Theoretical Analysis on Agile Security Architecture Model

M.Upendra Kumar
Associate Professor CSE MGIT Hyderabad India

Abstract: *In this Paper, discussion of theoretical analysis of designing of MDA extended Agile Modelled Layered Security Architectures is provided. Initial case study validations on simple secure Web Services design using proposed agile security model using pair programming and test driven development.*

Keywords— *Theoretical Analysis, Agile Modelling, Security Architecture*

1 INTRODUCTION

THEORETICAL ANALYSIS

1 MDA EXTENDED AGILE SECURITY ARCHITECTURES USING LAYERED DESIGN

Model Driven Development or Model Driven Architecture or Model Driven Software development is an innovative modeling paradigm used by most of the web applications. Here we designed models with a focus of scalability, reusability at the overall system level. Here we have a layered approach having Computational Independent Models (CIM) for requirements gathering, Platform Independent Models (PIM) for system functionality irrespective of execution platform, Platform Specific Models (PSM) is a combination of PIM with a specification of implementation on a particular platform. Essentially PIM gets transformed into PSM with a process of mapping at various layers levels.

MDA is an iterative process and also incremental process which provides a feedback mechanism in between development of architectures and their applications. MDA can be extended to Agile Modeling because of its property such as runnable software acceptance tested by all the stakeholders including the users which are a common goal for MDA. Complex software requires is MDA extended agile techniques for requirements management. The micro activities of MDA process can be integrated and extended by the agile methodology. Agile Teams assigns roles as its builds on the performance of individuals. MDA focused on importance of models domain architectures is a concept of extending MDA are integrating Agile for both modeling and implementation of application. Important Agile Techniques are eXtreme Programming (Pair Programming), Test Driven Development etc. Model is a simplification of anything that needs to be modeled, it has a purpose or intent and it focus on the relevance of its purpose.

Figure 1 provides important models and layers in MDA. (CIM, PIM, PSM, PSI Platform Specific Implementation). As discussed earlier CIM nature focus is on functionality of the software. PIM focus is on the important components of a system and is respective and without any details of its internal platform executable environment. PSM focuses on platform specific execution environment. PSM execution is done as PSI. The important artifact dealt by CIM is a use case. Use cases acts as significant artifact for requirements. The important artifact in PIM is a class. It performs the required functionality by using collaborations and interaction diagrams and behavioral diagrams as specified by UML. Transformation from CIM to PIM will be performed by an expert modeler having good domain expertise of the application being developed. PSM deals with platform specific requirements and functionality of the system for achieving optimized performance of the system. Generally design solutions are obtained from refinement of analysis solutions. Transformation from PIM to PSM happens by applying required design patterns or security patterns to PIM. PSM again in turn is created at 3 layers of abstraction, architectural level, mechanistic level and detailed level as discussed in next figure. PSI is all about the code implemented at PSM level for a specific PIM for a particular platform.

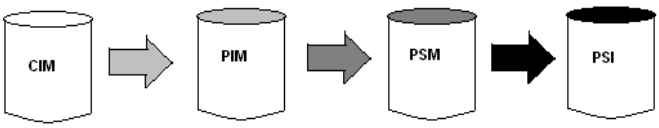

Fig 1 Key models in MDA

Figure 2 provides levels of design. High level design layer architectural design is done where scope is the entire software system being built. Middle level design mechanistic design is done where scope is collaboration. Low level design called detailed design is done whose scope of operation is a class.

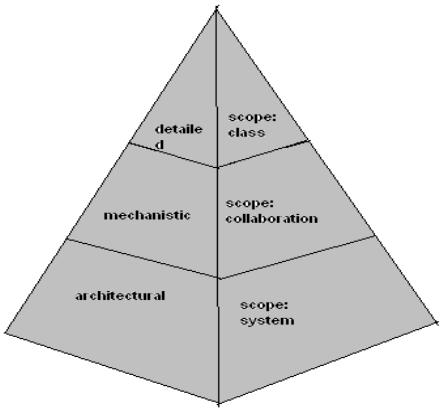

Fig 2 Levels of design

Figure 3 provides meta model transformations. Here we create PIM and transform them into PSM. Meta model is model or artifact which has information about other model artifacts. Model rules specified in the meta model has access to source language to PIM meta model and target language PSM meta model, based on which transformation can be done.

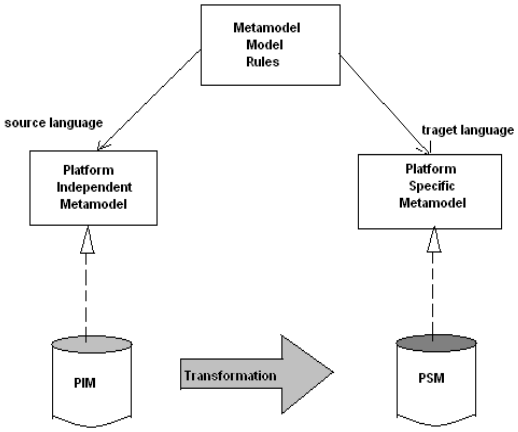

Fig .3 Meta model Transformations

Figure 4 provides MDA transformations of models. PIM can be transformed into PSM by a two step process by creating an intermediate Marked PIM (obtained from marking the significant elements of PIM based on model rules used for mapping).

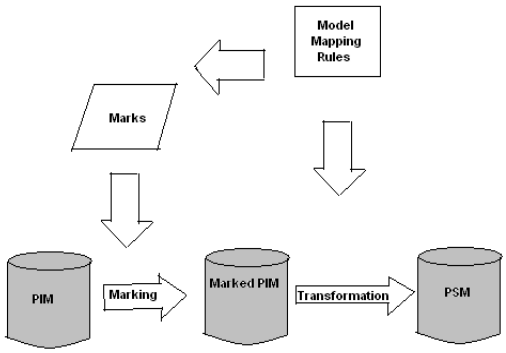

Fig 4 MDA Model Transformations

Figure 5 provides design pattern model transformations. Marked PIM can be transformed to PIM by modeling and transforming design patterns obtained from design patterns rules of mapping of MDA Transformations can also be done using UML, SysML etc. UML is a De facto Modeling tool used for web engineering applications. SysML is an extension of UML to specific application domain systems. UMLSec supports stereotypes and constraints to incorporate security, mechanisms and technologies by extending the core UML. Secure UML extends UML for extending security authorizations and authentications like RBAC (Role Based Access Control) and extensively uses Object Constraint Language (OCL) which uses stereotypes like user, role, permissions etc. Finally benefits of MDA include high portability, better re-usability, and abstraction from platform specific technologies.

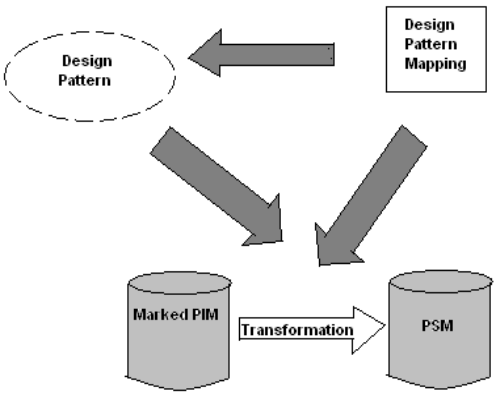

Fig 5 Design Pattern Model Transformations

2 LAYERED APPROACH AND SECURITY REQUIREMENTS

Agile Modeling popular development methods are eXtreme Programming (XP), importantly using Test Driven Development (TDD). MDA extended agile methodology uses a layered approach for security requirements at various layers like Source Code Model, Idiom level model, Designer level.

3 INITIAL CASE STUDY VALIDATIONS – NGSWEA

Based on the theoretical analysis performed in this paper, initial web services are designed using MDA extended agile modeling, using eXtreme programming, with pair programming and TDD.

Table 1 provides Web Services Security Threat Framework.

Table 1 WEB SERVICES SECURITY THREAT FRAME WORK

Layers of Web Services	Threats and Attacks
Web Services Transmission	WS-Routing Security
Web Services core processor	XML Parsing Attacks
Web Services Operation	Authentication and Certification
Web Services Code	Format String

Figure .6 provides Class Diagram for Web Services Application Agile Design. For Web Service authentication use logins with his username and password credentials by contacting the web service administrator after authentication validation and the authorization of privileges of the user, the user gets the facility to access the service.

Figures 7 and 8 provides Class diagram and sequence diagram for Role Based Access Control (RBAC) for web services security policies. Each user is assigned a role like Administrator or security personnel who will dictate the access control mechanisms, the job process run by the user at the administrator of web service. Administrator initially creates user and assigns a role which will later on execute a job for providing authorization.

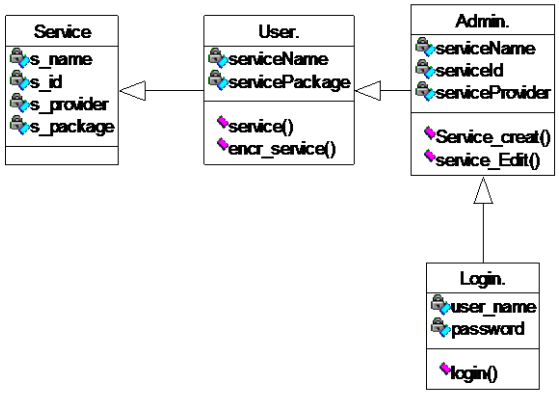

Fig 6 Class Diagram for Web Services Application Agile Design

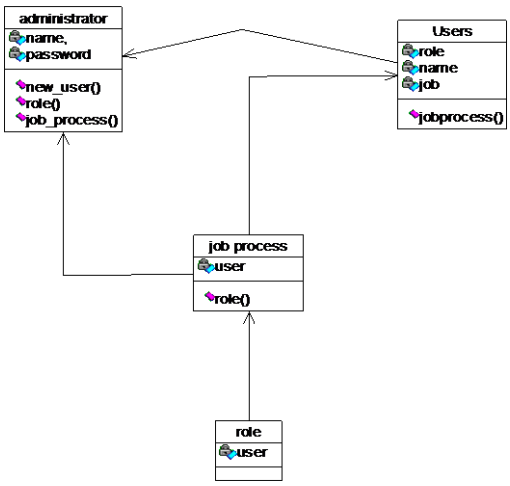

Fig 7 Class Diagram for RBAC Web Services Security Policies

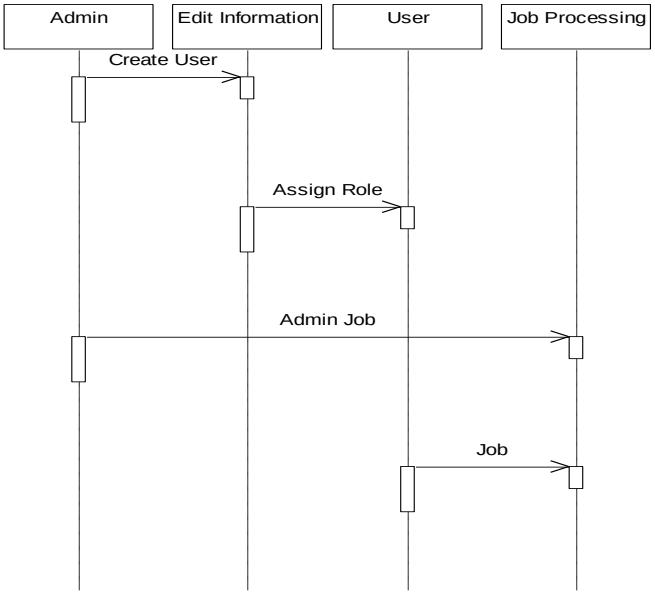

Fig.8 Sequence Diagram for RBAC Web Services Security policies

4 SUMMARY AND CONCLUSION

In this paper theoretical analysis of designing of MDA extended Agile Security Architectures using layered design is provided. Initial case study validations on simple secure Web Services design using proposed agile security model using pair programming and test driven development. In the next paper Design and Implementation of our proposed model for Agile Modeled Layered Security Architectures for Security Requirements. We had validated on case study Next Generation Secure Web Engineering Applications, using Agile Modeling for Web Services, Web 2.0 Services Authentication and Authorization.

REFERENCES

[1] Maria Jose Escalona and Gustavo Aragon,"NDT. A Model-Driven Approach for Web Requirements", IEEE Transactions on Software Engineering, Vol. 34, No. 3May/June 2008, , PP. 377-390.

[2] Anders Mattsson, Bjorm Lundell, Brian Lings, and Brian Fitzgerald, "Linking Model-Driven Development and Software Architecture: A Case Study", January / February 2009, IEEE Transactions on Software Engineering, vol. 35, no. 1, PP. 83-93.

[3] Robert France, Bernhard Rumpe, "Model driven Development of Complex Structure: A research Roadmap", PP. 1-12.

[4] Huy Tran, Taid Holmes, Uwe Zdun, Schahram Dustdar, "Using Model Driven Views and Trace Links to relate requirements and architecture: A Case Study", PP. 1 – 25

[5] Alistair Cockburn, Jim Highsmith, "Agile Software Development: The People Factor" IEEE Xplore, November 2001, PP. 131 – 133.

[6] Paul Clements, James Ivers, Reed Little, Robert Nord, Judith Stafford, "Documenting Software Architectures in an Agile World", Technical Note, CMU/SEI-2003-TN-023, July 2003, PP. 1-24.

[7] Lowell Lindstrom and Ron Jeffries, "Extreme Programming and Agile Software Development Methodologies", Information Systems Management 2004, PP. 1-4.

[8] Jeff Langr,"Agile Java Crafting Code with Test Driven Development", Prentice Hall, ISBN 0-13-148239-4, 2005, PP. 116.

[9] John Hunt," Agile Software Construction", Springer London, ISBN 1-85233-944-6, 2006, PP. 5.

[10] Martin C Robert, Martin Micah," Agile Principles, Patterns, and Practices in C#", Prentice Hall, ISBN 0-13-185725-8, 2006, PP. 12.