Interaction Multi-Agent Models' Automatic Alignment with MDA Higher Abstraction Level

https://doi.org/10.3991/ijoe.v19i02.37047

Nassim Kharmoum[1,2(✉)], Sara Retal[3,2], Karim El Bouchti[2], Wajih Rhalem[4],
Soumia Ziti[2]
[1]National Center for Scientific and Technical Research (CNRST), Rabat, Morocco
[2]Intelligent Processing & Security of Systems (IPSS) Research Team, Faculty of Sciences,
Mohammed V University in Rabat, Morocco
[3]SmartiLab, Moroccan School of Engineering Sciences (EMSI), Rabat, Morocco
[4]Electronic Systems Sensors and Nanobiotechnology (E2SN) Research team, ENSAM Rabat,
Mohammed V University in Rabat, Morocco
nassim.kharmoum@um5r.ac.ma

Abstract—With the massive growth of the software sector as well as the erratic needs of end users, agent-based information systems and Model Driven Architecture (MDA) approach are among the liveliest and significant fields of experimentation and improvement to emerge in the recent decade. In this vein, we suggest in this research an innovative method that automates the construction and the generation processes of the interaction multi-agent models from the business requirements engineering models at the MDA highest abstraction levels. So, our defiance is to align the Agent Modeling Language (AML) Communicative Interaction diagram with the E3value model dealing with the MDA approach. The ATLAS-Transformation Language (ATL) is applied to automate the model alignment process. The goal is to reduce project effort, time, and development costs as all alignment process is automatically done, boosting the chances of being more competitive in the software business.

Keywords—Model-Driven Architecture, automatic alignment, multi-agent modeling, AML Communicative Interaction Diagram, E3value

1 Introduction

Agent-based information systems and Model Driven Architecture (MDA) approach are among the most active and major disciplines of experimentation and development to emerge in the last years [1–3]. The aim is to track the massive growth of the software business as well as the capricious needs of end users. In this regard, this paper tackles the generation of our information system models, enlightening the multi-agent interaction models from the MDA higher abstraction level. In this vein, we deal with the MDA, which is an approach spearheaded by the Object Management Group (OMG) [3], and delivers for this approach three levels of abstraction: higher, medium, and lower levels.

We start with the higher-level anointed Computation-Independent Model (CIM), that presents the business requirements models, and its models do not treat the technical details. The medium level is called Platform-Independent Model (PIM), which forms an abstract design model without knowing any execution details. However, since Platform-Specific Model (PSM) is directly connected to the platform execution, it is the lowest abstraction level [4,5].

After highlighting all the OMG proposed levels, and since one of the MDA approach purposes is the model creation and generation between these different levels; we have in the literature that some researchers escape the first transformation from the highest levels (CIM and PIM levels) since these levels contain higher abstraction model character, most of which do not have defined standards [6]. Our challenge in this contribution is to propose an innovative solution that puts much focus on the construction and the generation of the MDA's Highest levels; to align the Agent Modeling Language (AML) Communicative Interaction diagram [2] with the E3value model [7].

In this paper, we generate the AML Communicative Interaction diagram, as it is a semi-formal graphical modeling language in the field of Artificial Intelligence (AI) [8] that is founded on the UML 2.x architecture [9], and it was within the scope of Agent-Oriented Software Engineering (AOSE) [10]. In this modeling language, the agent is the main element, which is a hunk of a concrete or virtual entity located in an environment, has resources, and is possess various characteristics; especially, we notice the autonomy, learning, and interaction. The generated diagram models the social interaction of the communication message between different multi-agent environments, considering the time order of several communication messages. Furthermore, the E3value is a model that represents graphically and unifies the business value model and designs how economic value is produced, exchanged, and consumed in a network of multiple actors. The purpose is to ensure a rigorous definition and common understanding of the business model [11]. Also, the E3value model introduces the engineering models of value-based requirements [7, 12].

In our proposal, the model alignments are done automatically by using the ATLAS-Transformation Language (ATL) [13]. ATL is based on the subsequent OMG standards; XML Metadata Interchange (XMI) [14], Object Constraint Language (OCL) [15], and Meta Object Facility (MOF) [16]. The automatic alignment will allow us to hasten the development of the information system process, as It reduces effort and time, lowering costs and ensuring competitiveness in the software sector. Hence, it helps to narrow the gap between the business value model and the multi-agent interaction model. Also, supporting stakeholders who do not comprehend AML Communicative Interaction diagram creation to produce their diagrams.

To get to the heart of our method, this paper has been divided into five sections. The following section describes the related work. The third section explains our proposal; it demonstrates the source and target descriptive metamodels. The mapping rules are explained in the fourth section. Section five, illustrate our proposed method. The sixth section offers an analysis and discussion of all obtained results. The last section concludes our contribution and emerges our upcoming work.

2 Related work

In this section, we look at various methods dealing with agent modeling in the context of the MDA in the last decade. Unfortunately, few studies focus on our chosen modeling purpose, but we were able to find some.

First, Maalal and Addou (2012) [17] generated automatically the Java Agent Development (JADE) [18] code in the PSM level from the AUML (Agent UML) Class diagram in the PIM level dealing with AndroMDA tool [19].

In the same vein, Elammari and Issa (2013) [20] generate the UML Class diagram through the substantiation of the JADE platform in the PSM level from the UML Class diagram in the PIM level and UML Use Case diagram and Use Case map in CIM level based on Protégé [21] and Poseidon [22] tools.

A Model-driven approach to generate the Agilla agents automatically using Acceleo, from the UML Activity diagram, is proposed by Di Marco and Pace (2013) [23].

Besides, Nouzri and El Fazziki (2015) [24] suggest an automatic mapping from the BPMN model to the JADEX model by employing the ATL transformation language.

Wautelet and Kolp (2016) in [25], deliver the CIM level with the I* (I-Star) [26] to generate the BDI multi-agent system diagrams in the PIM level.

Furthermore, Fernàndez-Isabel and Fuentes-Fernàndez (2017) [27] extend a generic traffic model to specific agent platform requirements, focusing on the UML model for Road Traffic at the PIM level to generate the UML model for Road traffic automatically through the instantiation of the JADE platform.

Kouissi et al. (2018) [28] based on the AndroMDA tool to generate the JADE model automatically in the PSM level from the AUML Class diagram in the PIM level.

Finally, Çam et al. [29] generate the RePast Flowchart Diagram [30] from the SysML [31] diagram automatically dealing with ATL transformation language. For the same study, a reverse transformation process is given for the same chosen models.

After studying the different works, we have that most papers have a graphical representation for representation of their source and target models, and most of them are based on metamodels for their models' generation, which is recommended by the MDA approach. Also, we have most of the papers dealing with a transformation language to automate their models' generation, others use their own tool or are based on a manual way for their transformation, and the majority define their mapping rules. Thus, most of the methods are based on a case study to evaluate their methods.

So, in this contribution, we suggest an innovative and validated method that automatically aligns the AML Communicative Interaction Diagram with the E3value model for a graphical representation and benefits from the metamodel definition. Thus, all mapping rules are defined. Moreover, the following section will present and detail our proposed method.

3 Proposed method

In this section, we concentrate on our innovative method that generates the multi-agent behavior and its social interaction models. The goal is to automatically produce and generate the highest MDA levels of our information system. We emphasize that our proposal will profit from the experience we gained while generating models at MDA's highest levels [32–34].

Our challenge is to generate automatically the AML Communicative Interaction diagram at the PIM level from the E3value at the CIM level; applying the vertical and exogenous transformation. The source E3value model expresses the value-based requirements engineering models, as the requirements engineering plays a significant role in all types of software development processes. The generated diagram performs the multi-agent behavior and social interaction model at the PIM level and is used to achieve interactions between these AML elements, especially "Agents" and "Entity Roles". The AML Communicative Interaction diagram is based on the UML 2.x Interaction diagrams suchlike the UML Sequence diagram.

To evolve the proposed method under the MDA approach, we track the steps depicted in Figure 1 that explains the generation process. We first determine the metamodels; the objective is to create and generate the right models and to offer automated model alignments [35]. Therefore, the E3value source metamodel illustrates the structure of the business value model on the higher abstraction level. After that stage, we define the target AML metamodels depicting the structure of the generated multi-agent behavior and social interaction model at the highest abstraction level. Thus, we establish the set of mapping rules that will be used by ATL to produce the selected target models automatically from the source model.

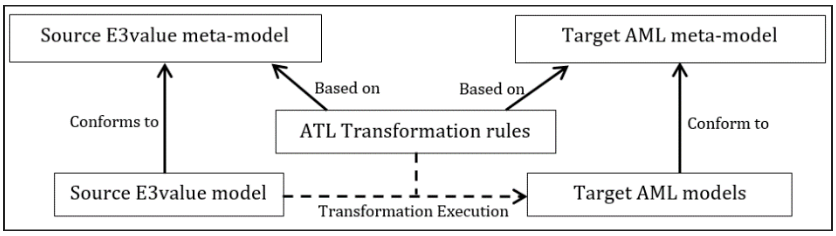

Fig. 1. The transformation process of the multi-agent method

3.1 Source metamodel: E3value metamodel

The proposed source metamodel explains the structure of our E3value model, which must adhere tothe E3value elements [7], as well as the value viewpoint representation of the information system (Figure 2).

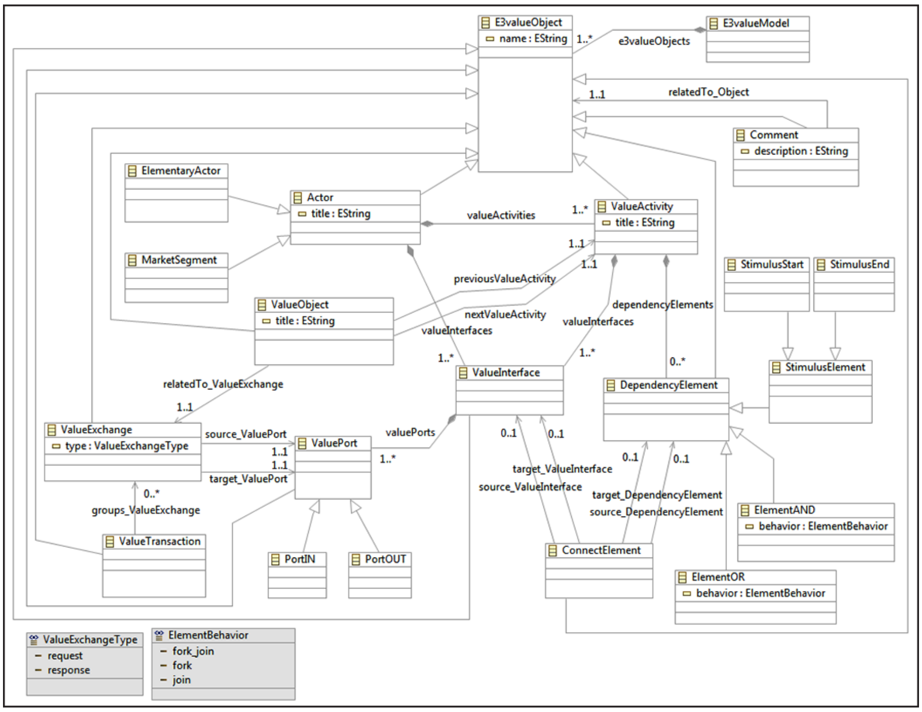

Fig. 2. The E3value metamodel

3.2 Target metamodel: AML Communicative Interaction metamodel

In this part, we demonstrate the construction of the AML Communicative Interaction metamodel. The AML Communicative Interaction diagrams are generated automatically from the E3value source model. They describe the system behavior by reproducing, in a standard way, the communicative interaction messages in the multi-agent systems, taking into account the time order of different communication messages. The AML Communicative Interaction metamodel (see Figure 3) include the above elements that are chiefly deduced from [2].

3.3 Source metamodel construction rules

The creation of the E3value model must adhere to their proposed metamodel (Figure 2) as the only restriction for our source model.

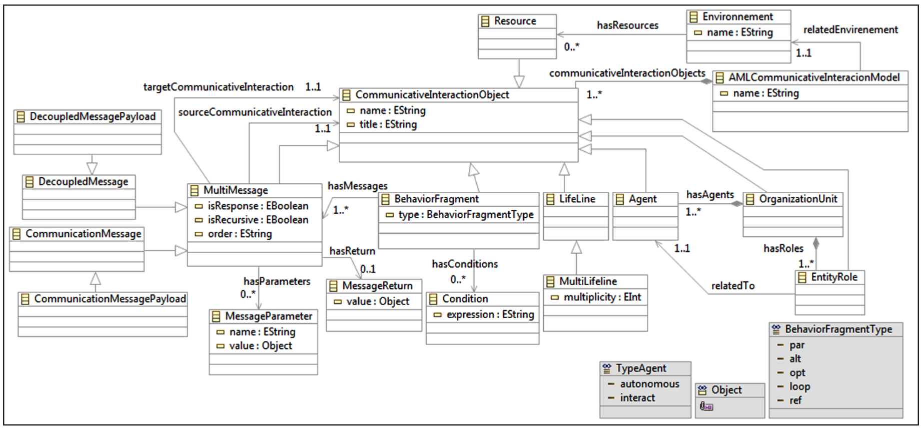

Fig. 3. The AML Communicative Interaction target metamodel

4 Mapping rules to align the AML Communicative Interaction diagrams with E3value model

This section specifies the mapping rules to align the AML Communicative Interaction diagrams with the E3value model. To build a well-detailed business information system, we will not produce all Communicative Interactions in a unique diagram. But we will produce different AML Communicative Interaction diagrams concentrating on the sub-function system. For our case, the E3value's "Value Activity" element serves as a sub-function for any business information system. Therefore, this part reveals the mapping rules suggested in our method via simple diagrams. The generation is automatically achieved with the ATLAS-Transformation Language (ATL), and it is based on its relevant source and target metamodels. Additionally, Figures 4 and 5 exhibit the set of mapping rules via ATL.

- **Rule 1:** we transform the E3value "Elementary Actor" or "Market Segments" which are subclasses of an "Actor" to AML Communicative Interaction diagram "Agent". We select the "Actor" who performs the selected "Value Activity"; via the helper "isActorInclude()", with all its providers using the helper "get-Providers()". Thus, the generated "Agents" provide the same "name" and "title" as the source "Actor" (see Figure 4);
- **Rule2:** allows generating "EntityRole" and a "MultiLifeline" from the selected "Value Activity". The "EntityRole" is related to the generated "Actor" in "Rule1", while "MultiLifeline", which is "Lifeline's" sub-type used to represent multiple participants in the interaction. And we concatenate the suffix "subsystem" to the

generated "MultiLifeline" for its "name" and "title". To show the interaction between AML Communicative Interaction diagram elements. We add four "Communication-Message" types to this rule in relation with the generated "EntityRole" and this "MultiLifeline", which are: "communicationMessageRequest", "communication-MessageRecursive", "communicationMessageResponse_favorable", and "communicationMessageResponse_unfavorable". Those "CommunicationMessages" have the subsequent attributes: "order" defines the execution sequence of the "CommunicationMessage", "name" provides the names of each "CommunicationMessage", "title" gives the name of each "CommunicationMessage", "sourceCommunicativeInteraction" specifies the source element of each "CommunicationMessage", and "targetCommunicativeInteraction" gives the target element of each "CommunicationMessage". The value of the "order" attribute is computed using two helpers: "getCounter()" and "getCounterForLastMessages()".

The "CommunicationMessage" "name" takes several prefixes: "need", "verify", "give", and "stop" as follows for the requested, recursive, favorable and unfavorable responses "CommunicationMessage". In addition, in "Rule 2", we create the "BehaviorFragment," which groups the produced two messages "communicationMessageResponsee_favorable" and "communicationMessageResponse_unfavorable", with "type" "alt" (alternative operator) (see Figure 4);

- **Rule3:** transforms "Value Activities" that link the selected "Value Activity" dealing with the "isValue-ActivityInclude()" helper, to "EntityRole" and we relate it to the generated "Agent" that matches (see Figure 5);
- **Rule4:** generates the "CommunicationMessage" between the generated "MultiLifeline" and "Entity-Role" in "Rules 2 and 3". We additionally generate four "CommunicationMessages" for that, which are: "communicationMessageRequest", "communicationMessageRecursive", "communicationMessage Response_favorable", and "communicationMessageResponse_unfavorable"; these "CommunicationMessages" contain the attributes: "order", "name", "title", "source-CommunicativeInteraction" and "targetCommunicativeInteraction". As the "Rule2" the "order" attribute value is computed relatively to the two helpers: getCounter() and getCounterWithoutIncrement(). Also, in the "Rule 4", we produce the "BehaviorFragment" which envelops the produced "CommunicationMessages", "communicationMessageResponse_favorable" and "communicationMessageResponse unfavorable", with "type" "alt" (see Figure 5);
- **Rule5:** generates the "Environnement" and "OrganizationUnit". The "OrganizationUnit" group all the generated "Agents" and "EntityRoles". At the end, this rule groups all generated elements in this sectionto the "AMLCommunicativeInteraction Model" (see Figure 5).

```
ATL transformation rules
Initialization: We initialize the name of «Value Activity» to be generated, for example in our case study we
choose: 'accessing_library_resources' :
helper def : selectedValueActivity : String = 'accessing_library_resources';
rule Rule1 {
   from
      actor : MME3value!Actor(actor.oclIsKindOf(MME3value!Actor)and actor.isActorInclude())

    to
      agent : MMAMLCommunicativeInteraction!Agent (
        name <- actor.name,
        title <- actor.title)

    do {
      for (act in actor.getProviders()){
        if MMAMLCommunicativeInteraction!Agent.allInstances() -> exists(agt | agt.name = act.name)
          then OclUndefined
          else thisModule.newAgent(act)
          endif;
        }
      }
}
rule Rule2 {
  from
     valueActivity : MME3value!ValueActivity(valueActivity.name = thisModule.selectedValueActivity)
   to
     entityRole : MMAMLCommunicativeInteraction!EntityRole (
        name <- valueActivity.name,
        title <- valueActivity.title,
        relatedTo <- valueActivity.refImmediateComposite()),

     multiLifeline : MMAMLCommunicativeInteraction!MultiLifeline (
        name <- valueActivity.name + '_sub-system',
        title <- valueActivity.title + ' sub-system'),

     communicationMessageRequest : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- thisModule.getCounter(),
        name <-'need_' + valueActivity.name,
        title <-  communicationMessageRequest.order + ': need ' + valueActivity.title,
        sourceCommunicativeInteraction <- entityRole,
        targetCommunicativeInteraction <- multiLifeline),

     communicationMessageRecursive : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- thisModule.getCounter(),
        name <- 'verify_' + valueActivity.name,
        title <- communicationMessageRecursive.order + ': verify ' + valueActivity.title,
        sourceCommunicativeInteraction <- multiLifeline,
        targetCommunicativeInteraction <- multiLifeline,
        isRecursive <- true),

    communicationMessageResponse_favorable : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- valueActivity.getCounterForLastMessages() ,
        name <- 'give_' + valueActivity.name,
        title <- communicationMessageResponse_favorable.order + ': give ' + valueActivity.title,
        sourceCommunicativeInteraction <- multiLifeline,
        targetCommunicativeInteraction <- entityRole,
        isResponse <- true),

     communicationMessageResponse_unfavorable : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- valueActivity.getCounterForLastMessages(),
        name <- 'stop_' + valueActivity.name,
        title <- communicationMessageResponse_unfavorable.order +': stop ' + valueActivity.title,
        sourceCommunicativeInteraction <- multiLifeline,
        targetCommunicativeInteraction <- entityRole,
        isResponse <- true  ),

      behaviorFragment : MMAMLCommunicativeInteraction!BehaviorFragment (
        name <-  communicationMessageRequest.name +  '_response',
        title <- 'need ' + valueActivity.title + ' response',
        type <- #alt,
        hasMessages <- Sequence{communicationMessageResponse_favorable,
                          communicationMessageResponse_unfavorable})
}
```

Fig. 4. ATL mapping rules from E3value model to AML Communicative Interaction diagram – part 1

```
ATL transformation rules
rule Rule3 {
   from
      valueActivity : MME3value!ValueActivity(valueActivity.name <> thisModule.selectedValueActivity
                       and  valueActivity.isValueActivityInclude())
   to
       entityRole : MMAMLCommunicativeInteraction!EntityRole (
         name <- valueActivity.name,
         title <- valueActivity.title,
         relatedTo <- MMAMLCommunicativeInteraction!Agent.allInstances() ->
                        select(agt | not agt.oclIsUndefined() and agt.name =
                        valueActivity.refImmediateComposite().name) -> collect(agt | agt))
}
rule Rule4 {
   from
      valueObject :  MME3value!ValueObject(valueObject.relatedTo_ValueExchange.type = #response
                   and valueObject.nextValueActivity.name = thisModule.selectedValueActivity )
    to
      communicationMessageRequest : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- thisModule.getCounter(),
        name <- 'need_' + valueObject.name,
        title <- communicationMessageRequest.order + ': need ' + valueObject.title,
        sourceCommunicativeInteraction <- valueObject.nextValueActivity,
        targetCommunicativeInteraction <- valueObject.previousValueActivity),

      communicationMessageRecursive : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- thisModule.getCounter(),
        name <- 'verify_' + valueObject.name,
        title <- communicationMessageRecursive.order + ': verify ' + valueObject.title,
        sourceCommunicativeInteraction <- valueObject.previousValueActivity,
        targetCommunicativeInteraction <- valueObject.previousValueActivity,
        isRecursive <- true),

      communicationMessageResponse_favorable : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- thisModule.getCounter(),
        name <- 'give_' + valueObject.name,
        title <- communicationMessageResponse_favorable.order + ': give ' + valueObject.title,
        sourceCommunicativeInteraction <- valueObject.previousValueActivity,
        targetCommunicativeInteraction <- valueObject.nextValueActivity,
        isResponse <- true),

       communicationMessageResponse_unfavorable : MMAMLCommunicativeInteraction!CommunicationMessage (
        order <- thisModule.getCounterWithoutIncrement(),
        name <- 'stop_' + valueObject.name,
        title <- communicationMessageResponse_unfavorable.order +': stop ' + valueObject.title,
        sourceCommunicativeInteraction <-  valueObject.previousValueActivity,
        targetCommunicativeInteraction <- valueObject.nextValueActivity,
        isResponse <- true),

       behaviorFragment : MMAMLCommunicativeInteraction!BehaviorFragment (
        name <-  communicationMessageRequest.name + 'response',
        title <- 'need ' + valueObject.title + ' response',
        type <- #alt,
        hasMessages <- Sequence{communicationMessageResponse_favorable,
                          communicationMessageResponse_unfavorable} )
}
rule Rule5 {
   from
      e3valueModel : MME3value!E3valueModel
   to
      environnement : MMAMLCommunicativeInteraction!Environnement (
         name <- e3valueModel.name),

      organizationUnit : MMAMLCommunicativeInteraction!OrganizationUnit (
         name <- thisModule.selectedValueActivity,
         title <- thisModule.selectedValueActivity,
         hasAgents <- MMAMLCommunicativeInteraction!Agent.allInstances(),
         hasRoles <-  MMAMLCommunicativeInteraction!EntityRole.allInstances()),

      AMLCommunicativeInteracionModel : MMAMLCommunicativeInteraction!AMLCommunicativeInteractionModel (
        name <- e3valueModel.name,
        relatedEnvirenement <- environnement,
        communicativeInteractionObjects <- MMAMLCommunicativeInteraction!OrganizationUnit.allInstances()
                  -> union(MMAMLCommunicativeInteraction!MultiLifeline.allInstances())
                  -> union(MMAMLCommunicativeInteraction!BehaviorFragment.allInstances())
                  -> union (MMAMLCommunicativeInteraction!CommunicationMessage.allInstances()))
}
```

Fig. 5. ATL mapping rules from E3value model to AML Communicative Interaction diagram – part 2

5 Case study

To illustrate and evaluate our proposed method, the "Service Delivery Laboratory management" case study gives an example of the proposed models' alignment. We mention that our case study is performed inside Eclipse Modeling Framework (EMF) [36], including its required plugins. For our case, we based on the "ATL Eclipse plugin" as a standard tool for transformation, Figure 6 demonstrates the practical case study in three main folders. This part connects each file in Figure 6 with their generated and created figures:

- "E3value_metaModel.ecore" with "E3value_metaModel.ecore" diagrams represent the E3value source metamodel (see Figure 2);
- "AMLCommunicativeInteraction_metaModel.ecore" with "AMLCommunicativeInteraction_metaModel.ecore diagram" represent the AML Communicative Interaction target metamodel (see Figure 3);
- "E3value_model.xmi" illustrates the XMI format of our proposed E3value-source model (see Figure 7), which is depicted graphically in Figure 9;
- "AMLCommunicativeInteraction_model.xmi" illustrates our generated AML Communicative Interaction diagram in XMI format (see Figure 8), which is depicted graphically in Figure 10;
- "E3valueToAMLCommunicativeInteractionTransformationRules.atl" and "E3valueToAMLCommunicativeInteractionTransformationRules.asm" illustrate our mapping rules from the E3value model to AML Communicative Interaction diagram. (see Figures 4 and 5).

We adopt the XMI format for our case study (see Figures 7 and 8) since it is an OMG standard; it also allows defining, exchanging, manipulating, and integrating XML data and objects, to better envision the generated model.

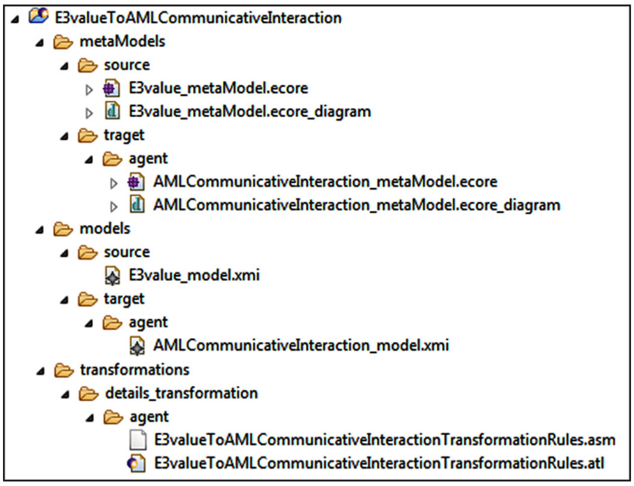

Fig. 6. Practical case structure for multi-agent method

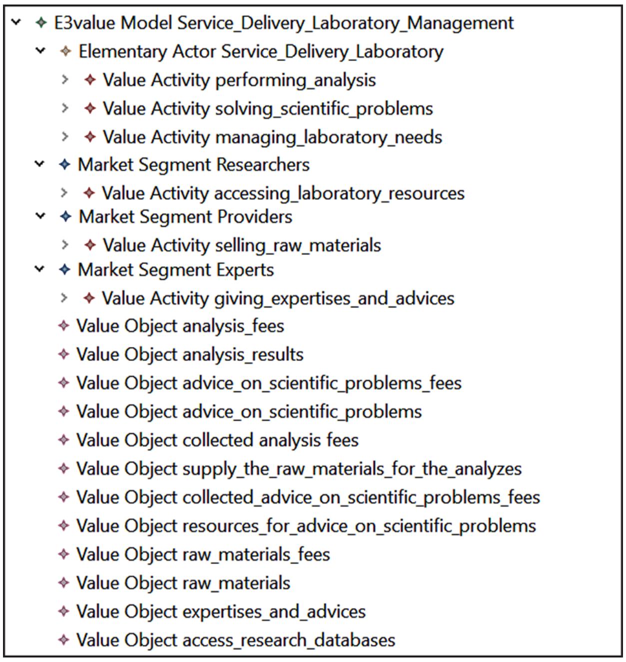

Fig. 7. Source E3value model in XMI format

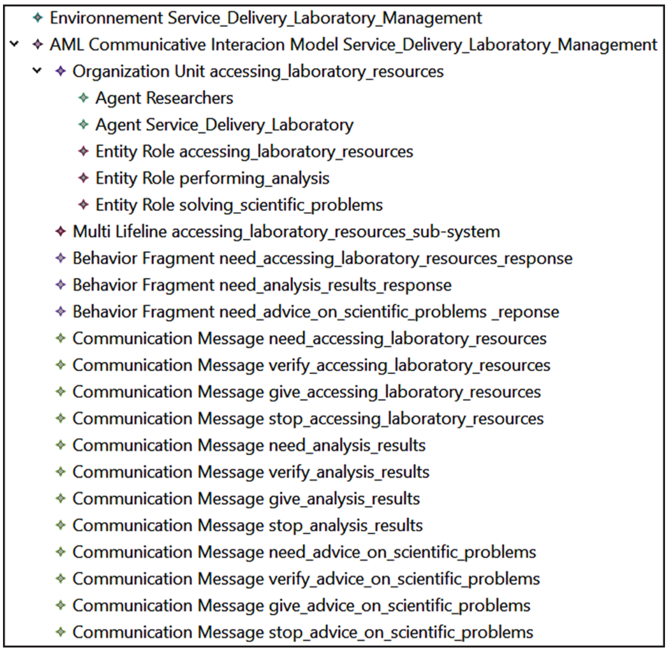

Fig. 8. AML Communicative Interaction diagram generated in XMI format

5.1 Proposed E3value model

We focus on the E3value model in the source model, which presents the business value models of the "Service Delivery Laboratory management," as shown in Figures 7 and 9.

The actors might be elementary actors like "Service Delivery Laboratory" or Market Segments like "researchers", "providers" and "experts". Each actor can participate in at least one value activity, in our model the researchers can access laboratory resources, the Service Delivery Laboratory can perform analysis, solve scientific problems, and manage laboratory needs, the Providers sell raw materials, while experts give expertise and advice. The value objects are fees of analysis, advice on scientific problems, training, raw materials, as well as expertise and advice.

For a thorough understanding of the business value model, the E3value model utilizes "value exchange" to define the direction of dependency paths. These latter, link E3value dependency elements, which can be "StimulusStart", "StimulusEnd", "ElementAND", or "ElementOR".

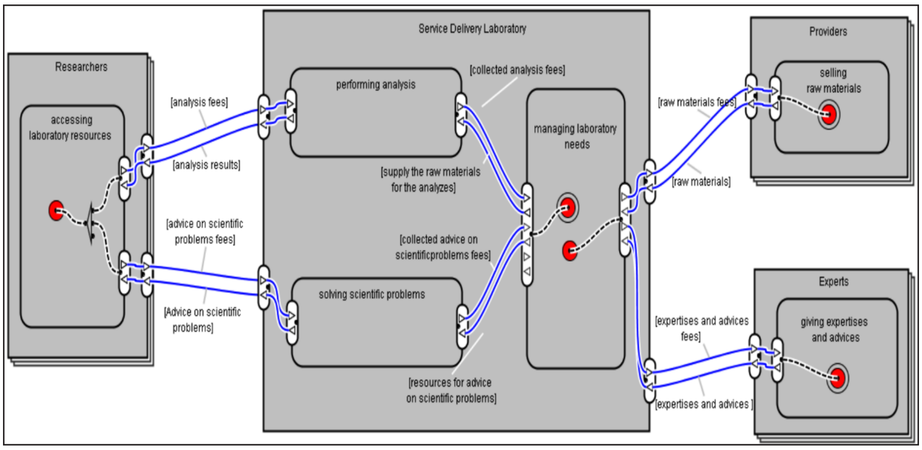

Fig. 9. E3value source model in graphical format

5.2 Generated AML Communicative Interaction diagrams

As previously indicated and to acquire thorough AML Communicative Interaction diagrams, we will obtain these diagrams based on their system sub-functions. We will align the AML Communicative Interaction diagram per E3value "Value Activity", therefore we can produce six AML Communicative Interaction diagrams, each of which corresponds to one of the following "Value Activities": "accessing laboratory resources", "performing analysis", "solving scientific problems", "managing laboratory needs", "selling raw materials", and "giving expertise and advice".

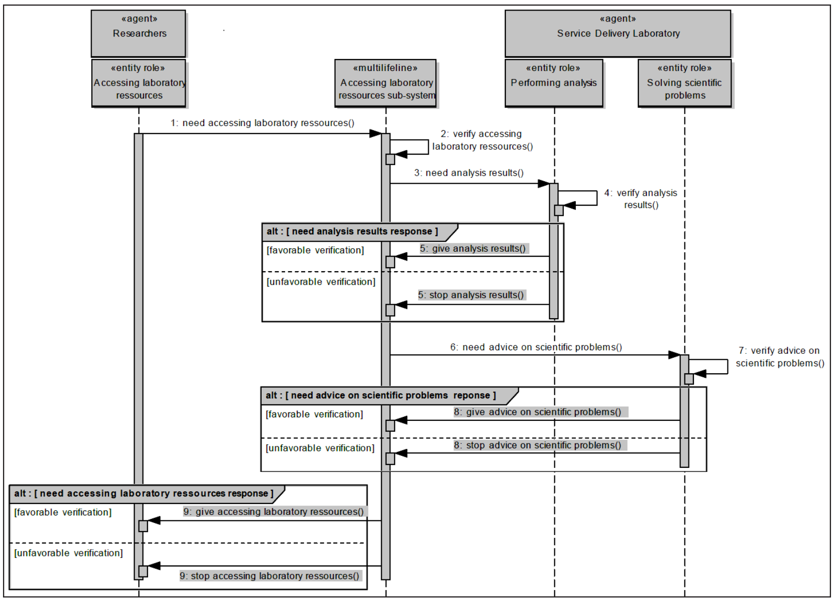

Fig. 10. AML Communicative Interaction diagram generated in graphical format

As we can see, all AML Communicative Interaction diagrams are generated in the same way. In this scenario, we highlight the first "Value Activity" "accessing library resources". Figure 8 illustrates the chosen UML Sequence diagram in XMI format, that we represented it graphically in Figure 10. Resultantly, the adopted AML Communicative Interaction diagram will be generated as follows: the "Actors": "Researchers" and "Service Delivery Laboratory" are transformed into "Agents", which are one of the interactive elements in our method. The chosen "Value Activity": "accessing laboratory resources" will then be aligned into the "Entity Role" of the same name and the "MultiLifeline" with the same name and to the "MultiLifeline" with the name "Accessing laboratory resources sub-system". "Value Activities" that join the selected "Value Activity": "accessing laboratory resources" are mapped to "Entity Roles", and in this case, we distinguish two "Entity Roles": "Performing analysis" and "Solving scientific problems". And we use "Communication Message" elements to ensure interaction between the created elements. In our proposal, we suggest the following "Communication Messages" types: we start with the service request, in which we add the prefix "need" for instance "need accessing laboratory resources". After, we add the prefix "verify" to the second "CommunicationMessage", to verify what is asked in the previous "Communication Message", for instance: "verify accessing laboratory resources". Thus, we group in one "Behavior Fragment" with an alternative operator (alt) the third and the fourth "Communication Message"; as this type of "Behavior Fragment" possess conditions. In this vein, we propose two conditions; the first is related to the third "Communication Message" that represents a favorable result such as "give accessing laboratory resources", while the second represents an unfavorable result such as "stop accessing laboratory resources" and is related to the fourth "Communication Message". Moreover, the four generated "Communication Messages" have the attribute "order" that is placed at the head of each message to consider the timing order of various communications.

6 Analysis and discussion

To analyze and discuss the proposed multi-agent interaction generation models' method, we shed more light on the studied methods in the related work section. In this regard, we concentrate on the OMG guidances [3] and the subsequent work [37,38] to define our evaluation criteria. As a result, we define our final criteria based on the model level construction, the model transformation model, and the use of assessment methodology.

Table 1 shows our evaluation criteria analysis of all studied methods with our proposal. For that, the table lines present studied papers, whereas the columns show the defined criteria.

Concerning the model construction criteria, all approaches position their models in the MDA approach. For that purpose, methods [20], [24], [25] as well as our proposal, start their generation from the CIM level, while the rest of the methods starting their models' generation from the PIM level except the paper [29] deal with the PIM level and also PSM level for their reverse transformation. However, most authors use the graphical representation for their source and target models, except the methods dealing with the generation of the PSM models, which generate textual models format. for the agent models representation, we have "AUML Class diagram" is used by [17] and [17] to generate their "JADE (Java Agent DEvelopment)" textual model. "UML Class diagram through the instantiation of the JADE platform" is used by [20] and [27], we have "Agilla agents" is used by [23], "BDI multi-agent system diagrams" is used by [25], "RePast Flowchart Diagram" is used by [29], whereas, "AML (Agent Modeling Language)" it was the object the [24] and our proposed method.

Referring to the transformation criteria, we distinguish that all methods are based on the metamodels, except the first transformation of the method [20]. Thus, for the transformation automation, we notice that some researchers transform their models relying on a transformation language; for instance, "AndroMDA" in [17] and [28]. We have also "Acceleo" in [23] and "ATL" in [24], [27], [29], and our proposal. However, we remark that in some methods, the models' transformation is performed via tool; such as "Protégé" [21] and "Poseidon" [22] in [20]. Furthermore, for the mapping rules definition, most of the studied work explains clearly the mapping rules excluding [17] and [28]. Concerning the evaluation approaches criteria, we note that all studied methods, except [29], employ a case study to evaluate their methodology.

Table 1. Studied papers comparison via evaluation criteria for multi-agent method

Studied papers	Source model			Target model			Transformation					Assessment Methodology
	Graphical representation	MDA level	Representation	Graphical representation	MDA level	Representation	Based Meta-models	Is Automatic	Type	Representation language	Mapping rules Definition	
Maalal and Addou (2012) [15]	Yes	PIM	AUML Class diagram		PSM	JADE (Java Agent DEvelopment)	Yes	Yes	Transformation language	AndroMDA		Case study
Elammari and Issa (2013) [18]	Yes	CIM	UML Use Case diagram Use case map	Yes	PIM	UML Class diagram			Human language			Case study
	Yes	PIM	Generated: UML Class diagram		PSM	UML Class diagram through the instantiation of the JADE platform	Yes	Yes	Tools	Protégé And Poseidon	Yes	
Di Marco and Pace (2013) [21]	Yes	PIM	UML Activity diagram		PSM	Agilla agents	Yes	Yes	Transformation language	Acceleo	Yes	Case study
Nouzri and El Fazziki (2015) [22]	Yes	CIM	BPMN (Business Process Model and Notation)	Yes	PIM	AML (Agent Modeling Language)	Yes	Yes	Transformation language	ATL	Yes	Case study
Wautelet and Kolp (2016) [23]	Yes	CIM	I* (I-Star)	Yes	PIM	BDI multi-agents system diagrams	Yes		Human language		Yes	Case study
Fernández-Isabel and Fuentes-Fernández (2017) [25]	Yes	PIM	UML model for Road Traffic		PSM	UML model for Road Traffic through the instantiation of the JADE platform	Yes	Yes	Transformation language	ATL	Yes	Case study
Kouissi et al. (2018) [26]	Yes	PIM	AUML Class diagram		PSM	JADE (Java Agent DEvelopment)	Yes	Yes	Transformation language	AndroMDA		Case study
Çam et al. (2020) [27]	Yes	PIM	SysML diagram	Yes	PSM	RePast Flowchart Diagram	Yes	Yes	Transformation language	ATL	Yes	
	Yes	PSM	RePast Flowchart Diagram	Yes	PIM	SysML diagram						
Our proposal	Yes	CIM	E^3value model	Yes	PIM	AML (Agent Modeling Language) Communicative Interaction Diagram	Yes	Yes	Transformation language	ATL	Yes	Case study

After reviewing all the studied approaches, we discovered that the paper [24] validates all the proposed criteria. The authors align their multi-agent diagram automatically, starting with the BPMN diagram at the CIM level dealing with ATL transformation language. They evaluate their method, focusing on a case study. In this method, we also validate all proposed criteria. Thus, we are the only ones who generated the AML diagram at the PIM level, starting with a straightforward model, which is the E3value model at the CIM level. Also, the executions of the proposed mapping rules are metamodel based; and, these diagrams are aligned automatically using ATL Transformation language. We relied on a case study for the assessment methodology.

7 Conclusion

Our objective in this paper was to submit an innovative method for business information systems. The proposed method aims to ensure the software industry's growth and to reply to end-user demands. To win this defiance, we align the AML Communicative Interaction diagram at the PIM from the E3value model at the CIM level. The source E3value model represents the business value model, whereas the generated AML diagram symbolizes multi-agent behavior and social interaction.

Conclusively, we notice that applying the proposed method in business information systems will let produce correct multi-agent behavior and social interaction models at the highest MDA level. We start with a simple source model which can be built by non-technical stakeholders who do not grasp computer modeling. On the one hand, it will reduce the effort, the time. Consequently, the development cost of business projects as all alignment is automated. This method increases the opportunity of being more rivals in the software industry. On the other hand, based on the generated AML Communicative Interaction diagram will promote the construction and the generation of different models for other lower levels, such as the PSM level. Our challenge in future contributions is to generate other models to continue the analysis and the design of the proposed business information system.

8 References

[1] M. Luck, P. McBurney, and C. Preist, Agent technology: enabling next generation computing (a roadmap for agent based computing). AgentLink, 2003. https://doi.org/10.1023/B:AGNT.0000038027.29035.7c

[2] R. Cervenka and I. Trencansky, The Agent Modeling Language-AML: A Comprehensive Approach to Modeling Multi-Agent Systems. Springer Science & Business Media, 2007.

[3] OMG-MDA, MDA Guide version 2.0. OMG, 2014.

[4] X. Blanc and O. Salvatori, MDA en action: Ingénierie logicielle guidée par les modèles. Editions Eyrolles, 2011.

[5] M. F. Amr, N. Benmoussa, K. Mansouri, and M. Qbadou, "Transformation of the CIM model into a PIM model according to the MDA approach for application interoperability: Case of the 'covid-19 patient management' business process," International Journal of Online & Biomedical Engineering (iJOE), vol. 17, no. 05, p. 49, 2021. https://doi.org/10.3991/ijoe.v17i05.21419

[6] Y. Rhazali, Y. Hadi, I. Chana, M. Lahmer, and A. Rhattoy, "A model transformation in model driven architecture from business model to web model." IAENG International Journal of Computer Science, vol. 45, no. 1, 2018.
[7] J. Gordijn and J. Akkermans, "Value-based requirements engineering: exploring innovative e-commerce ideas," Requirements Engineering, vol. 8, no. 2, pp. 114–134, 2003. https://doi.org/10.1007/s00766-003-0169-x
[8] J.-M. Alliot, T. Schiex, P. Brisset, F. Garcia, and J.-M. Alliot, Intelligence artificielle et informatique théorique. Cépaduès-éd., 1994.
[9] OMG-UML, Unified Modeling Language version 2.5.1. OMG, 2017.
[10] J. Lind, "Issues in agent-oriented software engineering," in International Workshop on Agent-Oriented Software Engineering. Springer, 2000, pp. 45–58. https://doi.org/10.1007/3-540-44564-1_3
[11] J. Gordijn, H. Akkermans, and J. Van Vliet, "Designing and evaluating e-business models," IEEE intelligent Systems, vol. 16, no. 4, pp. 11–17, 2001. https://doi.org/10.1109/5254.941353
[12] I. Garrigo's, J.-N. Mazon, N. Koch, and M. Escalona, "Web and requirements engineering," IET software, vol. 6, no. 2, pp. 83–84, 2012. https://doi.org/10.1049/iet-sen.2012.0044
[13] F. Jouault, F. Allilaire, J. Bézivin, I. Kurtev, and P. Valduriez, "Atl: a qvt-like transformation language," in Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications. ACM, 2006, pp. 719–720.
[14] OMG-XMI, XML Metadata Interchange version 2.5.1. OMG, 2015. https://doi.org/10.1007/978-1-4899-7993-3_902-2
[15] OMG-OCL, Object Constraint Language version 2.4. OMG, 2014.
[16] OMG-MOF, Meta Object Facility version 2.5. OMG, 2015.
[17] S. Maalal and M. Addou, "A new approach of designing multi-agent systems," arXiv preprintarXiv:1204.1581, 2012. https://doi.org/10.1109/ICMCS.2011.5945725
[18] F. L. Bellifemine, G. Caire, and D. Greenwood, Developing multi-agent systems with JADE. John Wiley & Sons, 2007, vol. 7. https://doi.org/10.1002/9780470058411
[19] G. Taentzer, D. Müller, and T. Mens, "Specifying domain-specific refactoring for AndroMDA based on graph transformation," in International Symposium on Applications of Graph Transformations with Industrial Relevance. Springer, 2007, pp. 104–119. https://doi.org/10.1007/978-3-540-89020-1_9
[20] M. Elammari and Z. Issa, "Using model driven architecture to develop multi-agent systems," Int. Arab J.Inf. Technol, vol. 10, no. 4, 2013.
[21] N. F. Noy, M. Crubézy, R. W. Fergerson, H. Knublauch, S.W. Tu, J. Vendetti, and M. A. Musen, "Protégé-2000: an open-source ontology-development and knowledge-acquisition environment." in AMIA Annual Symposium Proceedings, vol. 2003. American Medical Informatics Association, 2003, pp. 953–953.
[22] D. Gasevic, D. Djuric, V. Devedzic, and V. Damjanovi, "Converting UML to owl ontologies," in Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters. ACM, 2004, pp. 488–489. https://doi.org/10.1145/1013367.1013539
[23] A. Di Marco and S. Pace, "Model-driven approach to agilla agent generation," in 2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC). IEEE, 2013, pp. 1482–1487. https://doi.org/10.1109/IWCMC.2013.6583775
[24] S. Nouzri and A. El Fazziki, "A mapping from BPMN model to JADEX model." International Arab Journal of Information Technology (IAJIT), vol. 12, no. 1, 2015.
[25] Y. Wautelet and M. Kolp, "Business and model-driven development of BDI multi-agent systems," Neuro computing, vol. 182, pp. 304–321, 2016. https://doi.org/10.1016/j.neucom.2015.12.022
[26] J. Gordijn, E. Yu, and B. Van Der Raadt, "E-service design using i* and e/sup 3/value modeling," IEEE software, vol. 23, no. 3, pp. 26–33, 2006. https://doi.org/10.1109/MS.2006.71

[27] A. Fernàndez-Isabel and R. Fuentes-Fernàndez, "Extending a generic traffic model to specific agent platform requirements." Comput. Sci. Inf. Syst., vol. 14, no. 1, pp. 219–237, 2017. https://doi.org/10.2298/CSIS161010001F
[28] M. Kouissi, E. M. En-Naimi, A. Zouhair, and M. Al Achhab, "New approach of designing and developing multi-agent systems," in Proceedings of the 3rd International Conference on Smart City Applications. ACM, 2018, p. 72. https://doi.org/10.1145/3286606.3286849
[29] S. Çam, B. K. Gorur, J. Ledet, H. Oguztuzun, and L. Yilmaz, "Transformation from SysML to repast and back," Procedia Computer Science, vol. 170, pp. 845–850, 2020. https://doi.org/10.1016/j.procs.2020.03.146
[30] Repast, "Repast simphony." [Online]: https://repast.github.io, accessed: 2022-11-11.
[31] M. Hause et al., "The SysML modelling language," in Fifteenth European Systems Engineering Conference, vol. 9, 2006, pp. 1–12.
[32] N. Kharmoum, S. Ziti, Y. Rhazali, and O. Fouzia, "A method of model transformation in mda approach from e3value model to BPMN2 diagrams in CIM level." IAENG International Journal of Computer Science, vol. 46, no. 4, 2019.
[33] N. Kharmoum, S. Retal, S. Ziti, and F. Omary, "A novel automatic transformation method from the business value model to the UML use case diagram," in International Conference on Advanced Intelligent Systems for Sustainable Development. Springer, 2019, pp. 38–50. https://doi.org/10.1007/978-3-030-36671-1_4
[34] N. Kharmoum, S. Ziti, Y. Rhazali, and F. Omary, "An automatic transformation method from the e3value model to ifml model: An MDA approach," Journal of Computer Science, vol. 15, no. 6, pp. 800–813, 2019. https://doi.org/10.3844/jcssp.2019.800.813
[35] A. Rodriguez, I. G.-R. de Guzmàan, E. Fernàandez-Medina, and M. Piattini, "Semi-formal transformation of secure business processes into analysis class and use case models: An MDA approach," Information and Software Technology, vol. 52, no. 9, pp. 945–971, 2010. https://doi.org/10.1016/j.infsof.2010.03.015
[36] D. Steinberg, F. Budinsky, E. Merks, and M. Paternostro, EMF: Eclipse Modeling Framework. Pearson Education, 2008.
[37] Y. Rhazali, Y. Hadi, and A. Mouloudi, "A methodology of model transformation in MDA: from CIM to PIM," International Review on Computers and Software, vol. 10, no. 12, pp. 1186–1201, 2015. https://doi.org/10.15866/irecos.v10i12.8088
[38] M. Hanine, M. Lachgar, S. Elmahfoudi, and O. Boutkhoum, "MDA approach for designing and developing data warehouses: A systematic review & proposal." International Journal of Online & Biomedical Engineering (iJOE), vol. 17, no. 10, 2021. https://doi.org/10.3991/ijoe.v17i10.24667

9 Authors

Nassim Kharmoum, Ph.D. and Assistant Professor. He received his diploma of doctor degree in Computer Science from the Mohammed V University in Rabat. Currently, he is an Assistant Professor in Software Engineering at the National Center for Scientific and Technical Research in Morocco. He proposed different validated methods in scientific articles published in international journals, books, and conferences. His research interest focuses on artificial intelligence, data science, analysis and conceptual modeling, software requirements, multi-agent systems modeling, and modernization of legacy systems.

Sara Retal received her Ph.D. in Computer Science at the faculty of sciences of Rabat and is a full time Professor at the Moroccan School of Engineering Sciences in Rabat. She participated in the ICT Maghreb Technology Platform project co-funded by the European Community and in other research projects in Aalto University. Her research interests include artificial intelligence, data science, and software engineering.

Karim El Bouchti is currently a researcher at department of computer science/ Faculty of Sciences at Mohammed V University. His researches are in fields of computer security, trading financial market, analysis and conceptual modeling, and SQL/ NoSQL database security using encryption. Recently, He works on intelligent systems. He has served as invited reviewer for many indexed journals and conferences. Besides, he is the president and the founder of "Trading Research Club".

Wajih Rhalem, Ph.D. and university professor. He obtained his doctoral degree in Electrical Engineering from Mohammed V University in Rabat. Currently, he is president of the Moroccan Society of Digital Health and professor of electrical engineering at the National School of Arts and Crafts under the University Mohammed V of Rabat. He has proposed different methods validated in scientific articles published in international journals, books and conferences. His research interests focus on artificial intelligence, digital health and bioinformatics. Since 2019, he has held the position of Head of the Scientific Research, Cooperation and Supervision Service at ENSAM-Rabat.

Soumia Ziti is a Franco-Moroccan teacher-researcher lady at the Faculty of Sciences of Mohammed V University in Rabat since 2007. She holds a PhD in computer science in the field of graph theory, and a diploma in advanced studies in fundamental computer science, both obtained at the University of Orleans in France. His areas of expertise and research are graph theory, information systems, artificial intelligence, data science, software development engineering, modeling of relational databases and big data, cryptography and numerical methods and simulations in spintronics. She has over than sixty publications in high-level international journals and conferences in several research areas. In addition, she coordinates or participates in several educational or socio-economic projects.

Article submitted 2022-11-25. Resubmitted 2022-12-29. Final acceptance 2023-01-17. Final version published as submitted by the authors.