A Tool for Generating Test Case from BPMN Diagram with a BPEL Diagram

Chaithep Nonchot and Taratip Suwannasart

***Abstract*—Currently, software development has used Business Process Model and Notation (BPMN) to explain functional behaviors of a software. Mostly, instant services from the third parties are used in software projects in order to shorten the duration of software development. Previous research[1] proposed an approach of test case generation from BPMN. They have not focused on the instant services used in software development and use a lot of time to assign boundary values of inputs in BPMN diagram by users. This research focuses on test case generating from a BPMN with a BPEL diagram which is used to explain the service behaviors. Thus, we proposed a tool to generate test cases from a BPMN with a BPEL diagram and XSD Schema.**

***Index Terms*—Software Testing, Test Case, BPMN, BPEL, XSD schema**

I. Introduction

Business Process Model and Notation (BPMN)[2] is an important diagram to explain business processes and behaviors of a software. This diagram is usually designed and created in the early stage of software development life cycle. However, test case generation can be made parallel to software development in order to reduce time and effort. Thus, testers will have time to pay attention to test the software before delivery.

In the previous studies on the test case generation from BPMN[1] [3], The researches have intended to focus only on test case generation from a BPMN diagram. Moreover, boundary values and constraint language was use to generate test cases from a BPMN diagram. These researches have not concerned an instant service which is service task in a BPMN diagram.

In this paper, we present a tool for generating test cases from a BPMN with a BPEL diagram. The tool allows users to import a BPMN diagram that explains functional behaviors of software and how it process, and to import a BPEL diagram that explains the functional behaviors of a service task and its process, and to import a XSD Schema[4] to define boundaries of input a BPMN diagram.

The rest of this paper is organized as follows: Section 2 describes background of this research. Section 3 describes the tool which is developed by our approach. Finally, section 4 provides the conclusion and a plan future work.

C. Nonchot and T. Suwannasart are with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand e-mail: Chaithep.N@student.chula.ac.th,Taratip.S@chula.ac.th

II. Background

A. Business Process Model and Notation (BPMN)[2]

The BPMN is a diagram that conforms to OMG standard for describing functional behaviors of a software to be developed. The structure of BPMN diagram is saved in XML format. Groups of notations in BPMN diagram [1] consist of five main groups: Flow Object, Data, Connecting Object, Swimlane, and Artifact.

B. Business Process Execution Language (BPEL)[5]

BPEL is a diagram that conforms to Organization for the Advancement of Structured Information Standards (OASIS) for describing functional behaviors of services used in software development based on the structure in XML format. BPEL has procedure tags which describe functional behaviors of instant services from a BPMN diagram as follows:

1. <invoke>: it is used for calling another operation via <porttype>.
2. <receive>: it is used for receiving a parameter from client to use instant service.
3. <assign>: it is used to assign value of a variable which involves instant service.
4. <reply>: it is used to send output of an instant service to client.
5. <if>: it is used to define condition of an instant service operation.

C. XSD Schema[4]

XSD Schema is a definition of syntax which stores in XML format to provides definition of input data type and input boundary value in a BPMN diagram as shown in Fig. 1. The XSD schema can define 3 definition types as follows:

1. Default: it is used to define a default value of an input.
2. Fixed: it is used to define a constant value of an input.
3. Restriction: it is used to define a boundary of an input.

```
<?xml version="1.0" encoding="UTF-8"
standalone="yes"?>
<xs:schema version="1.0" >
   <xs:element name="age" type="xs:integer"
minInclusive="0" maxInclusive="120"/>
   <xs:element name="lang" type="xs:string"
fixed="EN"/>
<xs:element name="color" type="xs:string"
default="red"/>
</xs:schema>
```

Fig.1 An example of XSD schema in XML format

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

III. DEVELOPMENT OF THE PROPOSED TOOL

This section describes our approach for developing a tool to generate test cases from a BPMN with a BPEL diagram Fig. 2 shows the schematic representation of our proposed approach.

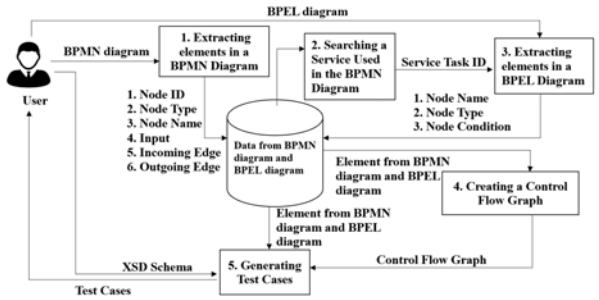

Fig. 2. Our approach for Test case generation

A. Extracting elements in a BPMN Diagram

A user can import a BPMN file into a tool for extracting BPMN elements. In Fig. 3 is a "check Permission" element which is a part of a BPMN diagram. We use XML Parser to extract the BPMN elements. After that, The BPMN elements are stored in the structure that they will be used in creating control flow graphs and test case. A BPMN element structure is shown in Table 1 and Fig.4 are result of BPMN elements that extracted by the tool.

```
<task isForCompensation="false" startQuantity="1"
completionQuantity="1" name="check Permission"
id="_17_0_4_2_fe4035d_1422165312113_321985_11745">
<incoming>_17_0_4_2_fe4035d_1422165332896_680537_11
791</incoming>
<outgoing>_17_0_4_2_fe4035d_1422165698605_310292_11
917</outgoing>
      <property name="check Permission_pin_out"
id="_17_0_4_2_fe4035d_1422165312113_321985_11745_pin
_out"/>
         <dataInputAssociation
id="_17_0_4_2_fe4035d_1422166151447_45185_12107">
<sourceRef>_17_0_4_2_fe4035d_1422166147990_449711_1
2083</sourceRef>
<targetRef>_17_0_4_2_fe4035d_1422165312113_321985_11
745_pin_out</targetRef>
         </dataInputAssociation>
   </task>
<dataStoreReference
dataStoreRef="_17_0_4_2_fe4035d_1422166147990_449711
_12083" name="employeeCode"
id="_17_0_4_2_fe4035d_1422166147996_493028_12089"/>
```

Fig. 3. An example of the BPMN element.

Table 1. A BPMN element structure

Node ID	_17_0_4_2_fe4035d_1422165312113_321985_11745
Node Type	task
Node Name	check Permission
Input	employeeCode
Incoming Edge	_17_0_4_2_fe4035d_1422166147990_449711_12083
Outgoing Edge	_17_0_4_2_fe4035d_1422165312113_321985_11745

From Table 1, BPMN elements are as follows:

1. Node ID: ID of the BPMN element.
2. Node Type: type of the BPMN element.
3. Node Name: name of the BPMN element.
4. Input: input data of the BPMN element.
5. Incoming Edge: connecting edge between a previous BPMN element and this BPMN element.
6. Outgoing Edge: connecting edge between this BPMN element and the target BPMN element.

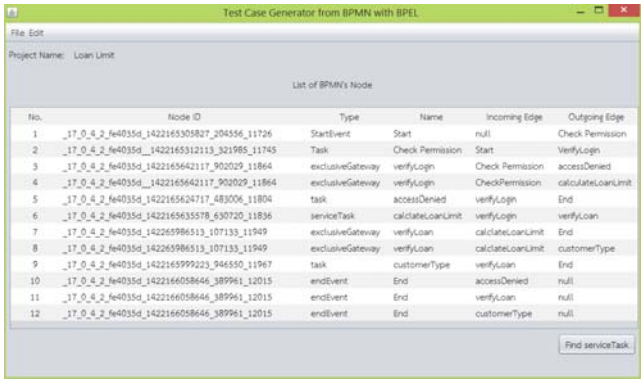

Fig.4. A result of BPMN elements

B. Searching a service Used in a BPMN Diagram

In this step, the tool uses BPMN elements from the previous step in order to search a service used in the BPMN diagram. The tool uses Node ID and Node Type to search a service used in the BPMN diagram. If we find "serviceTask", there will be a service used in the BPMN diagram. The output of this step is a service used in the BPMN diagram in order to allow a user to import a BPEL diagram. Fig. 5 shows an example service in XML format. Fig. 6 is a screen of the tool after a service is found from the BPMN diagram.

```
<serviceTask implementation="##WebService"
isForCompensation="false" startQuantity="1" completionQuantity="1" name="loanCalculate"
id="_17_0_4_2_fe4035d_1422165635578_630720_11836">
   <incoming>_17_0_4_2_fe4035d_1422165653899_107641_11873</incoming>
   <outgoing>_17_0_4_2_fe4035d_1422165941006_100775_11936</outgoing>
</serviceTask>
```

Fig. 5. An example service used

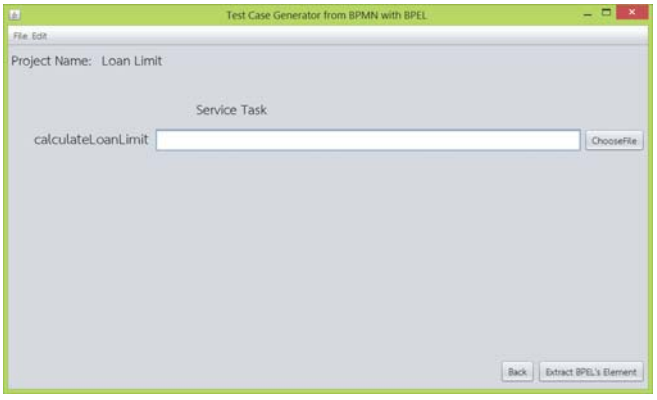

Fig.6. A screen of the tool display a service found

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

C. Extracting elements in a BPEL Diagram

In this step, a user can import a BPEL diagram that describes behaviors of the service used in the model. Fig. 7 is a "receiveID" element which is a part of a BPEL diagram. The tool uses the service task id from step 2 and a target namespace in the BPEL diagram to verify BPEL elements extraction. If the service task id from step 2 and the target namespace in BPEL diagram are both matched, the tool will extract elements from the BPEL diagram. The BPEL elements are stored in the structure that they will be used in creating control flow graphs and test cases. A BPEL element structure is shown in Table II. Fig. 8 is a result of BPEL elements that are extracted by the tool.

```
<bpel:receive name="receiveID"
partnerLink="client"
portType="tns:calculateLoanLimit"
operation="process"
inputVariable="ID"
createInstance="yes"></bpel:receive>
```

Fig. 7. An example of a BPEL element

Table II. A BPEL element structure

Node Name	receiveID
Node Type	receive
Condition	-

From Table II, the data of BPEL elements as follows:

1. Node Name: Name of the BPEL element.
2. Node Type: Type of the BPEL element.
3. Condition: Condition for the functioning of the BPEL element.

Fig. 8. A result of BPEL elements

D. Control Flow Graph Creating

In this step, we use a list of elements from step 1 and step 3 to create a control flow graph.

To create of a control flow graph from a BPMN diagram, we use a list of BPMN elements from step 1 to create a graph. Each element will be represented by a node, incoming edges, and outgoing edges. Fig. 9 is an example of BPMN diagram that is used to create control flow graph as shown in Fig. 10.

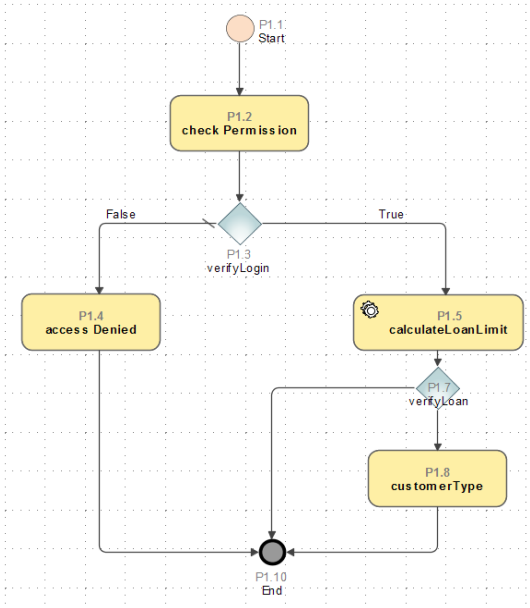

Fig. 9. An example a BPMN diagram

From Fig. 9, we can create 8 nodes as follows:

1. startEvent
2. task check Permission
3. exclusiveGateway verifyLogin
4. task accessDenied
5. serviceTask calculateLoanLimit
6. exclusiveGateway verifyLoan
7. Task customerType
8. endEvent

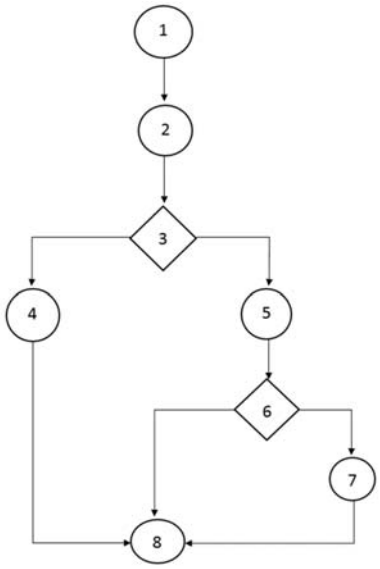

Fig. 10. A control flow graph from BPMN

To create of a control flow graph from a BPEL diagram, we use a list of elements from step 3 to create a graph. Each BPEL element will be represented by a node and a relationship between two nodes is represented with a connection. Fig. 11 is an example of a BPEL diagram that is used to create a control flow graph as shown in Fig. 12

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

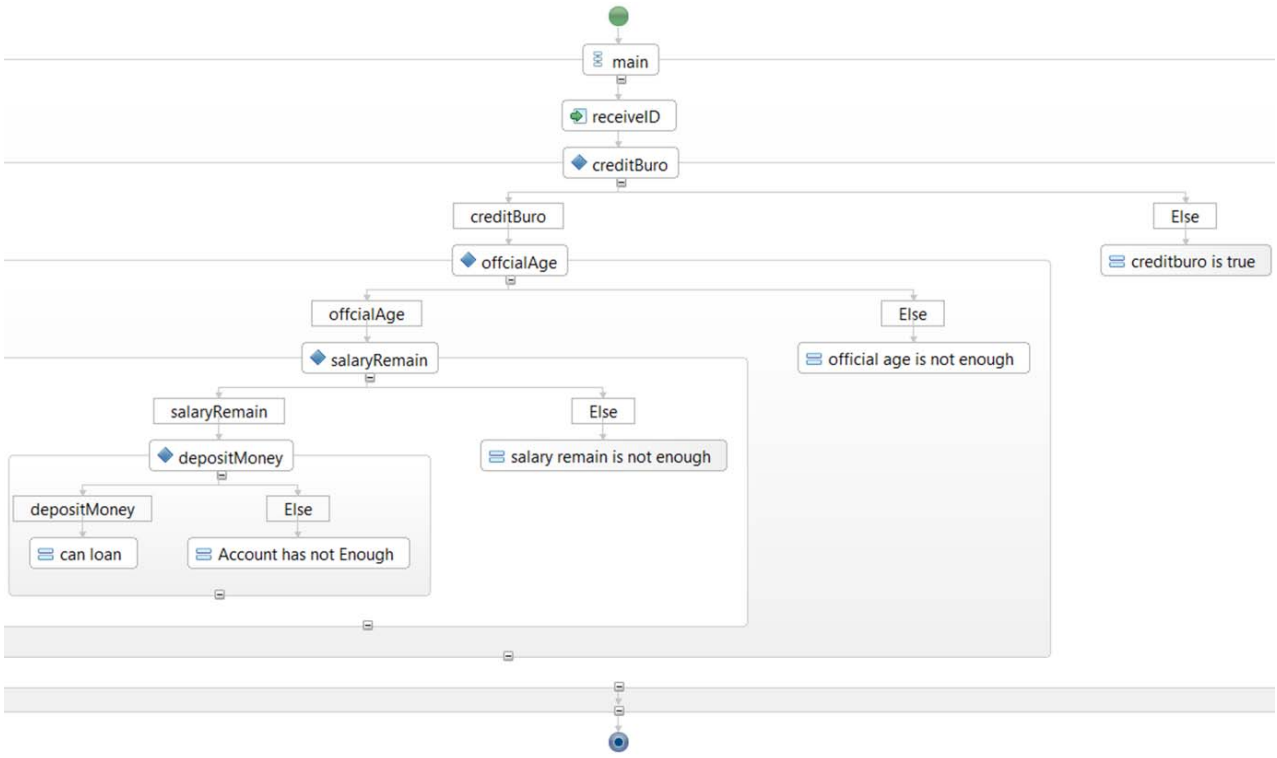

Fig. 11. An example of a service used in a BPMN diagram

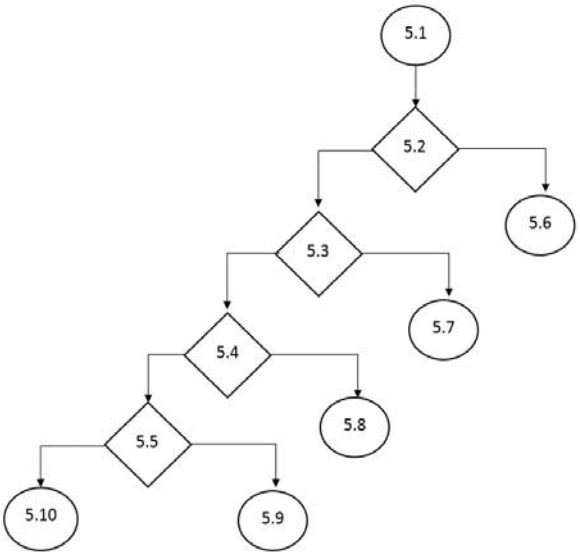

Fig. 12. A control flow graph created from a BPEL diagram

From Fig. 11, we can create 10 nodes as follows:

5.1 receiveID
5.2 creditBuro
5.3 officialAge
5.4 salaryRemain
5.5 depositMoney
5.6 creditBuro is true
5.7 Official Age is not enough
5.8 Salary remain is not enough
5.9 Account has not enough
5.10 can loan

E. Test Cases Generating

In this step, we merge the control flow graphs crated from the BPMN diagram and the BPEL diagram from the previous steps in order to create test cases from the merged graph. Fig 13 shows the result of graph merging from Fig 10 and Fig 12. The depth first search algorithm is used to find paths that cover all edges to achieve branch coverage. Fig 14 illustrates 11 paths that are derived from Fig 13 using the depth first search algorithm. Considering node 5.1 from Fig 13 using conditionExpression of the service element, we find that not every path in Fig 14 is feasible. Fig 15 shows the feasible paths.

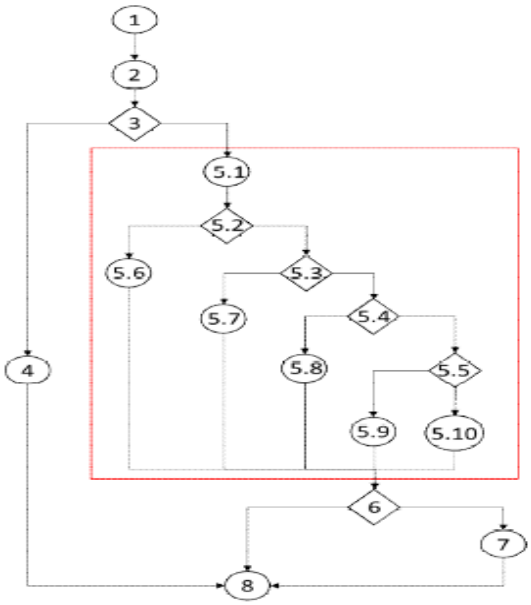

Fig. 13. A merged graph between a BPMN diagram and a BPEL diagram

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Path No.	Scenario
1	Start, Check Permission, verifyLogin, accessDenied, End
2	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, salaryRemain, depositMoney, can loan, verifyLoan, customerType, End
3	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, salaryRemain, depositMoney, The depositMoney has not enough, verifyLoan, customerType, End
4	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, salaryRemain, The salaryRemain has not enough, verifyLoan, customerType, End
5	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, The officialAge has not enough, verifyLoan, customerType, End
6	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, creditBuro is true, verifyLoan, customerType, End
7	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, salaryRemain, depositMoney, can loan, verifyLoan, End
8	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, salaryRemain, depositMoney, The depositMoney has not enough, verifyLoan , End
9	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, salaryRemain, The salaryRemain has not enough, verifyLoan, End
10	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, officialAge, The officialAge has not enough, verifyLoan, End
11	Start, Check Permission, verifyLogin, calculateLoanLimit, receiveID, creditBuro, creditBuro is true, verifyLoan, End

Fig. 14. Paths generated from the depth first search

Table III. A test case

Testcase No.	Input Data					Expected Scenario
3	employeeCode	CreditBuro	OfficerAge	SalaryRemain	DepositAccount	1. Start 2. Check Permission 3. verifyLogin 5. loanCalculate 5.1 receiveID 5.2 creditBuro 5.3 officerAge 5.4 salaryRemain 5.5 depositAccount 5.9 The Deposit Money has not enough 6. verifyLoan 8. End
	1237645891	False	5	6000	90000	

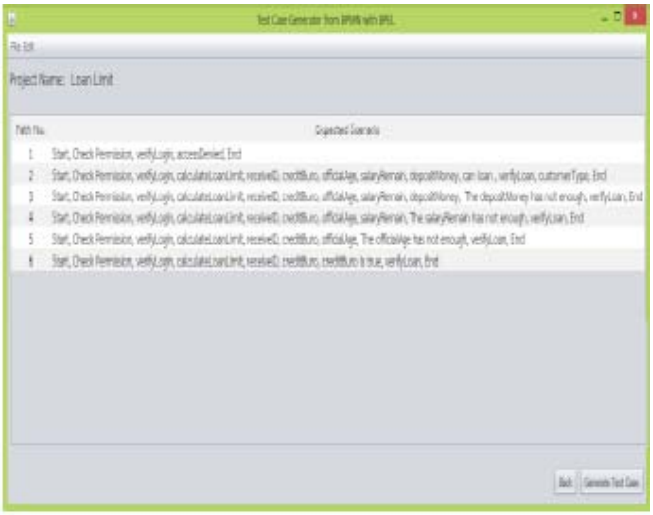

Fig.15. A display showing feasible paths

To generate test cases, we use inputs that are defined from the BPMN diagram and XSD schema file to generate test input data for each feasible path. We consider boundary values of each input and randomly generate test input data. Table III is an example of a test case that is generated from path 3 of Fig 15.

IV. CONCUSION AND FUTURE WORK

In this paper, we have presented the tool to generate test case from BPMN diagram with BPEL diagram. Our approach, an imported BPMN diagram is represented as a software, an imported BPEL diagram is represented as a service used in a software, and an imported XSD schema is represented as boundary input data. As a result, test cases will be generated. Finally, the benefit of this tool is to reduce cost and time for generating test cases.

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

REFERENCES

[1] P. Yotyawilai and T. Suwannasart, "Design of a tool for generating test cases from BPMN," in Data and Software Engineering (ICODSE), 2014 International Conference on, 2014, pp. 1-6.

[2] H. Völzer, An Overview of BPMN 2.0 and Its Potential Use, in Business Process Modeling Notation, 2010.

[3] A. Jimenez-Ramirez, R. M. Gasca, and A. J. Varela-Vaca, "Contract-based test generation for data flow of business processes using constraint programming," in Research Challenges in Information Science (RCIS), 2011 Fifth International Conference on, 2011, pp. 1-12.

[4] W3Schools. "Introduction to XSD Schema.".[Online]. Available: http://www.w3schools.com/schema/default.asp

[5] OASIS. (2010). Web Services Business Process Execution Language 2.0[Online]. Organization for the Advancement of Structured Information Standards. Available: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)