Sudan University of Science and Technology

College of Graduate Studies

Master of Computer Science

Model Driven Architecture Based Development Method for Effective Integrated Service Oriented Architecture

منهجية معمارية اشتقاق النماذج لتطوير معمارية الخدمة الموجهة المتكاملة

A Thesis Submitted in Partial Fulfillment of the Requirements of Master Degree in Computer Science

Prepared by:

Elaf Mohamed.Alyamani Abd.Alla

Supervisor:

Dr.Abdelhamid Mansor

May 2018

ABSTRACT

Service Oriented Architecture (SOA) and Model Driven Architecture (MDA) have emerged as a major evolutionary step in enterprise integration. Significant advances have been seen in SOA and MDA. Several literatures have been published about the web service being the solution to interoperability challenges within the SOA framework. The aim of this literatures was to find out, if the SOA infrastructure will actually solve these challenges. Based on Unified Modeling Language (UML), the MDA supports applications over their full lifecycle starting with design and moving into coding, testing, and deployment, maintenance, and eventually to evolution to new platform. In this research, a method to integrate services is introduced to ensure productivity and flexibility and automation of service integration by using model driven architecture, a method illustrated using "Car's Robbery Report Service" as case study. As a result of using this method, a web services integration developed in productive way, and WSDL code generation using automated code generation tool in small time and lower cost.

المـســتـــخـــلـــص

معمارية الخدمة الموجهة (SOA) ومعمارية اشتقاق النماذج (MDA) ظهرا كخطوة تطورية رئيسية في دمج المشروع. التقدم الهام تمت رؤيته في معمارية الخدمة الموجهة ومعمارية النمذجة. العديد من الدراسات تم نشرها حول ان خدمات الويب تكون الحل لتحديات التكاملية ضمن إطار معمارية الخدمة الموجهة. الهدف من هذه الدراسات ان تكتشف إذا كانت بنية معمارية الخدمة الموجهة ستحل هذه التحديات في الحقيقة. استناداً على لغة النمذجة الموحدة (UML)، تدعم معمارية اشتقاق النماذج (MDA) التطبيقات على دورة حياتهم الكاملة بدأً بالتصميم وتنتقل إلى الكود، الاختبار، النشر، الصيانة، أخيرا إلى التطور إلى بيئة جديدة. في هذا البحث تم تقديم طريقة لدمج الخدمات ولضمان الإنتاجية والمرونة في دمج الخدمات بطريقة آلية باستخدام معمارية اشتقاق النماذج (MDA)، تم توضيح المنهج باستخدام "خدمة تقارير سرقة السيارات" كدراسة حالة. كنتيجة من استخدام منهج معمارية اشتقاق النماذج (MDA) تم دمج خدمات الويب بطريقة فعالة، وتم توليد كود لغة وصف خدمة الويب (WSDL) بطريقة آلية في مدة زمنية بسيطة وتكلفة أقل.

ACKNOWLEDGEMENT

First of all, I should show my thankfulness for Allah's blessings and help, that without him nothing of this would be achieved. I would like to thank my thesis advisor Dr. Abdalhamid Mansour. He steered me in the right the direction whenever he thought I needed it.

would also express my very profound gratitude to my parents, and my brothers, and my sisters, and to my friends for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ATL	ATLAS Transformation Language
BPM	Business Process Modeling
BPMN	Business Process Model Notation
CIM	Computation Independent Model
EA	Enterprise Architecture
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
GReAT	Graph Rewrite And Transformation language
IBM	International Business Machines
IS	Information System
JTL	Janus Transformation Language
M2M	Model-to-Model
MDA	Model Driven Architecture
MOL	MOdel transformation Language
NFR	Non Functional Requirement
OMG	Object Management Group
PIM	Platform Independent Model
PSM	Platform Specific Model
QVT	Query/View/Transformation
SOA	Service Oriented Architecture
SOAML	Service Oriented Architecture Modeling Language
SOAP	Simple Object Access Protocol
SOEA	Service Oriented Enterprise Architecture
UDDI	Universal Description, Discovery, and Integration
UML	Unified Modeling Language
W3C	World Wide Web Consortium

WSDL	Web Service Description Language
XML	Extensible Markup Language
XSD	XML Schema Definition

CHAPTER 1

INTRODUCTION

1.1 Background of the Problem

From a technological point of view, in the last years has grown the number of projects based on software services to improve flexibility and integration. This kind of development, founded on software services, changes the traditional approach and allows implementing applications with small software components [1].

Service-Oriented Architecture (SOA) [2] is a software architecture consisting of a collection of loosely coupled services that communicate with each other through open standard interfaces. The World Wide Web Consortium (W3C) refers to SOA as "a set of components which can be invoked, and whose interface descriptions can be published and discovered" [3].

Model Driven Architecture (MDA) [4] approach is defined by the Object Management Group (OMG) [5]. MDA provides an approach for deriving value from models and architecture in support of the full life cycle of physical, organizational and I.T. systems. The MDA approach represents and supports everything from requirements to business modeling to technology implementations. By using MDA models, we are able to better deal with the complexity of large systems and the interaction and collaboration between organizations, people, hardware, software [6]. The practical benefits of SOA are widely recognized, relatively easy to describe, but more challenging to implement.

Using a model-driven approach, enterprises can define business models without consideration for the underlying technical implementations. However, SOA and MDA are distant in terms of the way they address the issues surrounding modeling. SOA focuses on the stereotypical roles of models based on separation of concerns. MDA focuses on levels of abstraction, defining the role of models within a process [7].

1.2 Problem Statement

When systems are integrated there are many problems are expected to arise, these problems may affect non-functional requirements (such as interoperability, portability, and compatibility). The current infrastructure for information integration is fragile, expensive, and difficult to maintain. And spend a huge amount of time and money to solve SOA problems.

(i) Interoperability**:** The ability of two or more systems (or components) to exchange and subsequently use that information. So interoperability is concerned with the ability of systems to be communicated. Moreover, it requires that the communicated information can be understood by the receiving system. However, it is not concerned with whether the communicating systems do anything sensible as a whole.

(ii) Compatibility: is concerned with the ability of more than one system or components to perform their required functions while sharing the same environment. Such components (or systems) do not need to communicate with each other, but simply be resident on the same environment. so that compatibility will not be concerned with interoperability.

(iii) Portability: is concerned with the ease of moving components or systems among different environments (hardware and/or software environments).

1.3 Research Question

According to state problem, research question:

"How to integrate services using model driven architecture?"

1.4 Objectives of the Research

The objectives of this research are:

(i) To evaluate the current state of the art on service oriented architecture approaches.

(ii) To propose a method to develop an effective service oriented architecture.

(iii) To validate the applicability of the proposed method.

1.5 Scope of the Research

This research focus on integration of systems based on service oriented architecture (SOA) technique using model driven architecture (MDA). focusing on the general process of model driven architecture (Computation Independent Model, Platform Independent Model, Platform Specific Model, and Code Generation) to modeling SOA illustrated using "Car's Robbery Report Service" as case study.

1.6 Importance of the Research

Service oriented architecture is used to achieve loosely coupling in enterprise application integration by providing everything work using services. The services are independent on each other so they lead to achieve loosely coupling in the electronic business. It promotes the idea of assembling application components into a network of services that can be loosely coupled to create flexible, dynamic business processes and agile applications that span organizations and computing platforms [8].

MDA is a new architectural approach that provides companies with the tools necessary to integrate all the various middleware technologies (such as CORBA, EJB, XML, SOAP and .NET). MDA addresses the complete life cycle of designing, implementing, integrating and managing applications and data using open standards. MDA provides an architecture that assures portability, cross platform interoperability, platform independence, domain specificity, and productivity [9].

MDA unifies and simplifies modeling, design, implementation, and integration of applications.

1.7 Structure of the Research

Chapter 2 (Literature Review) This chapter introduces the general concepts in service oriented architecture (SOA) and general concept of model-driven architecture (MDA).

Chapter 3 (Research Methodology) This chapter presents the research methodology, operational framework, assumptions and limitations.

Chapter 4 (Proposed Method) This chapter introduces the proposed method.

Chapter 5 (Results Evaluation) This chapter presents the result of using the proposed method to developing SOA.

Chapter 6 (Conclusion) This chapter presents the conclusion and future work.

CHAPTER 2

LITERATURE REVIEW

This chapter is a review for the existing literature on service oriented architecture and model driven architecture to understand the concepts that is related to the research.

2.1 Services

Services are reusable components that represent business or operational tasks, such as customer lookup, credit card validation, weather lookup, or line-of-sight calculation. Reusable is a key element of this definition because it is what enables the creation of new business and operational processes based on these services. Services expose their capabilities via well-defined, standard service interfaces. In a service-oriented environment, service interface definitions are available in some form of service registry [10].

The services in SOA are governed by a set of key principles that include loose coupling, autonomy, abstraction, reusability, compos ability, statelessness, discoverability and adherence to a service contract. These principles enable the services to evolve independently. SOA results in the creation of business solutions that consist of inherently interoperable services. An XML-based (Extensible Markup Language), vendor-neutral communications framework established by web services-driven SOA enables cross platform integration and intrinsic interoperability among the services [11].

2.2 Service Oriented Architecture (SOA)

Service Oriented Architecture (SOA) is a software architectural style that uses services as the main building component. A service, as a software component, is a mechanism to enable access to one or more capabilities [12].

SOA sanctions companies to reuse available components/services and build flexible systems that implement changing business processes expeditiously. SOA guarantee efficacious business-IT alignment, amended business suppleness and lower integration costs through greater interoperability and reuse of shared business services [13].

In fact, SOA is an approach for designing and developing Information System (IS). By this approach, design, development, and implementation of IS are possible due to the web technology. However, interoperability of services in SOA is not limited to web services, the web services are the most suitable technology for successful SOA. In IS based on SOA, information source and business functions can be converted into modular services units for control and management. Moreover, these units are shared over a network and collaborated in enterprise information system [14].

SOA's paradigm is widely adopted for the development of software systems through the dynamic composition of network accessible loosely-coupled services [15]. Service-oriented architecture (SOA) is a paradigm for the realization and maintenance of business processes that span large distributed systems. It is based on three major technical concepts: services, interoperability through an enterprise service bus, and loose coupling [2].

(i) A service is a piece of self-contained business functionality. The functionality might be simple (storing or retrieving customer data), or complex (a business process for a customer's order). Because services concentrate on the business value of an interface, they bridge the business/IT gap.

(ii) An Enterprise Service Bus (ESB) is the infrastructure that enables high interoperability between distributed systems for services. It makes it easier to

distribute business processes over multiple systems using different platforms and technologies.

(iii) Loose coupling is the concept of reducing system dependencies. Because business processes are distributed over multiple backends, it is important to minimize the effects of modifications and failures. Otherwise, modifications become too risky, and system failures might break the overall system landscape. Note, however, that there is a price for loose coupling: complexity. Loosely coupled distributed systems are harder to develop, maintain, and debug.

2.3 The SOA Life Cycle

International Business Machines (IBM) clients have indicated that they think about SOA in terms of a life cycle as shown in figure [2.1] [16]. They start in the model phase by gathering business requirements and designing their business processes. After processes are optimized, they implement them by assembling new and existing services to form these business processes. They then deploy these assets into a highly secure and integrated services environment. After the business processes are deployed, IBM clients manage and monitor these business processes from both an IT and a business perspective. Information gathered during the manage phase is fed back into the life cycle to enable continuous process improvement. Underpinning all of these life-cycle stages are governance and processes that provide guidance and oversight for the SOA project [16].

2.4 Web Service

Web services are the most suitable technology for successful SOA, although it is not limited to web services. For instance, Common Object Request Broker Architecture (CORBA) and Message-oriented Middleware systems such as the IBM Message Queue Series and Java Messaging Service (JMS) can be applied, but web services in comparison with others have more loosely coupled interfaces [14].

The Web services standards define XML-based protocols and interface descriptions that enable services to interact. Basic Web services standards include Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI) [17].

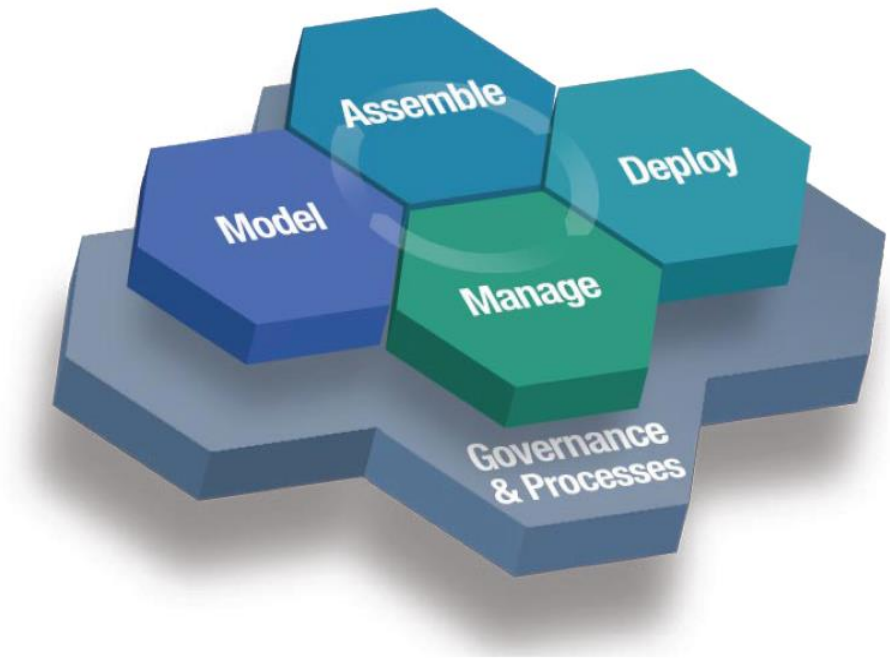

Figure [2.1] The SOA Life Cycle

2.5 SOA Approaches

An approaches for modeling and integrating Service Oriented Architectures.

2.5.1 Business Process Modeling and BPMN

Business Process Modeling (BPM) [18] represents and describes the activities and interactions in a business process of a company. BPMN (Business Process Model Notation) is a graphical notation for modeling business process which appeared to describe business process in a unified way. The principal goal of BPMN is to provide a notation that is readily understandable by all business users. This includes the business analysts who create the initial drafts of the processes to the technical

developers responsible for implementing the technology that will perform those processes [19].

2.5.2 Model Driven Architecture (MDA)

OMG is promoting MDA as a way to develop systems that more accurately satisfy customers' needs, and that offer more flexibility in system evolution. The MDA approach provides a comprehensive interoperability framework for defining interconnected systems [7].

2.5.3 Service-Oriented Architecture and SOAML

The Object Management Group (OMG) proposed Service Oriented Architecture Modeling Language (SoaML) [20] in 2009 for representing SOA artefacts using Unified Modeling Language (UML) as a core modeling standard. Moreover, a meta-model and a UML profile are provided in SoaML for the specification and design of service to SOA (meta-model for modeling the requirement for a service and UML for specifying services) [14].

2.6 Model Driven Architecture (MDA)

OMG's MDA unifies and simplifies modeling, design, implementation, and integration of applications – including large and complex ones – by defining software fundamentally at the model level, expressed in OMG's standard Unified Modeling Language (UML). An MDA application's base model specifies every detail of its business functionality and behavior in a technology-neutral way [4]. One fundamental aspect of MDA is its ability to address the complete development lifecycle, covering analysis and design, programming, testing, component assembly as well as deployment and maintenance [21].

As an OMG process, the MDA represents a major evolutionary step in the way the OMG defines interoperability standards. For a very long time, interoperability had been based largely on CORBA standards and services. Heterogeneous software systems interoperate at the level of standard component interfaces. The MDA process, on the other hand, places formal system models at the core of the interoperability problem. What is most significant about this approach is the independence of the system specification from the implementation technology or platform. The system definition exists independently of any implementation model and has formal mappings to many possible platform infrastructures (e.g., Java, XML, and SOAP) [22].

MDA goes through different levels of abstraction, from the very business model (the so-called CIM, Computation Independent Model) to the final executable code. In the middle, a PIM (Platform Independent Model) and a PSM (Platform Specific Model) are designed. The CIM describes the context and requirements of the system but does not address its structure or processing. The PIM considers the operational capabilities of the application in a platform-neutral way showing only those parts that can be abstracted out of any platform. A PSM adds to a platform independent model the details relating to the use of a specific platform or set of platforms [12] as shown in figure [2.2].

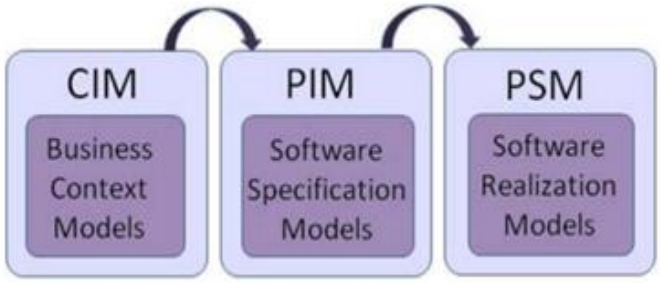

Figure [2.2] Model Driven Architectures Levels

2.6.1 Computational Independent Model (CIM); A CIM is also often referred to as a business or domain model because it uses a vocabulary that is familiar to the subject matter experts (SMEs). It presents exactly what the system is

expected to do, but hides all information technology related specifications to remain independent of how that system will be (or currently is) implemented. The CIM plays an important role in bridging the gap which typically exists between these domain experts and the information technologists responsible for implementing the system [21].

2.6.2 Platform Independent Model (PIM); ideally, software application design should be appropriate for all type of execution platforms (different operating systems, hardware, network protocols, programming languages, etc.) To achieve this Platform Independent Model (PIM) has been defined which provides a formal definition of the functionality of system without addressing any specific operating platform. A platform independent modeling language, such as UML, is used to design PIM model. The PIM model defines data, dependencies and architectural realizations. The model elements should provide enough information to make accordant code generation possible in next step. Based on platform independent model [23].

2.6.3 Platform specific model (PSM); model of the technical platforms. It mainly serves as a base for generating an executable code on the chosen technical platform. It indicates how the product will be used on these platforms. A good PSM must incorporate enough features and concepts (data types, classes, interfaces, patterns, etc.) of the platform chosen to make the code generation easy [24].

One of the great advantages of creating models is the ability to manipulate them to produce outputs, thus saving time and reducing the possibility of errors. Enterprise Architect implements Model Driven Architecture (MDA) transformations using a flexible and fully configurable template system. The templates act as instructions to a machine that takes a model as input and transforms it to a more resolved model as output. The input could be a large and complex model or a single

element and one input model could be transformed to a variety of output models [25] as shown in figure [2.3].

Model-Driven Architecture (MDA) is a best choice to address how SOA and Web services should be designed, developed and integrated in order to achieve integration. MDA provides specifications for an open architecture appropriate for the integration of systems at different levels of abstraction and through the entire information systems life-cycle [3]. MDA lets us design Web services at a more abstract level than that of technology-specific implementations. Systems built using MDA exhibit more flexibility and agility in the face of technological change—as well as a higher level of quality and robustness, due to the more formal and accurate specification of requirements and design [26].

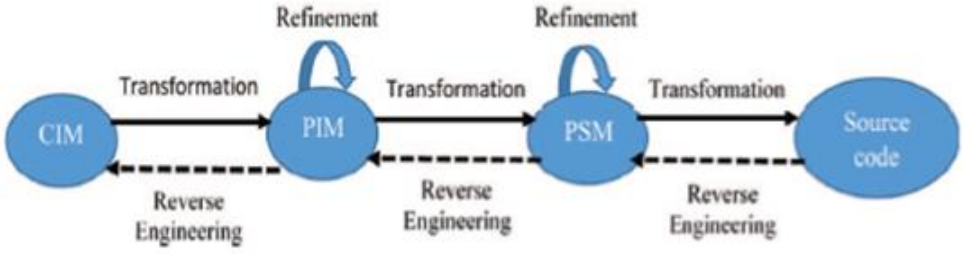

Figure [2.3] MDA Model Transformation Process

2.7 Related Work

Enterprise Architecture (EA) has become an important means to acquire and maintain knowledge about the structure and behavior of the enterprises and to develop the required IT systems. Two main goals in architecture - high flexibility and low complexity- are addressed in a new architectural style called Service Oriented Architecture (SOA). In this paper, discussed an approaches to have SOA models in the enterprise architecture framework. Based on an analysis of these approaches, proposed an extension to the Zachman Framework called SOEAF, in order to provide the framework with the capability to include service oriented artifacts representing services aspect of the enterprise. Then will proposed a Model Driven Approach

(MDA) to Service Oriented Enterprise Architecture (SOEA), in which, MDA concepts, standards and tools help us achieve a semi-automated service-based analysis and design of the enterprise IT systems. The proposed approach can be very beneficial for increasing efficiency to reach enterprise-wide goals [27].

On Demand Business concept, from IBM, shows how to coordinate the transformation of enterprise organization and its technological infrastructure to bridge the gap between business and technology. This problem appears in most of the information systems. This concept is based on Service Oriented Architecture (SOA) to improve the connection between business services and software services. If companies can reach this On Demand Business environment they could improve the software development process by applying standards like Model Driven Architecture (MDA), to separate business logic from software and technological platforms, and Business Process Management (BPM) to define business processes as Computation Independent Models (CIM). In this article proposed a new MDA approach that combines BPM, SOA and the On Demand Business to improve the initial phase of the software development process. Starting on the creation of business processes models, classified like CIM, can associated them to initial software models, based on software services, classified like Platform Independent Models (PIM) [1].

2.8 Summary

In this chapter introduced service-oriented architecture concept and brief description of service oriented architecture approaches, and comparative between these approaches, from this comparative evaluation concluded that MDA to integrate SOA services have main benefits such as increasing productivity and automation of integration services, Model-driven architecture concept has introduced and life cycle model to develop and integrate systems using MDA has mentioned, and an approach to integrate service-oriented services which we will use them to integrate the proposed method.

CHAPTER 3

RESEARCH METHODOLOGY

This chapter presents methodology to be adopted in continuing this research. Research design and procedures, operational framework, assumptions and limitations and research schedule are included.

3.1 Research Design and Procedures

Four main phases describe the research procedure: problem identification, development, validation and results. Figure [3.1] illustrates the methodology of research.

3.1.1 Problem Identification

This stage identifies the problem by talking about literature review in chapter 2 that provides the theoretical concepts of service oriented architecture (SOA), model driven development. Moreover, the chapter investigated on the different life cycle models that used in the process of developing service oriented. A comparative evaluation between those models was introduced.

3.1.2 Development

This section contains:

(i) Selected approaches based on literature review introduced in chapter 2.

(ii) The comparative evaluation base on approaches mentioned in chapter 4.

(iii) The proposed method based on the previous studies and research which will be introduced in chapter 4.

3.1.3 Validation

This section aims to make sure that the developed method satisfies the desired results or not, validation operation has been mentioned in chapter 5.

3.1.4 Results

This section contains the result of apply the proposed method discussed in chapter 5.

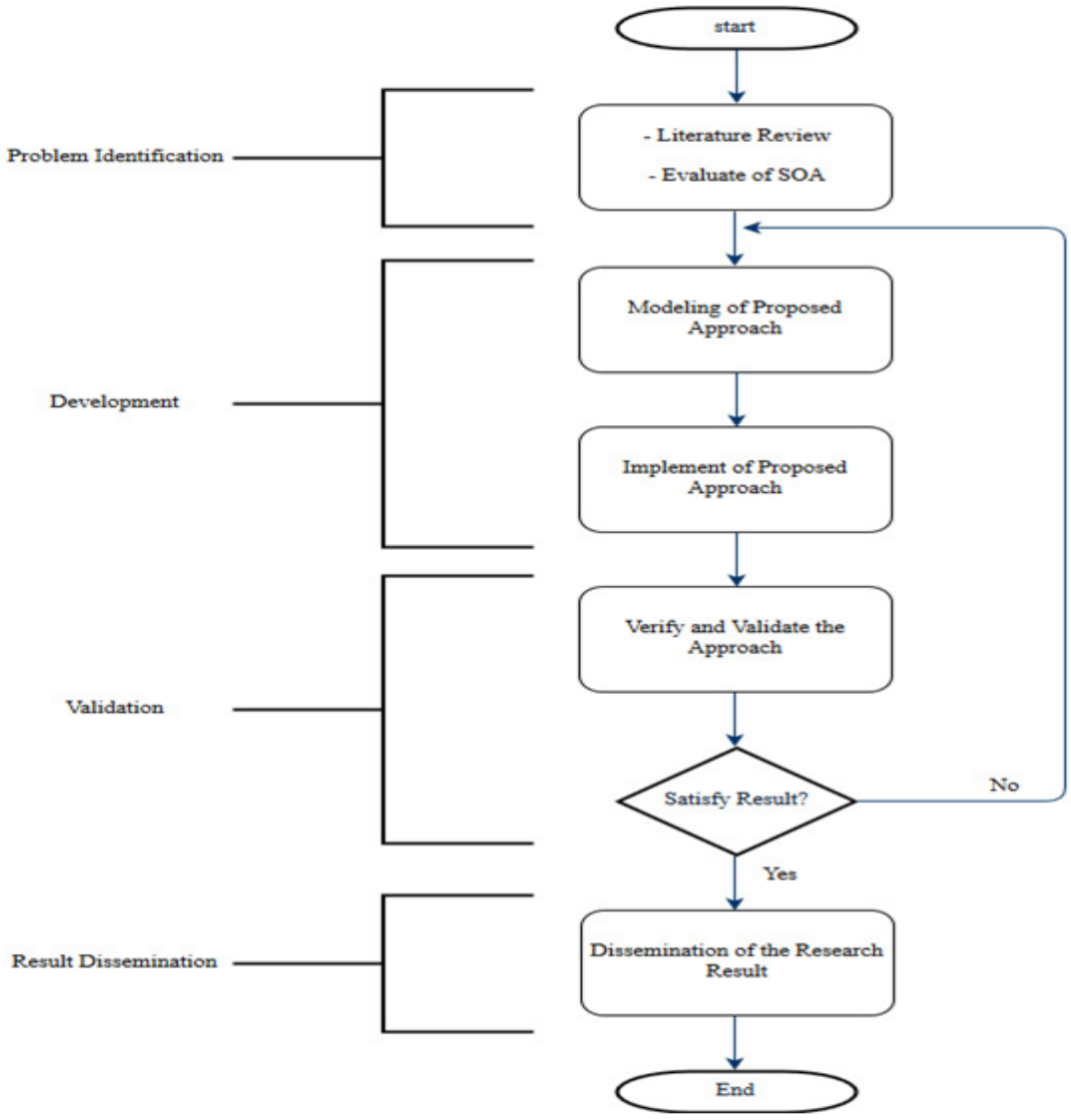

Figure [3.1] Research Methodology

3.2 Operational Framework

Table [3.1] Operational Framework

NO	Research Question	Objective	Activity(s)	Deliverable(s)
(i)	What are service oriented architecture approaches and how they work?	To evaluate the state of the current modeling and development approach.	Literature Review.	Comparative evaluation.
(ii)	How to improve SOA approach?	To propose a method to develop service oriented architecture system.	Work design and Coding.	Service development approach.
(iii)	How to validate proposed method?	To validate the applicability and effectiveness of the proposed method.	Work design and Coding.	Model Driven method.

3.3 Instrumentation

To develop the proposed method, we need to use:

(i) Enterprise Architecture Software
(ii) Unified Modeling Language.
(iii) WampServer.
(iv) HTML, CSS, PHP.
(v) MS Access.

3.4 Case Study

The process of "Cars Robbery Report" selected as a case study to explain how can represent this process by the different service oriented architecture techniques. This process was chosen because it has a simple process and a clear scenario and

integrates the Civil Registry System, Traffic Police System, and the Police Forces System.

To use the service, the admin user opens the site and he needs to login by using username and password, if the information is valid he will login to the system, if isn't he will reject and try again. After login the services provided by system will appear including "Report Form" Or "Display Reports". When the admin user selects "Report Form", a form containing the details of the robbery report will be displayed, and he will need to enter the national number of user and board number of car, after that the national number verify by Civil Registry and board number verify by Traffic Police, if they matching, the remaining user information and car information will be retrieved, and the admin user will see the information. After that he enter the report data and submit the report to the report section.

If admin user selects "Display Report", all the robbery report will be displayed. If he wants to see the details of some report, he need to enter the number of report and click to display details, then he sees the details of the report.

CHAPTER 4

THE PROPOSED MDA METHOD

This chapter characterizes the proposed service-oriented model-driven method. The levels of the method are described.

4.1 Development Method

The Model Driven Architecture (MDA) methods a clear separation of the business logic from the implementation logic that's less stable. It uses the models that are more perennial than codes. It puts the models at the center of the development of software and of the information systems. The MDA method consists at, firstly developing the CIM model, Secondly, obtaining the PIM model from the CIM, and finally generating the PSM model from the PIM which facilitates the generation of code for chosen technical platform.

4.1.1 CIM Definition

In this phase the functional requirements of the web service are collected. The CIM is defined by two models: the use case diagram and activity diagram.

i. **Use Case Diagram;** as shown in figure [4.1] Use Case Diagram used to show functions of system, is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in the system. A use case diagram can identify the different types of users of a system.

ii. **Activity Diagram;** as shown in figure [4.2] Activity Diagrams are graphical representations of workflows of activities and actions of the system, activity diagram is intended to model both computational and organizational processes. Activity diagrams show the overall flow of control.

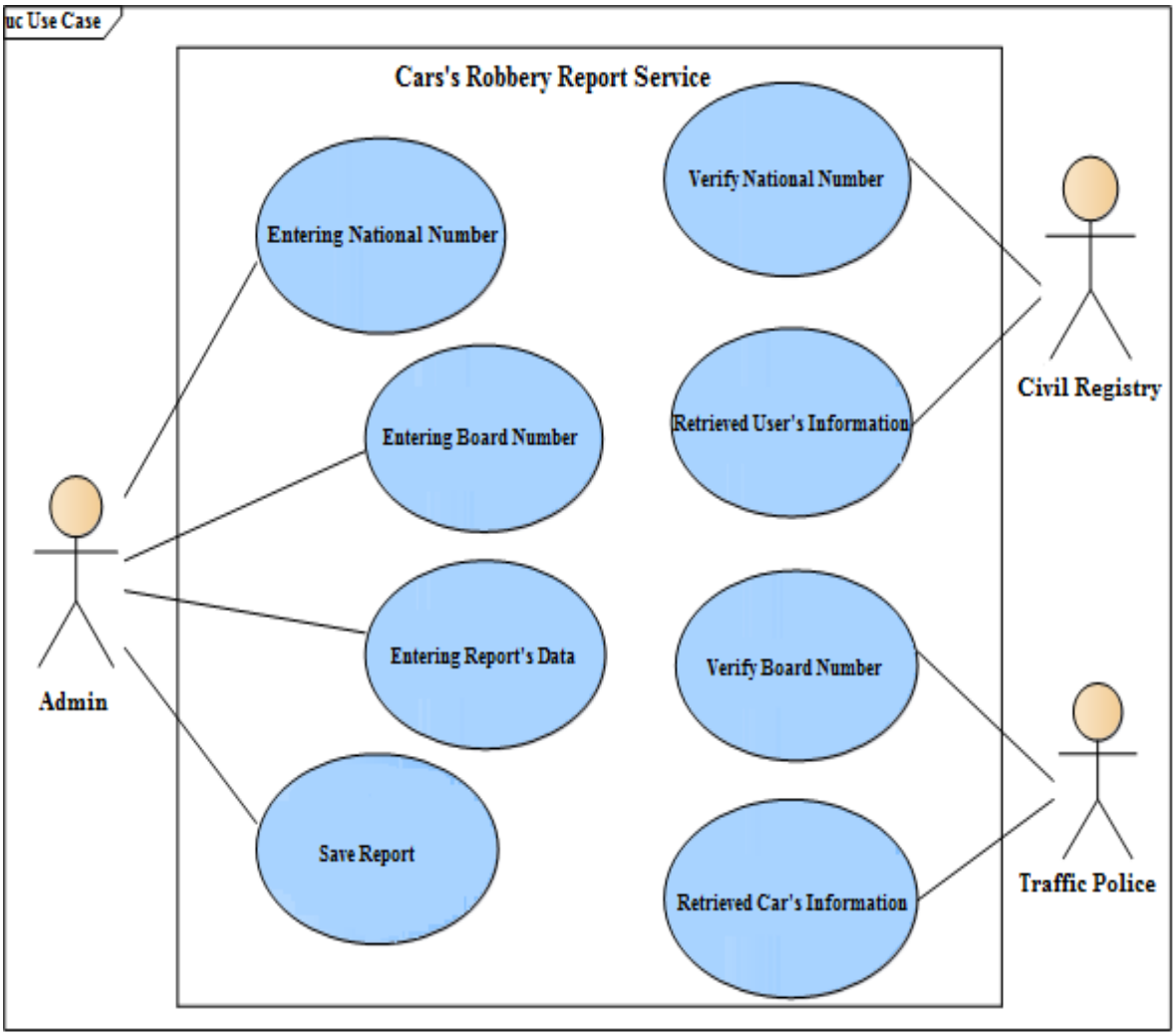

Figure [4.1] Main Functions of Web Service Use Case Diagram

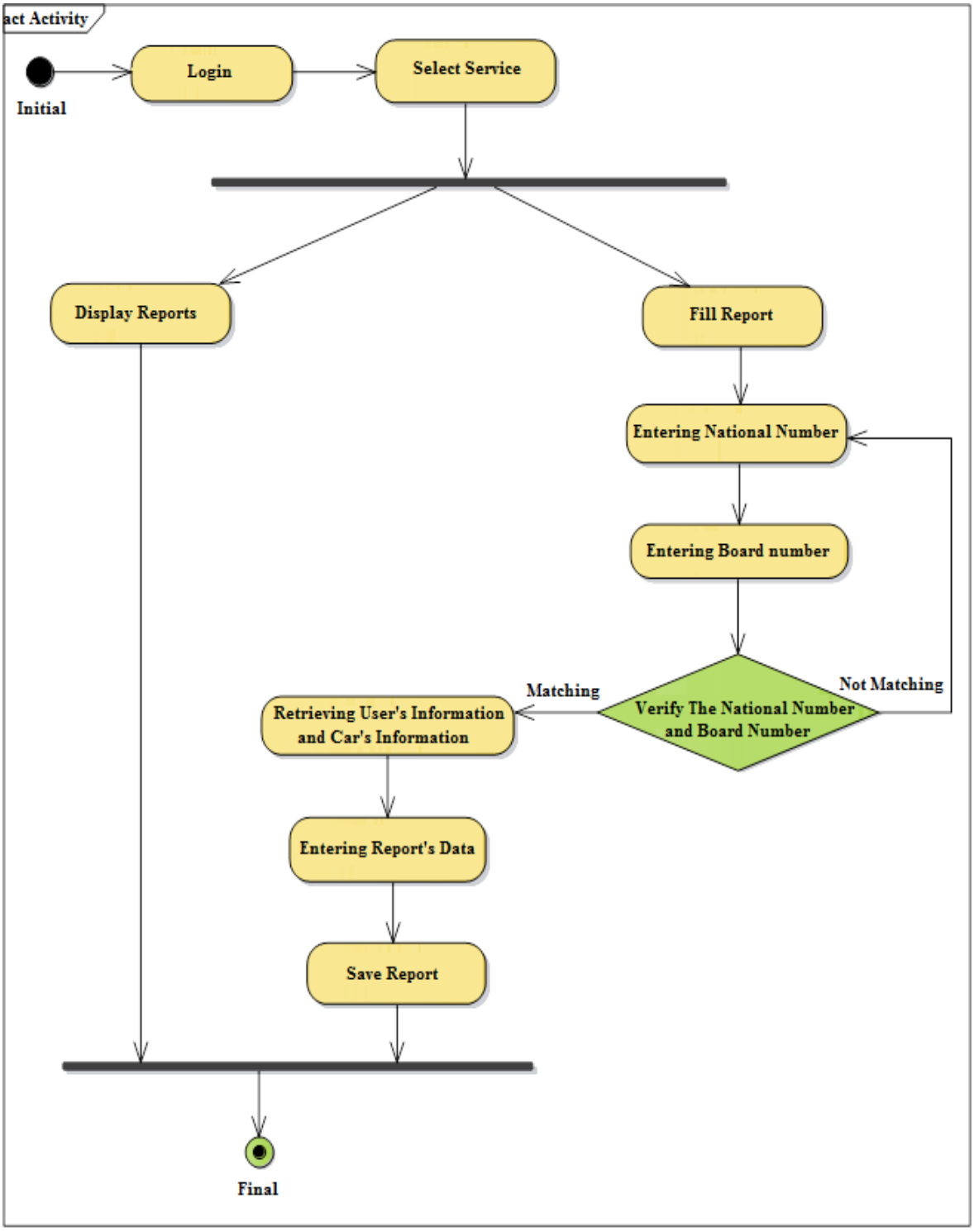

Figure [4.2] Web Service Flow of Control Activity Diagram

4.1.2 PIM Development:

In this phase PIM for system represent by using class diagram as shown in figure [4.3]. Class Diagram describe the structure of a system by showing the system's classes, their attributes, operations (or methods), and relationships between objects. The class diagram is used for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code, class diagram can also use for data modeling.

PIM for system explain main entities in web service which are Police_Forces, Traffic_Police, and Civil_Registry class and relationships between classes considered in diagram.

4.1.3 PSM Development:

In this phase PSM generated from PIM to develop web service using PHP language for models. Transformation process done as shown in figure [4.4].

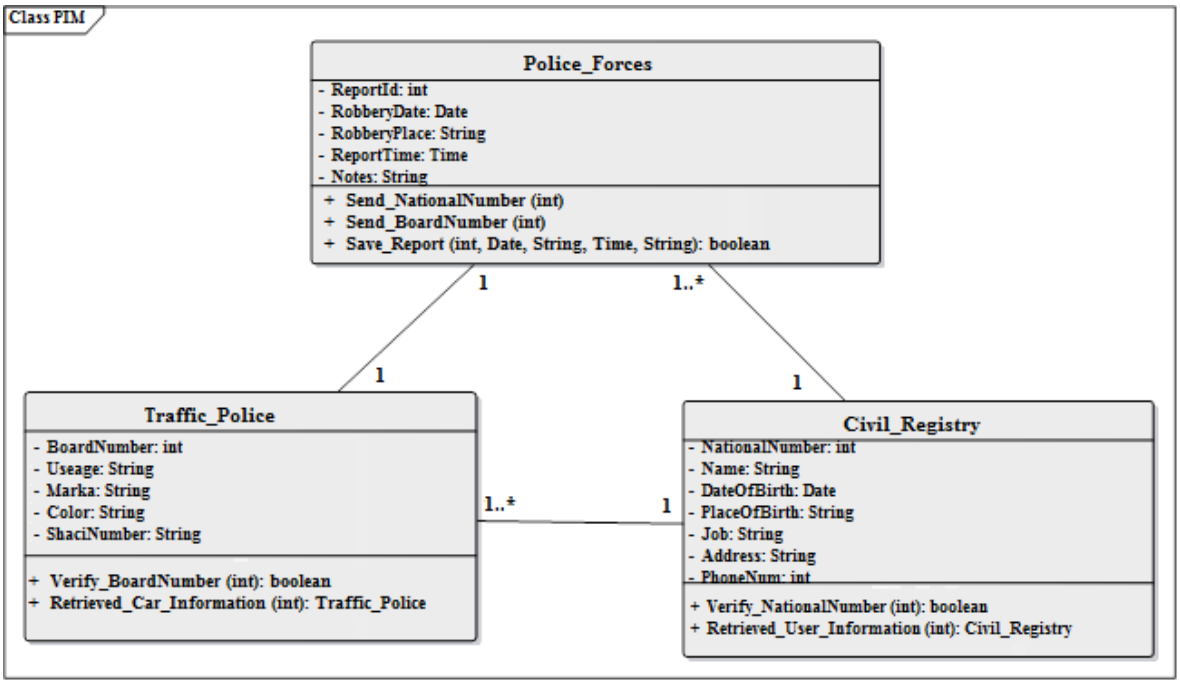

Figure [4.3] PIM for Web Service

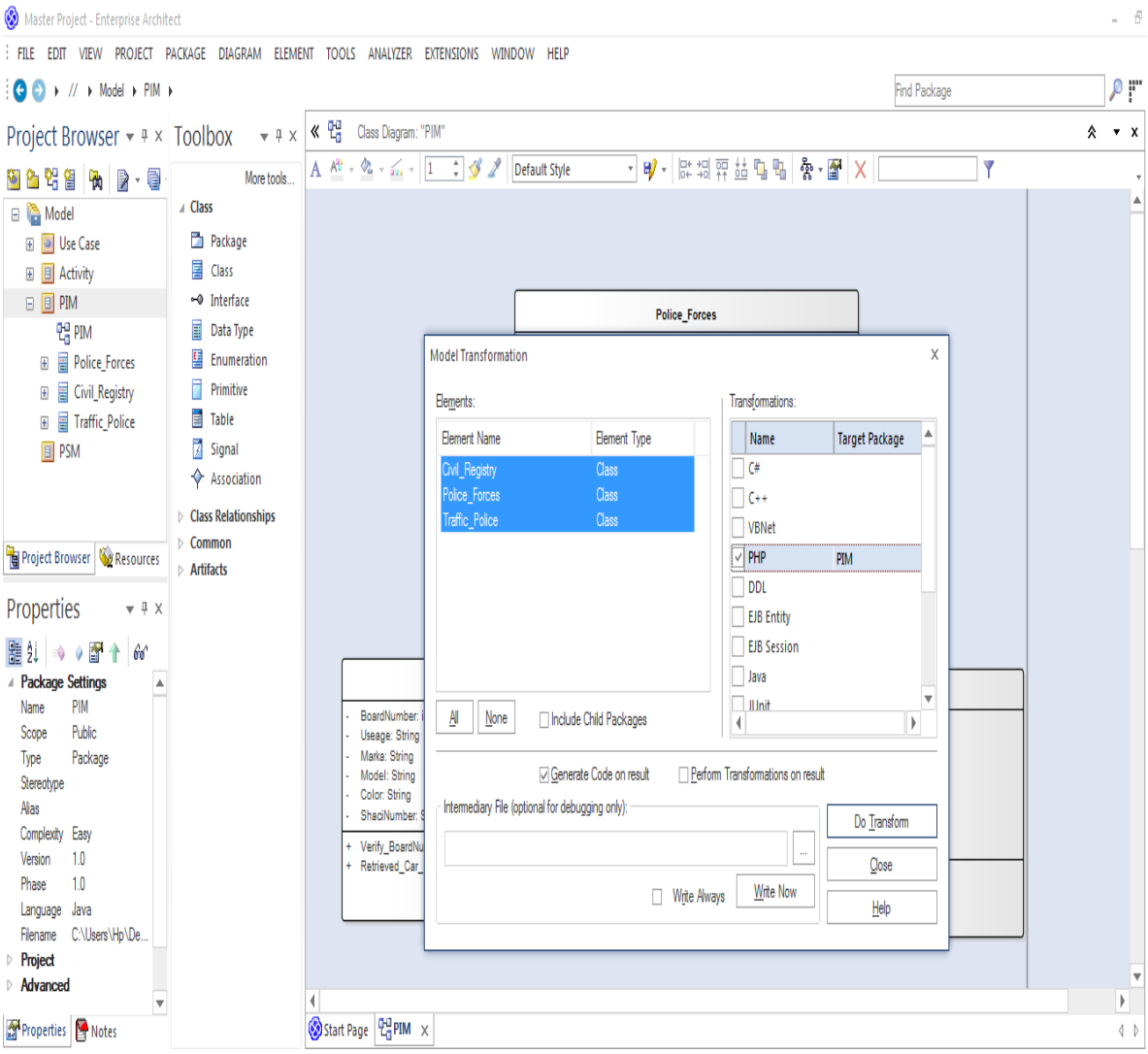

Figure [4.4] Transformation PIM to PSM

PSM models has generated for web service PIM as in figure [4.5], Generated PSM for PIM which transformed to model based on specific platform (PHP language). PHP language has selected because it simple and clearly, and most used to design web service.

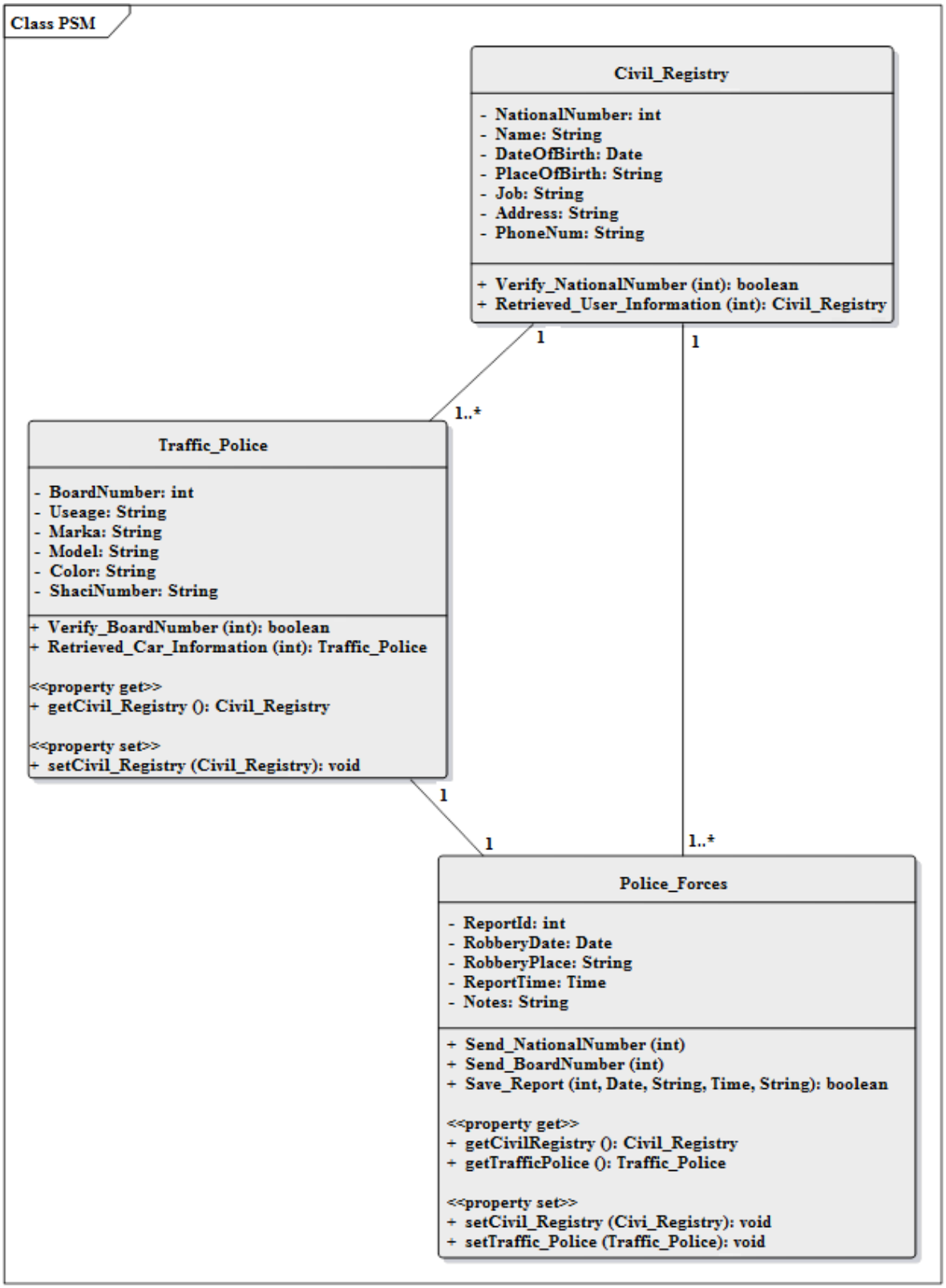

Figure [4.5] PSM for Web Service

4.1.4 Code Generation:

PHP Code has generated from PSM model using transformation tool, generated code is not complete and you may need to complete it manually. Code generation process done as in figure [4.6].

4.1.5 WSDL Diagram:

WSDL diagram contain from these diagrams: Messages, Port Types, Bindings, and Services. WSDL transformation is to create from a simple model an expanded model of a WSDL interface that is suitable for generation.

- Messages Model:

  Contains the WSDL Messages, modeled as UML Class marked with the stereotype WSDLmessage. Represent messages for message type mentioned in service as in figure [4.7].

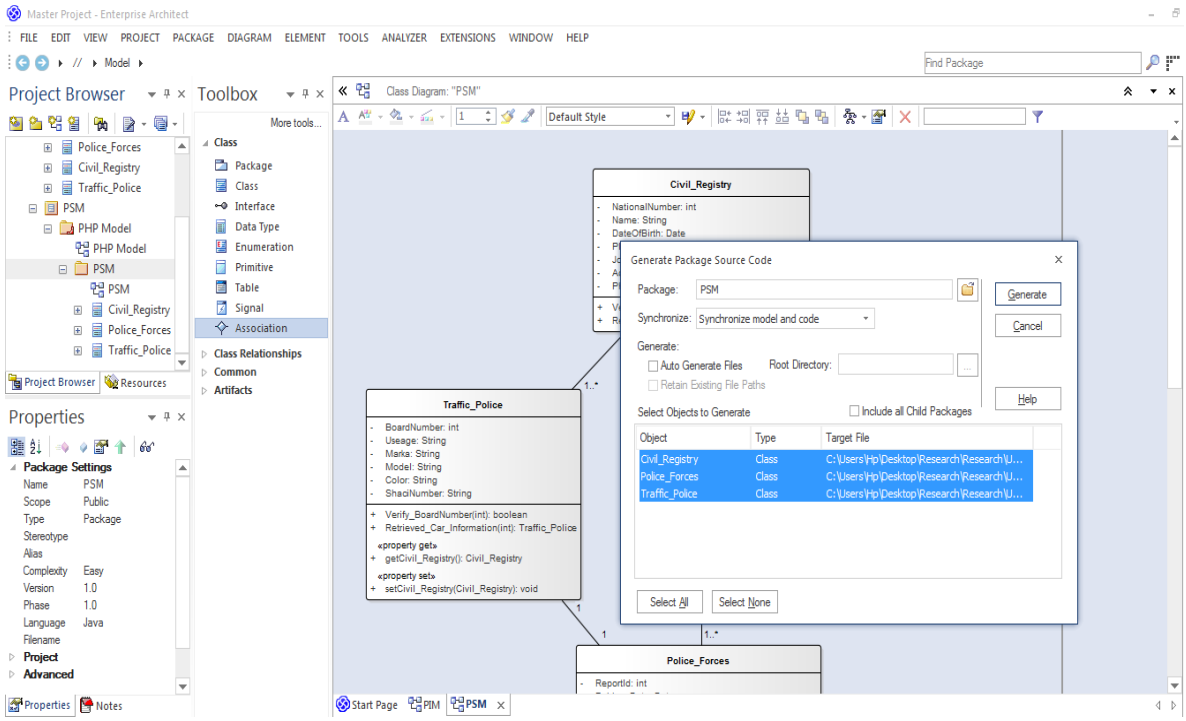

Figure [4.6] PHP Code Generation

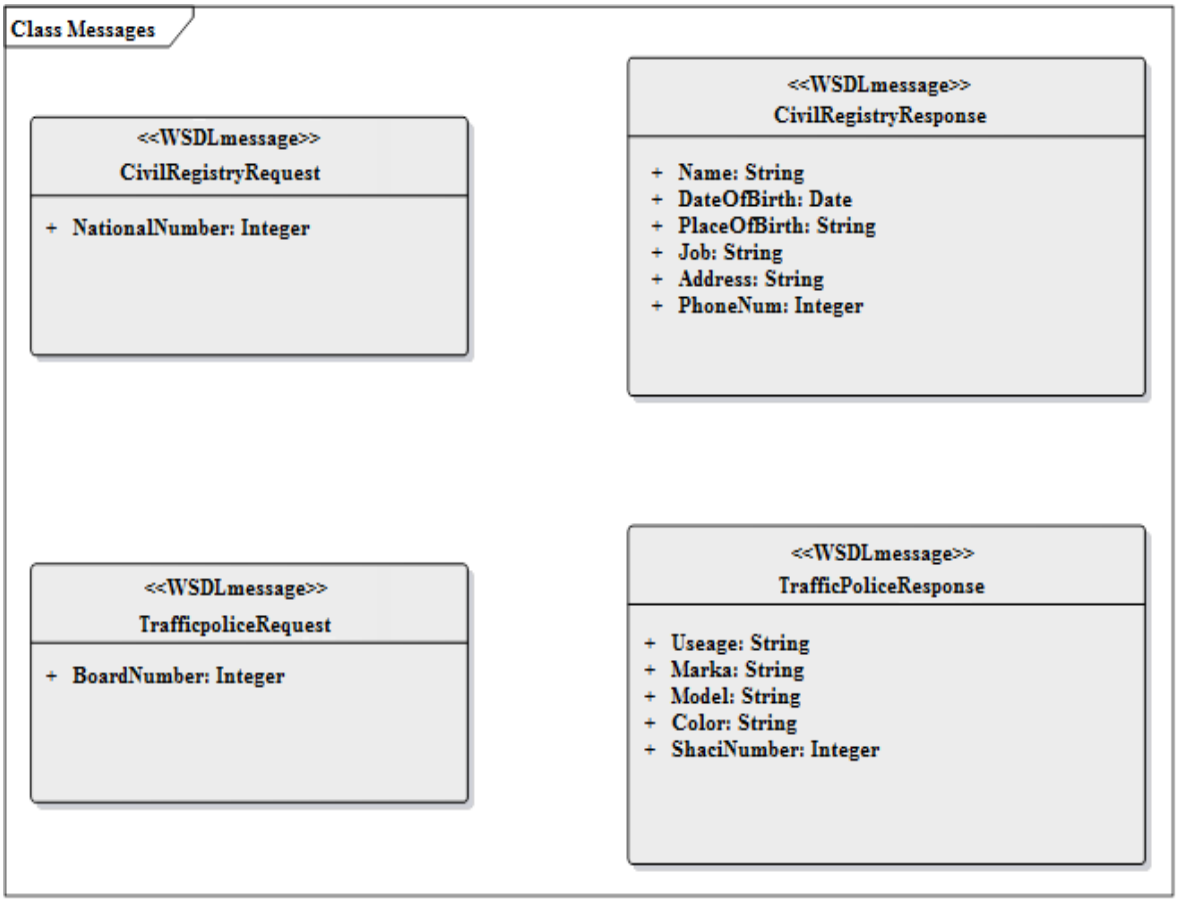

Figure [4.7] WSDL Messages

- Port Types:

  Contains the WSDL Port Types, modeled as UML interfaces marked with the stereotype WSDLportType. Represent port types in service as described in figure [4.8], and determine port operation types (OneWay, Request-Response) and messages (input, output).

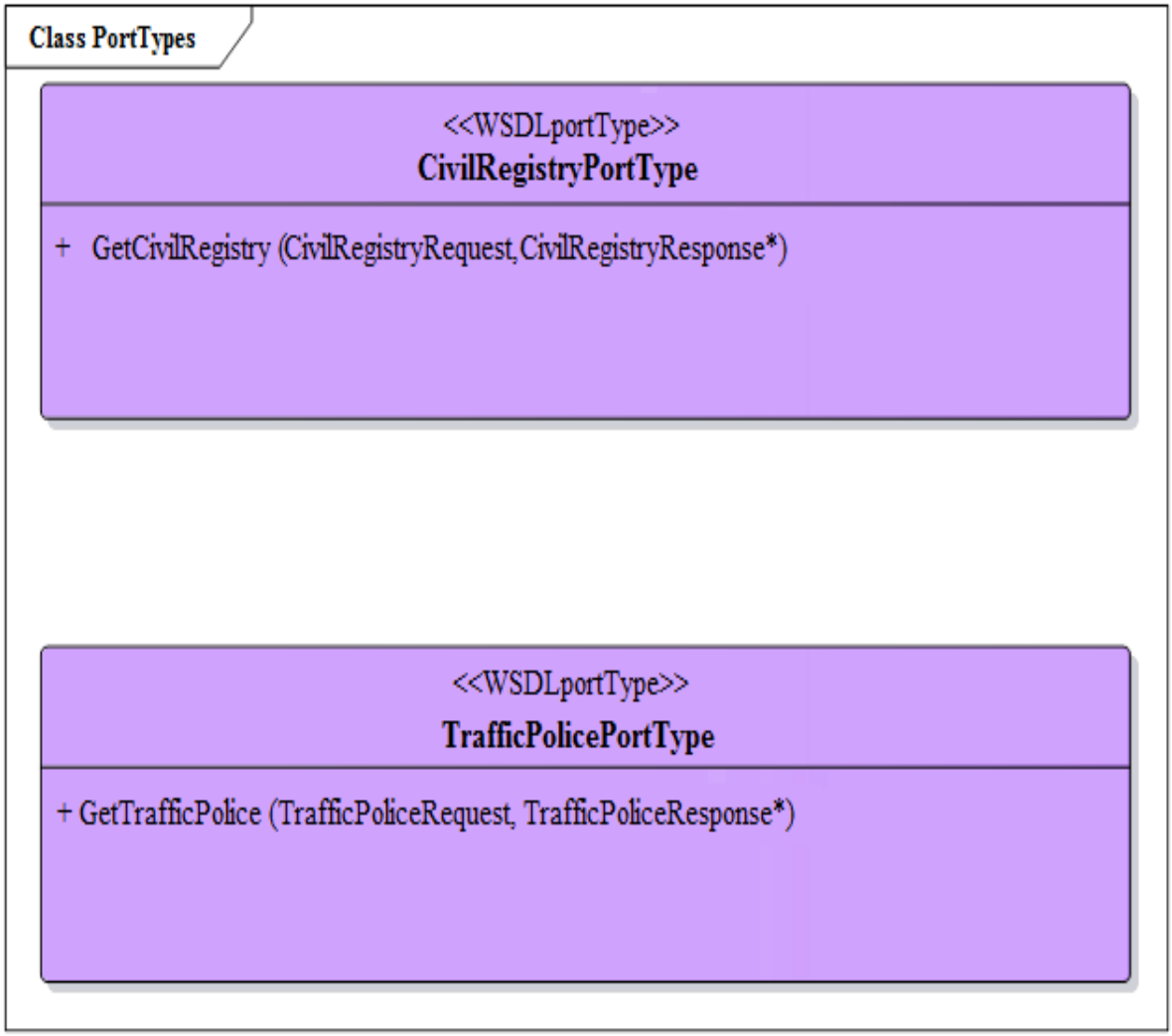

Figure [4.8] WSDL PortTypes

- Bindings:

  Contains the WSDL Bindings, modeled as UML Class that realize the PortType. Represent bindings in service (SOAP Binding, HTTP Binding) as shown in figure [4.9].

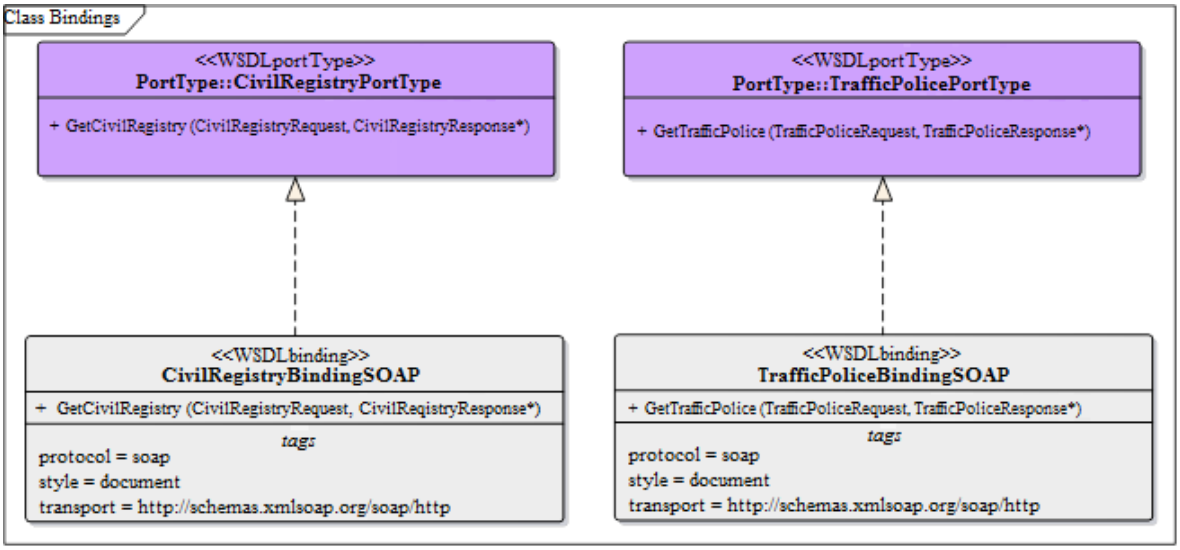

Figure [4.9] WSDL Bindings

- Services:

  Contains the WSDL Services, modeled as UML interfaces with associations to each exposed Binding. Represent services and methods and operations in service as shown in figure [4.10].

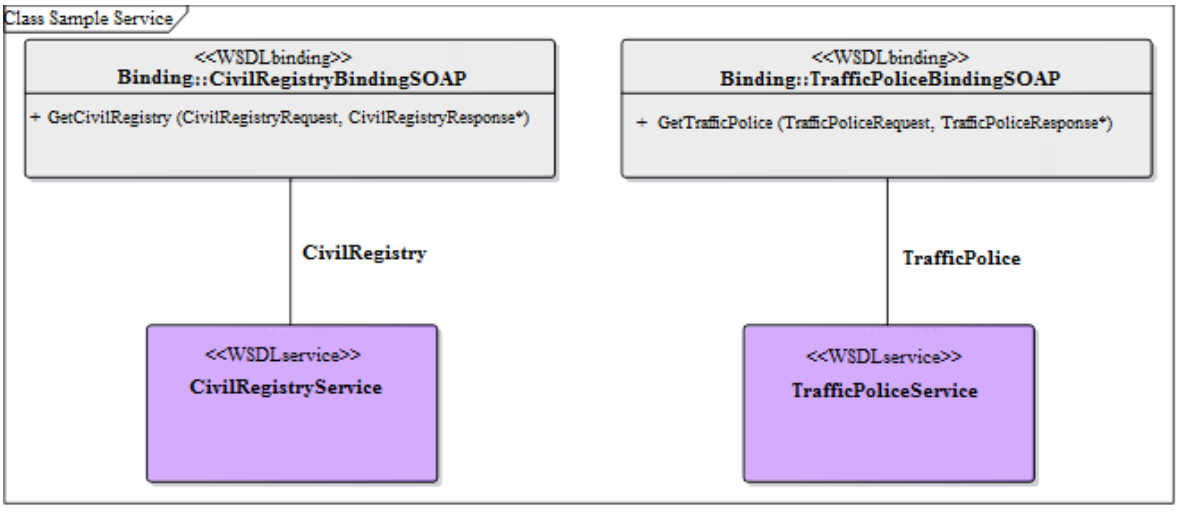

Figure [4.10] WSDL Services

- WSDL Diagram:

  Contains all mentioned models and WSDL file as shown in figure [4.11].

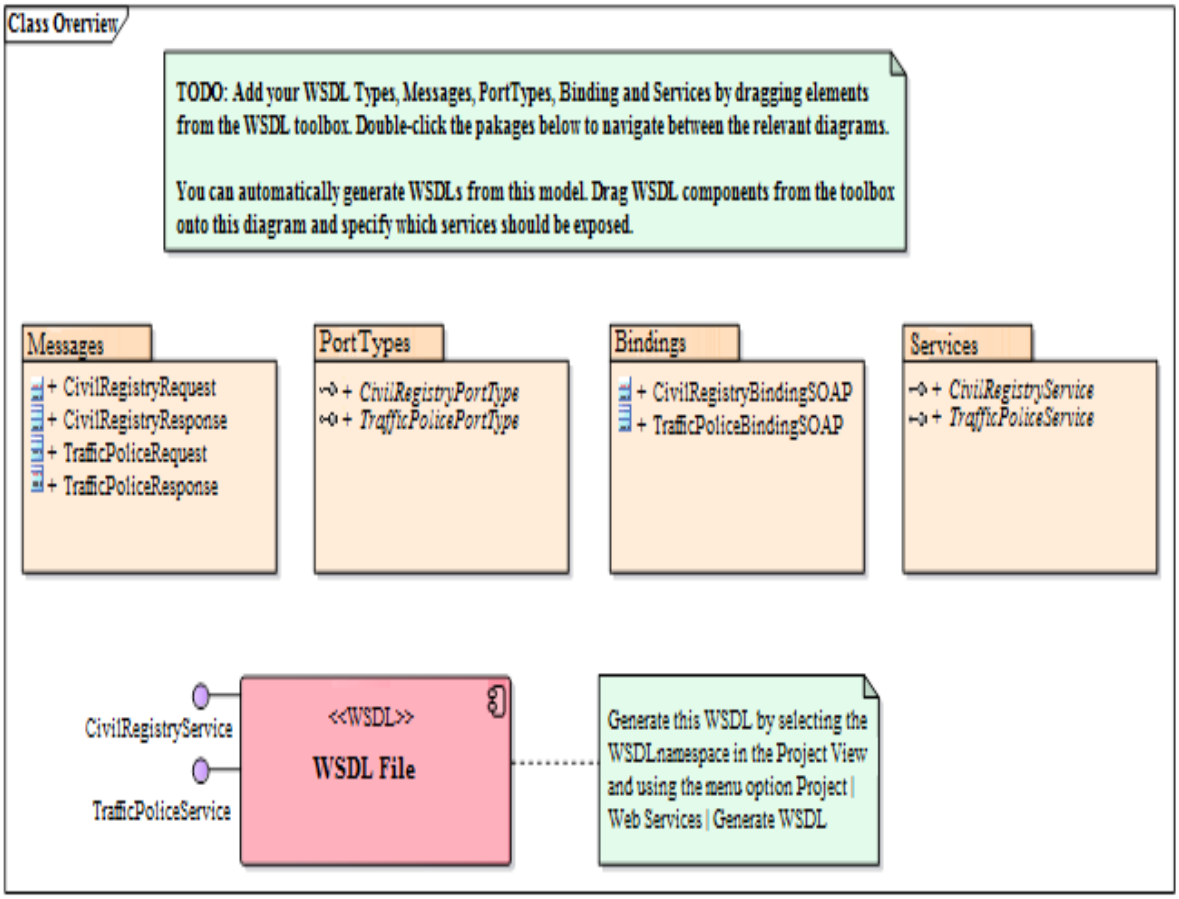

Figure [4.11] WSDL Diagram

4.1.6 WSDL Code Generation:

Generate WSDL code from WSDL file as shown in figure [4.12].

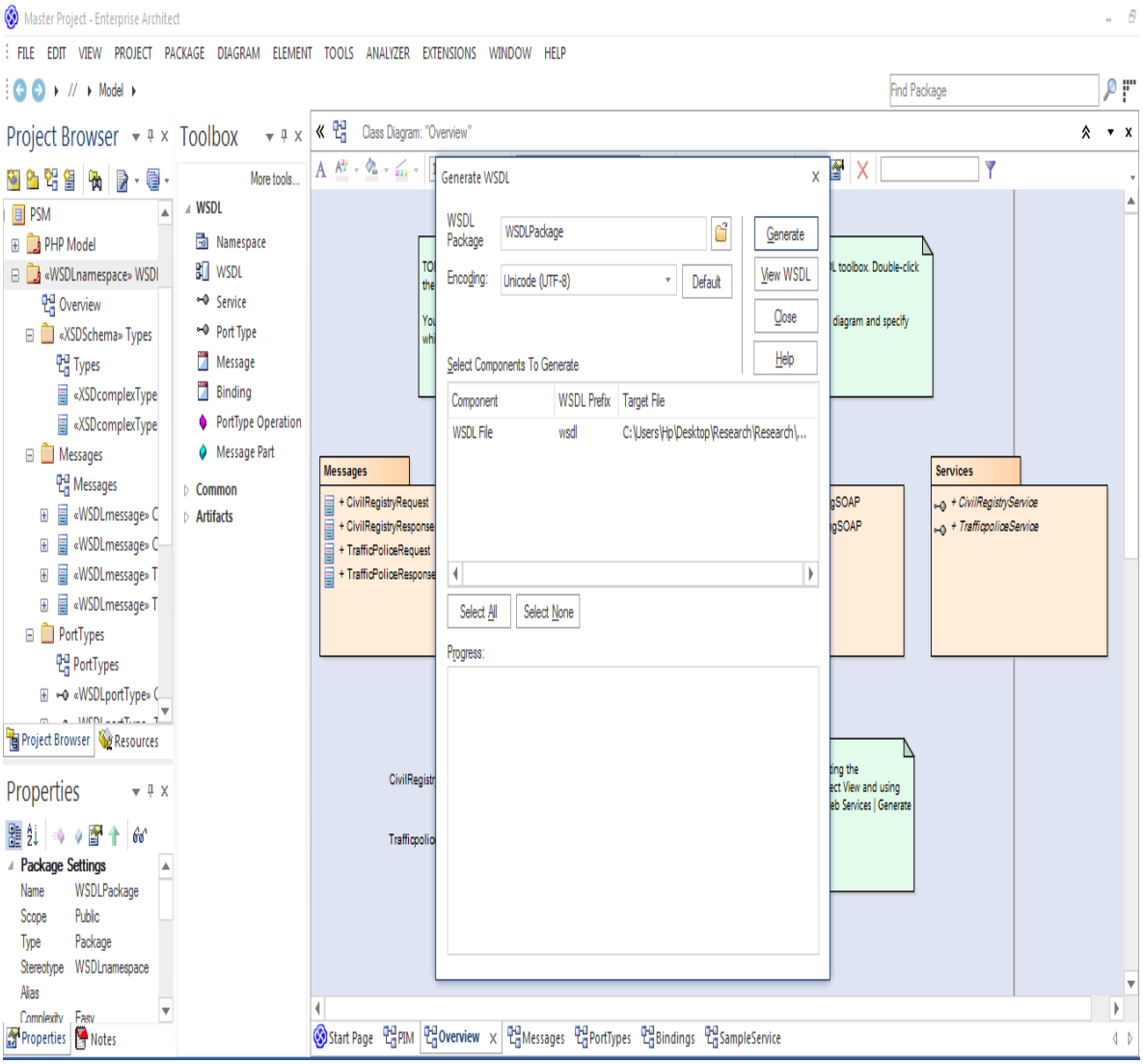

Figure [4.12] WSDL Code Generation

Generated WSDL code shown in figure [4.13].

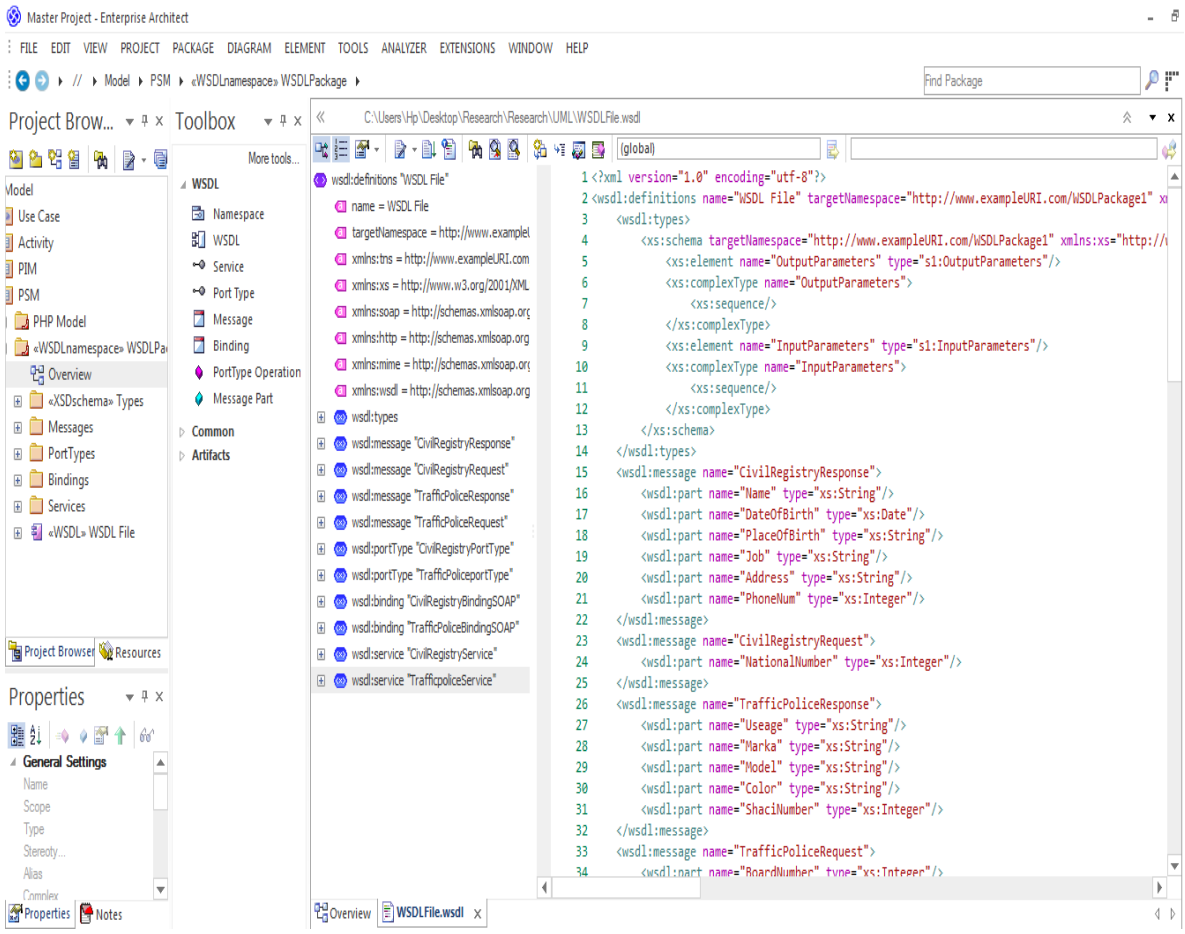

Figure [4.13] WSDL Code

4.2 Testing and Deployment

Here testing for web service functionality considered by take test cases and determine if service passes test case or not as shown in table [4.1]. Testing the below cases proved that the service worked correctly and satisfy its requirements.

Table [4.1] Web Service Test Cases

ID	Test Case	User Input	Criteria
Case_1	Get Data	Enter Wrong National Number and Wrong Board Number	Permission denied, Not Retrieved Data.
Case_2	Get Data	Enter Wrong National Number and Correct Board Number	Permission denied, Not Retrieved Data.
Case_3	Get Data	Enter Correct National Number and Wrong Board Number	Permission denied, Retrieved Civil Registry Data Just.
Case_4	Get Data	Enter Correct National Number and Correct Board Number	Permission granted, Retrieved Civil Registry Data and Traffic Police Data.

CHAPTER 5

EVALUATION

This chapter presents an evaluation or validation applicability of the proposed method which presented in previous chapter.

5.1 Evaluation

Because the integration of SOA is an essential part for building or development of service oriented in organization. And there are numerous of SOA techniques, it was necessary provide some recommendation to help developers to select the suitable SOA techniques.

This research studied and evaluated the previous studies. There are a lot of studied about SOA integration using MDA.

For example, Wiranto Herry Utomo [28] Based on the consideration and comparison of some integration methods, MDA method from OMG is chosen as a method to help dealing with SOA weaknesses. Model-driven based MDA method enables business level functionality to be modeled by UML language modeling that is separated from low level implementation (code level). Therefore, SOA used in MDA approach can be expressed using UML modeling language. This study proves that SOA-MDA method has been successfully used to perform analysis, design and implement of enterprise integration.

Also, Shuangxi Huang, Yushun Fan [3] SOA and MDA have emerged as a major evolutionary step in enterprise integration. Significant advances have been seen in SOA and MDA. It uses MDA as the philosophy of system development and SOA as the infrastructure of system implementation. After that, the technological framework of the methodology is presented to support the realization of integration in both model and service level. The framework comprises the meta model, the model

driven development frame, and service orient executing environment. Finally, the functionalities of the SOA and MDA based enterprise integration platform is introduced, which provides the integrated development and operational environment for enterprise integration. In this research, after evaluated previous studies selected MDA methods to integrate SOA. And case study that was discussed in chapter three used to explain integration process by SOA-MDA method. Finally, SOA-MDA method help developers in integration process in organization were provided.

5.2 Discussion

Proposed method integrate SOA with MDA method to produce SOA-MDA method to integrate services, development process is easier and models are portable and service integration is more flexible and developed in small time and with lower cost. Testing in proposed method applied and produces all test cases for service integration to develop quality and effective services integration.

5.3 Web Service Execution Interfaces

Figure [5.1] Login Window

Figure [5.2] Select Service Window

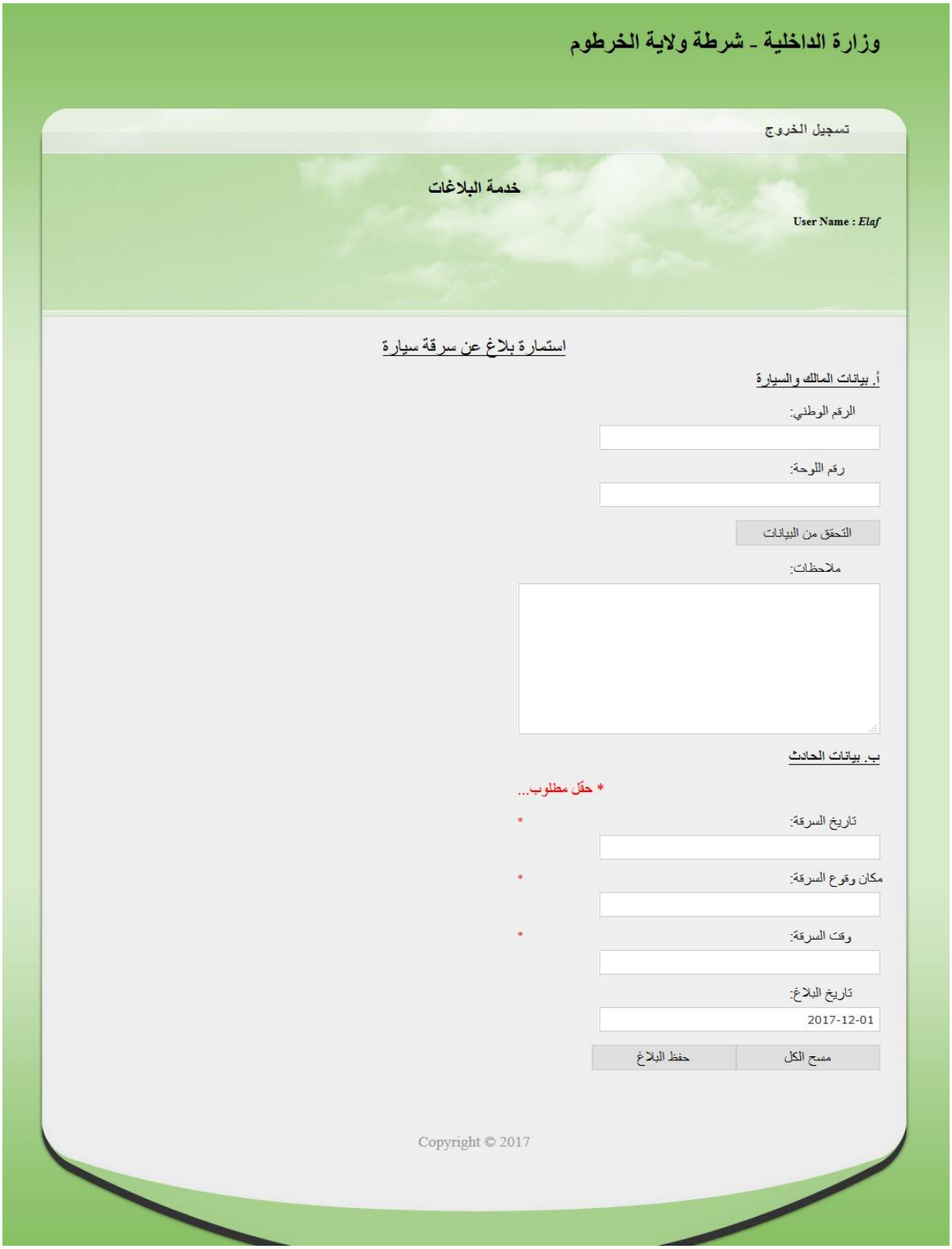

Figure [5.3] Report Form Window

Figure [5.4] Entering Wrong Data and check them

Figure [5.5] Entering Correct Data

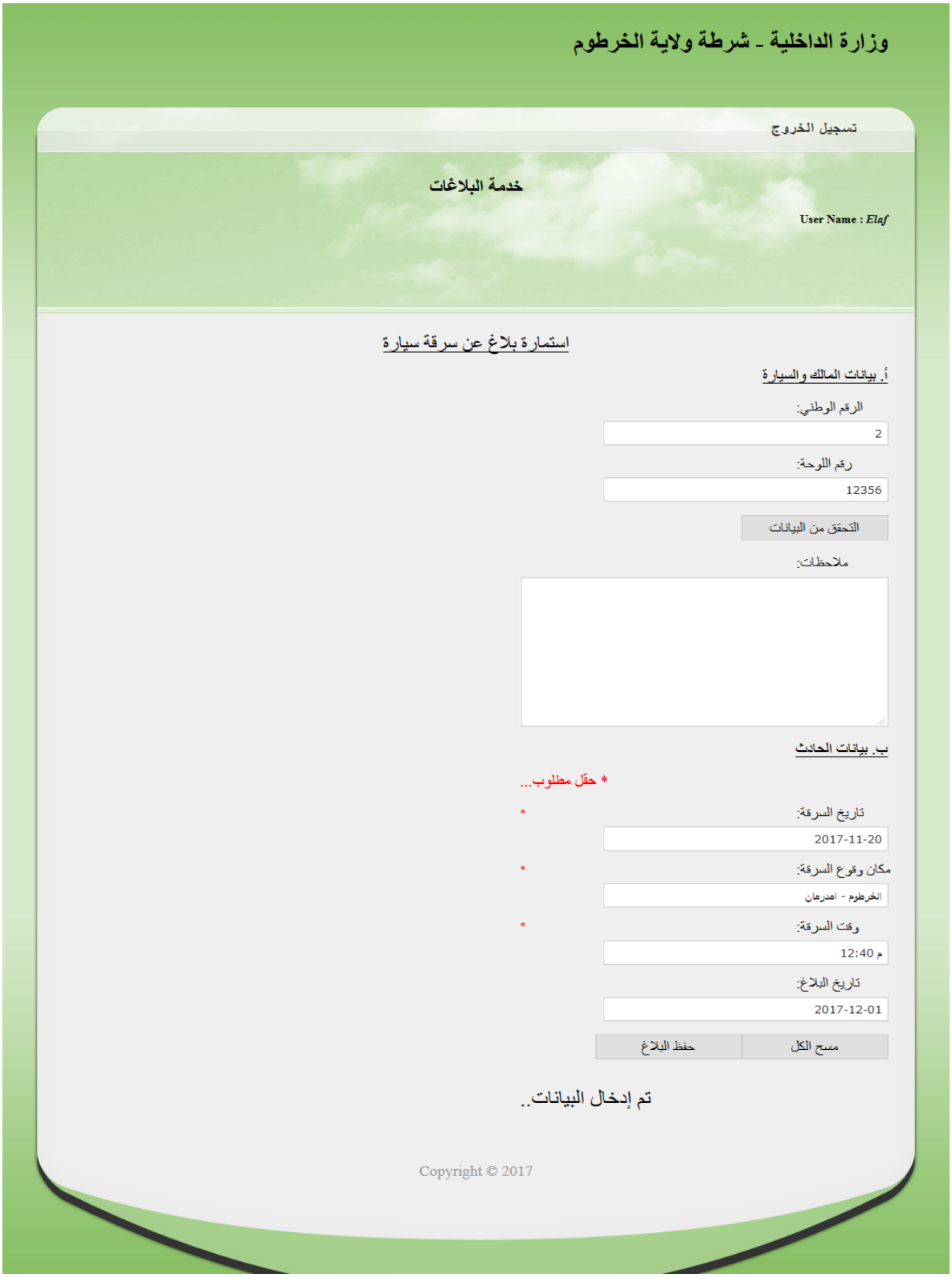

Figure [5.6] Save Data Window

Figure [5.7] Display Reports Window

Figure [5.8] Report Details Window

CHAPTER 6

CONCLUSION AND FUTURE WORK

This chapter presents the conclusion of this research and identifies some areas for future work.

6.1 Conclusion

This research aims to develop an integration of services using MDA to model and integrate services that increase productivity and decrease complexity by making developer more productive in development process and make development processes and integration automated. The combination of SOA and MDA can enhance the agility, flexibility, and robustness of integration by using SOA-MDA method.

MDA gives the opportunity to bring the services definition to a higher level of abstraction, due to the more formal and accurate platform independent specification of the services requirements and design.

To achieve this aim the current models for services development was discussed in chapter two and comparative evaluation for this models was set. The result from the comparative evaluation were the base for integrate the services.

The proposed method using SOA-MDA to reduce the complexity of the development process and make it easier and productive and portable, and help in managing the productivity and complexity of development method. MDA reduce the time and cost of the service development and increase the software quality.

6.2 Future Work

The future work for this research is to use the proposed method to integration larger services and deployed in real environment to be tested to check its effectiveness. The deployment of the service in real environment will provide a good feedback to improve the service. Can Using other modeling language to integrate services to provide many options to be available to help developers.

REFERENCES

[1] M. A. S. Vidales, A. M. F. García, and L. J. Aguilar, "A new MDA approach based on BPM and SOA to improve software development process," 2008.

[2] N. M. Josuttis, SOA in Practice. 2007.

[3] S. Huang and Y. Fan, "Model Driven and Service Oriented Enterprise Integration--- The Method , Framework and Platform."

[4] J. Siegel, building distributed applications is hard. 2002.

[5] I. Sacevski and J. Veseli, "Introduction to Model Driven Architecture (MDA)," 2007.

[6] O. M. G. Document, "Object Management Group Model Driven Architecture (MDA)," 2014.

[7] P. Saharan and C. Martinez, "Service-Oriented and Model Driven Architectures."

[8] N. V. Patil and M. C. Kshirsagar, "Enterprise Application Integration using Service Architecture with Web Service Aggregation Pattern," 2015.

[9] M. W. Chowdhury and M. Z. Iqbal, "Integration of Legacy Systems in Software Architecture."

[10] G. Lewis, "Getting Started with Service- Oriented Architecture (SOA) Terminology," 2010.

[11] R. Sharma, M. Sood, and D. Sharma, "Modeling Cloud SaaS with SOA and MDA."

[12] D. Ameller, X. Burgués, O. Collell, D. Costal, X. Franch, and M. P. Papazoglou, "Development of Service-Oriented Architectures using Model-Driven Development : A Mapping Study," 2015.

[13] M. Saroha and S. Sahu, "REVIEW ON ' SERVICE GRANULARITY IN SERVICE ORIENTED ARCHITECTURE ,'" 2014.

[14] M. Mohammadi and M. Mukhtar, "A Review of SOA Modeling Approaches for Enterprise Information Systems," 2013.

[15] N. A. Nabeeh, H. A. El-Ghareeb, and A. M. Riad, "Integrating Software Agents and Web Services in Service Oriented Architecture Based Cloud Services Discovery Framework," 2015.

[16] "IBM SOA Foundation : p roviding what you need to get started with SOA .," 2005.

[17] B. Gold-Bernstein and G. So, "Integration and SOA," 2006.

[18] G. Hook, "BUSINESS PROCESS MODELING AND SIMULATION," 2011.

[19] A. Bouain, A. EL FAZZIKI, and M. Sadgal, "Integration of non-functional requirements in a service-oriented and model-driven approach," 2014.

[20] C. Casanave, "Enterprise Service Oriented Architecture Using the OMG SoaML Standard," 2009.

[21] F. Truyen, "The Fast Guide to Model Driven Architecture The Basics of Model Driven Architecture," 2006.

[22] J. D. Poole, "Model-Driven Architecture : Vision , Standards And Emerging Technologies," 2001.

[23] T. Nodehi, S. Ghimire, R. Jardim-goncalves, and A. Grilo, "On MDA-SOA based Intercloud Interoperability framework," 2013.

[24] A. Kriouile, N. Addamssiri, and T. Gadi, "An MDA Method for Automatic Transformation of Models from CIM to PIM," 2015.

[25] S. Systems, "Model Transformation," 2017.

[26] D. Frankel and J. Parodi, "Using Model-Driven Architecture ™ to Develop Web Services," 2002.

[27] S. Khoshnevis, F. S. Aliee, and P. Jamshidi, "Model Driven Approach to Service Oriented Enterprise Architecture," 2009.

[28] W. H. Utomo, "Implementation of MDA Method into SOA Environment for Enterprise Integration," 2011.