it is probable that an intrusion can be successful [5]. The security violation defiantly cause losses, therefore it is necessary to secure the whole system. If we talk about SOA security then it is not sufficient to just protect a single point, a comprehensive security policy is required [1]. Security measures implemented in SOA systems are viewed from two different levels; first at high level security objectives, which are basically abstract representation of the security goals and the second at detailed security policies [4].

Security must be unified with the software engineering process but in practice it is considered afterthought and implemented in ad-hoc manner [5]. Furthermore it is left to the developer and added when the functional requirements are met or at the time of integration of distributed applications which is not a realistic approach [6]. SOA applications are cross-domain and coupled over various network technologies and protocols; just adding security code to software applications is not a realistic approach because all required security information are not available at the downstream phases[6, 7]. This approach degrade implementing and maintaining security of the system [8].

During the past few years, several SOA security protocols, access control models and security implementations have emerged to enforce the security goals [6, 9]; however focus of the SOA security standards and protocols are towards technological level; which do not provide high level of abstraction and mastering them is also a daunting task [1, 10]. Dealing security only at implementation stage will leads to security vulnerabilities, which justify increasing effort in defining security in pre-development phases, where finding and removing a bug is cheaper [11]. Business process modeling is the most appropriate layer to describe security requirements and to evaluate risks [1]. Business process modeling is normally performed in a modeling language such as Unified Modeling Language (UML) or Business Process Modeling Notation (BPMN). These modeling languages do not support specification of security requirements [12]. Some security extensions are proposed to annotate the business process model with security goals [13, 14] and the work is in progress. [13]

Model Driven Security (MDS) and automatically developed software having security configuration is a topic of

interest among the research community and different research groups across the globe are trying to solve the security problems for SOA based applications by presenting MDS Frameworks [6, 9, 12-16].

Business process modeling can be performed from different perspectives; security expert, business analyst and end user perspectives; and at different levels of abstraction [5]. Both experts i.e. business domain expert as well as security expert; work side-by-side while designing a business process model and defining security requirements [9]. Empirical studies shows that those, who model the business process i.e. business domain expert are able to specify security requirements at high level of abstraction [5]. It is evident that business domain expert must define the security requirements at business process model [17]. However in practice, business domain expert mainly focus on the functionality of the system and often neglect the security goals. It may be happened due to many reasons e.g. business domain expert is not a security expert [5] and no currently available process modeling notation have ability to capture security goals[17]. Furthermore system model and security models are disjoint and expressed in different ways i.e. system model is represented in a graphical way in a modeling language like Unified Modeling Language (UML) while security model is represented as a structured text [5]. Incorporating security goals into a business process model is a challenging task due to many reasons [18]:

- There is not a clear identification of security requirements to be modeled.
- Absence of notations to express the security requirements.
- Difficulty in integrating security requirements into business processes modeling

Our aim is to facilitate business process expert to add security goals while modeling business process for SOA based systems. Security annotative business process model will facilitate the security expert while defining concrete security implementation. In our work:

- We have provided detail analysis of basic security intents for modeling security objectives in a business process model i.e. confidentiality, integrity, availability auditing.
- We have presented a Domain Specific Language (DSL) to express these security requirements. We have used UML-profiling mechanism to extend the UML and proposed security stereotypes.
- As a proof of concept; we have projected our work to a real world business process model.

A business domain expert is facilitated to use modeling language which is equipped with the vocabulary for specifying security objectives at PIM level of abstraction. Hence he/she only need to understand the security concepts in the UML-based security design language and don't have to expertise in target security technologies [8]. Being able to express security requirements in a widely used design notation like UML; helps to save time and effort during the implementation and verification of security in system [19].

2 Related Work

A language is required for modeling security during designing the software system which provides syntax and semantic as provided by the UML and BPMN. To fulfill the security requirements in modeling languages, different extensions are proposed in the modeling languages to the model security. To model the security objectives related to different system's aspects different security extensions are

proposed by different authors. Mostly authors represent the abstract syntax of their DSL by a meta-model using Meta Object Facility (MOF) framework and concrete syntax by UML profile [5, 11, 15, 20]. Related work exists almost along all type of software development models, following is its brief descriptions:

System Models*:* Static structure of the system is represented by UML class diagram and UML state diagram [21]. Basin David et al. [8] presented SecureUML to model the security requirements for modeling static structure of the system. Basically it is a separate language based on protocol of Role Based Access Control (RBAC). Afterwards SecureUML can be integrated with any system modeling language like UML or BPMN to model the security in the system design. They have presented a meta-model for abstract syntax and used UML profile for concrete syntax and security constraints are added through Object Constraints Language (OCL).

Work Flow Model: UML activity diagram and BPMN are used to represent the business process work flow. This is the most important aspect of a system and most of the security extensions are proposed related to this aspect.
Rodriguez A. et al. created a meta-model for their security extensions and defined security stereotypes and developed a DSL. They also assign different symbols to these security stereotypes. They used the same DSL for extending the BPMN [5] as well as UML Activity diagram [11]. Another extension is made by Christian Wolter et al. [17], they incorporate security stereotypes in BPMN. Another research group lead by Ruth Brue et al. has presented [15] security stereotypes in UML activity diagram.

Deployment Diagram*:* UML component diagram is used for the representation of deployment of a system [21]. UMLSec presented by Jürjens, J. [22] also support the secure modeling of UML component diagram.

Interaction Diagram: UML Sequence diagram is used to represent the flow of control between the object of the system [21]. Jürjens, J. [22] defined UMLSec by extending the UML and developed a UML profile to incorporate security to represent the secure interaction.

3 Literature Study

3.1 Service Oriented Architecture (SOA)

SOA paradigm makes the software application development easy by coupling services over intranet and via the Internet [6]. SOA paradigm has changed the Internet from being repository of data to repository of service [13]. SOA is an architectural style in which software applications are comprised of loosely coupled and reusable services by integrating these services through their standard interface. Services are independent of language, platform and location and may be locally developed or requested from the provider. A business process can be realized as a runtime orchestration of set of services. Software applications are often comprised of numerous distributed components such as databases, web servers, computing nodes, storage nodes etc. and these components are distributed across different independent administrative domains. Services are used but not owned by the user and they reside on provider side. The reusability,

agility, cost effectiveness and many other attributes of SOA paradigm has attracted the organizations to adopt it for software development [23-25]. SOA is also called a "*Find, bind and invoke paradigm*" [4, 26] as shown in Figure 1

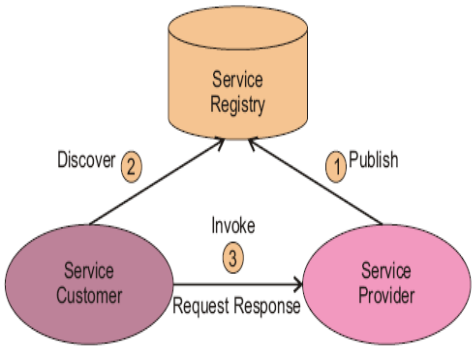


Fig. 1*:* Collaboration of Services in SOA environment

The basic building block of a SOA paradigm is a service. "*A service is an implementation of a well-defined piece of business functionality, with a published interface that is discoverable and can be used by service consumers when building different applications and business processes*" [27]. SOA paradigm can be implemented with different technologies like CORBA, Web Services, JINI etc.; however Web services technology is a widespread accepted instantiation of SOA [25, 28].

3.2 Web Services (WS)

Web Services are defined as "*self-contained, modular units of application logic which provide business functionality to other applications via an Internet connection*" [28]. Software applications are developed by integrating different web services either newly built or legacy applications by avoiding difficulties due to heterogeneous platforms and programming languages by exploiting the XML (Extensible Markup Language) and the Internet technologies [28, 29]. Web service enable the dynamic connections and automation of business processes within and across enterprises for EAI (Enterprise Application Integration) and B2B (Business-to-Business) integration. There are several XML based standards which lies the foundation of the Web Services technology e.g. UDDI (Universal Description, Discovery and Integration), SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) etc. [25, 30]. WSDL is used for the service interface description, SOAP messages are used for the communication between services and UDDI is used for the description and discovery of services into/from service registry. Service provider; publish the description of service in a service registry. Service consumer/user search the service according to their description in the registry and use the services if found [4].

Fig. 2 : Protocol Stack of Service Discovery, Description, and Invocation [30]

3.3 Business Process Modeling

Business Process Modeling is gaining more and more attention in an organization because it is the foundation

to describe the organizational workflow [1]. An effective business process model will facilitate the stakeholders of the business to understand the different aspects of the business system and provide a platform to discuss and agree on key fundamentals for achieving the business goals [5]. A business process is defined as "*a set of procedures or activities which collectively pursue a business objective or policy or goal*" [5]. It can also be defined as "*a set of activities and execution constraints between these activities*"[1]. Different techniques are used for business process representation; Damij, N. in [31], group them in two categories; diagrammatic and tabular. Christian Wolter et al. in [17] described different popular diagrammatic business process modeling notations like BPMN, UML, XPDL, Jpdl; among these two languages, UML and BPMN are considered as industry standards [5].

3.4 Model Drive Architecture (MDA) and Model Driven Security (MDS)

Currently software engineering is greatly influenced by a new MDA paradigm which work at model and meta-model level [32]. In MDA approach software systems are specified and developed through models; transformation functions are automatically performed between models at different levels of abstractions as well as between models to code [8].. Model based design methodology is being widely accepted in development of electronics systems due to their flexibility and tool support. To organize landscape of model, meta-modeling techniques are emerged; theories and methods are provided for the development of coordinated representation suitable for heterogeneous environment such as SOA [33].
MDS specializes MDSD towards information security [34]. MDS is a technology where security requirement are defined as a model during designing phase and concrete security configuration files can be generated by model transformation [7].

4 Organizational Security Goals

Security is an abstract concept which can be defined by specifying a set of security goals. These security goals can be further subdivided, specialized or combined [17]. Security objectives describe the most basic security need of an asset [30] and they can be defined as " *a statement of intent to counter identified threats and/or satisfy identified organizational security policies and assumptions*" [35]. Many names can be found in literature for security objectives like security properties, security aspects, security concern, security intents or security states [36].

Computer security is also defined as "*the protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resource*." [37]. These security principles are applicable to all information system irrespective of their technology platform, communication channels, size of the organization etc. Security is a composite notion, comprised of , confidentiality (the prevention of unauthorized disclosure of information), integrity (the prevention of unauthorized amendment or deletion of information) and availability (the prevention of unauthorized withholding of information) [38]. Conceptually; three basic security principles are Confidentiality, Integrity and Availability [39] and also known as CIA (Confidentiality, Integrity and Availability) [40]. CIA are termed as pervasive in nature and fundamental to all information systems [41] and for SOA applications these basic security principles are unchanged.

4.1 Security Objectives in related work

Different research groups are focusing different security goals for their DSLs [5, 6, 11, 17, 30]. In [34] Michal Hafner et al. defined the three security goals naming confidentiality, integrity and availability. They defined access control as confidentiality and availability is used in the meaning of no-repudiation. In [5, 11] Alfonso Rodríguez et al. extended the UML and BPMN by defining DSLs and focusing on five security goals: access control, integrity, privacy, attack-harm detection and non-repudiation. In [17]. Christian Wolter et al. presented a security policy model by focusing six security goals: authentication, authorization, confidentiality, integrity, availability, auditing. Michal Menzel et al. also used security policy model in their work [1] and defined security extensions to the BPMN. In [7] Yuichi Nakamura et al. defined three security intents for their work: authentication, integrity and confidentiality and defined a UML profile. In [6] Yuichi Nakamura et al. addressed four business level security intents as they are easy to be understood by business user and presentation of them is discussed in UML: Authentication, Integrity, Non-repudiation and confidentiality. Basically they picked some of the security intents defined in [42] and their names are changed according to WS-Security's terminology.

Among the security objectives mentioned above, we believe following are the essential security objectives which should be modeled in a business process model of SOA applications; which are focused by different authors either as it is or with some different name or by merging them.

1. *Confidentiality:*

It specifies the system's state where only authorized entities can access the information. Access control is maintained by authentication and authorization. Authentication is a mechanism to verify the identity of an entity. Authorization is based on some specific security model, how to grant various privileges to various entities on different resources [34]. Many authors treat confidentiality, authentication and authorization as a separate security goals [1, 5, 11, 17]. However; Ruth Brue and Michal Hafner in their work [34] keep authentication and authorization under the umbrella of confidentiality and we agree with their work because by enforcement of these access control mechanism one can achieve confidentiality.

2. *Integrity:*

It identifies an authorized subject to alter information in authorized ways. It ensures the integrity of data (properness of information) as well as integrity of origin [34] . Transferred, processed or stored data can only be modified with proper rights [17]. Basically it ensures that the transferred data between parties must be guaranteed to reach the recipient in the same form and with the same content [6].

3. *Availability:*

It is an important aspect of reliability and in SOA environment, it is interpreted as non-repudiation. A user may use a resource or call a service and this usage or service call must not deniable. Basically it is a system state where provision of a specific resource is guaranteed [30, 34]. It ensures that the information must include the digital signatures of the parties related to the document [6].

4. *Traceability and Auditing:*

It is a process of verification of all actions performed in an information processing system [17]. It underlies each security requirement and will automatically be understood when a security requirement is specified in a model [5], so there is no need to model it separately in a business process model.

5 Extending a Modeling Language According To a Particular Domain: Domain Specific Language (DSL)

General purpose modeling languages like UML are very successful and they also provide the tool support ranging from requirement engineering to code generation. However they does not render the superfluous of DSLs; furthermore it is very clumsy for tasks that can benefits from the integration of the domain-specific restrictions [20]. DSLs are small and provide basis for domain-specific formal analysis; furthermore DSLs use those notions which are familiar to domain experts [20]. DSL is used to formalize a modeling language capable of formalizing different business domains (like e-government, e-education), system aspects (like security, real-time) or concrete technologies(such as EJB or .NET [8]). Extending a modeling language according to a particular domain and defining DSL is a common practice e.g. UML extensions according to specific domains like data warehousing[43], Business intelligence[44] and real-time systems [33]. Following are the three alternatives for defining a DSL [8, 45].

1. Defining DSL using Extension Point Provided by Language itself:

The easiest way of defining a DSL is the usage of the extensions points provided by the language itself [45]. DSL can be defined directly in UML in a lightweight way by using stereotypes and tagged values known as "labels" resulting *UML profile*. To introduce new language primitives (elements), *stereotypes* are used by extending the semantics of existing types in the UML meta-model. Stereotypes are represented by double angle brackets e.g. <<*stereotype*>>. To formalize the properties of these new language primitive, *tagged values* are used which are written within curly brackets e.g. {Tag, Value} [46], which associate data with model elements. Model elements are assigned to these new language primitives and labeled them with corresponding stereotype. If some additional restrictions are required on the syntax of these new language primitives; Object Constraints Language (OCL) constraints is used. OCL is a specification language provided by UML, based on first order logic. Normally OCL expressions are used for various purposes such as invariant for classes, pre and post conditions for methods and guards for state diagram. Set of such definitions i.e. stereotype, tagged values and OCL constitutes the UML profile [8].

Most of the currently available UML modeling tools can readily be used because they support the definition of custom stereotypes and tagged value. Because of having tool support this approach is widely used [8, 20, 22]. Normally DSLs are defined by UML-Profiles when the "domain" may be combined with other domains, in an unpredictable way and the model defined under the domain may be interchanged with other domains [20].

It is very clumsy to add domain-specific restrictions in large languages like UML; furthermore for formal analysis, large languages usually lack detailed formal semantics [20]. Visualization of the complicated security intents might be confusing; furthermore, many modeling languages do not provide extension points [45].

2 Defining DSL by defining a Meta-Model:

Remaining two extension techniques are meta-model based techniques and known as heavy weight extension mechanism. Meta-model based technique of defining DSL is mostly used when the "domain" is well defined and has accepted set of concepts; there is no need to combine the domain with other domains and the model defined under the domain is not transferred into other domains [20].

A. DSL can be defined by using MOF by extending the meta-model of existing modeling languages like UML. Concept of stereotype is used to formally extend the meta-model of an existing modeling language. At modeling level, stereotypes are manipulated as annotation on model elements. In this way of DSL definition, an existing meta-model is reused and specialized.

Limitation is that the extended and customized meta-model is based on the entire meta-model of existing modeling languages and may be complex. Furthermore to support the DSL; CASE (Computer Aided Software Engineering) tool may also require extension to accommodate these new language primitives in particular storage component (repository) and visualization component [8, 20, 33]. Furthermore; extensions are defined and integrated according to a particular domain into a specific modeling language based on its meta-Model [45].

B. A new DSL for modeling the domain of interest or particular problem is created by a fully dedicated meta-model using MOF having no dependency on existing modeling languages. The resulting DSL have much more concise vocabulary than the vocabulary of existing modeling languages e.g. UML. For querying and manipulating meta-data of these DSL, interface would be more simple then the UML Interfaces. Abstract syntax is represented by the meta-model and notions (concrete syntax) of the DSL are specified with the UML profile [8]. This way of extension is optimally suited for the problem at hand [33].

Limitation is, sometime it does not provide the well-defined mapping between the UML model with which developer work, to the instances of meta-model of DSL that define the meaning of this model [20].

To gain the benefits of DSL and general purpose modeling language, DSLs are defined in terms of general purpose modeling language like UML or BPMN [20]. Current practice of defining a DSL by different researchers [5, 8, 11, 17, 22] is; abstract syntax is represented by a meta-model and concrete syntax (notion) is represented by a UML profile. We are also working along this approach.

6 Proposed Domain Specific Language

To gain the benefits of DSL and general purpose modeling language, DSLs are defined in terms of general purpose modeling language like UML or BPMN [20]. In our research work our domain is "*modeling the security in SOA system*". General purpose modeling language like UML can easily be customized by the extension mechanism provided by the language itself and DSL can be defined according to the domain of interest by extending the general purpose modeling language. In case of UML the extension mechanism is known as *UML Profile*. Tools are available for the general purpose modeling languages which support the definition and usage of DSL. In our case we have focus the domain of "*SOA Security*" and we have extended the general purpose modeling language UML by providing a DSL. We have used MagicDraw tool for UML modeling which support the definition and usage of DSL. The whole phenomenon can be explained by the Figure 3.

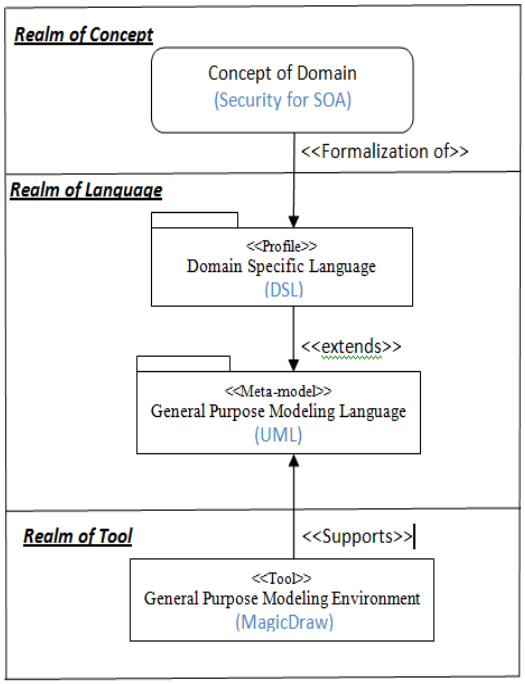


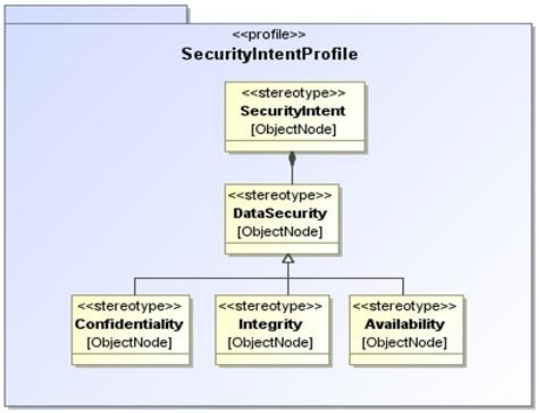
Fig.3. Definition Process of a Domain Specific Language [33]

Abstract syntax of our DSL is defined by a meta-model and concrete syntax by providing stereotypes. Afterwards UML profiling mechanism is used to apply our DSL into UML.

Each extension of the elements of UML meta-model is formally captured under the concept of stereotypes. Properties and/or modeling constraints of the target domain are associated with the stereotypes which results the UML profile. The most difficult task is the identification of elements of the meta-model of a modeling language which must be extended i.e. in case of UML, identification of UML meta-classes for which the stereotypes will be defined. In our case we are extending UML meta-class Object-Node. After the definition of domain specific UML-profile, general-purpose modeling tool can easily be specialized and these domain specific stereotypes are made available at the modeling level in the form of annotation [33]. Figure 3 explain the whole concept.

6.1 Abstract Syntax

Abstract syntax of our DSL is presented by a met model. The UML profile that describes our met model is described as UML package with the stereotype <<profile>> as shown in Figure 4. We are using package for the creating of our DSL as discussed in [47]. Our DSL is based on the security intents disused in previous section. The most difficult task is the identification of elements of the meta-model of a modeling language which must be extended for example in case of UML, identification of UML metaclasses for which the stereotypes will be defined [33]. In our case we have extended UML meta-classes *ObjectNode* and *ActivityNode* i.e. these are the metaclasses to which stereotypes will be assigned.



- **Fig.4.** Abstract Syntax of Proposed DSL

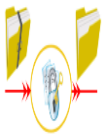
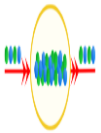
• 6.2 Concrete Syntax

- Each extension of the elements of UML meta-model is formally captured under the concept of stereotypes. Properties and/or modeling constraints of the target domain are associated with the stereotypes which results the UML profile. After the definition

of domain specific UML-profile, general-purpose modeling tool can easily be specialized and these domain specific stereotypes are made available at the modeling level in the form of annotation [33]. For concrete syntax we have presented following stereotypes as shown in Table 1.

- **7 Case Study**

- To demonstrate our work, a case study of *"Online Flight Booking System"* is presented. It describes the web services based interaction between the participants and enables them to work through the Internet. The whole process has to be realized in a peer-to-peer fashion and would integrate security requirements.

S/No	Security Stereotype	Symbol	Description
1.	<<Confidentiality>>		Idea behind the symbol is that, initially information are inaccessible to user and will only be access able to him/her when he/she provides the desired security credentials. In BPD it can be specified in Pool, Lane, Activity or Group. Idea is to restrict the access to authorized user only.
2.	<<Integrity>>		Idea behind the symbol is that before transformation, information contents are in particular form; during transformation it may change its form however it must be in the same form on its receipt. In BPD it is specified over the Message Flow
3.	<<Availability>>		Basically it is based on the idea of no-repudiation i.e. whenever a user uses some resource or service then his/her signature will be stored with the document along with date and time information. In BPD it can be specified over the message flow, it means it means the interactions cannot be denied.

7.1 Business Scenario

In today's era travel agencies provide online services to travelers for booking the flights. Traveler submits the trip information to the travel agency, containing the personal information of travelers; start date, end date, origin, destination and price rang etc. After having this information travel agency search for the suitable airline and routes accordingly and prepare itinerary and send it to traveler. If traveler accepts the itinerary then he/she will make payment into the bank specified by the travel agency. The bank; upon receiving payment send receipt of payment to both i.e. traveler as well as travel agency. After receiving conformation of payment, travel agency will order ticket from airline, which will send the ticket to the traveler.

7.2 Stakeholders:

In the case-study; services from the four stakeholders are involved i.e. traveler, travel agency, airline and bank.

7.3 Security Requirements of the system

In online flight booking system a traveler needs to perform different tasks i.e. fill in the trip information form, viewing the itinerary, make payment into the bank, view the ticket etc. Necessary permissions are assigned to him/her on different objects to perform these tasks i.e. travels require update information on trip information payment form, read permission on itinerary information and ticket. To perform these operations traveler's personal information are involved at different places e.g. passport number while filing the trip information, credit card information while making payment to bank etc. Therefore *confidentiality* is required i.e. proper *access control* mechanism with *authentication* and *authorization* is required to access this information. Furthermore, traveler has to submit the trip order to the travel agency, traveler must sign it with his/her signature so he/she may not be able to deny that he/she has not submitted the trip order. *Availability (Non-repudiation)* is required in this use-case between the traveler and travel agency. Travel order form is submitted online, therefore secure information flow i.e. *Integrity* is required to successfully perform this use-case. These three security requirements i.e. *Confidentiality*, *Availability*, and *Integrity* are identified and modeled for other stakeholders of the case-study like travel agency, airline and bank. Figure 4 shows the security enhanced business process model of the flight booking system use case.

Meaning of a particular security symbol at a specific place is discussed below.

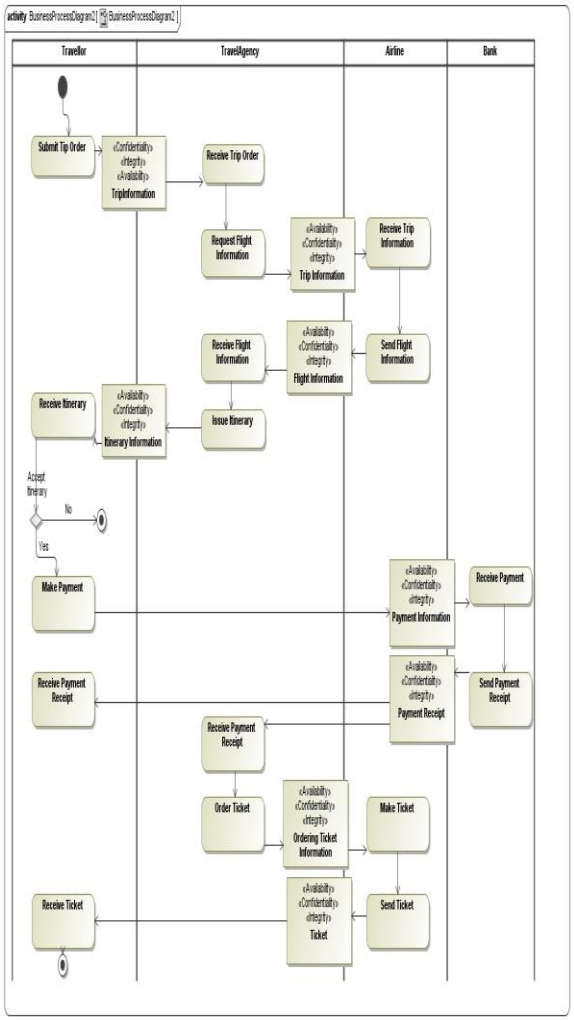
Confidentiality: Whenever some information are sent or received they are consider as confidential i.e. we show confidentiality requirement on data objects.

Availability (Non Repudiation): Whenever some information would be sent or received between the stakeholders; then availability security requirements would be modeled to ensure the non-repudiation. It represents that sending person would include additional information like digital signature, time and date along with the message, so the interactions cannot be denied.

Integrity: This security requirement is modeled whenever some transmission of information is takes place. It represents

integrity of the information transmitted over the Internet. In the case study whenever stakeholders interact with each other through sending messages; integrity symbols would be modeled over the message flow to ensure the integrity of information flow.

fig.4. Security Annotated UML Activity Diagram



(Business Process Model) of the case study

8 Conclusion and Future Work

Incorporating security requirements during early stages of software development will improve the important aspect "Security" of SOA based Information Systems. A security DSL is presented to model the security along with the business process model. We have facilitated the business process expert in modeling the security requirements along with the business process model. This security annotated business process model will facilitate the security expert in specifying concrete security implementation. We believe our effort is a contribution towards stressing to incorporate security requirements during business process modeling for SOA applications.

We are in the process of enhancing our DSL to incorporate more security intents which are essential to be modeled during business process modeling for SOA applications.

References

1. Menzel, M.T., I. Meinel, C. *Security Requirements Specification in Service-Oriented Business Process Management*. in *International Conference on Availability, Reliability and Security, 2009. ARES '09*. 2009.
2. Dr. Ulrich Lang, R.S., *Top SOA Security Concerns & OpenPMF Model-Driven Security*. ObjectSecurity white-paper, Topics Cloud Computing and Security Management, 2009.
3. Firesmith, D.G., *Engineering Security Requirements.* Journal of Object Technology, 2(1):53-58, 2003.
4. Xie, D.Y., Shi Zhang, Tao Jia, Xiang-Yang Liang, Zao-Qing Yao, Jun-Feng. *An Approach for Describing SOA*. in *International Conference on Wireless Communications, Networking and Mobile Computing*. 2006.
5. Rodriguez, A. and E.F.-M.M. Piattini, *A BPMN Extension for the Modeling of Security Requirements in Business Processes.* IEICE - Trans. Inf. Syst., 2007. E90-D(4): p. 745-752.
6. Nakamura, Y.T., M. Imamura, T. Ono, K. *Model-driven security based on a Web services security architecture*. in *IEEE International Conference on Services Computing, 2005*. 2005.
7. Satoh, F.N., Y. Mukhi, N. K. Tatsubori, M. Ono, K. *Methodology and Tools for End-to-End SOA Security Configurations*. in *IEEE Congress on Services - Part I, 2008.* 2008.
8. David Basin, J.D., Torsten Lodderstedt, *Model driven security: From UML models to access control infrastructures.* ACM Trans. Softw. Eng. Methodol., 2006. 15(1): p. 39-91.
9. Christian Wolter, M.M., Christoph Meinel, Andreas Schaad, Philip Miseldine, *Model-driven business process security requirement specification.* J. Syst. Archit., 2009. 55(4): p. 211-223.
10. Alam, M., *Model Driven Security Engineering for the Realization of Dynamic Security Requirements in Collaborative Systems*, in *Models in Software Engineering*. 2007. p. 278-287.
11. Rodríguez, A., E. Fernández-Medina, and M. Piattini, *Towards a UML 2.0 Extension for the Modeling of Security Requirements in Business Processes*, in *Trust and Privacy in Digital Business*. 2006. p. 51-61.
12. Menzel, M.M., C. *A Security Meta-model for Service-Oriented Architectures*. in *IEEE International Conference on Services Computing, SCC '09'*. . 2009.
13. Jurjens, J., *UMLsec: Extending UML for Secure Systems Development-Tutorial*, in *Proceedings of the 5th International Conference on The Unified Modeling Language*. 2002, Springer-Verlag.
14. Torsten Lodderstedt, D.A.B., Jürgen Doser, *SecureUML: A UML-Based Modeling Language for Model-Driven Security*, in *Proceedings of the 5th International Conference on The Unified Modeling Language*. 2002, Springer-Verlag.
15. Michal Hafner, R.B., Berthold Agreiter, *SECTET: an extensible framework for the realization of secure inter-organizational workflows.* Emeral Internet Research, 2006. Vol.16 No. 5 , Pag: 491-506: p. pp.491-506.
16. Mukhtiar Memom, M.H., Ruth Breu, *SECTISSIMO: A Platform-independent Framework for Security Services.* MODSEC08 Modeling Security Workshop, 2008.
17. Wolter, C., M. Menzel, and C. Meinel, *Modelling Security Goals in Business Processes.* Proc. GI Modellierung 2008, GI LNI 127, Berlin, Germany, March 2008. pp. 197 - 212.
18. Baresi, L., et al., *Incorporating Security Requirements into Service Composition: From Modelling to Execution*, in *Service-Oriented Computing*. 2009, Springer Berlin / Heidelberg. p. 373-388.
19. Jurjens, J., *Developing Secure System with UMLsec From business process to implementation.* Computing Laboratory University of Oxford GB, 2001.
20. Achim D. Brucker, J.u.D., *Metamodel-based UML Notations for Domain-specific Languages*. 4th International Workshop on Language Engineering (atem 2007), pp. 1-{??}, 2007.

21. Mikael Åkerholm, I.C., Goran Mustapić *Introduction for using UML* 2004.
22. Jürjens, J., *UMLsec: Extending UML for Secure Systems Development*, in *«UML» 2002 — The Unified Modeling Language*. 2002. p. 1-9.
23. Lewis, G., A. Morris, E., Simanta, S., Wrage, L. *Common Misconceptions about Service-Oriented Architecture*. in *Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, 2007. ICCBSS '07.* . 2007.
24. Asit Dan, P.N., *Dependable Service-Oriented Computing. IEEE Internet Computing*, March/April 2009, pp. 11–15, 2009.
25. Philip Bianco, R.K., Paulo Merson, *Evaluation of Service-Oriented Architecture.* Software Engineering Institute/ Carnegie Mellon, 2007. Technical Report, CMU/SEI-2007-TR-015, September 2007.
26. Papazoglou, M.P. *Service-oriented computing: concepts, characteristics and directions*. in *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on*. 2003.
27. Liam O'Brien, L.B., Paulo Merson *Quality Attributes and Service-Oriented Architectures* Software Engineering Institute/ Carnegie Mellon, *September 2005* Technical Note : CMU/SEI-2005-TN-014
28. Antonio Bucchiarone and S. Gnesi, *A Survey on Services Composition Languages and Models.* International Workshop on Web Services Modeling and Testing (WS-MaTe 2006), 2006.
29. van der Aalst, W.M.P., M. Dumas, and A.H.M. ter Hofstede. *Web service composition languages: old wine in New bottles?* in *Euromicro Conference, 2003. Proceedings.* 2003.
30. Michal Hafner, R.B., *Security Engineering for Service-Oriented Architectures.* Springer-Verlag Berlin Heidelberg, 2009, .
31. Damij, N., *Business process modelling using diagrammatic and tabular Techniques.* Business Process Management Journal, 2007. 13(1): p. 70-90.
32. Rodríguez, A., E. Fernández-Medina, and M. Piattini, *Towards CIM to PIM Transformation: From Secure Business Processes Defined in BPMN to Use-Cases*, in *Business Process Management*. 2007. p. 408-415.
33. Passerone, R.D., W. Ben Hafaiedh, I. Graf, S. Ferrari, A. Mangeruca, L. Benveniste, A. Josko, B. Peikenkamp, T. Cancila, D. Cuccuru, A. Gerard, S. Terrier, F. Sangiovanni-Vincentelli, *Metamodels in Europe: Languages, Tools, and Applications.* Copublished by the IEEE CS and the IEEE CASS, 2009. 26(3): p. 38-53.
34. Michal Hafner, R.B., *Security Engineering for Service-Oriented Architectures.* BOOK, 2009 Springer-Verlag Berlin Heidelberg.
35. *Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model*. CCIMB-99-031, Version 2.1, 1999.
36. M. Schumacher, E.F.-B., D. Hybertson, F. Buschmann, P. Sommerlad, *Security Patterns : Integrating Security and Systems Engineering (Wiley Software Patterns Series).* John Wiley & Sons,, March 2006.
37. Marianne Swanson, B.G., *Generally Accepted Principles and Practices for Securing Information Technology Systems.* National Institute of Standards and Technology Technology Administration U.S. Department of Commerce, 1996.
38. Algirdas Avizˇ ienis, J.-C.L., Brian Randell, *Fundamental Concepts of Dependability*. Research Report N01145, LAAS-CNRS, April 2001.
39. Ajay Tipnis, I.L., *Security - A major imperative for a service oriented architecture.* EDS White Paper, June 2008.
40. JONNAGANTI, V., *An Integrated Security Model for the Management of SOA.* Master Thesis University of Gothenburg, May 2009.
41. Poore, R.S. (1999) *Generally Accepted System Security Principles*. by International Information Security Foundation.
42. Johnston, S., *Modeling security concerns in service-oriented architectures.* IBM developerWorks, 2004.

43. Luján-Mora, S., Trujillo, Juan, Song, Il-Yeol, *Extending the UML for Multidimensional Modeling*, in *UML 2002, LNCS 2460, pp. 290-304, 2002*. 2002, Springer-Verlag Berlin Heidelberg 2002. p. 265-276.
44. Stefanov, V., B. List, and B. Korherr, *Extending UML 2 Activity Diagrams with Business Intelligence Objects*, in *Data Warehousing and Knowledge Discovery*. 2005. p. 53-63.
45. Menzel, M. and C. Meinel. *SecureSOA Modelling Security Requirements for Service-Oriented Architectures*. in *Services Computing (SCC), 2010 IEEE International Conference on*. 2010.
46. Saleem, M.Q., J. Jaafar, and M.F. Hassan, *Model Driven Security Frameworks for Addressing Security Problems of Service Oriented Architecture*. ITSim 2010, 2010.
47. OMG., T., *Meta Object Facility (MOF) 2.0 Core Specification.* OMG Available Specification,, 2005.