IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 3, Ver. II (May – Jun. 2015), PP 56-60
www.iosrjournals.org

Abstraction and Automation: A Software Design Approach for Developing Secure Data Management Application.

C. Bharathipriya

Gkm College Of Engineering And Technology, Chennai

Abstract: *Security plays a vital role in data management application. For this purpose while developing, three different views of applications are modeled namely conceptual, security and GUI model. It formalizes the objects of problem, applications security and Graphical user interface respectively. A model transformation process occurs, which shifts policy specified in security to GUI model. To ensure model's correctness validation process is carried out and as a final step, a code generator will generate a multi-tier application along with access control support. The applications are built using our related tool.*

Index terms: *conceptual model, validation, model driven security, authorization constraints*

I. Introduction

Building up of models is the heart of design system. In many engineering disciplines it is off course a true fact and it is specially increasing in the case of Software Engineering. Model driven engineering which is a software development methodology aims to build models of different system from which artifacts such as code gets automatically generated. The abstraction and automation approach is one such model driven development approach that shifts software development from code to modeling. The model driven approach can deliver its full potential: that mean, it's capable of producing secure Graphical user interface for data management application. In particular the data management applications focus on CRUD action: that is CREATE, READ, UPDATE and DELETE ACTIONS. All applications contain these operations as their building blocks. For example dynamic website users used to create their own account, read information from it, update information, store and also delete the information and data. In such cases the data's of the users has to be secured against unauthorized access. For restricting unauthorized access, providing access control policies is standard approach. But if the access control policies are simple, it may be possible to formalize. In such cases protecting data's is a difficult task. In multi-tier system, usually Role based access control is built on application server. In contrast fine grained access control policies depends on two facts namely user credentials and satisfaction of certain constraints on persistence layers state. In these cases mentioned above, the authorization checks are encoded in appropriate places programmatically. But the true fact is these programmatic addition leads to some error, inflexibility and poor scale in performance. Moreover auditing and maintaining is difficult in these cases. This is because the authorization checks are spread throughout the code. And change in security policy requires code changes.

To overcome these, we propose an approach for model driven development in order to develop secure data management application. . Using our methodology a secure application is modeled using three communicable models namely:

1. A conceptual model defines the activities of conceptual design like finding classes, specifying attributes, operations.
2. A security model defines applications security. It provides an authorized access to data's on conceptual model.
3. Thirdly a graphical user interface model specifies applications GUI.

The main innovations are an expressive modeling language for modeling applications graphical user interface and a model transformation that lifts policy specified in security model to GUI model and validation process that validates the models completeness and correctness which further enhances the security of the application. Then finally a code generator for generating codes. It stipulates the access control policy and provides a fine grained access control in the application. We propose the idea of using model transformations to lift security policy. Here we improve and generalize the previous work and we provided updated presentation of modeling languages, the toolkit, and the example applications that we developed.

Let us expand on main generalization with respect to previous work. In our work we now check authorization before executing each events data action and we provide events with transaction semantics: either all of the data actions are executed in given order, or none of them are executed at all. Over all contribution of our work first, our methodology offers Model Driven Architecture's purported benefits [1], [11] for data management applications. By working with models, designers can specifically focus on application's data, behavior, security and presentation. Second the model transformation process leads to modularity and separation of concerns this avoids problems that are mentioned with fine grained hardcoded security policies that are difficult to maintain and audit. Finally our methodology supports validation process to ensure models correctness. Since the quality of the generated code depends on the quality of the source model, validation plays a vital role in our approach. This makes the Sculpture toolkit more powerful and securable for application development than the previously witnessed toolkit discussed in related work

The secure data management application is developed using sculpture tool. Using Sculpture application components are modeled and this model is transformed to deployable components. As a model driven development approach, sculpture tool represents the next logical step in application development. It aims at facilitating the automatic construction of a software solution from a high level domain specific specification. This approach seeks to promote productivity, maintainability, expressiveness, and to aid in the management of complexity by supporting higher levels of abstraction and the systematic reuse of domain specific assets.The objective is to show that one can use our approach and sculpture toolkit to generate non toy secure data management application with access control policy. The methodology and tool that we report on constitute a substantial further development in [6],[5].

Applications

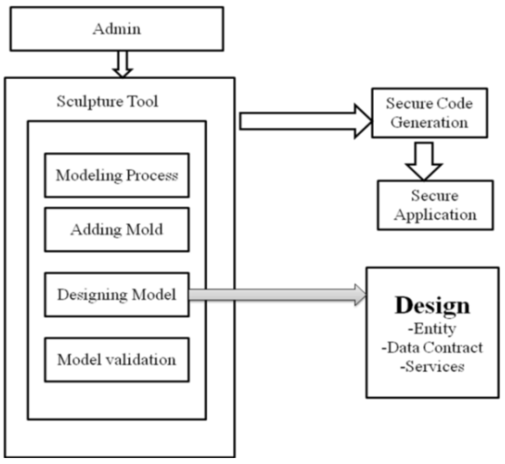
We report here on three web applications that we developed using Secure GUI tool known as Sculpture. The main motive is to show that our methodology and toolkit supports for developing highly secure web application in all aspects. We begin by briefly describing the application.

a) Volunteer Management (VM App).This is a web application for managing care centers volunteer management. Using this application, the program's coordinators can take actions such as: introduce new volunteers; create, edit, and modify the tasks; and propose these tasks to the volunteers, based on the volunteer's time availability and preferences. The access control policy stipulates, that is the volunteers are only authorized to edit their own personal information, such as their preferences and time availability, and to accept or reject their own task.
b) Electronic Health Record Management (e HRM App).This is a web application for managing Electronic health records. It allows users with the appropriate roles to: register new patients in hospital and assign to them clinicians (doctor, nurses, etc); retrieve patient information; register new nurses and doctors in hospital and assign them to respective wards; change doctors and nurses from one ward to another; and move patients to different practice. The access control policy stipulates, in particular, access to patient's highly sensitive records. These records are allowed only to access by the respective doctors.
c) Meal Service Management (MSM App) :This is a web application for managing meal service in students residential. Using this application, a resident can notify the administration weather he will have a meal at the residence's cafeteria, in which of the available time slots, and if he will bring a guest. A resident shall only edit his own meal selection and within a specific time window, which depends on the selected meal. Administrators can create new resident accounts, and list the meal requested foe each available time slot

This paper is organized as follows. System description is provided in Section II, Implementation of proposed work is described in Section III, and the algorithm is described in Sections IV and Finally in Section V experimental results are analyzed.

II. System Description:

The proposed work consists of the following system architecture. Here the modeling, design and secure application development process is described with the help of the figure below.



III. Implementation And Working

Sculpture is a Model-Driven Development code generation framework, ideal for creating and managing Enterprise Applications. With Sculpture the application components are modeled and then this model is transformed to deployable components. The sculpture model is added to the development environment. It contains diagram surface, toolbox, sculpture explorer, sculpture details and sculpture properties. It provides full tool support for modeling and secure code generation**.**

Adding Mold: Sculpture comes with a host of readymade molds. Sculpture uses guidance package for building own molds. The Sculpture Core Engine does nothing alone; just it needs to plug in to add the required molds

Mold Base: Is the base of all the other Molds; it contains all the common properties and activities that are shared among the Molds. All Molds must be inherited from it directly or from another Mold.

SQL Server Mold: Concerns in all activities related to Microsoft SQL Server. The reverse engineering engine parses the database and translates it to a model, so you can start the project from a database. Additionally, any updates in the database schema can reflect on the model easily without losing any metadata. The reverse engineering engine supports building entities from views to generate a script for the CRUD stored procedures, and database schema, so application design starts from the model, the entities should be designed, then generate the script of the database, and the script of the CRUD stored procedures.

LINQ Mold: This Mold generates LINQ to SQL entities, and Data Context, then generates repository classes for each entity with the default data access methods (Get All, Get by Id, Find, Save) are additional actions supported by our reported tool

Web Service Mold: This Mold does not generate any code, it just gathers the common properties for child web service Molds (ASMX, WCF); and if we are in need to generate any kind of web service (ASMX or WCF), we must include this Mold in the model

Designing Model: It includes designing entities, data contracts and services. Here reverse engineering tool is used that is supported by SQL Server Mold to generate entities and its associations from the database of application that we are generating. Data contacts are sometimes called data transfer object or value object. Using sculpture data contract can be generated from entities. Data contracts are the objects that will propagate to the higher level user interface. Instead of creating data contract for each entity manually, sculpture provides an option to generate data contracts from entities automatically**.** As a next step, services have to be generated from the data access method. Instead of creating services for each entity manually, sculpture provides an option to generate services from entities automatically**.** Then the model is generated with entities, their relation with their operations

Model Validation: Once the model is generated, it has to be validated. This validation process ensures the completeness and correctness of the model. Since the quality of the generated code depends on the quality of the source model. If the models do not specify the systems intended behavior, the generated system is not expected to do either. So it is essential to validate the model. Model driven development tool called sculpture supports for validation before performing automatic code generation for database actions. Sculpture provides validation rules for the most common errors found in the model

Secure Code Generation

Moreover, sculpture includes a code generator which produces code for the respective database actions and it gets attached to the respective project.

Application Generation

A three tier application is developed .The generated application is highly secure providing fine grained access.

1. Presentation tier (also known as front-end): Users access web applications through standard web browsers, which render the content dynamically provided by the application server.
2. Application tier: The toolkit generates Web Application. It process client requests and, generate content, which is sent back to the client for rendering.
3. Persistence tier (also known as data tier or back-end): The generated application manages information stored in a database. This raises the security level in database.

IV. ALGORITHM

Model generation algorithm describes the generation of completely secure model which then forms a base for secure code generation**.**

```
1. Model Generation (Database object, entities, data contracts, services, and model)
2. Initialize DBO to set of Data Base Object
3. For each of the DBO
4. Entity created ←Selected object
5 Add to model ←created entity
6 End For
7 For each of the entities in model
8 Generate data contract from entities
9 Add to model ← data contract
10 End For
11 For each entity
12 Add services to entity
13. Generate the services
14 Add to the model ←services
15 End For
16 generate Complete Model
```

Algorithm Description

As a first step, initialize DBO to set of Data Base Object. Then using Database object selector form, select each of the database objects, for which entity needs to be created.

Entity created ←Selected object

The created entity will gets added in the model. Then for each of the entities which are added to the model, generate data contracts from the respective entity. The data contracts are then added to the model.

Add to model ← data contract

The data contact establish mapping between the entity and its corresponding data contract. Then for each of the entities in the model services that are required are added and finally services are generated .The generated services are added to the model. There exist relation between the entity and the corresponding service.

Add to the model ←services

There exist relation between the entity and the corresponding service. Then the complete model is generated. The generated model has association between entity, its data contract and with its corresponding services

V. Experimental Result

The presented approach enables a gradual approximation to system modeling towards data management application by providing expressive modeling language for applications graphical user interface and behavior, and it promotes model to model transformation that enhances the security policy. We can show that it is possible to generate automatically complete deployable application.

Our methodology is supported by Sculpture toolkit. Real world application can be built using this toolkit. It shows its full potential. In our work, the tools robustness is witnessed. The tool supports for validation process, this shows that our work is a step ahead in model driven development paradigm. Moreover our proposed methodology provides improved model editors, which is not supported by other tools and methodology. Earlier tools and approaches contain simple model editors, so improved model editors which is supported by our related tool provides further enhancement to our proposed approach. Nevertheless, there is still much work ahead to turn toolkit support for GUI’s running on different platform like mobile devices. Further Sculpture tool helps in developing highly secure web application by handling privacy policies efficiently

VI. Related Work

Various extensions of model driven approach have been presented over the past 15 years. Among these, our work is closely related to UWE [7], [8],[9], and [4].UWE provides an augmented assistance to UWE modeler. These models have to be appropriately refined as discussed in [4].

Busch [9] extends UWE to combine with secure UML for enhancing security. But it doesn't work because this does not support model transformation and validation process which is supported by our proposed work. Similar to secure GUI, the ZOOM [14] approach allows modeler to specify widgets, their respective events and associated actions. Moreover ZOOM does not support for code generation. The approaches presented in [3],[2],[10] does not have language for modeling graphical user interface. In [10] the GUI's are derived from prototypical scenarios. There is other related work that falls between the two extremes of full GUI modeling and full GUI derivation. Both the OO-method[12],[13] and webML [14],[15] support building GUIs using UI-patterns. These approaches have the advantage of reducing the time required for modeling.

References

[1]. A.Kleppe, W.Bast, J.B.Warmer, and A.Watson,MDA Explained: The Model Driven Architecture-Practice and Promise. Addison-Wesley, 2003

[2]. A.M.R.da Cruz and J.P Faria," A metamodel-based Approach for Automatic user Interface Generation, "proc.13th Int'l Conf. Model driven Eng. Languages and systems:part1(MODELS'10),Oct.2010,pp.256-270

[3]. A.M.R.da Cruz, "Automatic Generation of User Interfaces from Rigorous Domain and Use Case Models," PhD disssertation, Univ.do Porto, Sept.2010.

[4]. C.Kroiss, N.Koch, and A.Knapp, "UWE4JSF: A Model-Driven Generation Approach for Web Applications," Proc. Ninth Int'l Conf.Wenb Engg. pp.493-496, (ICWE'09), 2009

[5]. D.A.Basin,M.Clavel,M.Egea, and M.Schlapter," Automatic generation of Smart,security aware GUI Models,"Proc.Second Int'l Symp.Eng .secure Software and systems(ESSoS'10),Feb.201,pp.201-217

[6]. D.A. Basin, M. Clavel, M. Egea, M.A.G. de Dios, C. Dania,G. Ortiz, and J. Valdazo, "Model-Driven Development of Security-Aware GUIs for Data-Centric Applications," Foundations of Security Analysis and Design VI, pp. 101-124, Springer, 2011.

[7]. [7] H.Baumeister, N.Koch, and L.Mandel, "Towards a UML Extension for Hypermedia Design," Proc. Second Int'l Conf. Unified Modeling Language: Beyond the Standard, pp.614-629 (UML"99), 1999.

[8]. M.Busch and N.Koch,"MagicUWE-A Case tool plugin for Modeling Web Applications," Proc.Ninth Int'l Conf.Web Engineering pp.505-508, (ICWE'09), 2009

[9]. M.Busch,"Integration of Security Aspects in Web Engineering,"master's thesis,Inst.fur Informatik,Ludwig-Maximilians-Univ.,2011

[10]. M.Elkoutbi, I.Khriss,and R.Keller, "Automated Prototyping of User Interfaces Based on UML Scenarios," Automated Software Eng., vol.13, pp.5-40,2006

[11]. Object management Group," Model Driven Architecture Guide v.1.0.1," technical report, OMG, http;//www.omg.org/cgi-bin/doc?omg/03-06-01,2003.

[12]. O.pastor,J.Gomez,E.Insfran, and V.Pelechano,"The OO-Method Approach for Information Systems modeling:From Object – Oriented Conceptual Modeling to AutomatedProgramming,"InformationSystems,vol.26,no7,pp.507-534,2001

[13]. P.J.Molina , S.Melia and O.Pastor "Just-UI: A User Interface Specification Model," Proc.Int'l Conf.Computer- Aided Design of User Interfaces, pp.63-74 (CADUI'02),2002

[14]. S.Ceri and P.Frternali,"The web Modeling Language-WebML,"http;//www.webml.org,2003.

[15]. Web models Company,"Web Ratio-You Think,You Get,"http;//www.webratio.com,2010.

[16]. X.Jia, A.Steele, L.Qin, H.Liu and C.Jones, "Executable Visual Software Modeling- The ZOOM Approach, "Software Quality Control , vol.15 pp.27-51, March 2007.