

Automatic Agile Transformation with Approach by Modeling from BPMN Collaboration Diagram Model to UML Use Case

Ouzayr RABHI

MATSI Laboratory, EST
Mohammed First University, Oujda, Morocco
o.rabhi@ump.ac.ma

Ibtissam ARRASSEN

Laboratory for Computer Science Research Faculty of Sciences
Mohammed First University, Oujda, Morocco
arrassen@yahoo.com

Mohammed ERRAMDANI

Department of Management, EST
Mohammed First University, Oujda, Morocco
mramdani69@yahoo.co.uk

Abstract

In this paper we propose an automatic and agile transformation method from the CIM level to the PIM level respecting the MDA approach and agile transformation rules. The proposal mainly focuses on the creation of a CIM Level from the defined rules that allow the realization of rich models containing all the important information to facilitate the transformation to the PIM level. Then, we propose to define the agile transformation rules from the CIM level to the PIM level, through passage rules and agile transformation rules, This method respects the MDA approach, by creating the business dimension at the CIM level through the use of BPMN via the collaboration and business process diagrams, defining the passage rules and agile transformation rules, to finally obtain a PIM recommended by MDA using UML use case and class diagrams.

Keywords

Agile transformation in MDA, CIM, PIM, BPMN, Scrum method

1. Introduction

MDE (Model Driven Engineering) [1] is an alternative approach to the development of information systems based on the creation of initial models and their transformation into various levels of abstraction up to automatic code generation . Its objective is to automate the software development process, which is performed manually by MDE experts can be considered as a general family of approaches, the most interesting and commonly represented variant being Model Driven Architecture (MDA) [2], supported by GMO. The MDA has the same principles as the MDE, but provides its own features defined by three levels of abstraction, defining certain requirements to be met and recommending the use of certain standards.

The first level of MDA is the CIM (Computation Independent Model), which is represented as a model used by managers and business analysts to describe business processes. The second level is PIM (Platform Independent Model), which defines the model used by software analysts and designers to perform independent analysis and design of software under development. The third level is the PSM (Platform-Specific Model), which is considered a code model used by software developers. These models contain all the necessary information for the runtime platform and are intended for use by software developers. The code is not an MDA model, but the end result of the MDA process.

Transformations between different MDA levels start with a CIM to PIM transformation, which aims to partially build a PIM model from the CIM model; the information that existed in the CIM model is rewritten to the PIM model; the PIM model is then converted to the MDA model. These transformations ensure that business information is transferred and respected throughout the MDA process. Secondly, the conversion from the PIM model to the MDA model adds PIM technical information associated with the target platform.

In practice, the automatic transformation starts at the second level of the MDA. However, the ultimate goal is to make the CIM layer productive, the basis for building the PIM layer through automatic processing, so that the business model is no longer just a document for communication between business professionals and software developers.

This document presents an approach to BPMN notation modelling at the CIM level, based on the BPMN notation for business process modelling at the CIM level (Figure 2, Figure 3, Figure 4, Figure 5). First, we identify the participants using BPMN coordination diagrams, represent the business processes as a set of sub-processes, and describe the overall structure of the business processes.

Based on the transformation rules, the actors and business sub-processes are transformed from the CIM level model to the utilization diagram model, which is a PIM level model that respects the flexible methodology in our approach. The rest of the work is organised as follows. Section II provides an overview of the concept of flexible transition from MCP to MIP. Section III introduces our approach and describes the model building rules at the MCP level, MCP-to-MCP conversion rules, and rules for agile conversion. In Section IV, we illustrate our proposal with concrete examples showing the construction of the MCP level, the transition to MCP level, and various sprinklers. Finally, Section V concludes by defining the results of this work and describing future work.

2. OVERVIEW OF CONCEPTS

2.1 Model Driven Architecture (MDA)

At the end of 2000, OMG, a consortium of more than 1,000 companies, first read the Model Driven Architecture document and decided to form a team of architects to develop a more formal explanation of MDA [3]. This approach focuses on the development of top-level abstraction models and promotes a model-to-model approach.

MDA addresses the challenges of today's highly interconnected and evolving systems by providing an architecture that ensures portability, cross-platform interoperability, platform independence, domain specificity and productivity [3]. Key to the MDA approach is the importance of patterns in the software development process. In MDA, the software development process is driven by the modelling activities of the software system.

In the field of software engineering, the OMG's MDA approach classifies the software design models it represents into four types: computational

Independent Model (CIM), Platform Independent Model (PIM), Platform Specific Model (PSM) and Code; defined as follows :

- **CIM:** The goal is to create a requirements model for the future application. Such a model must represent the application in its environment in order to define the services offered by the application and which other entities with which it interacts.
- **PIM:** This model represents the system-specific business logic or the design model. It represents the functioning of entities and services. It must be durable and lasting over time. It describes the system, but does not show the details of its use on the platform.
- **PSM:** MDA considers that the code of an application can be easily obtained from these models. The main difference between a code model and an analysis or design model is that the code model is linked to an execution platform.
- The reason for organising the above model is to develop models of the system's business logic independently of the execution platforms and then automatically transform these models into platform-dependent models. The complexity of the platforms no longer appears in the business logic models, but is in the transformation [11].

The steps required in model-driven development using the UML approach can be basically divided into the following steps [12], at the beginning of building the CCM that captures the user's requirements. A CCM is then built in accordance with this CCM. This is followed by the conversion of the proposed CCM into one or more other CCMs. This type of transition from CCM to CCM and from CCM to SMP is called model-to-model (M2M) transformation. The last step is to transform the model generated into the platform code chosen in consideration of

the SMP. This transition is called Model-to-Text (M2T) transformation. Figure 1 shows how the transformations are performed.

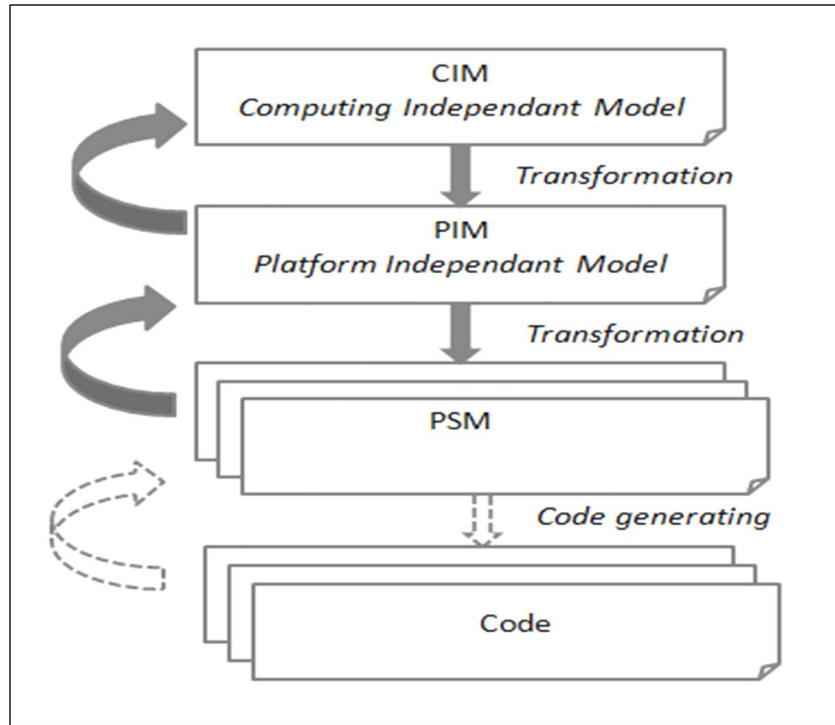


Figure. 1. Model Driven Architecture levels [SEP]

2.2 Business Process Management Notation (BPMN)

According to the OMG, BPMN is a specialized standard for business process modelling, all the benefits of most business process standards converge towards BPMN (OMG-BPMN, 2011). BPMN's main objectives are to standardize business process modeling, expand modeling resources and create a formal mapping between high-level modeling and execution languages. BPMN create a simple mechanism to develop business process models while also ensuring the inherent complexity of the processes. This notation was developed by the Business Process Management Initiative (BPMI) in May 2004, and in June 2005, BPMI merged with the Object Management Group (OMG). In February 2006, OMG adopted and officially published version 1.0.

One of the advantages of BPMN is the reuse of code, because according to the author, when a company builds different component models representing a specific implementation, these can be stored in its model libraries which can also be easily used in the future by one or more similar applications while importing the model code.

BPMN provides several basic diagrams to represent different business processes in a simple way, and at the same time are able to control the inherent complexity of business processes. The basic set can be divided into four main categories: Connection Objects (Fig. 2); Flow Objects (Fig. 3); Navigation Paths (Fig. 4); and Artifacts (Fig. 5).


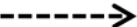

Element	Description	Icon
Sequence Flow	Used to show the order (sequence) in which activities will be carried out in a process.	
Message Flow	Used to show the message flow between two different participants who send and receive them.	
Association	Used to associate data, text, and other artifacts with flow objects. Associations are used to show inputs and outputs of activities.	

Figure 2. Connection Objects (source: Adapted from BPMI)




Element	Description	Icon
Event	Something that happens during a business process. These events affect the process flow and generally have a cause (trigger) or an impact (result). There are three types of events, based on how they affect flow: Start, Intermediate, and End.	
Activity	This is a generic term for a job done. Activity types are: Tasks and sub-processes. A sub-process is distinguished by a small cross at the bottom center of the figure.	
Gateway	This is used to control the divergence and convergence of a flow's sequence. It will thus determine traditional decisions like joining or dividing routes.	

Figure 3. Flow Objects (source: Adapted from BPMI)


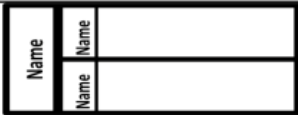
Element	Description	Icon
Pool	A pool represents a participant in a process. It acts as a graphic container to divide a set of activities from other pools, generally in the context of Business to Business situations.	
Lane	A lane is a subdivision in a pool used to organize and categorize activities.	

Figure 4. Swimlanes (source: Adapted from BPMI)

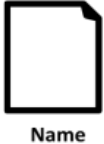

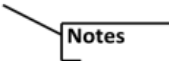
Element	Description	Icon
Data Objects	A data object is a mechanism to show how data is requested or produced by activities. They are connected to activities with the associations.	
Group	A group is represented by a rectangle and it can be used for documentation or analysis purposes.	
Annotations	Annotations are mechanisms for providing additional information to the reader of a BPMN diagram	

Figure 5. Artifacts Objects (source: Adapted from BPMI)

2.3 Agile Method

The agile method, which was founded in 2002, is based on dividing the project into iterations, also called "sprints". A sprint can have a duration that generally varies between two weeks and a month.

The estimation of the task in terms of time and complexity is done before each sprint using several methods to plan the deliveries and also to estimate the cost of each task for the customer.

Functionalities called "user stories" are the subject of the sprint and form a "sprint delay" which can be a deliverable at the end of the sprint.

There is a difference between the "product delay" of the sprint and the "product delay" of the sprint, which is the amount of functionality expected for the product in all sprints.

The Agile method is also characterized by a daily "scrum", called "morning" or "standing", in which employees (project managers, developers and function managers) take turns indicating the tasks they performed the day before, the difficulties they encountered and finally what they will continue to work on the next day. This makes it possible to assess the progress of the project and the resources where they are most needed, but also to provide support to employees who are encountering difficulties, if they have already met other team members.

2.3.1 Agility principles

Already applied in several information system and software development projects, the principles of agility help to solve some of the process modelling problems in projects. The agile principles were originally developed in response to the dissatisfaction with traditional development approaches. The agile manifest exposes the agile methods:

- Collaborating with clients during contract negotiations,
- Prefer a working software rather than a complete documentation,
- Reacting to different changes by following a plan
- Interactions and individuals on tools and processes.
- The Twelve Agile Principles (AP) which describe the agile philosophy in detail :
 - (PA1) Number one priority is customer satisfaction through valuable software delivery
 - (PA2) Accept all changes in customer needs, even if they are late. Adopt Agile Processes to apply change for the customer's competitive advantage.
 - (PA3) Delivery of functional software often within weeks to months, referring to a short period of time.
 - (PA4) Clients, designers and developers must work together daily throughout the project period.
 - (PA5) Projects need to be built around highly motivated people to give them the environment and support they need, and also trust them to get the job done.
 - (PA6) The best method of getting information to and from a development team is simply a face-to-face conversation.
 - (PA7) The main measure of progress is the working software,
 - (PA8) An agile process always promotes sustainable development. Sponsors, users and developers together must be able to maintain a constant rhythm indefinitely.
 - (PA9) Continuous attention to technical excellence and good design improves agility.
 - (PA10) The art of maximising the amount of work is Simplicity.
 - (PA11) The best architectures, requirements and designs arise from self-organisation, the best architectures, requirements and designs are born.
 - (PA12) Each team thinks of a way to become more efficient at regular intervals, then agrees on a way to adjust its behaviour.

While the agile paradigm has influenced the software engineering industry over the past 15 years, its impact on BPM has not been as strong or as direct. With the proliferation of agile approaches, in particular Scrum, the emphasis has shifted somewhat. Agile", which is an application of the Scrum approach, has been adopted not only in

project management but also in product development and management. In recent years, the term "agile" has become increasingly important in the BPM, often in the context of social media and electronic commerce.

The agile approach encompasses all the characteristics of agility, in addition to the attributes of responsiveness, proactivity and a positive attitude to changes in the environment. The definition of agile defines the term agility, which is the sum of flexibility and responsiveness. From the results of the literature on agile problems and principles in BPM we derive which problems in BPM projects can be solved by which agile principles.

3 TRANSFORMATION CIM TO PIM USING AGILE METHOD

A business model is the abstraction of the way the company operates. Depending on our modeling objectives, we can, more or less, create different models to describe the same reality. We can also model in order to control business processes, to improve the communication process with customers or partners, based on agility, or to create an information system. In this document, our objective is to design business process models as a first step in the development process of an information system. Our proposal takes into account the business dimension at the CIM level by using real high-level business models to preserve the business knowledge during the transformation at the PIM level in order to obtain a quality information system.

In this approach, we use the BPMN Collaboration Diagram and Process Diagram, the standard business process modeling, to leverage each diagram to achieve a rich and concentrated CIM level, which simplifies our transformation to the PIM level. MDA recommends the use of UML at the PIM level, we present this level with a model use case diagram that shows the functional viewpoint. The PIM level model will go through an automatic transformation of the CIM level, via well-defined and concentrated transformation rules. Below, we present the construction rules of the CIM level, the transformation rules at the PIM level and the agile transformations.

To combine MDA and the agile method of Scrum, we can use in each project sprint the principles of MDA, i.e. in each sprint we apply our approach of generating source code from the business needs, in the scrum we have a sprint backlog that describes the characteristics of the system, the combination can be described as follows :

- **Sprint 1:** Initially, the transformations are done at the use case level only.
- **Sprint 2:** After transforming the use cases from the CIM level "tasks", the "sequence flows" will be transformed into an "inclusion" relationship.
- **Sprint 3:** At the end the "decision nodes" will be transformed into an "extended" relationship.

3.1 Rules of construction CIM level using the Model of Collaboration Diagram BPMN

The rules for building the BPMN collaboration diagram template:

- R1: Define sub-processes of average sizes.
- R2: Each sub-process must consist of 5 to 8 tasks.
- R3: Avoid the representation of tasks, provided you only show the manual tasks.
- R4: The model should not represent all possible cases, ie do not use the gateways in this model,
- R5: Based solely on the presentation of the sub-processes and their relationships.
- R6: Use the "group" notation to group the sub-processes that belong to the same category

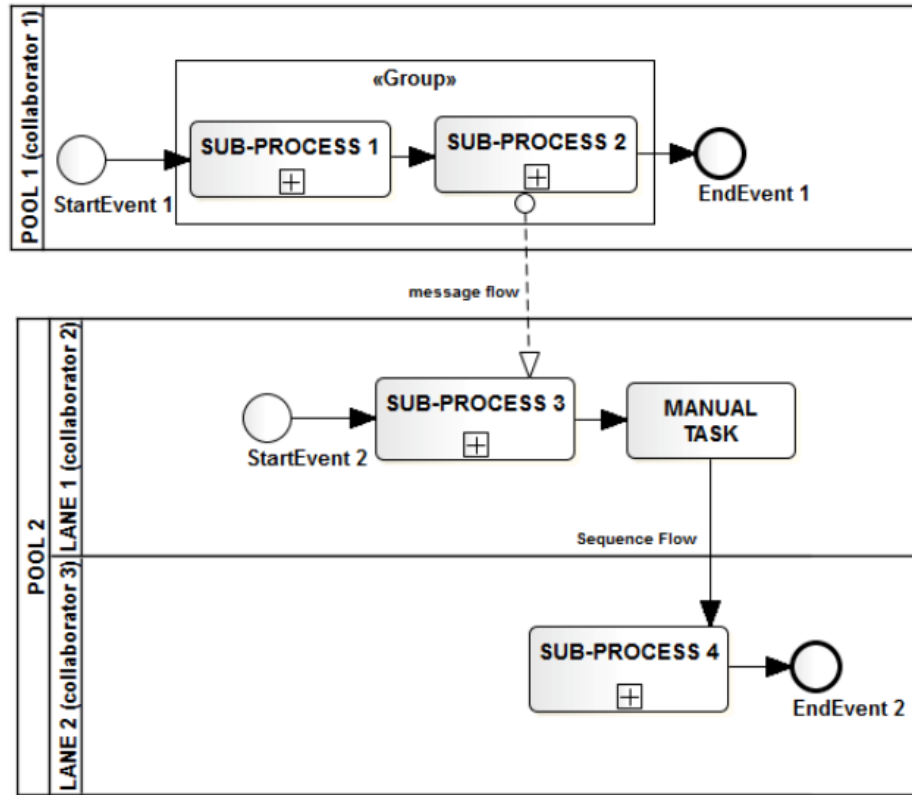


Fig 6. Generic Model of the BPMN Collaboration Diagram

3.2 The construction rules of the BPMN business process map model (Figure 6)

- R7: Detail each sub-process individually into several tasks (the task is the fundamental unit in the BPMN business process diagram).
- R8 : Avoid manual tasks.
- R9: Represent the gateways in this model.
- R10: Show the most exceptional paths, this time using gatways.
- R11: Add a data object containing the status of the object in the output of each task.

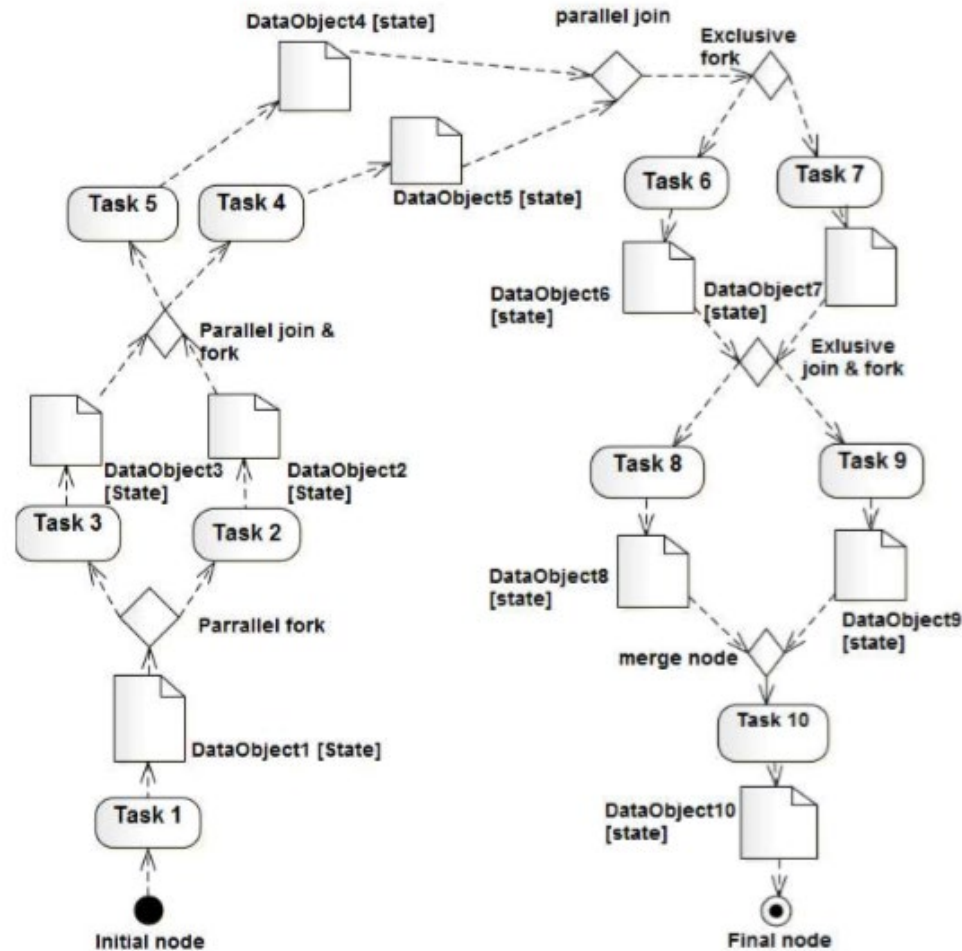


Figure 7. Generic BPMN business process diagram model

3.3 Transformation rules from CIM level to PIM level Models

Transformation rules from BPMN models to the use case model of UML:

- TR1: Each "task" corresponds to a feature of the system is transformed into a "use case".
- TR2: Each "decision node" between two "tasks" becomes an "extend" relationship between two "use cases".
- TR3: Each "collaborator" becomes an "actor".
- TR4: Each "sequence flow" between two "tasks" becomes an "include" relationship between two "use case".
- TR5: Each "sub-process" is transformed into a "package".

3.4 Agile Transformation

Agile Transformation rules from BPMN models to the use case model of UML:

- ATR1: The "task" transformed into "use case" corresponds to sprint 1.
- ATR2: The "sequence flows" between two "tasks" transformed into an "inclusion" relationship between two "use cases" correspond to sprint 2.

- ATR3: The "decision nodes" transformed into an "extended" relationship between two "use cases" correspond to sprint 3.

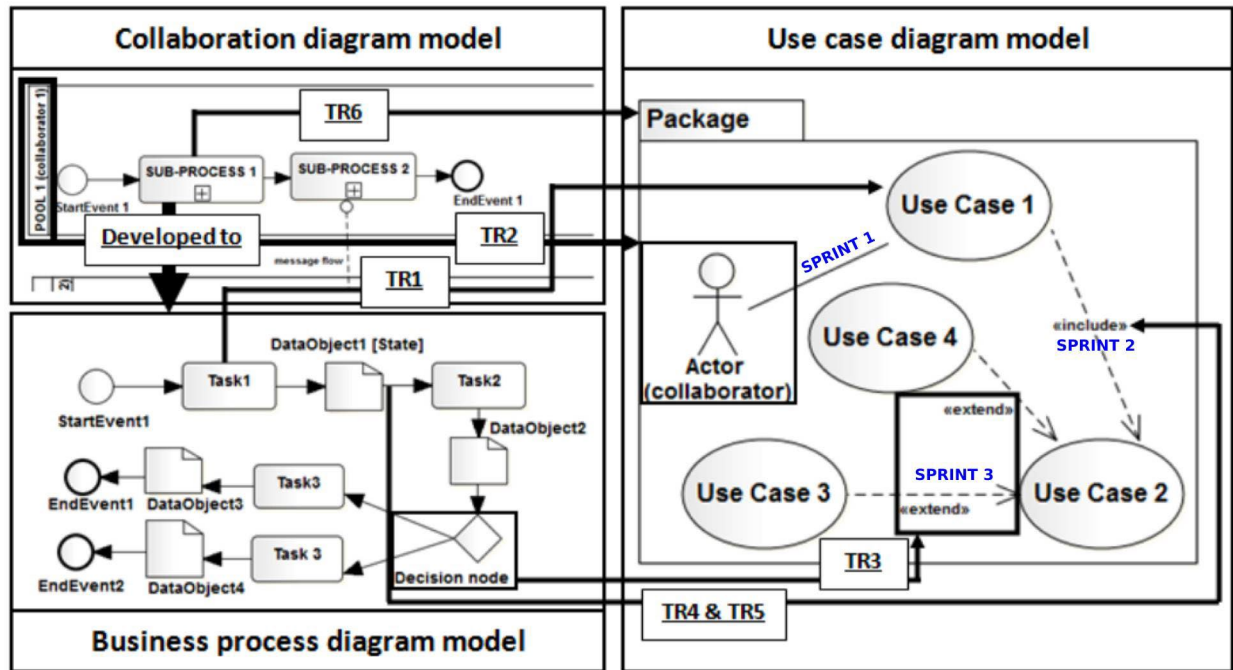


Fig 8: Transformation Rules from BPMN Models to Case Model of use of UML

4 CASE STUDY

In this part of our article, we present a special case study for sales via an e-commerce site to showcase an agile and automatic approach to transforming from the upper CIM level to the lower PIM level.

On the e-commerce site 5 users are identified: the customer, the order agent, the assembly worker, the assembly team leader and the delivery person.

- **User 1:** The customer
 - Behavior 1: A customer can browse the list of products available on the E-Commerce site.
 - Behavior 2: The customer can consult the detailed information on each product, then he has two possibilities, either he decides to put the quantity of product he wants in the cart or he does nothing.
 - Behavior 3: At any time, the customer can change the quantity he has chosen or even remove the product from the cart.
 - Behavior 4: If the products dirty the needs of the customer are, he can launch an order, afterwards he must represent this credit card information, and the address of the delivery.
- **User 2:** The order agent
 - Behavior 1: Once he receives the order. the order agent processes this order, and reserves the products ordered by the customer.
- **User 3:** The assembly worker
 - Behavior 1: Following the reservation of the order agent, the assembly worker retrieves all items ordered by the customer, manually, from the stock.
- **User 4:** The head of the assembly crew

- Behavior 1: Once the products are recovered by the assembly worker, the assembly team leader checks the products for quantity and quality.
- **User 5:** The delivery man
- Behavior 1: At the end of the process, the company's delivery man delivers the order confirmed by the assembly team leader to the customer who ordered the products.

4.1 Presentation of the CIM Level

a. Collaboration Diagram

The BPMN collaboration diagram, which represents the business process model in Figure 9. In this soft model we have tried to present as many collaborators as possible in order to define an ecommerce site process, where there is collaboration between several actors. However, we avoided identifying the gateways and tasks by simply specifying the sub-processes and their sequence in order to present the business process in general. Also, for the transition to the use case, the collaborators will be transformed into actors. However, we have presented the sub-processes of the supports. Thus, a customer would normally perform the activity "select products", then "launch order" and later the activity "present information", but since the "launch order" cannot contain more than three tasks, we merged with "select products" into one sub-process called "select products for order". At the end, we specify all manual tasks. Of course, we can make several improvements to the initial model in order to obtain a model that respects the rules we have defined.

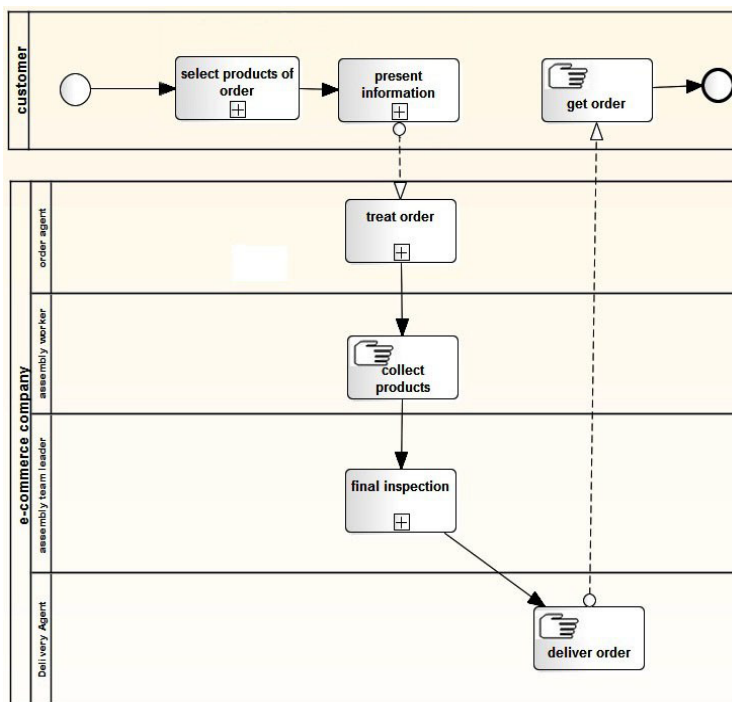


Figure. 9 Collaboration diagram model of “sales through e-commerce”

b. Business Process Flow

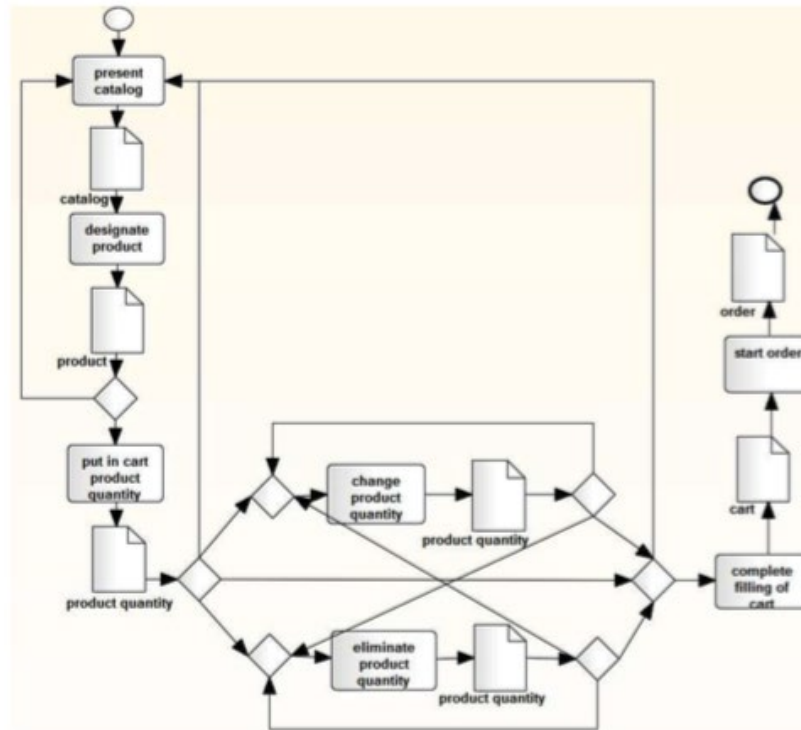


Figure 10. BPMN Business Process Flow Diagram Template of "selecting products from an order".

4.2 Presentation of the PIM Level

The use case model shown in Fig. 11 is transformed from the higher CIM level business models while respecting the sprints shown in Fig. 12.

- The sub-process "select a product for an order" of the collaboration diagram is transformed into a set.
- The "client" collaborator who executes the sub-processes in the BPMN collaboration diagram becomes an actor in the use cases.
- Each task detailing a sub-process in the collaboration diagram is transformed when used and presents the first sprint (Sprint 1).
- The sequence flow between two actions becomes the "include" relationship, which corresponds to the second sprint (Sprint 2). In our model, there are sequence flows that link two actions that are "present the catalog" and also "designate the product", and these two use cases are linked by an "include" relation.
- The "extended" relationships are the results of the exclusive gateways that link two tasks, which in our case correspond to the third sprint (Sprint 3).

In our model, there is an exclusive gateway linking two actions "designate product" and also "put in the cart the quantity of the product", so the two corresponding use cases connect via an "extended" relationship.

Because the use case focuses only on the identification of characteristics and in order not to complicate the model too much, in the latter there is no sequential flow that goes in reverse.

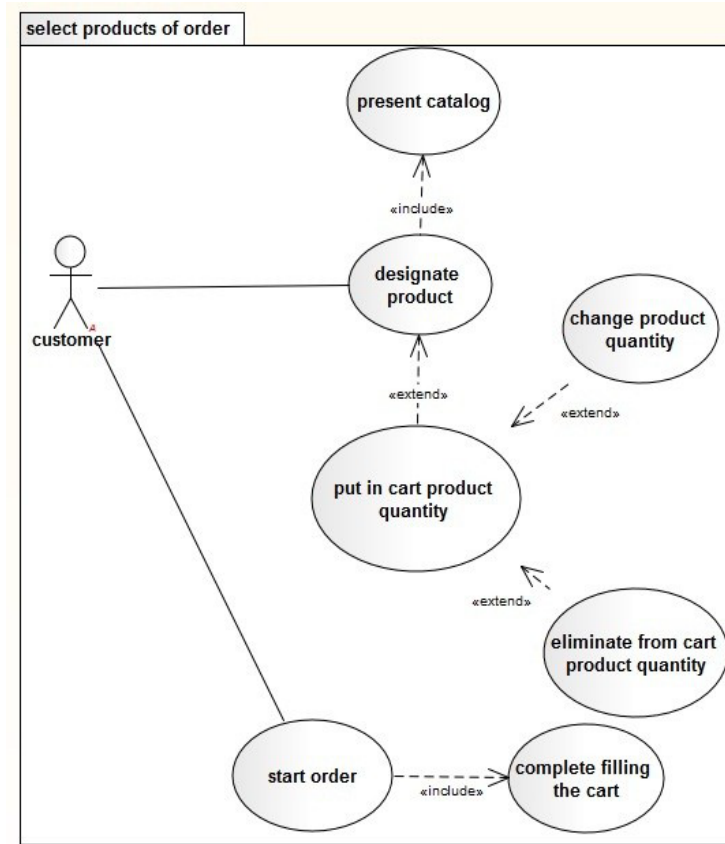


Figure. 11 Use case diagram Model of “choose products for order”

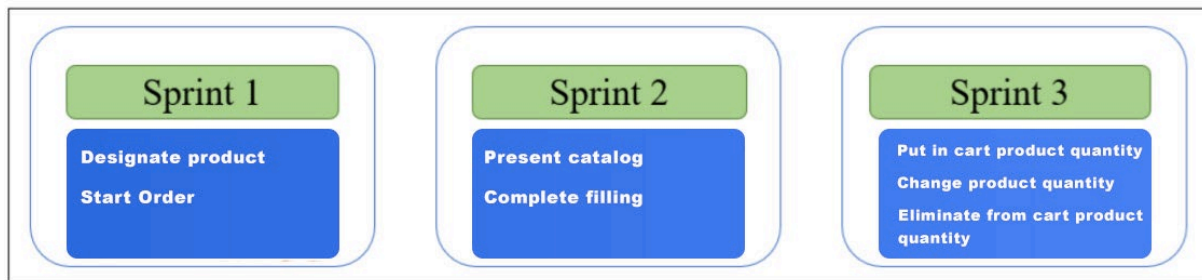


Figure. 12: Scrum RoadMap of sales through e-commerce [11]

5 Conclusion and future work

For software development, one of the main challenges is to define an approach that allows the transition from models that describe the functioning of a company to models that present the analysis and design of the software while remaining in permanent contact with the client to exchange and accept change requests at several stages of the process.

The objective of our article is to present an approach based on MDA models, and to provide a solution to the different transformation problems from the CIM analysis and design level presented in our article by the collaboration diagram, to the PIM modelled models level presented in our article by the use case diagram using an agile method presented by the different sprints in order not to burden the transformation process.

For our future work, we will work on the agile transformation from the PIM level to the PSM level, in order to generate the software source code in the end.

6 References

1. J. Miller, J. Mukerji, MDA Guide Version 1.0.1. Document No. omg/2003-06-01, 2003.
2. OMG, UML Superstructure 2.0. OMG Adopted Specification ptc/03-08-02, 2003.
3. C. Schmidt, Cover Feature Model Driven Engineering, 2006.
4. J. Miller, J. Mukerji, MDA Guide Version 1.0.1. Document No. omg/2003-06-01, 2003.
5. OMG, Business Process Modelling Notation, Version 2.0, 2011. <<http://www.omg.org/spec/BPMN/2.0/pdf>>.
6. C. Schmidt, Cover Feature Model Driven Engineering, 2006.
7. J. Gordijn, J.M. Akkermans, Value based requirements engineering:exploring innovative e-commerce idea, Requirements Engineering Journal 8 (2) (2003) 114–134.
8. V.D. Castro, E. Marcos, J.M. Vara, Applying CIM-to-PIM model transformations for the service-oriented development of information systems: Information and Software Technology 53 (2011) 87–105.
9. Amine Azzaoui, Ouzayr Rabhi, Ayyoub Mani, A Model Driven Architecture Approach to Generate Multidimensional Schemas of Data Warehouses (iJOE – Vol. 15, No. 12, 2019) <https://doi.org/10.3991/ijoe.v15i12.10720>
10. A. Rodríguez, I. García-Rodríguez de Guzmán, E. Fernández Medina, M. Piattini, Semi-formal transformation of secure business processes into analysis class and use case models: an MDA approach, Information and Software Technology 52 (9) (2010) 945–971.
11. S. Kherraf, E. Lefebvre, W. Suryn, Transformation From CIM to PIM Using Patterns and Archetypes : 19th Australian Conference on Software Engineering (2008) 338-346.
12. C. Hahn, P. Dmytro, K. Fischer, A model-driven approach to close the gap between business requirements and agent-based execution, in: Proceedings of the 4th Workshop on Agent-based Technologies and applications for enterprise interoperability, Toronto, Canada, 2010, pp. 13–24.
13. W. Zhang, H. Mei, H. Zhao, and J. Yang, "Transformation from CIM to PIM: A Feature-Oriented Component-Based Approach," presented at MoDELS 2005, Montego Bay, Jamaica, 2005.
14. B. Grammel, S. Kastenholz, A generic traceability framework for facetbased traceability data extraction in model-driven software development, in: Proceedings of the 6th ECMFA Traceability Workshop held in conjunction ECMFA 2010, Paris, France, 2010, pp. 7–14.
15. OMG, MOF 2.0 Query/View/Transformation (QVT), V1.0. OMG Document – formal/08-04-03, 2008.<<http://www.omg.org/spec/QVT/1.0/>>.
16. J.J. Gutiérrez, C. Nebut, M.J. Escalona, M. Mejías, I.M. Ramos, Visualization of use cases through automatically generated activity diagrams, in: 11th international conference on Model Driven Engineering Languages and Systems, 2008.
17. J. Mazón, J. Pardillo, J. Trujillo, A model-driven goal-oriented requirement engineering approach for data warehouses, in: Proceedings of the Conference on Advances in Conceptual Modeling: Foundations and Applications, ER Workshops, Auckland, New Zealand, 2007, pp. 255–264.
18. OMG, Service Oriented Architecture Modeling Language (SoaML) – Specification for the UML Profile and Metamodel for Services (UPMS).OMG document: ad/2008-08-04, 2009. <<http://www.omg.org/docs/ad /08-08-04.pdf>>. (Revised Submission).
19. M. Kardoš, M. Drozdová, Analytical Method of CIM to PIM Transformation in Model Driven Architecture (MDA) : JIOS, VOL. 34, NO. 1 (2010), PP. 89-99.