Palestine Polytechnic University
Deanship of Graduate Studies and Scientific Research
Master of informatics

An Automated Generation of Business Application using Entity Relationship Diagram and Business Process Model

Submitted by:

Sabha Issa Mohammad Abu Sabha

Thesis submitted in partial fulfillment of requirements of the degree "Master of Science in Informatics"
May, 2016

The undersigned hereby certify that they have read, examined and recommended to the Deanship of Graduate Studies and Scientific Research at Palestine Polytechnic University the approval of a thesis entitled: **An Automated Generation of Business Application using Entity Relationship Diagram and Business Process Model**, submitted by **Sabha I. Abu Sabha** in partial fulfillment of the requirements for the degree of Master in Informatics.

Graduate Advisory Committee:

Dr. Mahmoud Al-Saheb (Supervisor), Palestine Polytechnic University.

Signature:_____ Date:_____

Dr. Nabil Arman (Internal committee member), Palestine Polytechnic University.

Signature:_____ Date:_____

Dr. Rashid Jayousi (External committee member), Al-Quds University.

Signature:_____ Date:_____

Thesis Approved

Dr. Murad Abu Subaih
Dean of Graduate Studies and Scientific Research
Palestine Polytechnic University

Signature:_____ Date:_____

DECLARATION

I declare that the Master Thesis entitled **"An Automated Generation of Business Application using Entity Relationship Diagram and Business Process Model"** is my original work, and herby certify that unless stated, all work contained within this thesis is my own independent research and has not been submitted for the award of any other degree at any institution, except where due acknowledgement is made in the text.

Sabha Issa Mohammad Abu Sabha

Signature:_____ Date:_____

STATEMENT OF

PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for the master degree in Informatics at Palestine Polytechnic University, I agree that the library shall make it available to borrowers under rules of the library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgement of the source is made.

Permission for extensive quotation from, reproduction, or publication of this thesis may be granted by my main supervisor, or in his absence, by the Dean of Graduate Studies and Scientific Research when, in the opinion of either, the proposed use of the material is for scholarly purposes.

Any coping or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Sabha Issa Mohammad Abu Sabha

Signature:_____ Date:_____

DEDICATION

To my dear mother, Amna

To the soul of my father, Issa

To my husband, Munther

To my little girl, Alaa

To my mother in law, Khawla

and to my father in law, Mohammad

To my sisters and brothers

To all my family

To all my friends

To all teachers

To Palestine Polytechnic University

To Martyrs, Wounded, Prisoners from Islamic nations

And finally, To my home land Palestine

ACKNOWLEDGEMENT

I would like to acknowledge Palestine Polytechnic University for giving us the opportunity to complete a master degree in informatics.

Acknowledgement also goes to all the instructors in the Deanship of Graduate Studies and Research Units for their great impact in our education, especially my distinguished supervisor Dr.Mahmoud Alsaheb, and everybody who affected my education.

Likewise, I can't forget to acknowledge my mother and my husband for all support and patience. Finally, I would like to acknowledge my family, my friend.

الملخص

تتزايد الحاجة في السنوات الأخيرة إلى البرامج المحوسبة بشكل كبير، تكاد لا تخلو مؤسسة سواء أكانت صغيرة أم كبيرة، أم كانت هذه المؤسسة تعليمية، أو ترفيهة، أو تجارية من هذه البرامج. يستغرق بناء هذه البرامج وقتاً وجهداً كبيرين فيقع على عاتق الجهة المبرمجة أن تفهم ما يريده المستخدم في برنامجه الخاص ومن ثم الشروع لبرمجته. وقد يتجاهل بعضهم الحاجة إلى مرحلة فهم المشروع فيقع في مشكلة أكبر، إضافة إلى أن أغلب هذه التطبيقات المبرمجة تتشابه في الأساسيات وتختلف في الخصوصيات. ونلاحظ أنه في الآونة الأخيرة أصبح هناك اهتمام ببرامج وأدوات مساعدة تساعد المبرمجين في عملية البرمجة بإنشاء رمز (كود) برمجي تلقائي يساعد في البرمجة ويقلل من الوقت والجهد اللازمين للبرمجة.

نقوم في هذه الرسالة باستخدام نموذجين يتم استخدامهما أثناء مرحلة فهم المشروع وتصميمه وهي نموذج العلاقات و نموذج إجراءات الأعمال والقيام بعملية الربط بينهما وإنشاء تطبيقات محوسبة تشمل: إنشاء قاعدة البيانات، إنشاء أصناف خاصة بكل الكائنات المذكورة في نموذج العلاقات، إضافة إلى شاشات خاصة أيضاً بكل كائن مع امكانية التعامل مع قاعدة البيانات من إضافة وحذف وتعديل وبحث. والإضافة المميزة في هذه الرسالة إمكانية الربط مع نموذج الأعمال للتحكم في سير عمل هذه البرامج.

Abstract

Software development life cycle (SDLC) plays a significant role in building a good quality software. SDLC has six stages listed as follows: Requirement Analysis, Design, Development, Testing, and Maintenance. The success of any of SDLC stages, the success of building the software. The design stage mimics the user requirements to vision document using well known models such as UML Models. Recently, there has been a lot of attentions in the automatic code generation from models to codes. In this study, we use two models the Entity-Relationship (ER) diagram and the Business Process Model and Notations (BPMN). These models are used together because of the limitation of BPMN to represent data.

We have proposed a business application framework that maps ER and BPMN models to the business application code and make a connection between them. We mapped ER to the SQL file, LINQ to SQL classes files, and windows form application. We have also mapped from BPMN to windows workflow (WF). Furthermore, we suggested a connection method between BPMN and ER. In addition, We implmented the mapping method using windows form environments by building business application automatically that can select, insert, update, and delete from the database. And we developed a mapping from BPMN to activity in WF as well as developing the connection between ER and BPMN.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BA	Business Application
BAF	Business Application Framework
BP	Business Process
BPI	Business Process Incubator
BPMN	Business Process Model and Notations
Chen-ER	Chen Notations of Entity Relationship
CLR	Common Language Runtime
DBMS	Database Management Software
DTD	Document Type Declaration
EER	Enhanced Entity Relationship
EPC	Event-driven Process Chain
ER	Entity Relationship
GUI	Graphical User Interface
IDE	Integrated Development Environment
IT	Information Technology
LINQ	Language Integrated Query
OMG	Object Management Group
OS	Operating System
SDLC	Software Development Life Cycle

SE	Software Engineering
SQL	Structured Query Language
UI	User Interface
UML	Unified Modeling Language
VB	Visual Basic
VBA	Visual Basic for Application
WCF	Windows Communication Foundation
WF	Windows Workflow
WfMC	Workflow Management Coalition
WPF	Windows Presentation Foundation
XAML	Extensible Application Markup Language
xBML	Extended Business Modeling Language
XML	EXtensible Markup Language
XPDL	XML Process Definition Language
XSD	XML Schema

Chapter 1

Introduction

---

1. Chapter Outline:

    1.1. Problem of the Study

    1.2. Purpose and Deliverable

    1.3. Thesis Contributions

    1.4. Target Users

    1.5. Thesis Structure

1.1 Problem of the Study

Today, few business applications can exist without software. Software ranges from a small program, like music player in the mobile to the big enterprise application like online shopping system. So, the software is a collection of computer programs that are used to operate and manipulate computers and their peripheral devices [65]. Software products may be developed for an individual customer or may be made for a general market. Software products help ordinary people, employees, and a director of the company to perform their work with high efficiency. For example, word processing software helps users to create documents that contain text and graphics in a proper format. And database management software (DBMs) helps users to create, access and manage a database. Users can also add, change, delete, or retrieve data in the database. Users can create forms and reports as well.

The software can be categorized into two broad classifications: system software, and application software [65]. The operating system is an example of system software, but business software, graphics and multimedia, web browser and accounting applications are examples of application software. Application software is designed to help users to perform an activity, and to make the business activity more efficiently. The development of application software is not an easy work. That is to say, the effort of building any application is quite agile and easy, but behind this there is a big mind works which are developed by software teams. So we need a framework to help developers and teams to build their applications quickly and efficiently.

Before building the application, we should determine the requirements from users. These requirements need to be discovered, revised and reviewed. The process of determining the requirements called requirement engineering

and this process is an important step in the software development life cycle (SDLC)[76]. In requirement engineering, the designer should meet the user whether they are ordinary or business people to determine the requirements and write the requirements in unrestricted natural language. Then rewrite the requirement in a structured natural language with specific rules. After that, use a modeling language, such as unified modeling language (UML), business process modeling notation (BPMN), entity relationship model (ER) to integrate and facilitate multiple views for the system.

A modeling language [28] is used to express the structure of the system that is defined by a consistent set of rules; modeling language can be graphical or textual. Conceptual modeling and business process modeling are a modeling language, but conceptual modeling describes a data such as ER model while business process modeling describes a process (a group of activities) such as BPMN. BPMN is a more popular one to model a business process, but does not facilitate the modeling of data involved in the process, on other hand, ER used to model data and forfeit to model the process. Therefore, it's necessary to design a framework that provides a combination of conceptual modeling and business process modeling.

ER describes entities, attributes, and the relationship between them. We can get benefit from ER model to build a relational database to be created in different database management system such as Oracle, Microsoft access, Microsoft SQL server etc. We can also build classes and member variables for each entity and attribute. So, we can use these classes in any ER-based applications.

Since we can get a relational database from ER, we can build a business application whether this application is a desktop, client-server, or web application. But, the ER-application built in a traditional way push busi-

nesses into higher risk because of increasing competition, growing customer requirements, shift of technologies and other rapid market changes [35].

Business process modeling helps the organization to solve the risk of rapidly changing business application. There are many benefits for using business process modeling; for example, reducing process costs, increasing process quality, reducing process throughput times, and increasing forecast accuracy [90].

1.2 Purpose and Deliverable

This study aims to develop a framework that handles multiple modeling languages (data modeling and business process modeling) that are suitable to automate building a business application. The deliverable is, therefore, a business application framework (BAF) that take a business process and entity relationship model as input and automated generation of business application as output.

1.3 Thesis Contributions

The main contribution of this thesis is "To improve building automated generation of business application from modeling language. The used modeling languages are business process model and notations (BPMN) and entity-relationship (ER) models". These mapping includes; (i) mapping ER to SQL file, classes files, and windows forms, (ii) mapping BPMN to windows workflow foundation activity, and (iii) making a combination process between BPMN and ER models.

1.4 Target Users

The primary target users to the BAF are developers and companies. Developers can build their applications in a real time and effort. Companies can also deliver their applications in a reasonable time with the possibility of achieving the business goals.

The second target users are Researchers. BAF opens up real opportunities for researchers to conduct research in BA automation.

1.5 Thesis Structure

This thesis of eight chapters is organized as follows:

- Chapter 1
  - Chapter one introduces the study problem of the thesis. The chapter provides an introduction to the problem, purpose, deliverable, thesis contributions and the target users of the thesis.
- Chapter 2
  - Chapter two presents the basic concepts within automatic generation of business application. It covers the background of business process model and notation and entity relationship models. In addition it sheds light on the related technology used in this thesis.
- Chapter 3
  - Chapter three reviews the related literature that contributes to the study.
- Chapter 4

  - Chapter four introduces the research data and methodology used in this study.

- Chapter 5

  - Chapter five proposes a theoretical framework. It shows how to combine business process model notation and entity relationship diagram. Also, it describes the mapping process of building automated generation of business application.

- Chapter 6

  - Chapter six lists the implemented of some concept in theortical framework.

- Chapter 7

  - Chapter seven shows the evaluation of this study.

- Chapter 8

  - Chapter eight is a discussion of conclusions and recommenations

Chapter 2

Background

2. Chapter Outline:

    2.1. Automatic Code Generation

    2.2. Business Application

    2.3. Modeling

        2.3.1. Data Modeling

            2.3.1.1. Entity Relationship Diagram

        2.3.2. Business Process Modeling

            2.3.2.1. Business Process Modeling and Notation

    2.4. Study Related Technologies

This Chapter presents a definition of automatic code generation, common benefits and barriers of automating code, and the typically targeted issues to be achieved. It also defines business application and the general properties of these applications. Furthermore, this Chapter presents a background information about business process model and notations and entity relationship diagrams. Extra information about the technology used in the thesis is demonstrated.

2.1 Automatic code generation

Automatic code methods can provide a real value to the software development applications. Automatic code generation can be considered as a mapping process from textual or graphical formalism to a code in a specific program or application [39].
There are many of the benefits that can be gained from automating [78]. Some of the most general benefits are listed here:

1. The generated code (Repeatability) can be used repeatedly, from one application to another.

2. The generated code (Reliability) can be reduced the possibility of human error.

3. The generated code (Efficiency) will regularly be enhanced to the developer to make their tasks in a less time than write it manually.

4. The generated code (Leverage) enables to make multiple versions of the basic code with additional enhancement.

In spite of these benefits, the automation is not relying on the most organization because the developers focused on the core requirements and

neglected the design steps in a quick manner to complete the code. In addition, the lack of organizational and management support due to have a poor knowledge about automation concepts [78].

2.2 Business Application

2.2.1 Definition

Business Application (BA) is any application that is significant to running a business. There are many types of business application from a large line-of-business systems to the specialized tools [61].
Because the business application is a set of computer programs that perform various business functions, they may have the following common properties [60]:

- BA is based on user interaction.
- BA has a graphical user interface.
- Users can query-modify-input data and view results directly.
- Users can run reports instantaneously.
- Some BAs run in batch mode; application running based on event/time.
- Some BAs are built in house, and Some BAs buy from a vendor.
- BAs are installed in desktop or in big servers.

An enterprise application is a big business application. The cuurent enterprise applications environments are complicated, scalable, distributed, component-based, and mission-critical [60]. Figure 2.1 [82] show the evolution of the enterprise application.

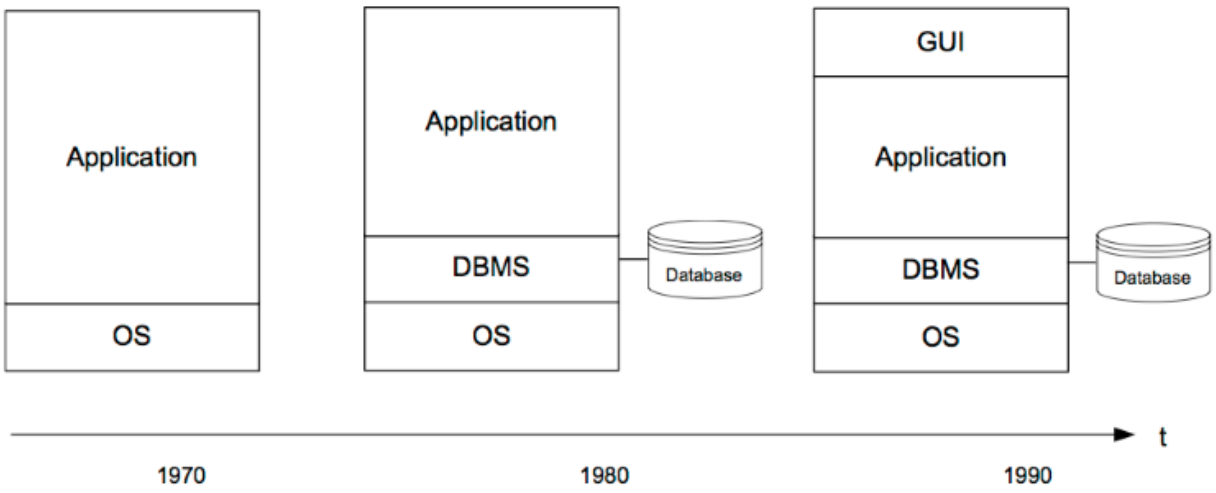


Figure 2.1: Early systems architectures

In the early infromation system, we show that the software categorized into two subsystems: system software and application software. Using interfaces as a connecting method between subsystems. With the evaluations of the separation of concepts and hiding information theory, system architectures evolved these concepts. Each subsystem has an identical set of functions that packaging with the interfaces in it. So local changes do not affect the overall system.

The next evolution of systems architectures recognizes the management of the data. To separate the structure of data and the data itself, the database management systems were introduced. This property is called physical data independence. The logical data independence is also supported by separating the logical organizations of data from applications. Use query language like Structured Query Language (SQL) to access a large amount of data without changing the data itself.

After that, graphical user interface (GUI) is appeared in the next evolution, which were developed to facilitate human interaction with application systems. It also helps the seperating of GUI from application structure.

2.2.2 Business Process (BP)

There are many definitions of business process. Hammer and Champys (1993) [30] defined the business process as: *" A collection of activities that takes one or more kinds of input and creates output that is of value to the customer"*. In 2008, Pant and Juric [68] defined business process as: *"a set of coordinated activities that are performed either by humans or by tools with an objective to realize a certain business result"*.

Business process definition as in BPMN specification (2011) [86]: *"a defined set of business activities that represent the steps required to achieve a business objective. It includes the flow and use of information and resources"*.

Lindsay and Lunn (2003)[45] attempted to collect all business process definitions and listed a brief history about their related models.

2.2.3 Business Process Management (BPM)

Weske (2007) [6] says that *"the business process management includes concepts, methods, and techniques to support the design, administration, configuration, enactment, and analysis of business processes"*.

Business process and business process management system are parts of a larger development that has been affecting the design of enterprise applications. The developments in the enterprise software and business process management has led to workflow management.

2.2.4 Workflow

The Workflow Management Coalition(WFMC) [83] was founded in 1993 to bunch workflow related activities by adopters, developers, as well as a university and research groups engaged in workflow and BPM.

Workflow [48] is the automation of a business process according to a set of rules. Workflow management systems is a software system that manages the executions of the workflows. Workflow technology is efficient for supporting the business processes within a given application system or between a set of application systems.

Today, most enterprise application systems host the workflow components that customize a business process within these systems. In this study, we use a single application workflow which consists of activates and their ordering that is comprehended by one common application system. Figure 2.2 shows single application workflow systems architecture.

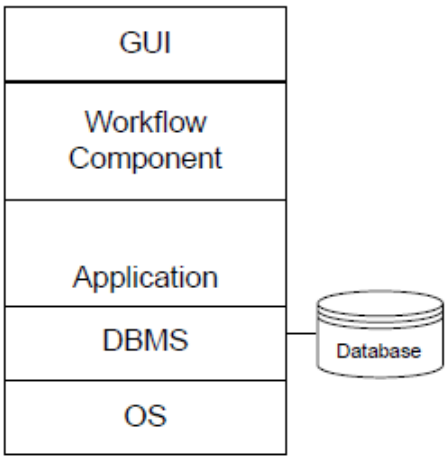


Figure 2.2: Single-application workflow systems architecture

2.3 Modeling

The specifications of something to be created are called models. Modeling is an important field in software engineering (SE). There are many types of models in SE. These include: process models, information flow models, design models, models of user interaction and data models [48, 9]. In the following subsections, we will explain the entity relationship diagram (ER)

which represents data modeling and business process model and notation (BPMN) which represents process modeling. These models are a core modeling throught in this study.

2.3.1 Data Modeling

Data modeling [69] is an analysis process of data requirements that are needed to support the business process. There are various types of data modeling; like conceptual, logical and physical data modeling. ER is a conceptual data modeling.

2.3.1.1 Entity Relationship Diagram

The ER [25] describes data as entities, relationships, and attributes. Entities are specific objects or things in the mini-world that have an independent existence. A relationship relates two or more distinct entities with a specific meaning. Attributes are properties used to describe any entity. Any entity will have a value for each of its attributes, and each attribute has a value set associated with it.
There are several types of the attribute in ER model: simple versus composite, single-valued versus multivalued, and stored versus derived.

- **Simple:** each entity has a single atomic value.
- **Composite:** the attribute may be composed of several components.
- **Single-valued:** a particular entity has a single value.
- **Multi-valued:** an entity may have multiple values for that attribute.

- **Complex:** composite and multi-valued attributes may be nested arbitrarily to any number of levels.
- **Stored:** actually stored attributes in the database.
- **Derived:** attributes that are not physically existed in the database and can be extracted from other stored attributes.

An entity type is a group of entities with the same basic attributes. An entity set is a collection of all entities of a particular entity type at a particular point in time. Entity type should have unique attributes called key attribute. The key attribute is deduced from the database mini world, and may be composite. An entity type may have more than one key.
The relationship of the same type is grouped into relationship type. The number of participating entity types is the degree of relationship type. A recursive relationship is an association of entity type and itself. The relationship may be binary, ternary, or more. The binary relationship between two entity types, the ternary relationship between three entities and so on. The maximum number of a relationship instances that make an entity can participate in the cardinality ratio. Cardinality ratio of binary relationship might make:

- One to one (1:1)
- One to many (1: N)
- Many to one (N:1)
- Many to many (N: N)

The existence of the entity depends on it is type related to another entity which called persistence constraints (total or partial).

- **Total participation:** every entity must relate to other entities in that relationship.
- **Partial participation:** every entity may relate to other entities in that relationship.

An entity type may take the form of a regular (strong) or weak entity. A weak entity is an entity that doesn't have a key attribute. It must use a foreign key in combination with its attributes to create a primary key. See table 2.1 [18].

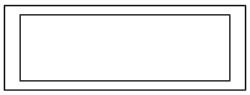
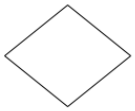
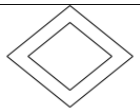
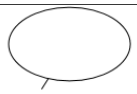
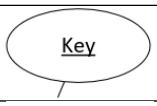
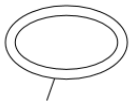
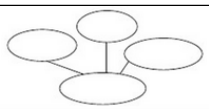
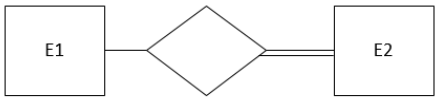
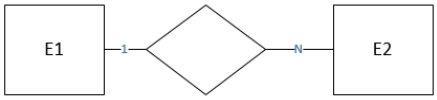
2.3.2 Business Process Modeling

Process modeling has been an active area from 1992 [71]. The business process modeling is an analysis of enterprise process activity. So, the process can be improved. Some business process modeling techniques are BPMN, Extended Business Modeling Language (xBML), Event-driven process chain (EPC) and Unified Modeling Language (UML) [33].

2.3.2.1 Business Process Modeling and Notation (BPMN)

In this study, BPMN is used. The two main reasons why BPMN is chosen in this study are: BPMN is currently a widely used notation to model business processes [16]. The BPMN notation is intended to bridge the communication gap between business process design and implementation [19].

Table 2.1: Entity Relationship Notations

Number	Symbol	Meaning
1		Entity Type
2		Weak entity
3		Relationship type
4		Identifying relationship type
5		Attribute
6		Key attribute
7		Multivalued attribute
8		Composite attribute
9		Derived attribute
10		Total participation of E2 in R
11		Cardinality ratio 1:N for E1:E2

BPMN was developed by object management group (OMG). BPMN have two versions, the first one released in 2004 (BPMN 1.0), the second version which was released in 2011 has been adopted in this thesis [66].

BPMN [86] are expressed using graphical notation. BPMN have four main categories of the element: Flow object, Connecting object, Swimlanes, and Artifacts. ***Flow objects*** are the building blocks of a business process; they consist events, activities, and gateways. See table 2.2.

Table 2.2: Flow Objects

Event	The Occurrence of states in the real world that are relevant to the business process. (start, intermediate, end)	
Activity	The work performed during a business process.	
Gateway	The split and join behavior of the flow of control between activities, events, and gateways.	

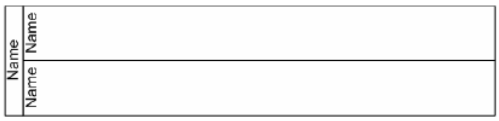
Connecting objects connect flow object, swimlanes, or artifacts. The connecting object includes sequence flow, message flow, and association. See table 2.3.

Table 2.3: Connecting Objects

Sequence Flow	Used to specify the ordering of flow objects.	
Message Flow	Describes the flow of messages between business partners.	
Association	Used to link artifacts to elements in business process diagram.	

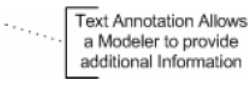
Swimlanes are used to represent organizational aspects in the business process diagram. Swimlanes are restricted to pool and lane. See table 2.4.

Table 2.4: Swimlanes

Pool	Represent organizations that participate in the interaction of multiple business processes.	
Lanes	Represent organizational entities.	

Artifacts are used to show additional information about the business process. BPMN support three type of artifacts: a data object, groups, and annotation. And these artifacts not directly relevant to the flow of the process. See table 2.5

Table 2.5: Artifacts

Data object	Used for documentation purposes.	
Groups	Provide additional text information for the reader.	
Text annotations	Used to group elements of a process.	

2.3.2.2 BPMN in detail

Events play an important role in the business process diagram. Events can be partitioned into three types: start, immediate, and end events. Start events are used to trigger the process. Immediate events can occur during the process. End events signal the end of the process. See figure 2.3

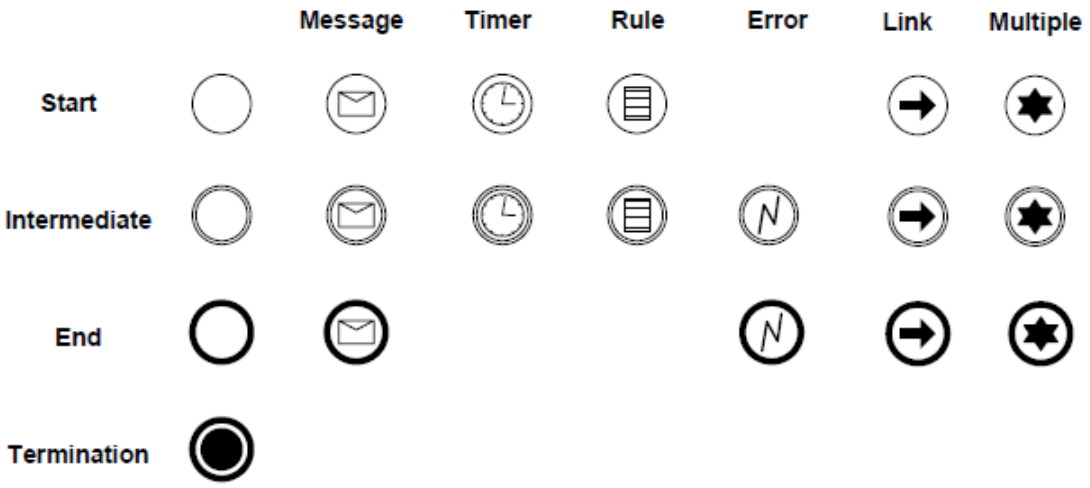


Figure 2.3: Event types in the BPMN, Object Management Group (2006)

Start event can have different trigger:

- None: No event trigger type is given. This is used when a subprocess is started.
- User: A user manually creates the start event of the process.
- Message: participants in BPMN can communicate using a message; one of the parties send a message, a message arrived as a message event in the recipient participant.
- Timer: the process starts when a specific date or a specific cycle will be triggered.
- Rule: if the rule evaluates to true, this event type will be triggered.

Intermediate events start during a business process. They are used to delay the execution of a process.

- None: can be used to signal a state change in the process.

- Message: the progress of the process depends on the message arrived from participants, when the message arrives, the process can continue.
- Timer: the event is triggered based on timer information.
- Error: the intermediate event can be used to represent exception handling. The exception may be thrown during the normal flow of the process. And this event is used to represent catching the exception. When the exception is caught, the related activity is started.

The purpose of the ***end event*** is obvious; the none marker refers to the completion of the process or subprocess. An error end event can be used to raise an exception. A termination end event is used to terminate all activities of a given process.
There are additional event types that relate to start events, intermediate events, end events.

- Link: it is a mechanism for connecting the end of one process to the end of another.
- Multiple: there are multiple ways of triggering the process.

The next main component in the BPMN is activity. Activity are unit of work in BPMN. There are two types of activities: atomic activity called task, or subprocess.
There are several types of tasks:

- Service tasks: is a task that using a web service or application program to software systems.
- Send tasks: is a task that sends a message.
- Receive tasks: is a task that waits to receive a message.

- User tasks represent traditional workflow tasks that involve user interaction.
- Script tasks: is a task that uses some scripting language expression in order to be performed.
- Manual activities: they are performed without the support of the software systems.
- Reference tasks: it provide a means to reuse tasks in a different business process.

A ***subprocess*** can either be expanded or to be collapsed, in which case a plus marker sites at the bottom of the subprocess.

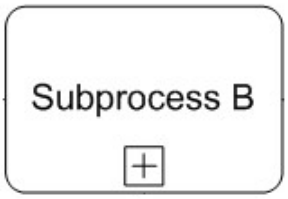


Figure 2.4: Subprocess Notation

Sequence flow used to connect flow object using solid arrows. There are different types of sequence flow: normal flow which represents expected the behavior of the process, exception flow represents exception situation and sequence flow induced by link events.

Gateways elements are used to control how sequence flows interact as they converge and diverge in a BPMN diagram. See figure 2.5.

Figure 2.5: Diffrent Gateway Notation

Each gateway works as a join or split node. Join nodes have at least two incoming arcs and one outgoing edge. Whereas split node has exactly one incoming arc and, at least, two outgoing edges.

There are different types of exclusive or splits depending on data and events. Data based exclusive is a gate with an associated condition. Once a gate condition is evaluated to true, the corresponding branch is taken, and the other condition is discarded. Event based exclusive or gateway enables multiple activities of type receive. If the first of these activities receive a message, the other activities are neglected. The inclusive or gateway is used in the situation where an arbitrary number of outgoing branches is selected. A complex gateway allows combine split and joins behavior.

BPMN is not restricted to single organization business process, but can be used to express interacting processes of multiple organizations. For interacting with processes purposes it uses swimlanes. Pools may represent specific process participants, or represent business entity roles. Lanes are used to represent organizational entities within participants.

2.4 Study Related technologies

In this section, we will present the technologies that have been adopted throughout the present study.

2.4.1 XML-based Modeling

Extensible markup language (XML) is one of the important models in computer science. XML simplifies the problem of data exchange between different applications [38]. In figure 2.6 [38] we show the problem of incompatibility between different applications which can be solved using XML as shown in figure 2.7 [38].

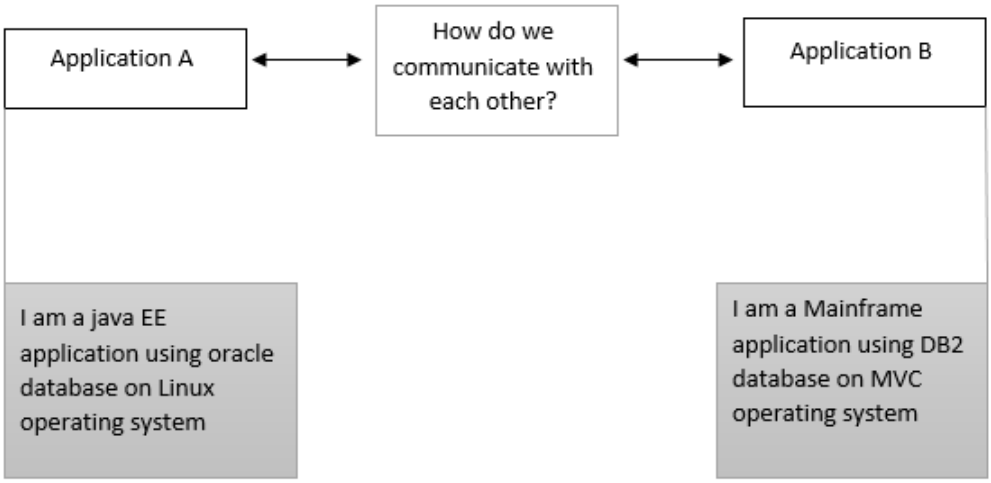


Figure 2.6: Problem of incompatibility between applications.

Figure 2.7: XML as the data exchange mechanism between applications.

2.4.1.1 XML Schema (XSD)

An XML Schema describes the structure of an XML document [31, 80]. The powerful features of XSD are: written in XML, support data type and namespace. We can use XML parser to parse XSD schema. The following presents XSD main elements:

1. The <schema>element [4]:

   schema is the root element of every XML schema. It has attributes like targetNamespace which indicates the XSD namespace.

2. The <element>element [5]:

   <element>is the main elements in schema. Names, data types, minOccurs, maxOccurs and default are the main attributes of element tags.

3. The <complexType>element [3]:

   <complex>element contains other elements with their specified attributes. <sequence>element can be used with complex element to indicate the order of child elements should appear in sequence as they declared.

We use XSD in this thesis to represent ER diagram.

2.4.1.2 XPDL

XML process definition interchange [84] is a standardized format by WFMC to interchange process definitions between different workflow products. WFMC defines the process definition as [20] *"The representation of a business process in a form that supports automated manipulation, such as modeling, or enactment by a workflow [or business] management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc. "* . Figure 2.8 shows the concepts of the process definition interchange. XPDL outlines a universal standard that enables products to support different internal representations of process definitions with an export/import function to map the standard at the product boundary. Metamodel describes the

entities contained in process definition, their relationship, and attributes.

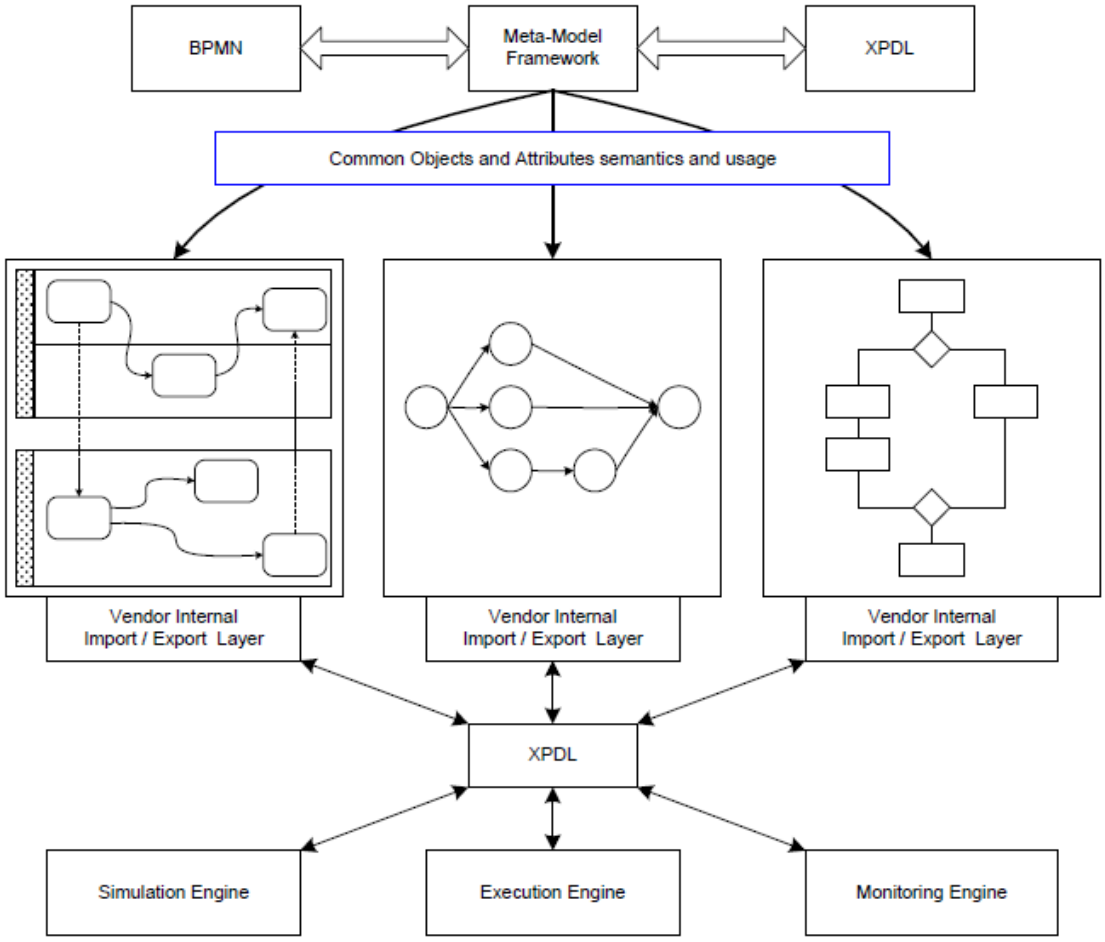


Figure 2.8: The Concept of the Process Definition Interchange.

The XPDL process and workflow process correspond to the BPMN Process. XPDL defines three conformance classes, SIMPLE, STANDARD, and COMPLETE. A modeling tool can choose one of these classes to import and understand all parts of a serialized BPMN to the class. All classes neglect vendor-specific extensions which can be used in Extended Attributes. The supporting BPMN in these classes exposes in the following lists.

1. **Simple class:**

   Task, collapsed subprocess, gateway (exclusive data-based, parallel), none start and none end events, pool, lane, data object, text anno-

tation, sequence flow (uncontrolled, conditional, default), and association.

2. **Standard class:**

   Task (User, Service, Send, Receive), collapsed and expanded subprocess, looping or multi-instance activity, gateway (inclusive, exclusive data-based, exclusive event-based, parallel),start events (None, message, timer), catching intermediate events in sequence flow (timer, message), throwing intermediate events in sequence flow (message), attached intermediate events (timer, message, error), end events (None, error, message, terminate), pool, lane, data object, text annotation, sequence flow (uncontrolled, conditional, default), and association.

3. **Complete class:**

   It includes all task types, all event types, and all gateway types, message flow, transactional subprocess, and ad hoc subprocess.

2.4.2 Windows Form Application:

Windows forms [62] is a platform for Microsoft windows development based on the .net framework. This framework provides a set of classes that can be used to develop windows application. A form is a rectangular screen, that we can use to show information to the user and to receive input from the user. Firstly, the Form is blank. Then we can add controls to the form like buttons, text boxes, menus, check boxes, radio buttons, etc.

When we create a form application in visual studio as a visual basic (VB) or C# project, there are three classes created in project folders. One of them is created to execute forms called "Program.cs" or "Program.vb". The other two classes ("Form.cs", "Form.Designer.cs") are created as a partial class.

Partial classes are used to split the definition of class into multiple source files [63].

2.4.3 Windows workflow foundation (WF):

A workflow in WF [17, 7] is an ordered series of steps that accomplish some work according to a set of rules. Workflows represent a declarative programming model. A declarative workflow model supports a good separation between what to do (real work) and when to do it.

We can solve a business problem using general purpose languages such as C# or Visual basic. Using a workflow add additional feature to implement enterprise domain specific language. For example, if the domain is banking and finance, we might refer to accounts, checks, loans, and so on. But if the problem domain is pizza delivery. for example, we might refer to menus, specials, phone numbers, address, and so on.

Workflows allow all companies to easily model system and human interactions. The workflow may be distributed in multiple computer or locations. Workflow is typically represented in a graphical manner.

Workflow in WF can be thought as the flow of processes or tasks that produce results. Workflow is concerned with a software system to define the flow of work, activities performed and rules employed.

WF is a framework support creating a running workflow applications on windows platform.WF contains a programming model, engine, and tools (including designers for visual studio). Workflow can be developed completely in a code or created in conjunction with XAML markup. A workflow instance is created and maintained by workflow runtime engine.

When a workflow model is compiled, it can execute inside any windows process including:

- ⋆ Console application.
- ⋆ Form-based application.
- ⋆ Windows services.
- ⋆ ASP.Net website.
- ⋆ Web services.

2.4.3.1 WF features and capabilities:

1. **Declarative activity model**

   The most important feature in WF. Activities and workflow build as hierarchal trees of activities. Some activities are leaf nodes; others act as containers for child activities. The order of executing activities depends on the location of activity in the tree. So the declarative model provides a separation between the work to be accomplished and the mechanism that coordinates that work. We will declare activities in XAML, an XML-based markup syntax that supports the declaration of objects and properties. However, WF also supports declare activity in code.

2. **Standard and custom activity**

   WF encourage a standard activity library. This library contains the most common building blocks that we will need to construct the activities and workflows. In WF, we can build custom activities not limited to standard activities library. custom activities can be implemented in procedurals code or declaratively using XAML. WF provides several base class that we can use to author custom activities. For example, the CodeActivity and AsyncCodeActivity.

3. **Workflow designers**

   Workflow designer is integrated into visual studio. This designer allows you to visually declare activities and workflow using familiar drag and drop interface. Standard and custom activity can be chosen from the Toolbox.

4. **Workflow services**

   Workflow services are windows communication foundation (WCF) services that are declaratively implemented as workflows. Client access the workflow view one or more endpoint. The workflow service is self-contained. WF supports a common message pattern of request/response, one-way, and duplex.

5. **Workflow hosts**

   The application code doesn't execute an activity directly. The runtime is responsible for the execution of the individual activities that have been scheduled for execution. Activities can be hosted within the application using one of the workflow host classes (WorkflowInvoker, WorkflowApplication, WorkflowServiceHost).

6. **Workflow extensions**

   To provide additional functionality to activity and workflow, WF provides an extension mechanism. These extensions are added to host classes.

7. **Persistence**

   Workflow persistence is the ability to save and reload the state of the workflow instance. This features can be used in the development of long-running workflows that may take minutes, hours, or days to complete.

8. **Bookmark processing**

   WF supports a bookmark which is a mechanism for temporarily suspend execution of a workflow and resume execution at a later time.

9. **Transaction support**

   WF provide the ability to declare a transaction within workflow model. The transaction allows ensuring consistency when performing update operations in a resource such as SQL server database.

10. **Compensation and exception handling**

    In WF, host application handles the exception within activity or workflow. Handling the exception locally provides a better opportunity to recover from failure.

11. **Workflow tracking**

    The most important feature of workflow tracking is that it is automatic. It is built into the workflow runtime infrastructure and can be enabled without changing workflow at all.

2.4.3.2 Activity

Activity is the basic element unit of the workflow. Everything in a workflow is an activity, including the workflow itself. We can consider a workflow as a collection of activities. Categorization of activities based on kinds of work:

1. **Atomic unit of work**

   This kind of activity performs an atomic unit of works that synchronously based on the workflow thread. WF provides a class named CodeActivity that should be used as the base class when developing custom activities of this kind.

2. **Asynchronous unit of work**

   This kind of activity performs an atomic unit of works that asynchronously depend on a separate thread. WF provides a class named AsyncCodeActivity that should use as the base class when to develop custom activities of this kind.

3. **Long running unit of work**

   This kind of activity performs works that may take a long time to complete. WF provides a class named NativeActivity that should be used as the base class when to develop custom activities of this kind.

4. **Control flow activities**

   This kind of activity doesn't perform work. Its a container of other activities such as Sequence, and While. WF provides a class named NativeActivity that should use as the base class when to develop custom activities of this kind.

Each activity has data that it works on. There are two kinds of the data:

1. **Argument**

   Arguments are the inputs and outputs of the activity. Input arguments are passed to the activity when it begins executions. Output arguments contain the results from activity. The InArgument <T >and OutArgument <T >generic classes are used to define argument.

2. **Variable**

   Variable defines internal storage within an activity. Variables are used to pass result from one activity to another. To define variable use Variable <T >generic classes.

   We can define a variable using a generic class or using CLR data type.

Activities usually differs If the activity is executed multiple times, each execution will point to a fresh set of variables. But if we use CLR variable, the value of member variable is maintained between executions.

The next tables show the standard activity that WF support.These standard activities can be find in a toolbox in visual studio. In the toolbox, the activities are categorized into groups. The group elements have a general common feature. So I explained these groups as they appeare in the toolbox. Table 2.6 shows the control flow, The activities in this category are used to control the flow of the execution within the workflow. Table 2.7 describes the state machine activity, state machine provide a modeling style with the workflow in an event-driven manner. Table 2.8 shows the flowchart. Table 2.9 describes the messaging that can be used in WCF. Table 2.10 shows the persist and terminate acivities. Table 2.11 display a basic activity such as Assign, Delay, and WriteLine. Table 2.12 shows the activity that are used in transactions process. Table 2.13 shows the activity that can be used to address a collection of data. In addition, Table 2.14 shows the activity used to handle the error.

Table 2.6: Standard Activity - Control flows

Control flows	
Sequence	Sequence activity is a container of other activity that implements a sequential execution pattern.
If	If the activity is used to model an if-then-else condition within the workflow.
Switch	Switch activity models C# switch statement.
While	While activity executes a child activity while a condition is true.
DoWhile	The dowhile activity works as while activity but the child activity executed at least once.
Parallel	Parallel activity schedules simultaneous execution of multiple child activities.
ForEach	ForEach activity executes a declared activity for each element in a collection.
ParallelForEach	ParallelForEach activity executes as ForEach but in parallel.
Pick and PickBranch	Pick activity executes one of the declared branches based on the recipient of an event. To use Pick activity, we declare multiple instances of PickBranch activity as children. Each PickBranch has Trigger and Action. Trigger determine an activity that waits for a continuation event after bookmark resumption. Action used to declare the activity want to execute when the trigger event is received.

Table 2.7: Standard Activity - State Machine

State Machine	
State	State in which a machine can be in.
Transitions	Transitions used to control the flow between states

Table 2.8: Standard Activity - Flowchart

Flowchart	
Flowchart	Flowchart activity is the basis for the flowchart modeling style. It maintains the collection of child nodes that have been declared.
FlowDecision	FlowDecision activity models a simple decision. FlowDecision has condition property.
FlowSwitch	FlowSwitch activity implements the logic in a switch activity for the flowchart modeling styles.

Table 2.9: Standard Activity - Messaging

Messaging	
ReceiveAndSendReply	This template models the service side of a request/response message exchange pattern. The template contains a correlated Receive and SendReply activity.
SendAndReceiveReply	This template models the requesting side of a request/response message exchange pattern. The template contains a correlated Send and RecieveReply activity.
Send	Send activity is used to send a request to a WCF endpoint.
Receive	Receive activity models the recipient of a one-way from a WCF client.
CorrelationScope	CorrelationScope activity is used to provide management of correlation between messaging activity.

Table 2.10: Standard Activity - Runtime

Runtime	
Persist	Persist activity is used to declaratively request that the workflow persists its current state.
Terminate	Terminate activity is used to declaratively terminate execution of the workflow.

Table 2.11: Standard Activity - Primitives

Primitives	
Assign	Assign activity allows setting a variable or argument to the new value.
Delay	Delay activity is used to set a timer within a workflow. When the timer expires, the Delay activity completes and execution of the remainder of the workflow.
InvokeMethod	InvokeMethod enables to call methods on an object instance or class.
WriteLine	WriteLine activity writes a string to the console.

Table 2.12: Standard Activity - Transactions and compensations

Transactions and compensations	
TransactionScope	TransactionScope activity protects the work that is done by a child activity with a transaction.
CompensableActivity	CompensableActivity is a container for child activities that require compensation. Compensation is the undoing of completed work.
Compensate	Compensate activity allows beginning the compensation process for an activity.
Confirm	Confirm activity allows beginning the confirmation process for an activity.

Table 2.13: Standard Activity - Collection Management

Collection Management	
AddToCollection	AddToCollection activity adds a new item to a collection.
RemoveFromCollection	RemoveFromCollection activity remove the item from a collection.
ExistsInCollection	ExistsInCollection activity is used to determine whether the item is found in a collection.
ClearCollection	ClearCollection activity clears all items from a collection.

Table 2.14: Standard Activity - Error Handling

Error Handling	
TryCatch	TryCatch activity models the familiar try/catch pattern in C#.
Throw	Throw allows to declaratively raise an exception within a workflow. It's similar in usage as the C# throw statement.

Table 2.15: Standard Activity - Migration

Migration	
Interop	Interop activity is used for interoperability with activities developed earlier WF.

2.4.4 LINQ

Language-Integrated Query (LINQ) [55] is a set of features in visual studio environments. LINQ extends robust query capabilities to the language syntax of C# and visual basic. LINQ introduce easily-learned patterns for querying and updating data, and the technology extended to support any kind of data store. Visual Studio contains LINQ provider assemblies that enable the use of LINQ with .NET Framework collections, SQL Server databases, ADO.NET Datasets, and XML documents.
In our thesis, we used two LINQ based technology, LINQ to SQL and LINQ to XML.

- **LINQ to SQL**[56]

  LINQ to SQL is a component of .NET framework that provides a runtime infrastructure for managing relational data as objects. The developer writes their query at object model expressed in a programming language. LINQ to SQL mapping the object model to the data model of a relational database. When the database returns the results, the LINQ to SQL translates back to the object model.

- **LINQ to XML** [57]

  LINQ to XML provides an in-memory XML programming interface that takes advantages of the .NET LINQ framework. LINQ to XML is an up-to-date, redesigned approach to programming with XML.

2.4.5 T4 Template

T4 text template [52] is a variety of text statements and control logic that can generate a text file. The control logic is written in a programming languages

such as C# or VB. The output (generated file) can be the text of any kind, for example, web page, resource file, or source code for any language.

There are two kinds of T4 text templates:

- **Run time T4 text templates:** the generated file outs after running the application.
- **Design time T4 text templates:** It is executed in Visual Studio to define part of the source code and other resources of an application.

In this study, we use the design time T4 template.

Chapter 3

Literature Review

3. Chapter Outline:

    3.1. ER related Mapping

    3.2. BPMN related Mapping

    3.3. BA related mapping

There are many attempts to map models into codes. This step is called automatic code generation. The generation step can reduce the time of building the business application. It can also increase the efficiency of developed code. Automatic code generation can be done as a direct way from models to codes. Alternatively, generate a code by using intermediate way such as XML. The usefulness of adopting XML as a middleware is working to the most of the programming languages after working with suitable XML parser. In the next section, we will mentioned recently papers that make a mapping between ER diagram to XML schema and vice versa.

3.1 ER related Mapping

In 2003, Arijit, Sriram and Rahul [74] described a method for translating from ER model to XML document type definition (DTD) and XML schema (XSD) and vice versa. The component of ER that described in XSD:

1. Entity mapped to an element in XSD; attributes to sub-element; but multi-valued attributes mapped to sub-element with minOccurs=1 and maxOccurs=unbounded; primary key mapped to attribute with type=ID.

2. They used sequence tag if it ordered entity attribute, else if the entity have unordered attributes they used all tag.

3. Relation mapped to an element with minOccurs and maxOccurs determine cardinality; but if the cardinality M: N they used complexType inside complexType.

4. Generalization mapped complexType inside complexType.

5. Weak entity, ternary relationship, and aggregation not listed in the paper due to that they had the same semantic as other concepts.

They implemented a prototype of the XER model using visual basic for an application (VBA) with Microsoft's Visio 2002.
Franceschet et al. [29] (2004) provided a methodology to create a specification of transaction programs and applications that independent of a run-time environment. In their methodology, they defined a formal specification to transaction programs and applications using CASE tool. Then they mapped the formal specifications to XML documents. After that, they applied XSL transformation onto XML document to produce the final document. In this way, the project developer can use any runtime environments.

Joseph and San (2004) [26] suggested a conversion method from relational schema into XML schema, the method that used for conversion seems to be:

1. Firstly, it convert from relation schema to extended entity relationship diagram (EER model) using classification table.

2. Secondly, they mapped EER model to XSD graph.

3. Finally, they map XSD graph to XSD.

They addressed the eleven rules to translation from EER model to XSD as in table 3.1. Before they mapped EER model to XSD schema, they choose the root of a tree (XML schema represented as a tree structure) and usually the root of a tree based on user input, then divide the entity into levels based on their cardinality. They used products of a factory as a case study.
There are other trials on this fields, but with different topics concentration. Anthony et al. (2005) [47] developed a visual tool that it is used in the mapping EER diagram to XML schema. The input of the tool is the relation schema of EER diagram. Users draw EER and make their modification on the constraints then create XSD file. However, Ivan et al. (2005) [49] provided a case tool to develop a complex database schema from conceptual

modeling. The CASE tool allows to declare the relation subschema. It then created a form type for each entity to modify the attribute, the relations, the keys ...etc. After that he recollects the database schema to be suitable to a database information system schema. Inspit of this, Chengfei and Jianxin (2006) [46] suggested algorithms and rules that mapped ER diagram to XSD schema. In this paper, authors focused on the quality of the output schema. They discussed several criteria in designing a good XSD schema.

Massimo et al. (2009) [27] proposed improved mapping method from ER to XSD schema. They focused on the following properties: the preservation of the information and the integrity constraints, the keeping of no redundancy and the different hierarchical views of the conceptual information. So the resulting XML structure is highly connected, and the design is reversible.

These papers [43, 21, 81] proposed a mapped method from ER to XML schema and vice versa. But they changed the attributes that they have addressed.

Some people build a tool based on the mapping from ER to XSD. Hsien and Shang (2006)[75] build XsEnvironment tool, which is a form based XML schema-driven. The input of XsEnvironment is XML schema that mimics the ER diagram. Then build another form based XML processg applications (XSD editor, database and web service components). The output of XsEnvironment is valid XML file or database. Figure 3.1 shows the major XsEnvironment components and their relationships.

Table 3.1: Mapping Rule between EER to XSD

Rule #	EER		XSD	Notes
1	Root relation		Root element	We can choose a relation that has 1:N cardinality and in isa a relationship.
2	Entity		Element	Multi valued attributes mapped to sub-element
	Attributes		Attributes	
3	Foreign key		Element/ sub-element	Primary key mapped to key tag, foreign key transformed to keyref.
4	Isa relationship		Element extension	Use complexType feature, and id needs some attribute from the super class we need restriction tag.
5	Generalization		Element extension	Tow type: overlap mapped to complexType with overlapped attributes. disjoint mapped to complexType with choice tag
6	Aggregation		Group sub-element, ref tag	create the whole relation included other sub-relation, then mapped sub-relation to the element and whole relation to element/sub-element nested and use ref tag to referencing for sub-relation
7	Categorization		Element/ sub-element	Inside the whole relation use choice and group tag and reference to sub-relation using ref tag.
8	Participation	Total	Element/ sub element	with minOccurs=1 and maxOccurs= "unbounded"
		Partial	Key/keyref	need these tag: selector and field
9	Cardinality	1:1	Element/ sub element	with minOccurs=1 and maxOccurs= 1
		1:N		with minOccurs=1 and maxOccurs= "unbounded"
		M:N		create new whole relation and reference to sub-relation using ref tag.
10	N-ary relationship		Group element	controls by type of cardinality
11	Unary relationship		Element/ sub element	create new element for self-relation and referenced to basic relation using ref tag and with maxOccurs= unbounded

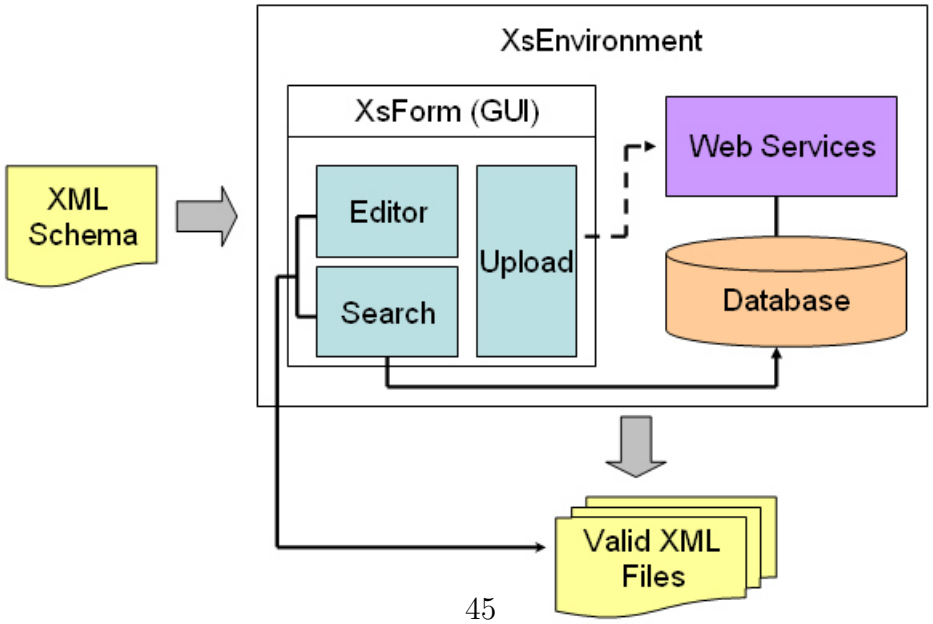


Figure 3.1: XsEnvironment components and their relationships.

Also, Lisa and Taratip (2012) [44, 32] suggested a tool that builds a database schema from the enhanced-ER diagram (EER). EER is converted to XML file using EDraw tool. Then it imports the XML file in their tool, which build the database schema and enables the designer to edit ER constraints. So the resulted schema is more applicable to the user requirements. Here, figure 3.2 shows the architecture of the tool.

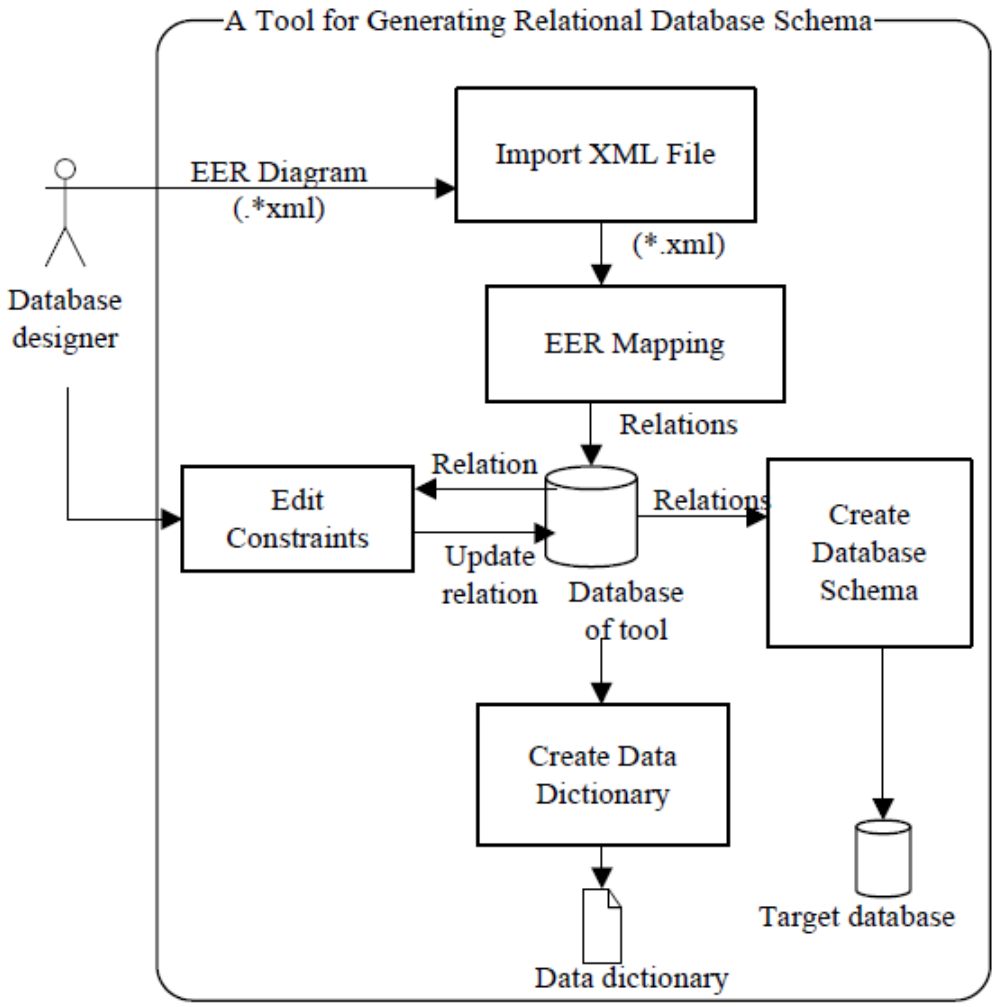


Figure 3.2: Architecture of Lisa and Taratip tool

ERDraw [88] is a tool for drawing ER visually then translating them to relational database schema automatically. ERDraw converts ER to XML file using ERML, a language based on XML and it converts ER to ERML format

which translates to relational schema.

The system architecture of ERDraw consists of five major modules: Graphical user interface to support the interaction between ER-diagram designers and ERDraw, ER-diagram object consists ER graphical object like rectangle, ER semantic object model: a data structure used to represents the complete semantic information of ER diagram, ERML object model: intermediate data structure that maps ER semantic model to ERML file, and finally the relational object model that is a result of translating from ERML format to relational database schema.

ERML language is based on XML and the structure of the document described using XML document type descriptor (DTD). The translating to database schema followed a set of the rule described in their paper.

3.2 BPMN related Mapping

Another important mapping is the mapping from BPMN to XPDL. WfMC provides XPDL documentation which describes in detail the mapping from BPMN to XPDL [84]. But Stephen [85] tried to simplify the mapping process as in table 3.2.

After exploring the mapped table between BPMN and XPDL, White take a customer order electronically as a BPMN workflow example because the business process diagram of this example has many features of BPMN so we can map many features to XPDL for explanation purposes.

XPDL proposed by WfMC was used in the WfMC-related applications, especially workflow whose concepts are currently compatible with those of business processes. Jung et al. (2004) [37] proposed a mapping method from BPMN to XPDL to support different process engine.

Table 3.2: Simplify Mapping BPMN to XPDL

#	**BPMN object**	**XPDL tag name**
1	Pool	<WrokflowProcess/>
2	Start/End,Event	<Activity> <Rout/> <Activity/>
3	Sequence Flow	<Transition/>
4	Task	<Activity> <Implementation> <Tool/> <Performer/> </Implementation> </Activity>
5	Sub Process	<Activity> <Implementation> <SubFlow/> </Implementation> </Activity>
6	Intermediate,Event	<Activity> <Implementation> <TransitionRestriction> <Split Type="XOR"> </TransitionRestriction> </Activity> Combined with a: <Transition> <Condition Type="EXCEPTION"> </Transition>
7	Decision	<Activity> <Route/> <TransitionRestriction> <Split Type="XOR"/> </TransitionRestriction> </Activity> Combined with a: <Transition> <Condition/> </Transition>

XPDL is based on workflow environments. There are several attempts to mapping the BPMN and XPDL to the current workflow environments

Table 3.3: Mapping BPMN to WF

Number	XPDL Model	WF model
1	Data Item or Data Object	Variable
2	Exclusive split	Flow Decision or Flow Switch
	Exclusive Merge	WriteLine
3	Parallel Split	Parallel
	Parallel Merge	WriteLine
4	Inclusive Split	Parallel containing If in its branch
	Inclusive Merge	WriteLine
5	Complex Split	Parallel containing If in its branch
	Complex Merge	IF
6	Data Type	
	INTEGER	Int
	FLOAT	Float
	BOOLEAN	Boolean
	STRING	String
	DATE, TIME, DATETIME	DateTime
7	Throwing compensation event	Sequence {WriteLine, Compensate}
	Catching Compensation event	Sequence {Body(contain the task that has compensation token), Compensation handler {event, task}}
8	Block subprocess	WriteLine,Flowchart
	Expanded subprocess	WriteLine,InvokeMethod, TargetType: ExecuteSubProcess, MethodName: CallProcess
9	Sequence flow connection	Flow step and Next

with a failure to adopt these mappings. Petia et al. (2006) [87] suggested an analysis paper that showed the elements in BPMN agree with workflow pattern. Then they mapped a BPMN to workflow pattern. We consider The following elements in the mapping process: basics BPMN, basics control flow, data element and references element. Wil MP (2003) [79] is another analysis paper, but the paper analyzes XPDL elements to workflow pattern. In 2014, Saran and Wiwat [89] proposed a methodology to map a BPMN to workflow patterns.

Additional significant mapping is outlining between some of the BPMN to

windows workflow patterns. J. Deng et al. (2012)[22, 23] suggested a methodology to transform from BPMN (represented by XPDL) to WF styles. They proposed a mapping system for the data object, data item, compensation event, task, subprocess, and data-based gateway (Parallel, Inclusive, Exclusive, and complex). See table 3.3.

3.3 BA related mapping

Business applications evolve dramatically, but their functionality slightly change. So we need a ready template for this functionality.
In 2005, Xiyong and Xingwang [91] suggested a template based framework for customization the service in e-business application. Software mass customization is a powerful business model; it focused on producing and monitoring software product. Mass customization of the software product has been always in the form of the software package, which fit nearly all business process in almost all organization.
E-business software mass customization involves: I) Functionality of e-business. II) Interface or presentation of e-business application. III) IT infrastructure like the operating system supporting the software. and IV) Other nonfunctional attributes such as data integrity.

Mass customization focuses on identifying common data and procedures of e-business. And the main techniques used for customization are standardization, modularization, automatic code generator, and modeling.
Authors claim that component based technique seems not to be an efficient way for mass customization, so they used service oriented architecture (SOA), which is a software model that compromise three primary parties: provider of a service, consumer of a service and a directory of service. The services

are used to divide the large application into smaller distinct modules. The services can also be integrated via composition mechanism to create a large application.

Template has nearly completed application with the capability of the reusable component. In this paper, they build a template based framework which contains five parts: business process modeling tools, business service specification tools, the template definition tools, the code generator, and a directory of application services. See figure 3.3.

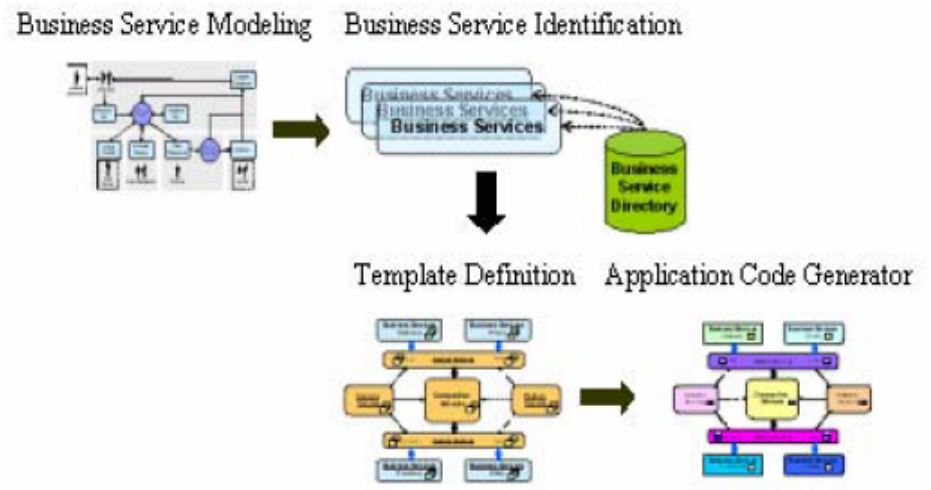


Figure 3.3: Template-based framework for e-business application mass customization

Kulkarni and Reddy (2008) [40] developed an approach to developing a business application based on modeling language. They classified the implementation code of business application into business logic, design strategies, architecture, technology platform, and GUI event handling. They developed a set of tools to give platform-specific implementation from models and high-level specification.

They used UML and UML metamodel to separate the function concerns from the technology concerns of business application. The business applica-

tion includes three classes of functionality, namely online, batch, and reports. Authors restricted their work on online functionality; since online functionality was implemented using layered architecture consists of the business logic layer, presentation layer, database layer.

The business logic layer is the part of a program that represents the real world business rule that determines how data is created, viewed, stored, and modified. To model the business logic, they used the following diagram:

- Class diagram to extract entity and relationship.
- Use-case diagram to extract business scenario.
- Activity diagram to describe business process flow.
- Extend class diagram in order to capture architectural information.

Then they designed a model aware high-level procedural language called Q++ to guarantee the business logic specifications to be consistent with the models.

Presentation layer is a layer related to design a screen that determines the data element of a business entity to display using controls, business service to invoke, etc. This means that the presentation layer deals with a set of user interaction patterns in business application e.g.; create/delete/update business entity screen. They used UML metamodel to specify presentation layer in terms of special abstraction control called window and windowType.

Database layer interested in the relational database schema and complex database accesses which can be specified using extended UML metamodel. They supported many object-relational mapping strategies in their models.

Kulkarni and Reddy (2002) used UML metamodel to combine the above three layers (business logic, presentation, and database) see figure 3.4. They also used the template-based technique to generate code from models.

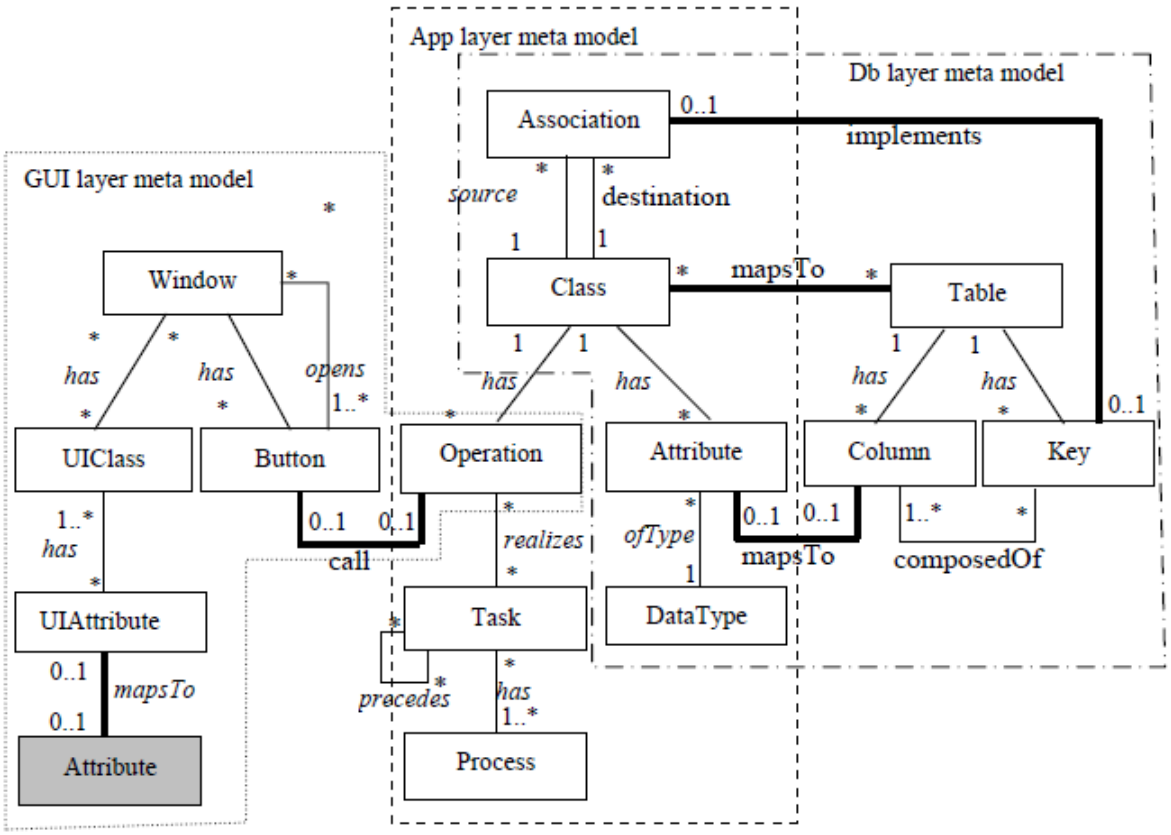


Figure 3.4: A subset of the unified meta-model.

Sonja et.al (2012) [73] provided an extension of IIS*Studio which was a form driven approach to generate the business application. They build IIS*UIModeler, the aim of this modeler to model user interface (UI) templates. Graphical user interface (GUI) is important aspects in a business application which contains static aspects (controls like button, screen) and dynamic linkage aspects (functionality like create database operation). GUI modeling language should support both aspects.

There are many approaches to describe GUI; it might be described by using the XML-based approach like XML user interface language (XUL) and extensible application markup language (XAML), or it may be described using extend UML metamodel. In their paper, they used user interface markup language (UIML) to model GUI.

GUI modeling language of IIS*studio has a specification of three classes: I) GUI displayable elements, II) Visual interpretation of displayable elements, III) Linkage structure between screen forms. GUI specification stored into IIS repository and this specification may be divided into three packages: core, business application, UI template. GUI static and dynamic aspects are stored on core and business application packages, but UI template contains attributes valued for common UI like screen size.
In modeling UI in IIS*studio, they depend on form type which was the abstraction for business form, form type that has a dual core; it may be an input to database or it may be a source for generation transaction program and its screen and report forms. So form type contained information to provide GUI code generation. To provide functionality specification between screen forms, they used business application diagram.
IIS*Studio generated UIML specification for the business application. Then they sloshed UIML specification into java AWT/Swing components using Java Renderer interpreter, the generated components represent GUI static aspects. To interpret the dynamic aspects of GUI, they used custom Java components.

Patrick and Kappel (2004) [42] suggested a transforming XML Schemas into Java Swing GUIs. Anup Kumar and RK (2009) [10] provided a formal framework to generate a code from activity diagram. For other works in BA mapping refer to [50, 14, 24, 15, 72, 41, 8, 70].
Automated generation of business application is very important to reduce the development time and to increase the efficiency of the application. It is obvious that previous related works lake a complete automated generation of business application. We can show attempts to convert models to code,

but these attempts still revolve about creating a database from the model and from the structure of the data in the database we can use to create an application. Also, the suggested automated generation of business application might restrict itself to describe the data without considered the business process model.

Chapter 4

Data and Methodology

4. Chapter Outline:

    4.1. Data Awareness

        4.1.1. ER models examples

        4.1.2. BPMN models examples

    4.2. Research Methodology

        4.2.1. Problem Statement

        4.2.2. Theoretical Framework

        4.2.3. Implemented Framework

        4.2.4. Evaluation

        4.2.5. Conclusion

This chapter describes the modeling examples that used to improve the goals of this thesis. The steps of the thesis project are also described .

4.1 Data Awareness

There are many models examples that can be used in this thesis, ER models examples, and BPMN examples.

4.1.1 ER models examples

1. Company example from [25] book. Figure 4.1 shows company ER models.

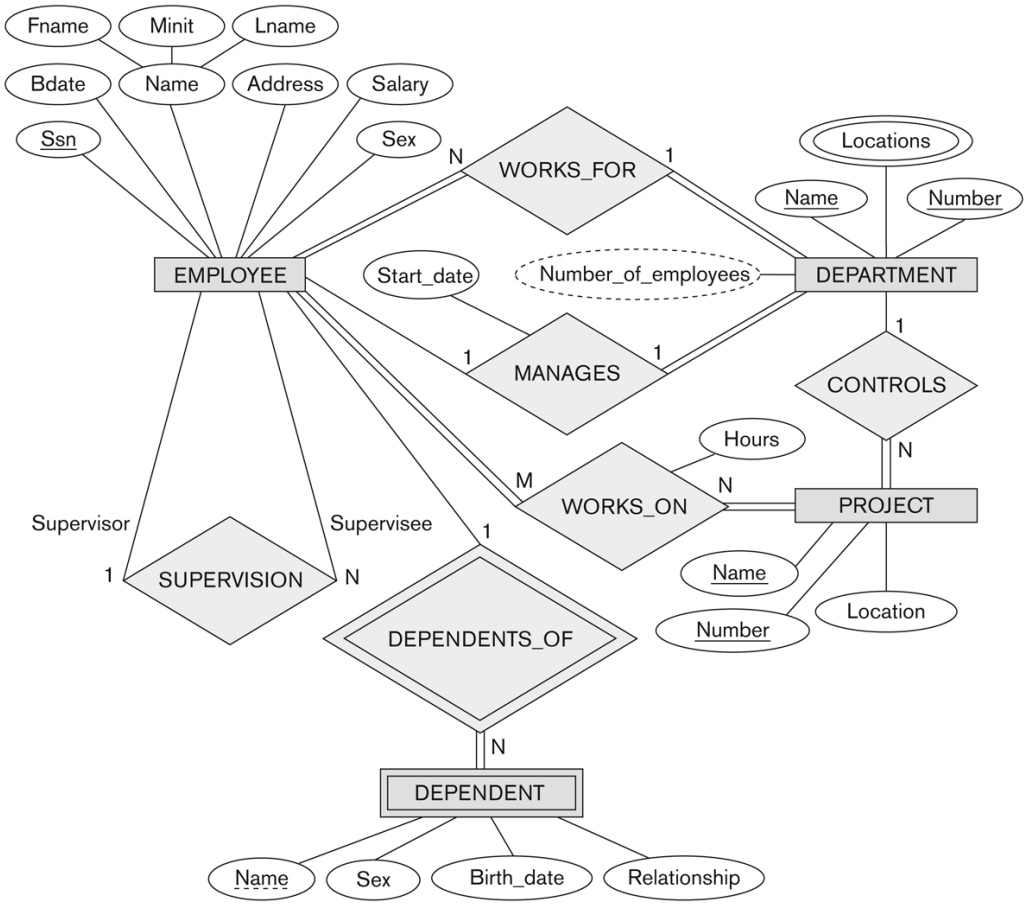


Figure 4.1: ER diagram for Company schema.

2. College Management System [64]. As show in figure 4.2.

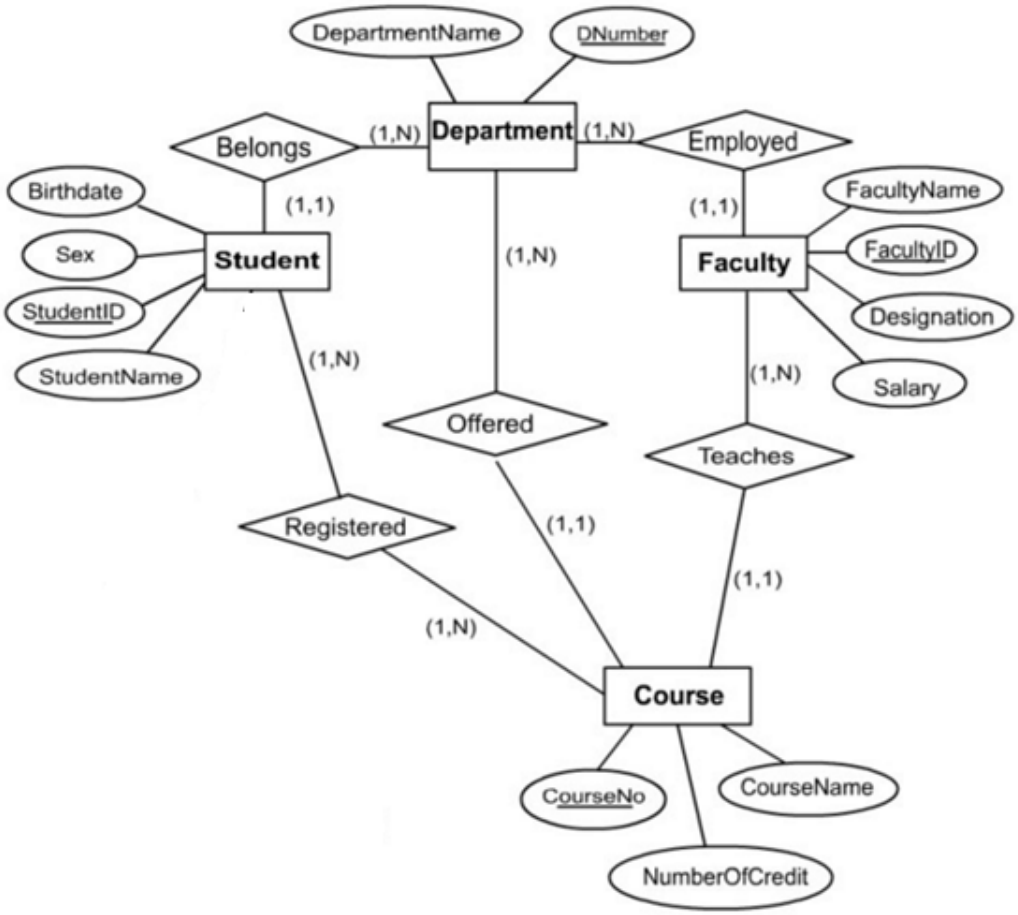


Figure 4.2: ER diagram for College Management System.

After we get the ER diagram, we convert this diagram to XML schema using existing tools such as Enterprise Architect [77], QSEE-Superlite [2] and ERDraw [88].

4.1.2 BPMN models examples

Thera are many business process examples, which can be found in Business Process Incubator [34], Bizagi [12], and BPMN Handbook [86]. Furthermore, there are many tools to convert from BPMN to XPDL such as Cloud apps from BPI [1], Bizag application [11], and Bonita software[13].

1. Basic BPMN example

We draw all basic elements (simple class of XPDL) BPMN in Bizagi tools as shown in figure 4.3. Then convert the diagram to XPDL.

2. Bank Account Opening.

This template is taken from BPI website [34]. Figure 4.4 describe the diagram.

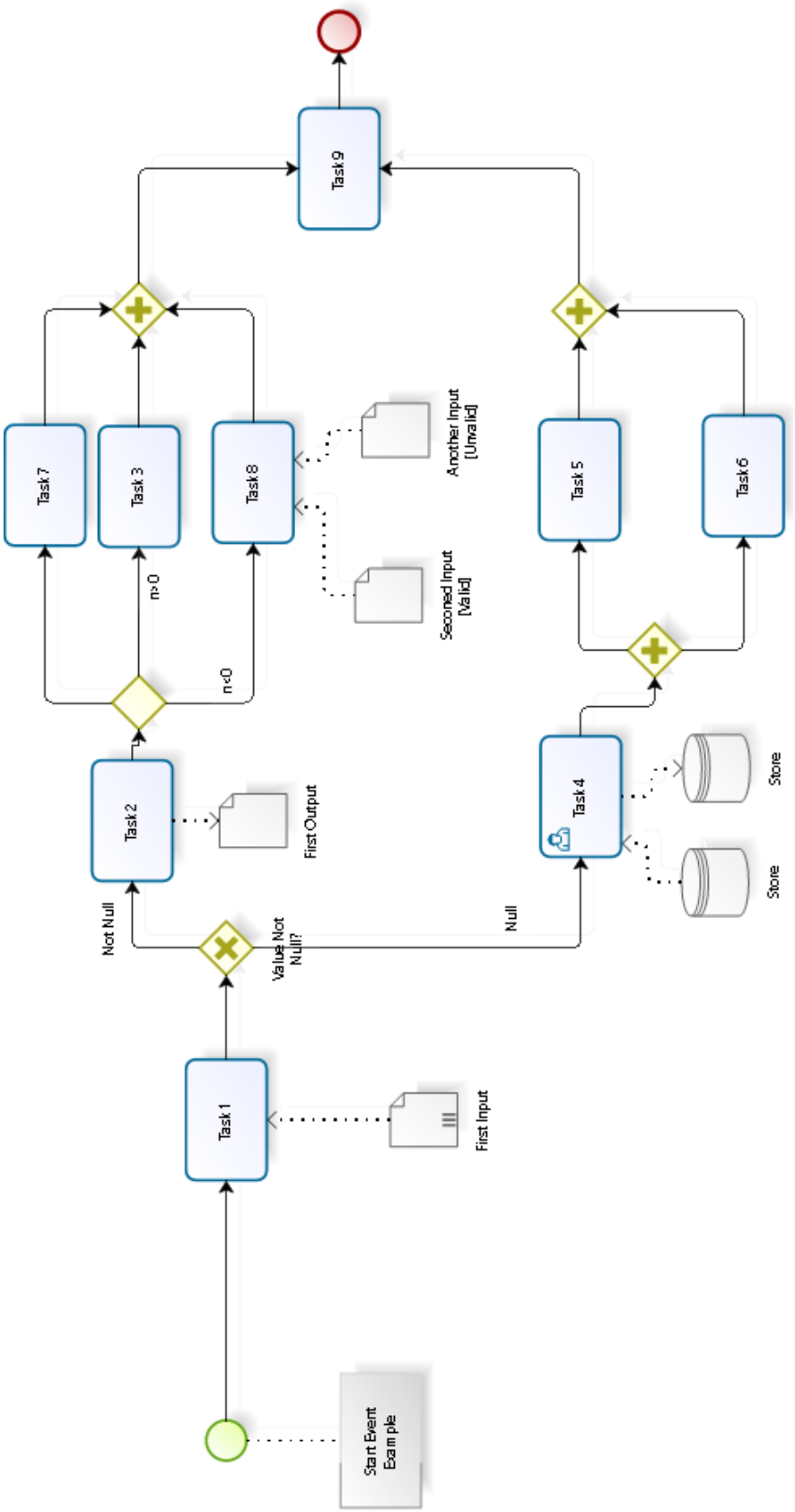


Figure 4.3: BPMN example- basic elements.

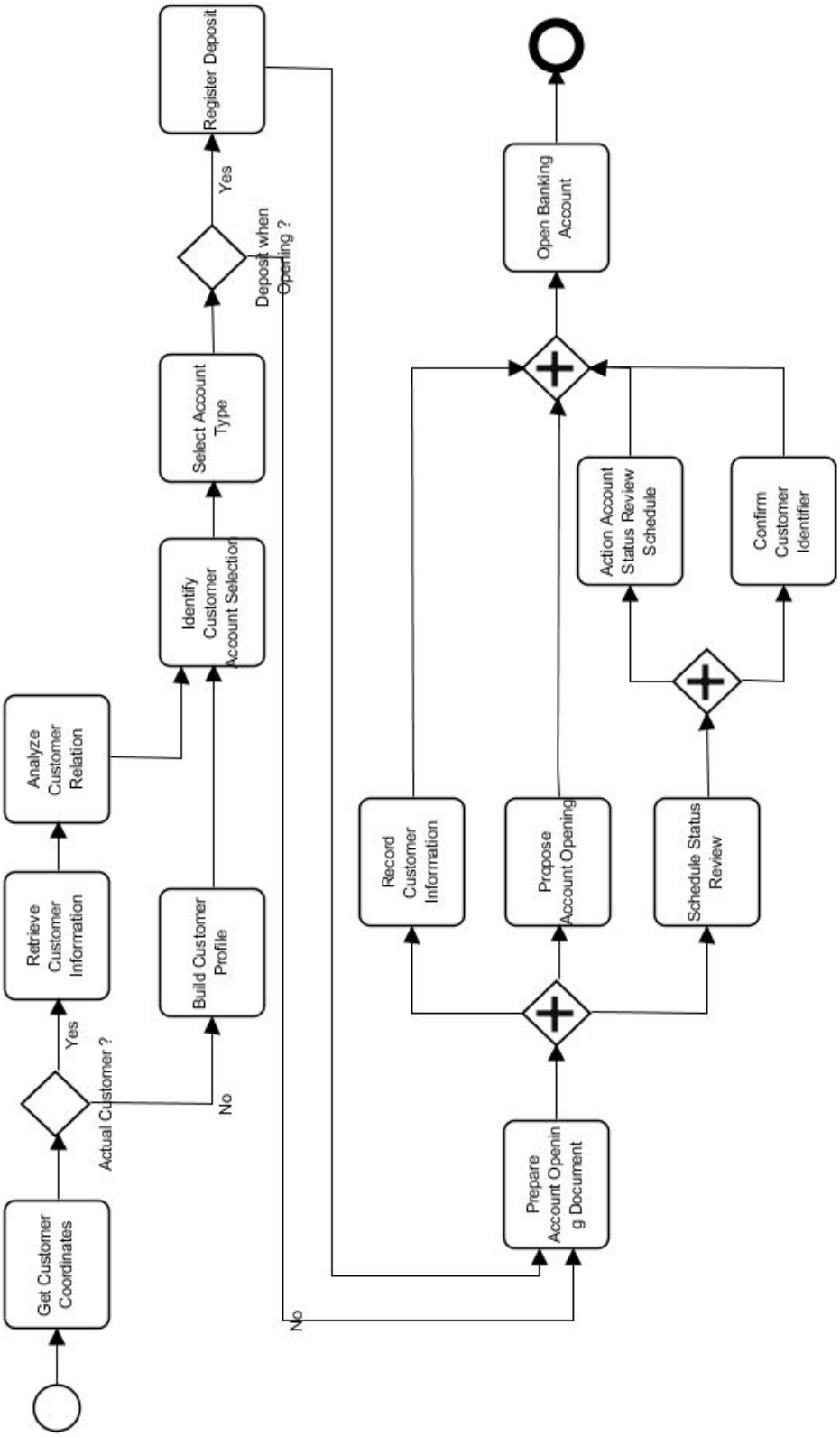


Figure 4.4: BPMN Bank Account Opening.

4.2 Research Methodology

The steps of the thesis methodology and the relations between them are shown in the figure 4.5. The scenario of each step in this thesis is explained as in the following subsections.

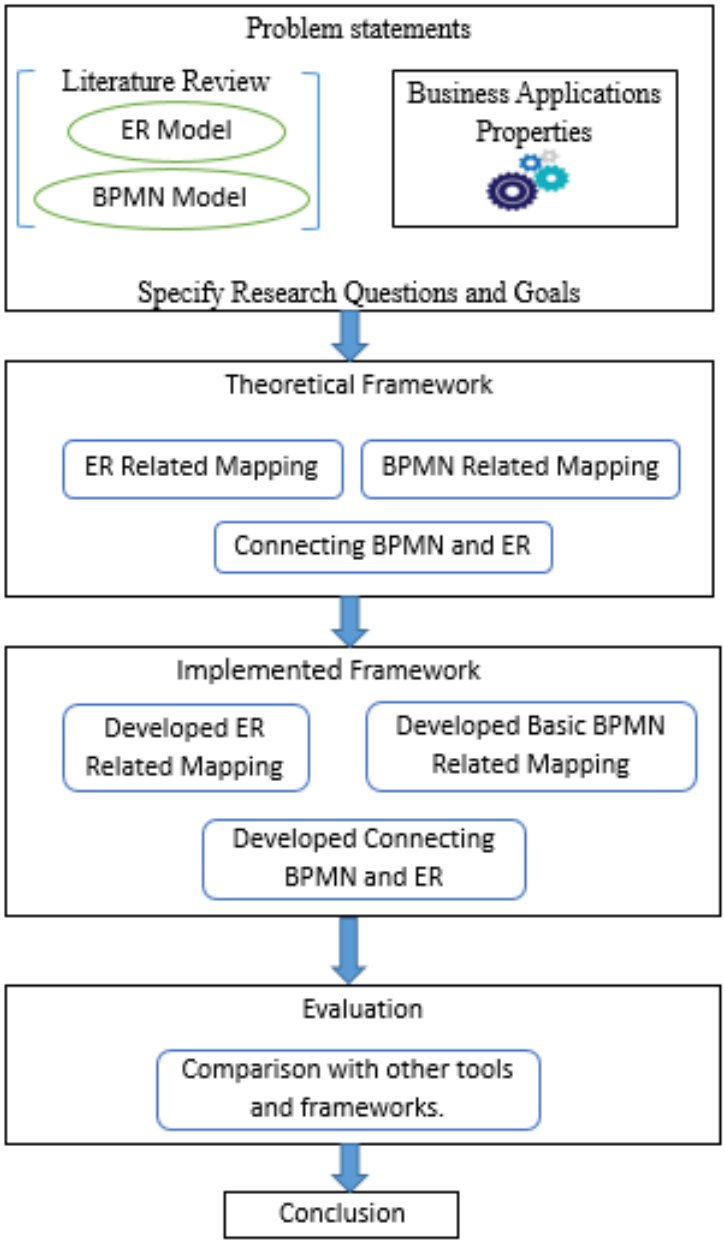


Figure 4.5: Thesis Methodology

4.2.1 Problem Statement

The first step in methodology is the perception of the problem, where the literature review relates to within the limits of this study. We make a literature review as follows:

- Understanding the business application by briefing the terms and concepts regarding to the properties and functions of business application.
- Gathering information about BPMN that are used to represent the business process modeling and ER model that are used to represent the conceptual modeling.
- Reviewing the related information to the automated code generation using modeling language. Making an awareness about mapping the Chen-ER and BPMN to XML-based technology.

After gathering information and making the literature review, we infer some of the questions about the study statement. The problem statement of this research can be defined through the following research questions:

- How can BPMN be improved to handle the data involved in the business process without increase the complexity of BPMN?
- How can BPMN benefit from ER model to represent data?
- How can we make a combination between BPMN and ER models to attain the process and their data handled inside the process diagram?
- How can we get benefit from a combination of BPMN and ER models to build a useful business application?

Data in BPMN is represented using artifacts, and data structures are discarded [16]. "Data object are considered artifacts because they dont have

any direct effect on the sequence flow or message flow of the process, but they do provide information about what activities require being performed and / or what they produce" as mentioned by the standard. And this statement is questionable because data object is important factors on the flow of the process.

4.2.2 Theoretical Framework

A suggestion of the approach is given in the second process step. This phase can be broken down into multiple sub-steps. Firstly, the theoretical framework is proposed. From the literature review step, we show that the BPMN has a problem in describing the data.
The data in the BPMN is represented as artifacts, which means that the data don't have any direct effect on the sequence flow or message flow of the process. But they provide information about what activities require being performed and / or what they produce. And this statement is questionable because the data object is important factor on the flow of the process.

Because the data is very important in building the business application and the BPMN consider the data as artifacts, we suggested to use the Chen-ER diagram in our framework. Chen-ER models provide a conceptual data about the business entities without increase the overhead to the BPMN. A combination process between BPMN and ER did by the mapping method as follows: mapping ER to SQL file, classes files, and windows forms. Then mapping BPMN to windows workflow pattern. Finally, make a combination between BPMN and ER using generated classes files.

4.2.3 Implemented Framework

The development framework is done after the theoretical framework is describing . We choose the C# windows forms as a development environment for many reasons [51] . Firstly, a C# is a robust .Net language. Secondly, there are many business applications developed using this environment. C# support workflow based application, and workflow service application.
The development step covers these theoretical details proposed in the previous step:

- Mapping ER to SQL file, classes files, and windows form.
- Mapping basic BPMN notations to workflow patterns.
- The combination between ER and BPMN.

4.2.4 Evaluation

The evaluation is being done after the development step. The first evaluation of the proposed framework is done by applying it in a real-world scenario to build a business application. The expression "real world scenario" means that the ER diagram and BPMN are chosen by real examples and projects. The second evaluations are done by comparing thesis work to other tools and frameworks.

4.2.5 Conclusions

The last step in the approach which involves the conclusions of the work. The conclusions will proof the goals and objective we demand to achieve them. Moreover, the future works of this thesis also is described.

Chapter 5

Theoretical Framework

5. Chapter Outline:

    5.1. General overview of theoretical framework

    5.2. Detail of the Suggested Framework

        5.2.1. ER related mapping

            5.2.1.1. Mapping ER to SQL Server File.

            5.2.1.2. Mapping ER to C# classes files.

            5.2.1.3. Mapping ER to GUI components.

        5.2.2. Mapping BPMN (using XPDL) to Workflow

This chapter describes the proposed framework. Initially, the general view of the framework and what layers it support is described. Then the architecture of the framework is presented and explanation of each step is illustrated. Furthermore, suggested algorithms about the mapping process of ER and BPMN is presented.

5.1 General overview of theoretical framework

Figure 5.1 shows the supported layers in this framework. This framework called ERWFFW . We have three layers: Data layer, Business process layer, and the presentation layer. Data layer has a conceptual model represented by the ER diagram which used to create database tables and constraints and GUI. Business process layer have a BPMN models to create WF and call GUI forms build by data layer. Finally, the presentation layer contains the GUI build by data layer and called by business process layer.

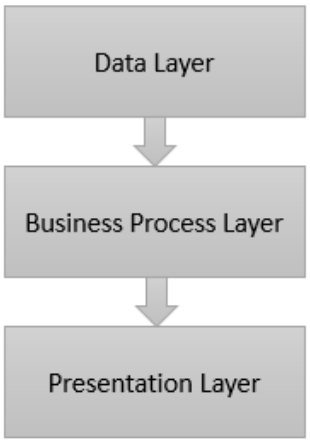


Figure 5.1: General Overview of ERWFFW Framework

5.2 Detail of Suggested Framework

In the figure 5.1, we show the general layer of the thesis framework. In this section, we will present the suggested framework in details as shown in figure 5.2.

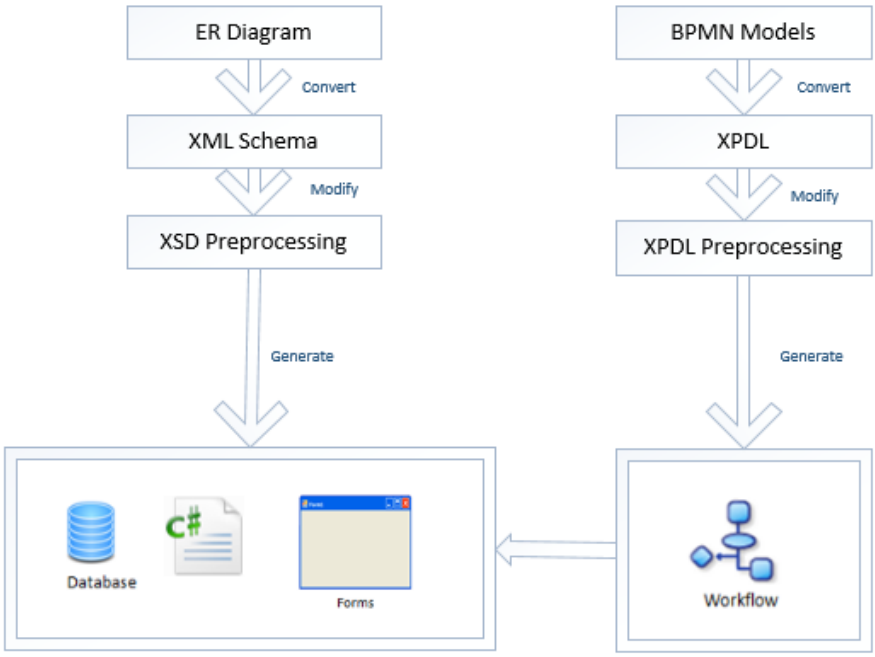


Figure 5.2: architecture of the thesis framework

From figure 5.2, we have two inputs to the system: Chen-ER diagram which represents the data models and the BPMN which represent the business process models. The Chen-ER, BPMN converted to XSD, XPDL in respective order. After that, the XML format of the model needs a preprocessing of some of their properties to be suitable in the framework. More information about the preprocessing steps will be shown in the implmented framework. This architecture focuses on three critical areas: Chen-ER related mapping, BPMN related mapping and combination process of BPMN and Chen-ER. The next subsections will describe these areas in detail.

5.2.1 ER Related Mapping

ER model has been proved to be a benificial technique but not only in the field of database design. ER is a graphical representation that is easy to understand among users and developers. This section shows a proposed method for generating SQL file, classes files and a graphical user interface from ER diagrams, written in XML schema format. Initially, we convert ER to XSD using existing tools that mentioned in the background chapter. Then the outputted XSD file undergos to the mapping rule suggested above in the study.

5.2.1.1 Mapping ER to SQL Server File

The steps to create SQL file are described in the following method:

1. For each regular entity in ER, create a table in SQL file.
2. For each regular entity attributes,
    a. If the attribute is simple, create a column in their tables in SQL file.
    b. If the attribute is composite, simplify the attribute, then create a column in their tables in SQL file.
    c. If the attribute is single-valued, create a column in their tables in SQL file.
    d. If the attribute is multi-valued, create a table in SQL file.
3. For each regular entity attributes, convert XSD datatype to SQL Server data type.
4. For each weak entity in ER, create a table in SQL file.

5. For each weak entity attributes, do the steps a to d.

6. for each primary key attribute, map to primary key constraints in SQL file.

7. For each foreign key attribute, map to foreign key constraints in SQL file.

8. For each unique attribute, mapped to unique constraints in SQL file.

9. For each M: N relation, refer to step 1-8.

The previous steps show that the mapping algorithm does not deal with the relationship between entities. We consider that this mapping is used when we convert ER to XSD file.

5.2.1.2 Mapping ER to C# Classes File

The generated classes in this work are not ordinary C# classes, but they are a LINQ to SQL classes. LINQ to SQL classes link the ordinary classes with the data source such as SQL server. More development criteria can be found in the chapter (6,7+8). The following steps show the mapping process from ER components to classes fields, properties, method, etc.:

1. For each regular entity in ER, create a C# Class file.

    a. The name of the entity is the name of the C# class file

    b. The name of the entity is the name of the class.

2. For each regular entity attributes,

    a. If the attribute is simple, create a field in their classes file.

b. If the attribute is composite, simplify the attribute, then create a field in their classes file.

c. If the attribute is single-valued, create a field in their classes file.

d. If the attribute is multi-valued, create classes as in step 1.

3. For each regular entity attributes, convert XSD datatype to C# data type.

4. For each weak entity in ER, create a class file.

5. For each weak entity attributes, do the steps 2.a to 2.d.

6. For each M: N relation, refer to step 1-8.

Other constraints used to modify these steps to "LINQ to SQL classes", which described in the development process.

5.2.1.3 Mapping ER to GUI components

There are four steps for mapping an ER diagram to GUI components as follows:

1. Mapping Regular entity

2. Mapping Weak entity

3. Mapping 1:1, 1: N, M: N binary relationships

4. Mapping of N-ary relationships

Details of mapping described in the next points.

1. Mapping Regular entities

A. A new Form is created for each regular entity

a. The name of the entity is the name of the form.

B. Attributes:

a. Simple Attributes: Create Labels and Textboxes for each attribute.

b. Composite Attributes: Grouping attributes with group box titled with main attribute and Create Labels and Textboxes for every component.

c. Single valued Attributes: Create Labels and Textboxes for each attribute.

d. Multi-valued Attributes: A new Form is created with labels and textboxes for each attribute.

e. Complex Attributes: Apply b+d.

2. Mapping Weak entity

A. A new Form is created for each weak entity.

B. Attributes: As in regular entity, but create combo box for the partial key attribute.

3. Mapping 1:1, 1: N, M: N binary relationships

A. 1:1 binary relationships

Proved in the conversion method to test referencing data. Such as the foreign key of the relation at one side contain the primary key of the relation at another side.

B. 1: N binary relationships

Proved in the conversion method to test referencing data. Such as the foreign key of the relation at side N contain the primary key of the relation at side 1.

C. M: N binary relationships

a. A new form is created.

b. Attributes: create labels and textbox's but the primary key of a relationship (foreign key of N and M side) should appear as a combo box or Listbox.

D. Mapping of N-ary relationships

Create a new form for new relation.

Now we will present the mapping process from BPMN to windows workflow pattern.

5.2.2 Mapping BPMN (using XPDL) to Workflow

There are three classes of XPDL; simple, standard, and complete. In a theoretical framework, we use a standard class to map BPMN to workflow.

1. Task

a. Normal Task

To every atomic task →create a custom activity that using the CodeActivity as a base class. The name of the activity in WF is the value of the name fields of the activity in XPDL file. Figure 5.3 shows the custom code activity against with BPMN.

```
public sealed class Task1 : CodeActivity
{
    // Define an activity input argument of type string
    public InArgument<string> Text { get; set; }

    // If your activity returns a value, derive from CodeActivity<TResult>
    // and return the value from the Execute method.
    protected override void Execute(CodeActivityContext context)
    {
        // Obtain the runtime value of the Text input argument
        string text = context.GetValue(this.Text);
    }
}
```

Figure 5.3: Maping from atomic task to CodeActivity.

b. User Task

To every user task →create a custom activity that using NativeActivity as a base class. Create a bookmark to handle the data needed in user interaction. The name of the activity in WF is the value of the name fields of the activity in XPDL file. Figure 5.4 shows the custom code native activity against with BPMN.

```
public sealed class Task1 : NativeActivity
{
    OutArgument<string> name;
    public OutArgument<string> Name
    {
        get { return this.name; }
        set
        {
            this.name = value;
        }
    }
    protected override void Execute(NativeActivityContext context)
    {
        context.CreateBookmark("input", new BookmarkCallback(this.OnBookmarkCallback));
    }
    void OnBookmarkCallback(NativeActivityContext context, Bookmark bookmark, object obj)
    {
        this.Name.Set(context, (string)obj);
    }
}
```

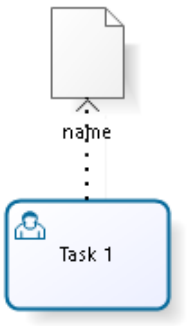


Figure 5.4: Maping from User task to NativeActivity.

c. Send Task:

d. Receive Task:

e. Service Task:

Service/ Send/ Receive →create WCF code to implements these tasks. The name of the file is the same name of the task. The extension of the WCF file is xamlx. Figure 5.5 shows the outputted file against with BPMN element.

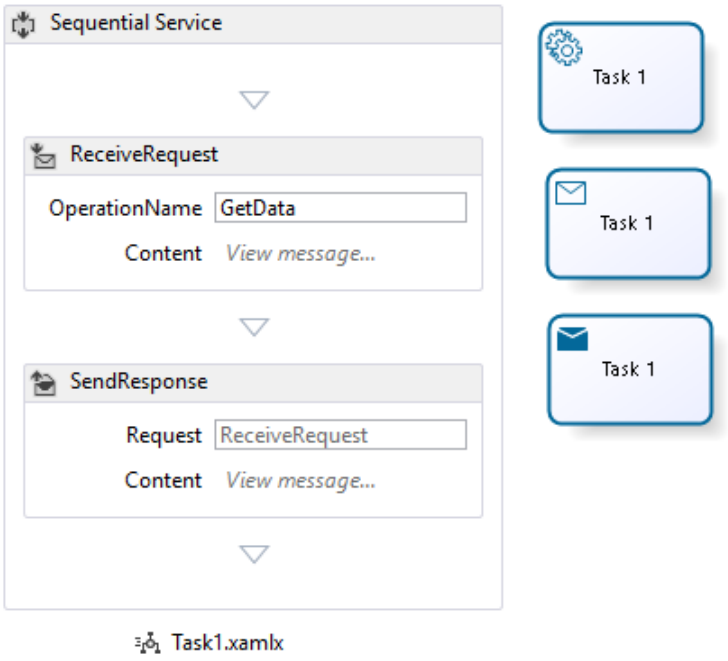


Figure 5.5: Maping from User task to NativeActivity.

2. Collapsed and expanded subprocess

Collapsed subprocess →the subprocess related to collapsed should create custom activity, then call it from mapped workflow .

Expanded subprocess →create a custom activity, and take the rule defined in the tasks mapping.

The custom activity creates the same rule of simple tasks in point 1.

3. Looping or multi-instance activity

Looping →build custom activity, and use while control flow to loop the activity. see figure 5.6 which get from [17]

Multi-instance →WF doesn't support multi-instance activity. It builds by the developer when the application needs a multi-instance workflow.

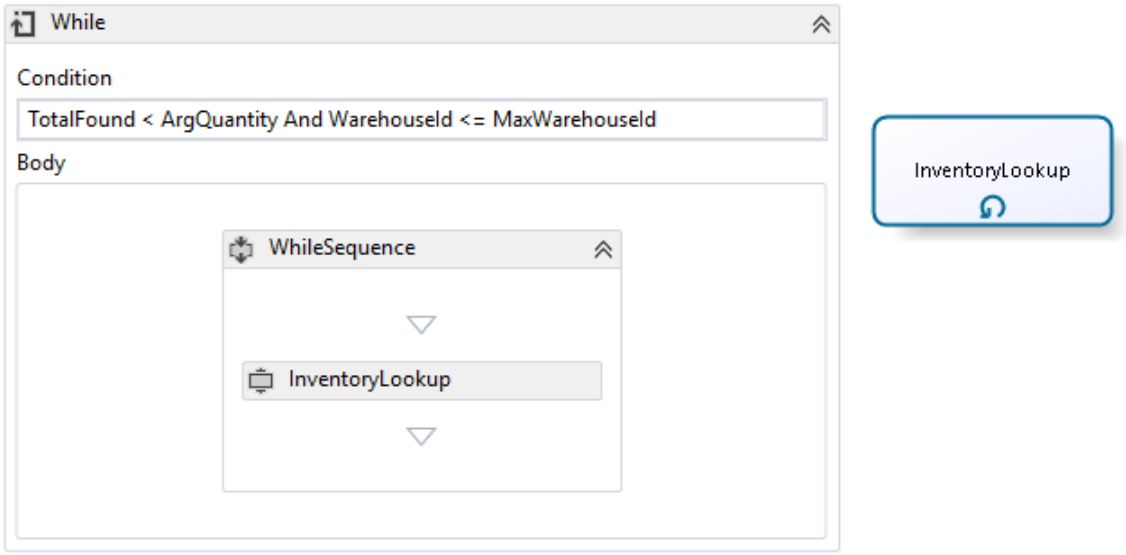


Figure 5.6: Maping from looping activity to while loop.

4. Gateway (split and join)

   If we have two split →use flowchart decision. example of this mapping shown in the figure 5.7.

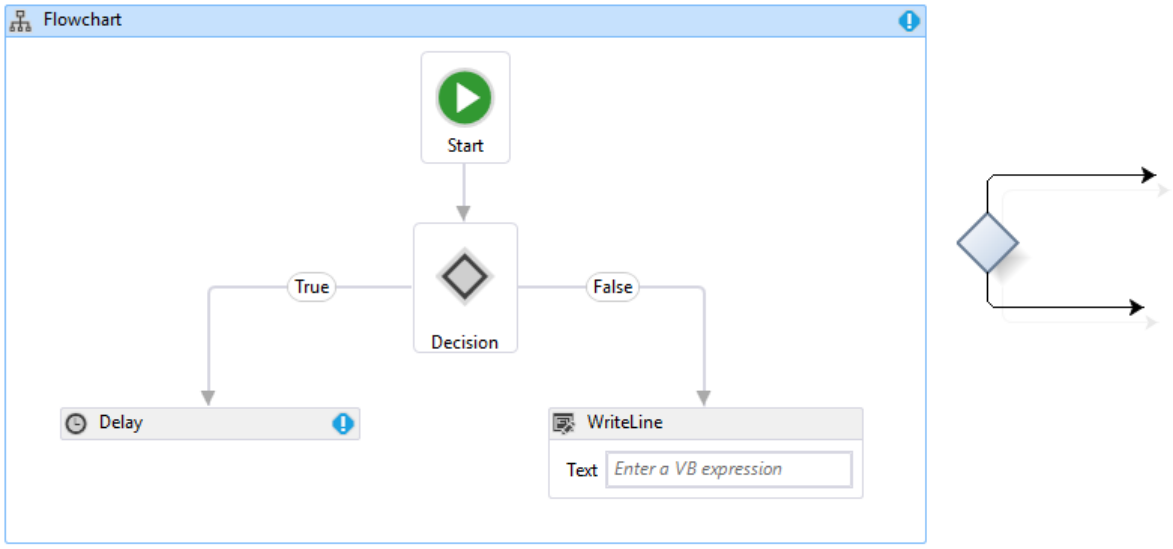


Figure 5.7: Maping from gateway split to flowchart decision.

Else if we have more than two splits →use flowchart switch. see example in the figure 5.8.

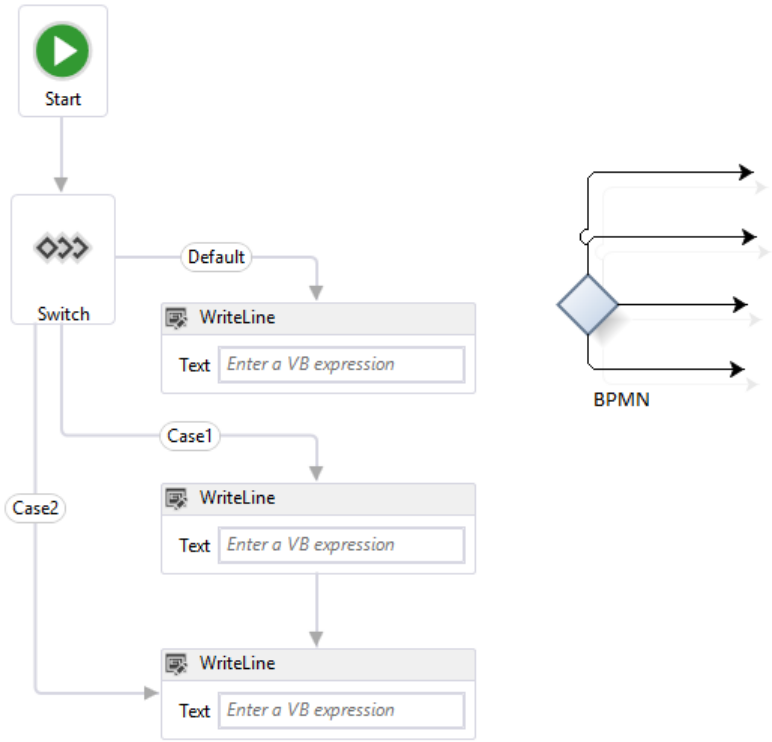


Figure 5.8: Maping from gateway split to flowchart switch.

To join activities →we can use simple activity such as WriteLine to join all activity. Figure 5.9 has example of this mapping.

Figure 5.9: Maping from gateway merge to writeLine.

a. Inclusive OR

   Every activity after inclusive gate →build a custom activity that use the AsynCodeActivity as a base class. The name of the activity is the value of the name attribute of the task. If one of the activity complete, neglect another activity result.

b. Exclusive OR-data based

   Every activity after exclusive gate →build a custom activity that use the AsynCodeActivity as a base class. The name of the activity is the value of the name attributes to the task. Use flowchart decision and switch to control the flow of this activity.

c. Exclusive OR-event based

   Every activity after exclusive gate →build a custom activity that use the AsynCodeActivity as a base class. The name of the activity is the value of the name attributes to the task. Use flow chart and pick activities to control the flow of this activity.

d. Parallel

   Every activity after exclusive gate →build a custom activity that use the AsynCodeActivity as a base class. The name of the activity is the value of the name attributes to the task. Use Parallel

to control the flow of this activity. Figure 5.10 desribed parallel mapping.

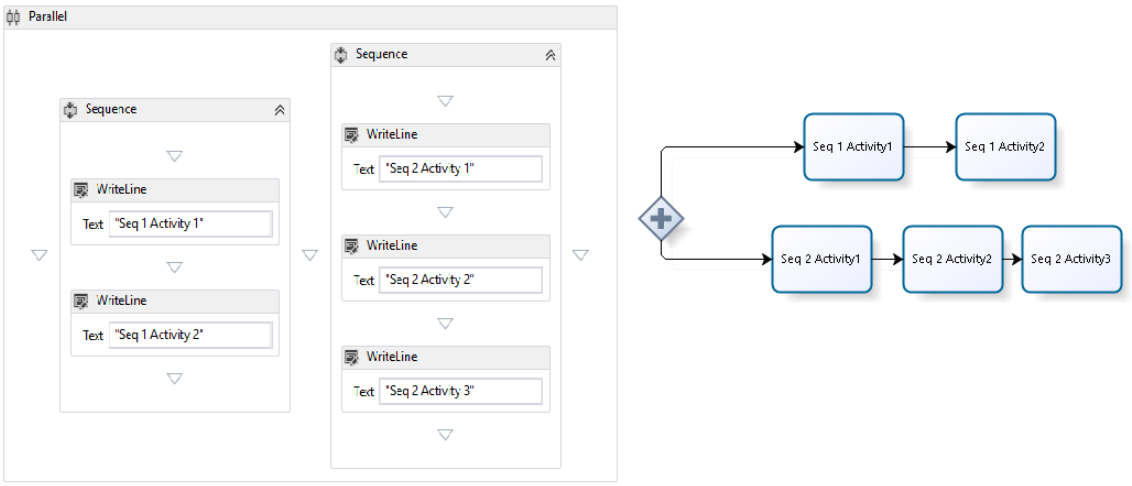


Figure 5.10: Maping from gateway parallel to WF parallel .

5. Event

   a. Start event (None, Message, Timer)

   b. Intermediate event (Timer, Message)

   c. End event (None, Message, Error)

   d. Terminate

For activity after event raised →use Pick and PickBranch to control the activity after the event and the event occurred. Put the event in Action property and put all activity after the event in Trigger property. Figure 5.11 shows pick activity that mapping from event.

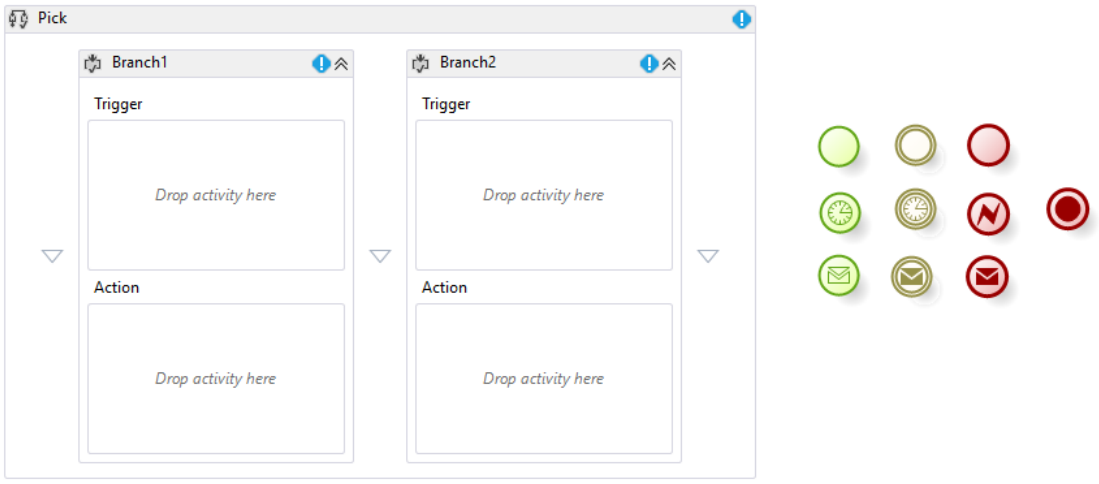


Figure 5.11: Maping from event activity to pick activity .

6. Data object

   Data Input →use In Argument to represent data input.

   Data Output →use Out Argument to represent output data.

   Data store →use classes that has been built in ER automation to represent the data store. Figure 5.12 shows the data object mapping

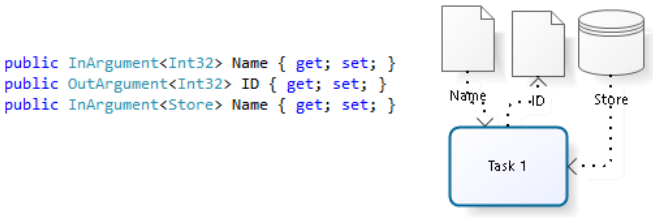


Figure 5.12: Maping data object to argument and classes

7. Text annotation

   Text annotation →use comments to take text annotations.

8. Sequence flow Use sequence flows in WF.

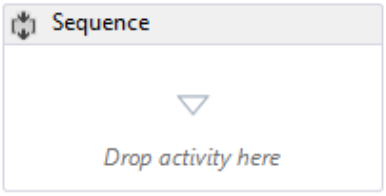


Figure 5.13: Maping from sequnce flow to sequnce activity .

9. Association

   Not supported in WF.

5.3 How to connect BPMN to Chen-ER model?

BPMN 2.0 provide advance notation than the previous version; from these notations, data object such as Data Store, Data Input, and Data Output. Data store can be used to connect BPMN and Chen-ER model as in the following rules:

Rule 1: the name of data store in BPMN should be the same name of entities in ER model. So, we can connect it to the form and C# that have the same name, (see figure 5.14)

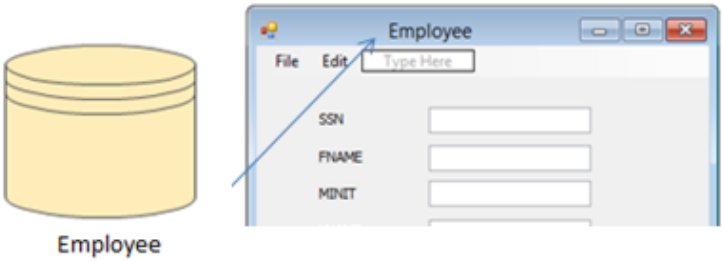


Figure 5.14: Maping from sequnce flow to sequnce activity .

Rule 2: the attribute of entity determined by data store and modified

by the user can be found in the XML file extracted from ER. (attribute such as FNAME, LNAME, SSN ...etc.)

Rule 3: in Data Input or Data Output cases, determine the direction of the data in every activity to determine if it In Argument or out Argument to be embedded inside the activity

Rule 4: Swimlanes (Pool and Lanes)

Rule 4.1 Lanes →Store the name of the lane in user account table.

Rule 4.2 Pools

→if it alone →store the name of the pool in the user account table

→If it has multiple lanes →rule 3.

Rule 5: we let that every participant can deal only with the activities refer to it. See figure 5.15

Pool 1 responsible for activity 1 only

Lane 1 responsible for activity 2 only

Lane 2 responsible for activity 3 only

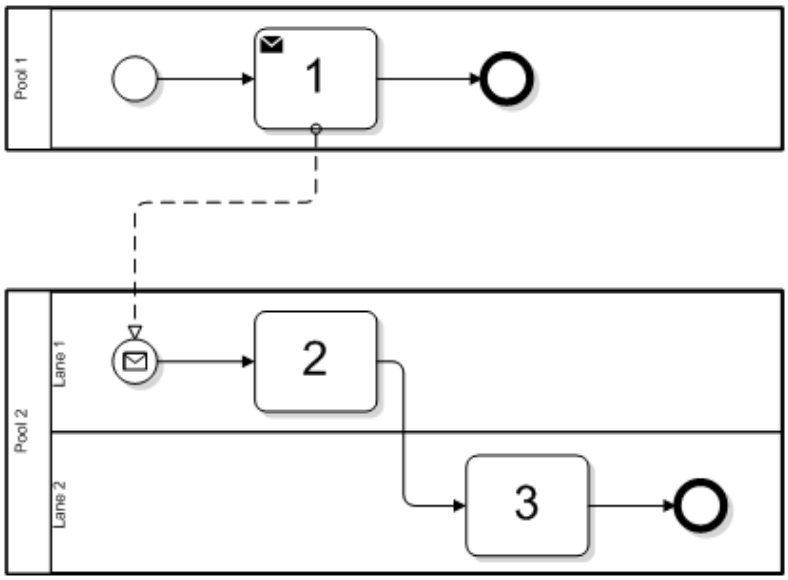


Figure 5.15: Pool-Lane responsiblity .

Chapter 6

Implemented Framework

6. Chapter Outline:

    6.1. General overview of the practical framework

        6.1.1. Development environment

        6.1.2. Create SQL file

        6.1.3. Create ”LINQ to SQL classes”

        6.1.4. Create Windows form

        6.1.5. Create Windows workflow

In this chapter, details of the development step are described. This is to extract the SQL file, the classes file, and windows form from ER diagram. Furthermore, it extracts the BPMN and mapping to C# workflow code and develope a connection between ER and BPMN.

6.1 General Overview of the Implemented Framework

Figure 6.1 shows the development environment used in this study. In addition, what are the input and the output to the system as well as what is the processing step in this work are dicussed. In the next subsections, we show the details of the system as shown in figure 6.1.

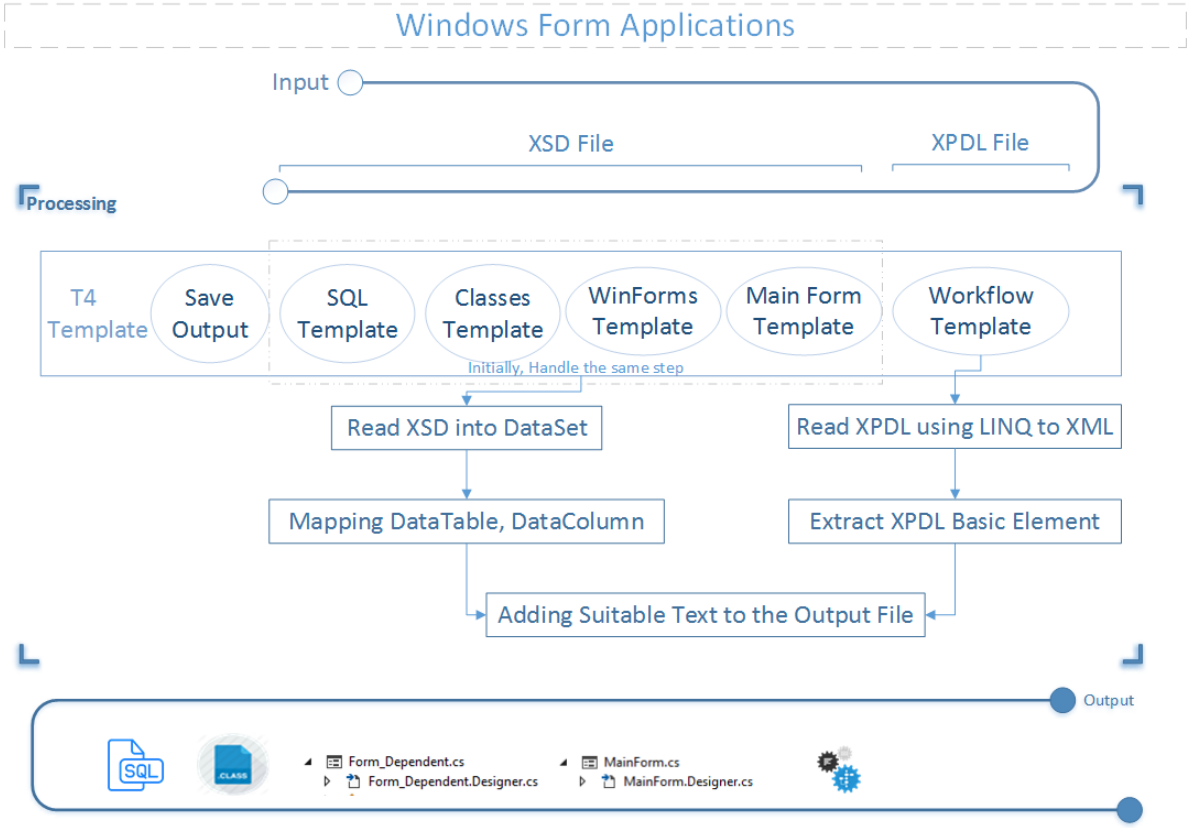


Figure 6.1: Development steps: Input, Output, and Processing of ERWFFW

6.1.1 Development Environment

The development environment used in this work is the Visual C# in the Visual Studio 2015 [59]. We use the following components in visual C# IDE: windows form, window workflow, T4 template, LINQ to SQL, LINQ to XML. Database tested using SQL server 2014. HP ProBook Core i5 used for the implemented work.

The inputs in this work are XSD file and XPDL file. The XSD file represented ER diagram. The XPDL file represented BPMN models.

The processing on the inputs to give the aimed goals is done as follows: we have six t4 template files to generate the appropriate files content. The first one is the save-output file which has only one function that receives one parameter of string type. This string parameter is the relative path which contains the file name and the subfolder to store the files. This function aims to generate all text statements in other templates on the specified relative path.

The second template is the SQL template file, which is used to create SQL file. The SQL file has all tables, columns, and constraints which read into the dataset. We read the XSD file into the dataset using (ReadXmlSchema) [58]. This method fills the dataset information (its table, columns, relations, and constraints) from the schema file. The XSD file pass through preprocessing step; which used to modify constraints to be readable in the dataset. For example, in the case of the primary key we add ' my data:PrimaryKey="true" ' to the key element. Also, the multivalued attribute, M: N relationship, and N relationship being processed when we convert from Chen-ER diagram to XSD file. Another important criterion, we add "ID" element to each entity. We use this field as AutoIncrement, Primary Key field. We use ID field in

query string.

In the next sections, we show an algorithm-1 which is used in the development process to create SQL File, to create LINQ to SQL classes, and to create windows forms.

We generate a function to make SQL file content. Then call saveOutput to write all the text in the specified path. The second t4 template is LINQ to SQL classes. Also, in this template file, read XSD into the dataset, then map each table and column to the classes fields, property, setter and getter, and constructor. The difference between LINQ to SQL classes [56] and the ordinary classes that connect directly to the SQL database tables and views regardless of the database context. Before writing class statement write this statement [Table (Name= Name of The Table in Database)] and before setter and getter write this statement [Column (DbType= data type of database fields).

Algorithm-2 shows the steps of the development process of generating LINQ to SQL classes.

6.1.1.1 Create SQL File

Algorithm 1 How to Create SQL File

for ***each*** *table in DataTable* **do**
    Write "Create Table" + table.Name + "("
    for ***each*** *column in DataColumn* **do**
        Convert column.dataType to SQL Server.dataType
        Write column.Name + column.dataType +
        if *the column not allow null value* **then**
            Write " NOT NULL "
        end
        if *the column is auto increment* **then**
            Write " IDENTITY (" + auto increment seed + auto increment step + ")"
        end
    end
    for ***each*** *PK of Primary Key in table.Name* **do**
        Write "CONSTRAINT" + PK.Name + "PRIMARY KEY CLUSTERED ("+ Pk.coulmn.Name +")"
    end
end
for ***each*** *table in DataTable* **do**
    for ***each*** *FK of foreign key in table.Name* **do**
        Write "ALTER TABLE" + table.Name
        "ADD CONSTRAINT [" + constraint.Name +"] FOREIGN KEY
        "+ ([fk.column.Name])
        "REFERENCES"+ [table.Name] ([fk.RelatedColumns])
    end
end
Call SaveOutput

6.1.1.2 Create LINQ to SQL Class File

Algorithm 2 How to Create LINQ to SQL Class File

```
for each table in DataTable do
    Create class file →Name of class file= table.Name
    Write header class content
    Write namespace statement
    Write "[Table (Name="+ table.Name.toUpper()+"]"
    Write "Class "+ table.Name
    Write "{"

    for each column in DataColumn do
        Write "public " + column.dataType + column.Name.toLower()+";"
    end
    Write "public "+ table.Name + " ( ) "
    for each column in DataColumn do
        if column.Name =ID then
            Write [Column (DbType= ConvertToSql(column.dataType), Is-
            PrimaryKey =true, IsDbGenerated =true)]
        end
        Write [Column (DbType= ConvertToSql(column.dataType)]
        Write "public "+column.dataType + column.Name.toUpper()
        Write " get;set;"

    end
    Write "}"
    Call SaveOutput
end
```

The third template file is windows form template. This template generates two files: Form.cs and Form.Designer.cs which represents ordinary C sharp windows form. The two files have a partial class of the same name of the form. The desired form that should be implemented as shown in figure 6.2. The name of the form is the name of the table with the initial character capitalised. In this form, we have a label with a location in the middle of the form with text value to equal the same name of the form. Moreover, we have two panels. The first one is used for tables and columns of the dataset that represents the entities and the attributes in the ER. In the developed step, we consider all attributes as labels and textboxes. The second panel has eight buttons. These buttons have a query statements inside the click event handler of the buttons. Finally, we have a DataGridView to show the data about each form entity.

The size and locations of these controls are fixed in this work: form, first and second panels, and DataGridView. The first panel has a true auto scroll property that can expand many created labels and textboxes. Each label and textbox is created based on the number of columns in the dataset. The locations of labels and textboxes are distributed from left to right and up to down directions. The size of the labels and the textboxes is usually fixed. Algorithm-3 shown the created form the process. The queries created in this file are shown in the next chapter.

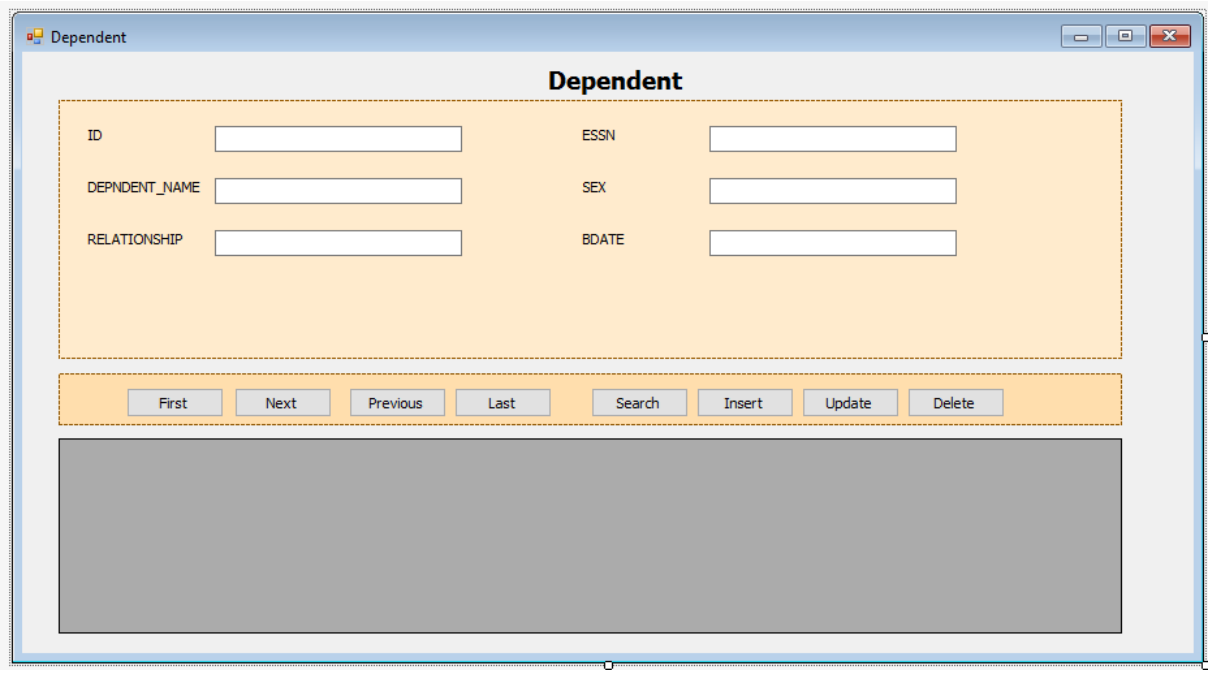


Figure 6.2: Template of created windows form.

6.1.1.3 Create Windows Form

Algorithm 3 How to Create windows form (Form.cs) File

for ***each*** *Table in DataTable* **do**

Table.Name →table.Name.UpperCaseFirst

Write the header of classes

Write the definition Data Context

Write the partial class statement

Write form constructor method

Write form load method

{

Write the connection string of data context

Write the creation object of data context

}

Write First query button click method

Write Next query button click method

Write Previous query button click method

Write Last query button click method

Write Search query button click method

Write Insert query button click method

Write Update query button click method

Write Delete query button click method

Write upgrade DataGridView method

Write end of class

end

Algorithm 4 How to Create windows form (Form.Designer.cs) File

for ***each*** *Table in DataTable* **do**
    Write the header of classes
    Write the partial class statement
    Write dispose method
    Write InitializeComponent method
    { Write the created controls instances
    Write the fixed controls instances
    for ***each*** *column in DataColumn* **do**
        Write the created labels and textboxes instances
    end
    Write the fixed panel properties and add the buttons control on it
    Write the auto scroll panel properties
    for ***each*** *column in DataColumn* **do**
        Write the addition of created labels and textboxes on it
    end
    Write the fixed controls properties
    for ***each*** *column in DataColumn* **do**
        Write the distribution of created labels and textboxes to right and left positions
    end
    }
    Write the declaration of fixed controls
    for ***each*** *column in DataColumn* **do**
        Write the declaration of created labels and textboxes
    end
    Write end of class
end

The fifth t4 template is the Main template. The main template file is used to create the first screen. The main screen is used to open the created forms and test its work. Figure **??** below shows the main form. The main form has one auto scroll panel and one exit button. The panel is used to create the buttons to open the created forms in the previous step. Exit button is used to close down the application. To generate this screen, we follow the same techniques in the previous step.

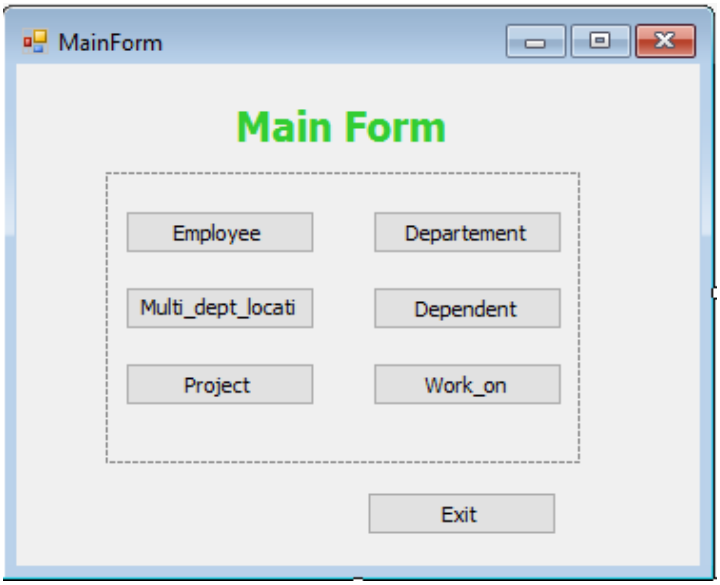


Figure 6.3: Template of created windows form application - Main Form

Finally, the six t4 template is the workflow template. In this template, we read the XPDL file using LINQ to XML mechanism. Parsing or querying the XPDL is a complicated process. It has a rich set of elements and child elements. To achieve the study goals regarding the workflow mapping, we use ordinary XML parsing method, but the LINQ to XML provides a more powerful technique. It helps to parse the XML using query based concepts. In the Implemented development of the workflow, we focus on the simple XPDL classes. Then we extract all basics XPDL elements and store them in lists, in order to complete the work in the future. And we concentrate on

the mapping point between Chen-ER and BPMN.

There are several elements and attributes that must be taken into accounts when extracting XPDL element. Table 6.1 display these elements and attributes.

Table 6.1: XPDL Element and their attributes.

Number	**Element**	**Related element or attribute**
1	Actual Parameters	Id, Target ID, Source ID
2	Artifacts	Id, ArtifactsType, Text Annotations
3	Associations	Id, Target, Source,direction
4	Data Associations	Id, From, To, Actual Reference Id
5	Data Fields	Id, Name, Is Array, Data Type
6	Data Objects	Id, Name, Data field Id
7	Data Stores	Id, Name, is unlimited
8	Data store References	Id, data store reference Id
9	Formal Parameters	Id, Data Type, initial Value, Mode, Is Required,Is Array, read only, Name.
10	Gateways	Id, Name,Expression, Gateway type, Exclusive type
11	Task Activities	Id, Name, description
12	Transitions	Id, Name,From, To, condition type, condition string

We create classes for each element mentioned in the table 6.1. Then, we

extract all element from XDPL file and store them in lists of these classes. We consider some of the constraints in this work and they worth mentioning here before illustrate the algorithm being used:

- The XPDL file have only one workflow process.
- The XPDL file havent multiple pools.
- Consider the simple class for extracting XPDL element.
- Neglect graphical based element.
- Neglect package, package header, Participants, pagesetc.
- Focused on the element mentioned in table 6.1. Especially the elements used to make a connection between ER and BPMN.
- Because we are dealing with the database based application, we consider all created activity based of "AsyncCodeActivity"
- The XPDL file have one start event and one end event.

Extract XPDL element: every element in the XPDL file have nearly the same algorithm to extract their child elements and attributes. Algorithm-5 used for extracting the element.

Algorithm 5 How to extract xpdl element

Load XPDL file using **XDocumnet**

Var EleVar = **from** element in doc.Descendants(XPDL Name Elemnt)

select element

for ***each*** *element in EleVar* **do**

  Declare instance object from element classes i.e. DataObject obj

  Extract element attributes

  Assign object instance field to the value of element attribute

  Add object instance to element list

end

In this work, we have created an activity for each task in the process. We take into account the data input, output, and store. Data input is mapped into In Argument in the activity connected with. Data Output is mapped to Out Argument in the activity connected with. Data store should have the same name of the created LINQ to SQL classes from ER developed based. If the data store is input to the task, it is mapped to In Argument of the class name such as InArgument <EMPLOYEE >.

When we work with the XDPL file, we find that we have multiple tags to describe the data. WfMC provides a meta-model [84] to describe the relations between them. But the meta-model has two problems, the first one, it is too general to work on this projects. And the second problem that the model does not have connection between these data elements. Figure 6.4 explains the relations between these data elements. We can show that the data object is used to refer to the data field and the data store reference is used to refer to the data store. To simplify this relation, we add data object reference Id to the data field element. And we add data store reference Id to the data

store element. So, we have only four data elements to work with.

The relation between the actual parameter and the data field as well as the relation between the actual parameter and the data store depends on the relation between the actual parameter and the formal parameter. The dependency is depending on the formal parameter mode, If the mode is "Out", the target Id of the actual parameter is the formal parameter Id, and the source Id of the actual parameter is the data field Id or data store Id. If the mode is "In", the source Id of the actual parameter is the formal parameter Id, and the target Id of the actual parameter is the data field Id or data store Id.

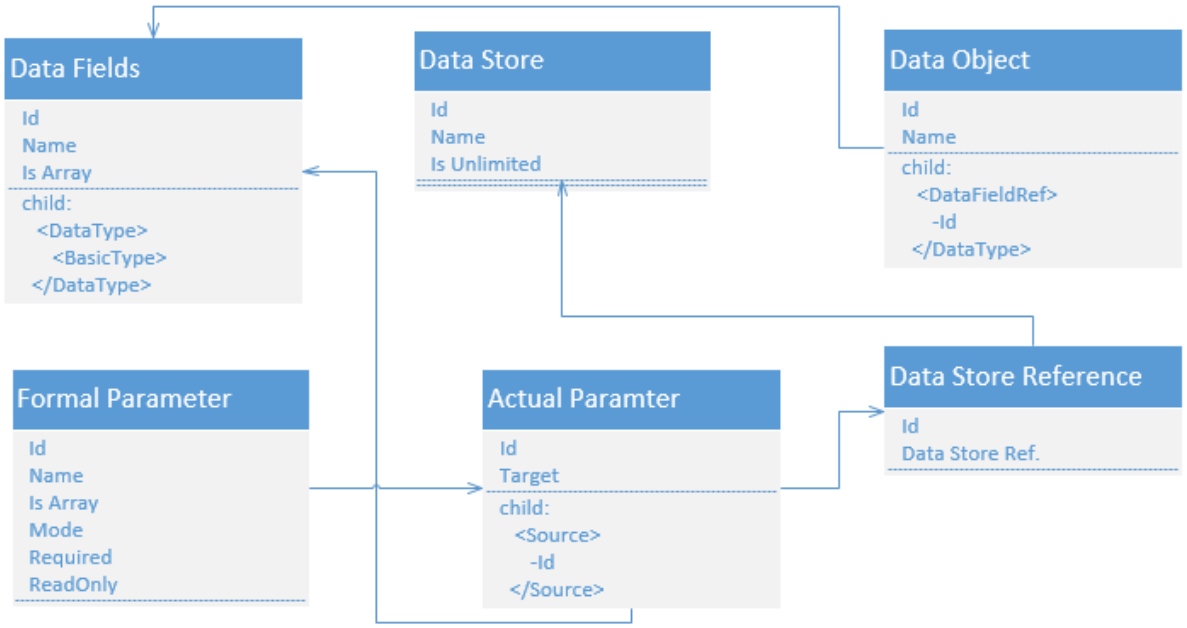


Figure 6.4: Relations between data elements

Algorithm-6 shows how to create the activity and make a connection statement inside each activity.

6.1.1.4 Create Mapping between BPMN and ER

Algorithm 6 Create activity and mapping step between BPMN and ER

```
for each activity in XDPL file do
    if the activity has implementation tag then
        Create custom activity that inherit AsyncCodeActivity
        The name of the custom activity →the name of task activity
        Write header text
        for each datafield of Data field in activity do
            for each ap of Actual parameter in activity do
                for each fp of Formal parameter in activity do
                    if datafield.ID = ap.sourceID then
                        Write "public "+UppercaseFirst(fp.mode.tolower()) +
                        "Argument<string >" + datafield.Name +" {get; set;}
                        "
                    end
                    if datafield.ID = ap.targetID then
                        Write "public "+UppercaseFirst(fp.mode.tolower()) +
                        "Argument<string>" + datafield.Name +" {get; set;}
                        "
                    end
                end
            end
        end
        Write reminder text
    end
end
```

Chapter 7

Evaluations

7. Chapter Outline:

    7.1. Windows Desktop Application Guidelines

        7.1.1. Controls

        7.1.2. Commands

        7.1.3. Text

        7.1.4. Messages

        7.1.5. Interaction

        7.1.6. Windows

        7.1.7. Visuals

    7.2. Discusions

This chapter introduces the evaluation of our work. The evaluation deals with the generated forms and screens that appear after making an automated generation of business application. The business application is based on GUI. A good user interface can gain by following guidelines from experts. We build the business application as a Windows desktop application. In the next section, we shed light on the windows desktop application guidelines. Then we discussed what the guideline that we followed in this study.

7.1 Windows Desktop Application Guidelines

The guidelines [36, 53, 54] include: controls, commands, text, messages, interaction, windows, and visuals guidelines. I) Controls are UI elements that users interact with applications main window area. II) Commands are the actions of the users that he takes while using the app. III) Text is any text that the user can see in the application. IV) Messages are any kind of messages user need when he uses the app. V) Interaction in the variety of ways users interacts with the app, including touch, keyboard, mouse, and so on. VI) Windows are the main UI surfaces of the desktop app, including the main windows itself and pop-ups, dialogs, and wizards. VII) Visuals contain the visual elements, these guidelines help to make decisions about layout, fonts, color, icons, and so on in the app. And the Windows environment is the onscreen work area provided by Windows. Next subsections will discuss these guidelines in details.

7.1.1 Controls

It contains these UI components:

1. **Balloons** inform users of a usual problem or a special condition in a control.
2. **Checkboxes** allow users to make a decision between two or more different choices.
3. **Command buttons** allow users to perform an action.
4. **Command links** allow users to make choice among a set of related choices.
5. **Drop down lists and combo boxes** allow users to make choice among a list of mutually exclusive values.
6. **Group boxes** allow users to see relationships among a set of related controls.
7. **Links** allow users to navigate to another page or window.
8. **List boxes** allow users to select from a set of values in a list.
9. **List views** allow users to view and interact with a collection of data objects, using either single selection or multiple selections.
10. **Notifications** inform users of events that are unrelated to the current user activity.
11. **Progress bars** allow users to see the progress of a length operation.
12. **Radio buttons** allow users to make a choice among a set of mutually exclusive, related choices.
13. **Sliders** allow users to choose from a continuous range of values.

14. **Spin controls** allow users to change incrementally the value within its associated numeric text box.

15. **Status bars** display information about the state of the current window, background tasks, or other contextual information.

16. **Tabs** present users with related information on separately labeled pages.

17. **Text boxes** allow users to display, enter, or edit a text or numeric value.

18. **Tooltips** label is an unlabeled control.

7.1.2 Commands

We can add a command to any of previous controls. In addition, we can add a command to app menus, ribbons, and toolbars.

1. **Menus** are hierarchical lists of commands or options available to users in the current window. Menus may be fixed above form or hidden appear when mouse click or hover. Menus are displayed on a menu bar.

2. **Ribbons** are the modern way to help users use commands efficiently and directly with a minimum number of clicks. A ribbon is a command bar that organizes a program's features into a series of tabs at the top of a window.

3. **Toolbars** are ways that help the group commands for efficient access. We can use the toolbar in addition or in place of the menu bar.

7.1.3 Text

It includes:

1. **User interface text** appears on UI screen. This text includes:
    1.1. **Control labels** describe controls and are placed directly on or next to the controls.
    1.2. **Static text** provides users with detailed instructions or explanations so they can make informed decisions
2. **Style and Tone** are intended to create a response or emotion from the reader.

7.1.4 Messages

It has four types of messages:

1. **Error messages** alert users of a problem that has already occurred.
2. **Warning messages** alert the user of a condition that might cause a problem in the future.
3. **Confirmations messages** ask if the user wants to perform an action.
4. **Notification messages** inform users of events that are unrelated to the current activity.

7.1.5 Interaction

The user interacts with the app using one of the following ways:

1. Touch

2. Keyboard

3. Mouse and pointer

4. Pen

7.1.6 Windows

These guidelines about which surface to use.

1. **Windows management:** this guideline covers default position of windows when initially appeared on the screen, their order relative to other windows and their initial size.

2. **Window Frames:** in frames, the users can manipulate a window and view the title and icon to identify its contents. There are many types of frames: glass, hidden and custom.

3. **A dialog box:** is an extra window that allows users to perform a command, asks users a question, or provides users with information or progress feedback.

7.1.7 Visuals

These guidelines discuss layout (size, space, and placement of content within windows), fonts, color, icons, and so on in the app.

7.2 Discusions

The following tables present which guidelines used in this study. It has three columns: I) Design Guidelines for Windows desktop application II) Yes/No

Table 7.1: Control Design Guidelines

Design Guideline	**Yes/No**	**Notes**
Balloons	No	
Checkboxes	No	
Command buttons	Yes	In each form, we have eight command buttons (First, Next, Previous, Last, Search, Insert, Update, and Delete).
Command links	No	
Drop down lists and combo boxes	Yes	We used it when the attribute of the entity in ER is a foreign key.
Group boxes	Yes	We used two group boxes. The first group box used to gather each label and textbox for each attribute. The second group box used to group the eight commands buttons.
Links	No	
List boxes	No	
List views	No	
Notifications	No	
Progress bars	No	
Radio buttons	No	
Sliders	Yes	We used it if the labels and textbox of the first group box needs space more than specified
Spin Controls	No	
Status bars	No	
Tabs	No	
Text boxes	Yes	We used it to represent each attribute for specified entity in ER.
Tooltips	No	

column refers if this guideline used in our study or not used. II) Notes column to remark some point in design UI for our study.

Table 7.2: Commands Design Guidelines

Design Guideline	**Yes/No**	**Notes**
Menus	No	
Ribbons	No	
Toolbars	Yes	We put the eight commands button (First, Next, Previous, Last, Search, Insert, Update, and Delete) as a toolbar because it is more efficient than menu bars because they are direct access.

Table 7.3: Text Design Guidelines

Design Guideline	**Yes/No**	**Notes**
UI Text: Control labels	Yes	We used it before each text box to indicate the using of text box.
UI Text: static text	Yes	We used it on the top of each form to indicate the entity form.
Style and Tone	No	

Table 7.4: Message Design Guidelines

Design Guideline	**Yes/No**	**Notes**
Error	No	
Warning	No	
Confirmation	Yes	We used it when an insert, delete a row from the database to confirm the process.
Notification	Yes	We used it to notify the user that he reaches the last or first record.

Table 7.5: Interaction Design Guidelines

Design Guideline	**Yes/No**	**Notes**
Touch	No	
Keyboard	Yes	
Mouse and Pointer	Yes	
Pen	No	

Table 7.6: Windows Design Guidelines

Design Guideline	**Yes/No**	**Notes**
Windows management	Yes	We used the default position and size of the form.
Windows frames	Yes	We used glass frames with minimize, maximize and close buttons.
Dialog box	No	

Another important guideline can found in [67]. It discusses the basic

principles of the intuitive user interface. The really efficient looking UI based on these four factors:

- **Spacing and Positioning:** spacing between two controls is important. Figure 7.1 shows two textboxes that are too close but the list under them is too far away. Also, there is a lot of unused area on the form. A suggested solution to this form can see in figure 7.2, it was modified to align a label with the text baseline of the textbox or other control next to it. Other change can be made on figure 7.1 through appointing the textboxes and labels on proper position.

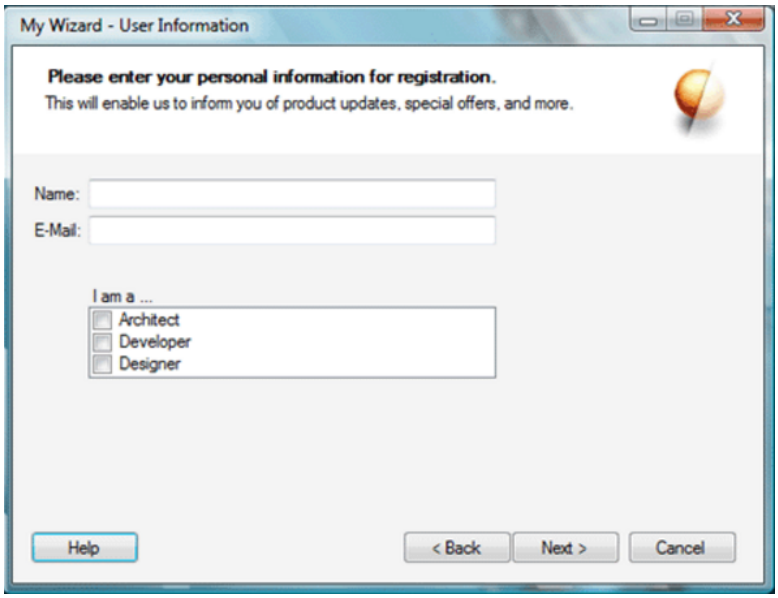


Figure 7.1: Poorly designed form

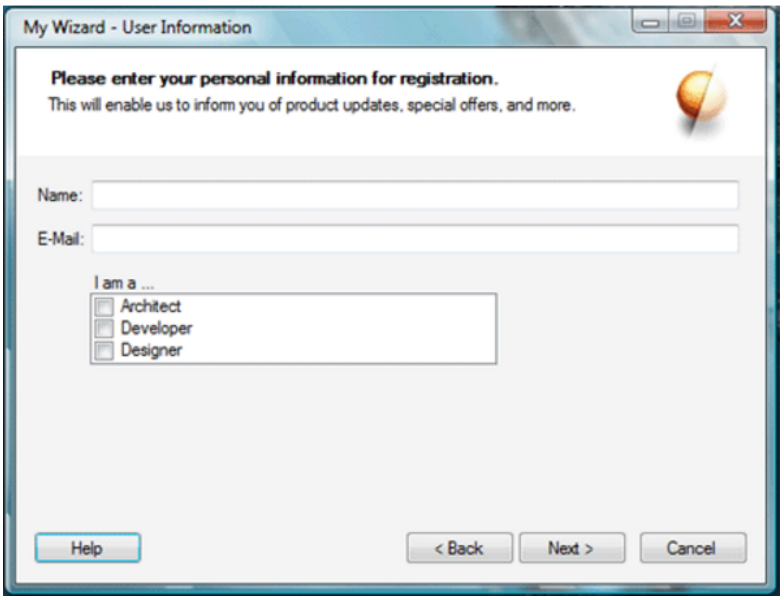


Figure 7.2: Properly designed form

- **Size:** When we drag a button from the toolbox to the form, it has the perfect height and width. In figure 7.2, we can see three buttons with the text of it indicates the proper size.

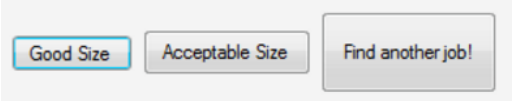


Figure 7.3: Button size comparison

- **Grouping:** any application has many controls. By using intuitive groups, we can make all these controls easier to use. Categorized grouping is done best by Tab controls. But controls that contribute to the same end result are best done by Group controls such as Panel controls.

- **Intuitiveness:** it is the significant attitude of a great user experience. The user knows what the controls do. Some important topics in intuitive design are color-coding, the text of controls in the application, and the placement of the OK/Cancel buttons.

Finally, we will explain some important point that deals with the visual guidelines (size, space, font and so on). Figure 7.4 exhibits one of the outputted forms.

1. Each form is a glass window frame which can minimize, maximize and close the form.

2. Each form has a bold, 14 font size label that indicates the name of the entity. The position of the label is (400, 10).

3. Under the previous label, we have the first panel, it has labels and text boxes for each attribute for the specified entity. The properties of this panels:

   - Auto scrolled
   - Color: blanched almond
   - Location: 28, 37
   - Size: 800,200

4. Each label and text box distributed left and right. The common labels and text boxes properties are:

   - Label size: 35,13
   - Label text: name of the attribute in ER.
   - Location of the first label in the left: 20,20 then increase 40 points for the y-axis.
   - Location of the first label in the right: 400,20 then increase 40 points for the y-axis.
   - Text size: 190,20

- Location of the first text box in the left: 120,20 then increase 40 points for the y-axis.
- Location of the first text box in the right: 500,20 then increase 40 points for the y-axis.

5. Then, each form has a second panel with the following properties:
   - Color: Navajo white
   - Location: 28, 247
   - Size: 800, 40
6. The command buttons in panel two have a text property that indicates the action of the button. With taking into account the size and the position of each command.
7. Data grid view size and position is relative to the panels as follows:
   - Size: 817, 150
   - Location: 28, 297

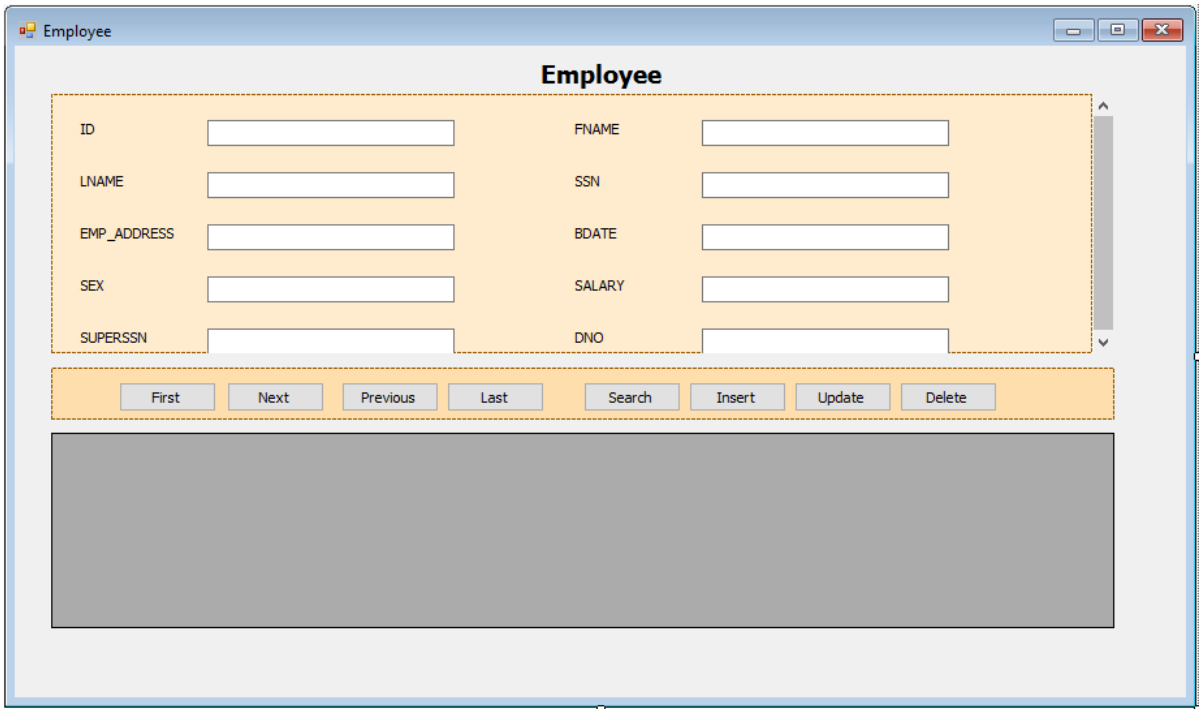


Figure 7.4: Employee Windows Form.

Chapter 8

Conclusions and Recommendations

8. Chapter Outline:

    8.1. Conclusions

    8.2. Future Works

8.1 Conclusions

This study shows the limitations of BPMN in describing the data. The data in BPMN is considered as artifacts. So, the study is set out to find the possibility to improve the BPMN to handle the data without increasing the complexity of BPMN. The study has also sought to show if we can benefit from ER models to represent the data in BPMN. Furthermore, this study suggests an improvement in building a business application from Chen-ER and BPMN models.

This work is proposed a framework that maps models to the codes. The raised mapping in the study as follows: mapping from ER to SQL file, Mapping from ER to "LINQ to SQL" classes files, mapping from ER to windows form, and mapping from BPMN to WF. Furthermore, we suggests a combination method of ER and BPMN.

In this study, firstly, we have studied the properties of the business application. Then, we have proposed the theoretical framework that maps from ER to SQL statements, classes, and GUI components. Also, the theoretical framework mapped BPMN to Windows workflow pattern. Finally, in a theoretical framework, a combination between Chen-ER and BPMN was proposed. The combination process is based on the BPMN 2.0 advance notations (Data Input, Data Output, and Data Store).

After that, we have developed some of these mapping methods such as mapping from ER to SQL file, Mapping from ER to "LINQ to SQL" classes files, mapping from ER to windows form, and create an activity for each atomic task in XPDL. In addition, we developed the combination method between ER and BPMN. The results of our experiments are a framework to build business application automatically.

Business application emulates real business entity represented in ER models. Each entity is represented in Windows form. Each form has labels and textboxes, buttons, and data grid view. We can browse the data in database using (First record, Next record, Previous record, and Last record) buttons. Furthermore, we can make a search, insert, update, and delete query using LINQ programming style. We established a route to automate a complete BPMN in the windows form by mapping BPMN to windows workflow. This can increase such enhancement by developing a connection between BPMN and Chen-ER.

Consequently, we state the following conclusions, once we use the framework of the automated generation of business application: i. We can help developers to save both their efforts and time. ii. Developers can enhance their business application rather than restricting themselves to customer's delivery iii. It encourages the enterprise to pass through the requirements analysis and design steps.

8.2 Future Works

Here are some of the important points for researchers to consider in the future. First, the complete class of the BPMN can be mapped to Windows workflow pattern. Secondly, the data flow diagram (DFD) can be searched to find the flow of the data through the business application. Thirdly, the researchers can use a UML class diagram instead of ER models.

Bibliography

[1] Cloud apps - bpi - the destination for everything process related. https://www.businessprocessincubator.com/cloudapps/. (Accessed on 01/05/2016).

[2] Qsee technologies — case and modeling tool specialists - leeds beckett university. http://www.leedsbeckett.ac.uk/qsee/. (Accessed on 01/05/2016).

[3] Xml schema complex elements. http://www.w3schools.com/xml/schema_complex.asp. (Accessed on 28/04/2016).

[4] Xml schema element. http://www.w3schools.com/xml/schema_schema.asp. (Accessed on 28/04/2016).

[5] Xml schema simple elements. http://www.w3schools.com/xml/schema_simple.asp. (Accessed on 28/04/2016).

[6] Workflow handbook 1997. chapter The Workflow Reference Model, pages 243–293. John Wiley & Sons, Inc., New York, NY, USA, 1997.

[7] K Scott Allen. *Programming Windows Workflow Foundation: Practical WF Techniques and Examples using XAML and C#*. Packt Publishing Ltd, 2006.

[8] Guntis Arnicans. Application generation for the simple database browser based on the er diagram. In *Databases and Information Systems, Proceedings of the Third International Baltic Workshop*, volume 1, pages 198–209, 1998.

[9] Joanne M. Atlee, Robert France, Geri Georg, Ana Moreira, Bernhard Rumpe, and Steffen Zschaler. Modeling in software engineering. In *Companion to the Proceedings of the 29th International Conference on Software Engineering*, ICSE COMPANION '07, pages 113–114, Washington, DC, USA, 2007. IEEE Computer Society.

[10] Anup Kumar Bhattacharjee and RK Shyamasundar. Activity diagrams: A formal framework to model business processes and code generation. *Journal of Object Technology*, 8(1):189–220, 2009.

[11] Bizagi. Bizagi - business process management (bpms) and workflow software. `http://www.bizagi.com/`. (Accessed on 01/05/2016).

[12] Bizagi. Bpmnbyexampleeng.pdf. `http://resources.bizagi.com/docs/BPMNByExampleENG.pdf`. (Accessed on 01/05/2016).

[13] Bonita. Bonitasoft. `http://www.bonitasoft.com/`. (Accessed on 01/05/2016).

[14] Marco Brambilla. Generation of webml web application models from business process specifications. In *Proceedings of the 6th international conference on Web engineering*, pages 85–86. ACM, 2006.

[15] Marco Brambilla, Stefano Ceri, Piero Fraternali, and Ioana Manolescu. Process modeling in web applications. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 15(4):360–409, 2006.

[16] Jan vom Brocke and Michael Rosemann. Handbook on business process management 1: introduction, methods, and information systems. 2014.

[17] Bruce Bukovics. *Pro WF: Windows Workflow in .NET 4*. Apress, 2010.

[18] Peter Pin-Shan Chen. The entity-relationship modeltoward a unified view of data. *ACM Transactions on Database Systems (TODS)*, 1(1):9–36, 1976.

[19] Michele Chinosi and Alberto Trombetta. Bpmn: An introduction to the standard. *Computer Standards & Interfaces*, 34(1):124–134, 2012.

[20] Workflow Management Coalition. Terminology & Glossary. Technical Report WFMC-TC-1011, Issue 2.0, Workflow Management Coalition, June 1997.

[21] Giuseppe Della Penna, Antinisca Di Marco, Benedetto Intrigila, Igor Melatti, and Alfonso Pierantonio. Interoperability mapping from xml schemas to er diagrams. *Data & Knowledge Engineering*, 59(1):166–188, 2006.

[22] Jian Deng, Bo Chen, and Jiazhi Zeng. Model driven development for business process model with nested compensations. *International Journal of Digital Content Technology & its Applications*, 6(16), 2012.

[23] Jian Deng, Bo Chen, and Jiazhi Zeng. Research on the transformation of gateways from business process model to flowchart in workflow foundation. *Journal of Convergence Information Technology*, 7(15), 2012.

[24] Demiano Distante, Gustavo Rossi, and Gerardo Canfora. Modeling business processes in web applications: an analysis framework. In *Proceedings of the 2007 ACM symposium on Applied computing*, pages 1677–1682. ACM, 2007.

[25] Ramez Elmasri and Shamkant Navathe. *Fundamentals of Database Systems*. Addison-Wesley Publishing Company, USA, 6th edition, 2010.

[26] Joseph Fong and San Kuen Cheung. Translating relational schema into xml schema definition with data semantic preservation and xsd graph. *Information and software technology*, 47(7):437–462, 2005.

[27] Massimo Franceschet, Donatella Gubiani, Angelo Montanari, and Carla Piazza. From entity relationship to xml schema: a graph-theoretic approach. In *Database and XML Technologies*, pages 165–179. Springer, 2009.

[28] Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 2002.

[29] Miro Govedarica, Ivan Luković, and Pavle Mogin. Generating xml based specifications of information systems. *Computer Science and Information Systems*, 1(1):117–140, 2004.

[30] Michael Hammer and James Champy. Re-engineering the corporation: A manifesto for change. *London: Nicholas Brealey*, 1993.

[31] Mary Holstege and A Vedamuthu. W3c xml schema definition language (xsd): Component designators. *W3C Candidate Recommendation CR-xmlschema-ref-20100119*, 2010.

[32] Shang-Hsien Hsieh and Hsien-Tang Lin. Xsform: A schema-driven form-based xml information processor. In *Proceedings of the 21st International Symposium on Automation and Robotics in Construction*, pages 21–25, 2004.

[33] Kayo Iizuka, Yasuki Iizuka, and Chihiro Suematsu. e-business process modeling issues: From the viewpoint of inter-organizational process efficiency and information sharing. *Procedia Computer Science*, 22:820–827, 2013.

[34] Business Process Incubator. Templates archives - bpi - the destination for everything process related. `https://www.businessprocessincubator.com/category/type/templates/`. (Accessed on 01/05/2016).

[35] Monique Jansen-Vullers and Mariska Netjes. Business process simulation–a tool survey. In *Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark*, volume 38, 2006.

[36] Caroline Jarrett and Gerry Gaffney. *Forms that work: Designing Web forms for usability*. Morgan Kaufmann, 2009.

[37] Moonyoung Jung, Hak Soo Kim, Myung Hyun Jo, Kyung Hyun Tak, Hyun Suk Cha, and Jin Hyun Son. Mapping from bpmn-formed business processes to xpdl business processes. In *ICEB*, pages 422–427, 2004.

[38] Atul Kahate. *XML & Related Technologies*. Pearson Education India, 2009.

[39] Andrew J. Kornecki and Sona Johri. Automatic code generation: Model-code semantic consistency. In Hamid R. Arabnia and Hassan Reza, editors, *Software Engineering Research and Practice*, pages 191–197. CSREA Press, 2006.

[40] Vinay Kulkarni and Sreedhar Reddy. A model-driven approach for developing business applications: experience, lessons learnt and a way forward. In *Proceedings of the 1st India software engineering conference*, pages 21–28. ACM, 2008.

[41] Vinay Kulkarni, R Venkatesh, and Sreedhar Reddy. Generating enterprise applications from models. In *Advances in Object-Oriented Information Systems*, pages 270–279. Springer, 2002.

[42] Patrick Lay and Stefan Lüttringhaus-Kappel. Transforming xml schemas into java swing guis. *GI Jahrestagung (1)*, 50:271–276, 2004.

[43] Dongwon Lee, Murali Mani, and Wesley W Chu. Effective schema conversions between xml and relational models. *Transformation for the Semantic Web KTSW 2002*, 21:3, 2002.

[44] Hsien-Tang Lin and Shang-Hsien Hsieh. Design of a schema-driven environment for facilitating application of construction information standards. 2006.

[45] Ann Lindsay, Denise Downs, and Ken Lunn. Business processesattempts to find a definition. *Information and software technology*, 45(15):1015–1019, 2003.

[46] Chengfei Liu and Jianxin Li. Designing quality xml schemas from er diagrams. In *Advances in Web-Age Information Management*, pages 508–519. Springer, 2006.

[47] Anthony Lo, Reda Alhajj, and Ken Barker. Virex: visual relational to xml conversion tool. *Journal of Visual languages & Computing*, 17(1):25–45, 2006.

[48] Jochen Ludewig. Models in software engineering–an introduction. *Software and Systems Modeling*, 2(1):5–14, 2003.

[49] Ivan Luković, Pavle Mogin, Jelena Pavićević, and Sonja Ristić. An approach to developing complex database schemas using form types. *Software: Practice and Experience*, 37(15):1621–1656, 2007.

[50] Ivan Luković, Sonja Ristić, Aleksandar Popović, and Pavle Mogin. An approach to the platform independent specification of a business application. In *Proceedings of the 23rd Central European Conference on Information and Intelligent Systems-CECIIS*, pages 449–456, 2012.

[51] MSDN: Microsoft. C# programming guide. `https://msdn.microsoft.com/en-us/library/67ef8sbd.aspx`. (Accessed on 01/05/2016).

[52] MSDN: Microsoft. Code generation and t4 text templates. `https://msdn.microsoft.com/en-us/library/bb126445.aspx`. (Accessed on 28/04/2016).

[53] MSDN: Microsoft. Guidelines (windows). `https://msdn.microsoft.com/en-us/library/windows/desktop/dn688964(v=vs.85).aspx`. (Accessed on 08/02/2016).

[54] MSDN: Microsoft. How to create the best user experience for your application. `https://msdn.microsoft.com/en-us/library/aa468595.aspx`. (Accessed on 08/15/2016).

[55] MSDN: Microsoft. Linq (language-integrated query). `https://msdn.microsoft.com/en-us/library/bb397926.aspx`. (Accessed on 28/04/2016).

[56] MSDN: Microsoft. Linq to sql. `https://msdn.microsoft.com/en-us/library/bb386976.aspx`. (Accessed on 28/04/2016).

[57] MSDN: Microsoft. Linq to xml. `https://msdn.microsoft.com/en-us/library/bb387098.aspx`. (Accessed on28/04/2016).

[58] MSDN Microsoft. Loading dataset schema information from xml. `https://msdn.microsoft.com/en-us/library/atchhx4f(v=vs.110).aspx`. (Accessed on 02/05/2016).

[59] MSDN Microsoft. Visual c#. `https://msdn.microsoft.com/en-us/library/kx37x362(v=vs.90).aspx`. (Accessed on 02/05/2016).

[60] MSDN Microsoft. What is an enterprise application?, apr 2016. (Accessed on 04/28/2016).

[61] TechNet Microsoft. Business application definition, 2010. (Accessed on 28/04/2016).

[62] MSDN:Microsoft. Introduction to windows forms. `https://msdn.microsoft.com/en-us/library/aa983655(v=vs.71).aspx`. (Accessed on 28/04/2016).

[63] MSDN:Microsoft. Partial classes and methods (c# programming guide). `https://msdn.microsoft.com/en-us/library/wa80x488.aspx`. (Accessed on 28/04/2016).

[64] NPTEL. Nptel :: Civil engineering - g.i.s in civil engineering. `http://nptel.ac.in/courses/105102015/36`. (Accessed on 11/05/2016).

[65] James A O'Brien. *Management Information Systems: Managing Information Technology in the Business Enterprise*. Information technology / McGraw-Hill Higher Education. McGraw-Hill/Irwin, 2004.

[66] OM OMG. Business process model and notation (bpmn) version 2.0. *Object Management Group*, 2011.

[67] Dax Pandhi. How to create the best user experience for your application. `https://msdn.microsoft.com/en-us/library/aa468595.aspx`, April April. (Accessed on 08/20/2016).

[68] Kapil Pant and Matjaz B Juric. *Business process driven SOA using BPMN and BPEL: From business process modeling to orchestration and service oriented architecture*. Packt Publishing Ltd, 2008.

[69] Mike Papazoglou, Stefano Spaccapietra, and Zahir Tari. *Advances in object-oriented data modeling*. MIT Press, 2000.

[70] Jorge-Luis Pérez-Medina, Sophie Dupuy-Chessa, et al. A survey of model driven engineering tools for user interface design. In *Task Models and Diagrams for User Interface Design*, pages 84–97. Springer, 2007.

[71] Jan Recker and Jan Mendling. Adequacy in process modeling: A review of measures and a proposed research agenda - position paper -. In Barbara Pernici and Jon Atle Gulla, editors, *The 19th International Conference on Advanced Information Systems Engineering (CAiSE'07)*, pages 235–244, Trondheim, Norway, 2007. Tapir Academic Press.

[72] Sonja Ristic, Slavica Aleksic, Ivan Lukovic, and Jelena Banovic. Form-driven application development. *Acta Electrotechnica et Informatica*, 12(1):9, 2012.

[73] Sonja Ristić, Ivan Luković, Slavica Aleksić, Jelena Banović, and Ali Al-Dahoud. An approach to the specification of user interface templates for business applications. In *Proceedings of the Fifth Balkan Conference in Informatics*, pages 124–129. ACM, 2012.

[74] Arijit Sengupta, Sriram Mohan, and Rahul Doshi. Xer-extensible entity relationship modeling. In *Proceedings of the XML 2003 Conference*, pages 140–154, 2003.

[75] Lisa Simasatitkul and Taratip Suwannasart. A tool for generating relational database schema from eer diagram. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, 2012.

[76] Ian Sommerville and Gerald Kotonya. *Requirements engineering: processes and techniques*. Worldwide series in computer science. John Wiley & Sons, Inc., 1998.

[77] SPARK. Enterprise architect - uml design tools and uml case tools for software development. `http://www.sparxsystems.com.au/products/ea/index.html`. (Accessed on 01/05/2016).

[78] Kitchens Tim. Automating software development processes, jan 2006. (Accessed on 28/04/2016).

[79] Wil MP van der Aalst. Patterns and xpdl: A critical evaluation of the xml process definition language. *BPM Center Report BPM-03-09, BPMcenter. org*, pages 1–30, 2003.

[80] W3C. Xml schema. `http://www.w3schools.com/xml/xml_schema.asp`. (Accessed on 28/04/2016).

[81] Chunyan Wang, Anthony Chiu Wa Lo, Reda Alhajj, and Ken Barker. Converting legacy relational database into xml database through reverse engineering. In *ICEIS (1)*, pages 216–221, 2004.

[82] Mathias Weske. *Business Process Management: Concepts, Languages, Architectures*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[83] WFMC. Workflow management coalition. Web page, feb 1993. http://www.wfmc.org/.

[84] WFMC. Xml process definition language version 2.2. (WFMC-TC-1025), August 2012.

[85] Stephen A White and S BEYOND. Xpdl and bpmn. *Workflow handbook*, pages 221–238, 2003.

[86] Stephen A White and Conrad Bock. *BPMN 2.0 Handbook Second Edition: Methods, Concepts, Case Studies and Standards in Business Process Management Notation*. Future Strategies Inc., 2011.

[87] Petia Wohed, Marlon Dumas, Arthur HM Ter Hofstede, and Nick Russell. Pattern-based analysis of bpmn-an extensive evaluation of the control-flow, the data and the resource perspectives (revised version. 2006.

[88] Shuyun Xu, Yu Li, and Shiyong Lu. Erdraw: An xml-based er-diagram drawing and translation tool. In *Computers and Their Applications*, pages 143–146, 2003.

[89] Saran Yamasathien and Wiwat Vatanawood. An approach to construct formal model of business process model from bpmn workflow patterns. In *Digital Information and Communication Technology and it's Applications (DICTAP), 2014 Fourth International Conference on*, pages 211–215. IEEE, 2014.

[90] Sira Yongchareon, Jian Yu, Xiaohui Zhao, et al. A view framework for modeling and change validation of artifact-centric inter-organizational business processes. *Information systems*, 47:51–81, 2015.

[91] Xiyong Zhu and Xingwang Zheng. A template-based approach for mass customization of service-oriented e-business applications. In *Proceedings of the 7th international conference on Electronic commerce*, pages 706–710. ACM, 2005.