E-ISSN 2281-4612
ISSN 2281-3993

Academic Journal of Interdisciplinary Studies
MCSER Publishing, Rome-Italy

Vol 3 No 6
November 2014

Bizagi Process Management Suite as an Application of the Model Driven Architecture Approach for Developing Information Systems

Oskeol Gjoni

PHD Student at "European University of Tirana", Tirana, Albania

Doi:10.5901/ajis.2014.v3n6p475

Abstract

Bizagi business process management is a suite for software development and process automation that implements the Model Driven Architecture (MDA) principles. The capabilities of the suite will be presented through a real case information system developed that represents a business process in a modern corporation; the user access management flow that employees follow in order to gain access into systems. The MDA approach to develop information systems is considered as the biggest shift since the move from Assembler to the first high level programming languages. At the heart of the MDA approach are models which represent the information system developed while the implementation in a certain technology is obtained through model transformation. Following the MDA paradigm, in Bizagi the starting point is the model of the business process. The model is enriched throughout the steps of process automation in order to produce an information system which can be immediately executed. As result an information system which automates the business process is obtained without the need to write any code. Bizagi business process management suite fully implements the MDA approach and principles in building process applications.

Keywords: *Bizagi, BPMN, model driven architecture, information systems, modeling.*

1. Introduction

The development of information systems has always been and still is a complex task. The bad news is that complexity continues to increase. This due to the fact that customer and business needs are more and more sophisticated. At the same time the resulting information systems are developed and implemented using highly dynamic, constantly changing technologies [20]. In order to survive in a world always more competitive, more difficult to predict, more connected and challenging, business environment changes dynamically. Regulatory compliance, mergers and acquisitions, joint ventures, outsourcing activities just to name a few, impose changes in the business model and therefore on the information system. At the same time the progress and introduction of new technologies impose changes in the business environment.

To create business information systems today is costly (highly paid resources over extended periods of time), too slow for modern business conditions and highly risky (difficult to control and high failure rates) [20]. The modern development process of information systems has not changed much over the past years. It is highly focused on the codification (the "how") rather than modeling (the "what") which affects in a negative way the productivity and the quality of the final product. Most of the efforts are spend in absorbing the technological complexity of a particular implementation rather than concentrating on understanding the business process for which the information system is being build [19, 12, 10].Programing is still the most important task in creating a software product. It is true that programming technologies have improved with the passing of years, but the problem is that if we consider the bad results programming-oriented technologies have achieved historically is logical to ask the following: is reasonable to search for better methods of creating information systems? It would be desirable to have a new approach in developing information systems that separates business logic from the implementation technology, which takes the software development process to a higher level of abstraction using expressions that are more close to the business logic.

In order to address the above concerns the Object Management Group (OMG) has defined a new framework for developing information systems, the Model Driven Architecture (MDA) [15, 18]. As per OMG definition of MDA, "*MDA separates business and application logic from underlying platform technology... No longer tied to each other, the business and technical aspects of an application can each evolve at its own pace – business logic responding to business need, and technology taking advantage of new developments – as the business requires*" [18]. At the center of the MDA approach are models; the process of software development is driven by constructing models that represent the software under development, creating software means creating the model. The specific code that represents the implementation of the model in a certain underlying technology is obtained using model transformation capabilities [19].

The MDA approach in developing information systems is a model-oriented approach since if focusses on the business logic (the "what") rather than on the specific implementation technology (the "how") on a particular programming environment such as Java, .NET etc [19, 11, 12]. This separation enables also portability, interoperability and reusability [17]. MDA makes models the authentic protagonist of information systems development, not the source code. This separation decreases the impact that technology evolution has on application development and also enables the possibility to profit from new implementation technologies. Once the model is created, it can be transformed automatically into code in different software platforms. Knowledge and intellectual property engaged in application development are moved from source code to models. Models are the most valuable assets since code is obtained from them through automatic transformations [20].

MDA specifies information systems from three viewpoints: computation independent, platform independent and platform specific. The computation independent viewpoint focusses on the environment of the system and its requirements (expresses the business model) without considering details of the structure and processing of the system [17]. Efforts to represent the business model independently of the computing platform are referred to as Computation Independent Model (CIM) [13, 19]. A CIM is a high level model which describes the main requirements of the business model. In general CIM's are designed by analysts and typically are independent on how the system is implemented. Parts of CIM may be supported by software, but in general CIM is software independent [17, 19].

The platform independent viewpoint focusses on the operation, functional capabilities of a system while hiding and not considering the details necessary for implementation in a particular platform. The resulting model is referred as PIM (Platform Independent Model) [18, 19]. PIM's describe the information system expressing the relationships between the concepts of the domain in question without making references to any specific computing platform [13, 17, 19].

The platform specific viewpoint enriches a platform independent viewpoint with details of a specific computing platform that will be used for the implementation. Such a model that contains details of the underlying implementation technology is referred to as Platform Specific Model (PSM) [15, 17, 18]. The main idea in the MDA approach is to start first developing a PIM which represents the main concepts and their relationships of the domain under study. The finalized PIM can be transformed into different PSM based on the underlying implementation technology of the chosen platform. Model transformation is the process of converting one model into another model for the same information system. A set of unambiguous specifications, the transformation rules, guide the transformation process with the goal to obtain a target model from an initial model [19].

Currently, there are different MDA solutions that are based on the same OMG guidelines. Some of the most known implementations both proprietary and open source solutions are; Abstract Solutions (http://www.abstractsolutions.co.uk/), AndroMDA (http://www.andromda.org), Virtual Enterprise (http://www.intelliun.com), Oliva Nova [20] etc. Some of the biggest companies like Lockheed Martin, Credit Suisse, Austrian Railways etc, have obtained success stories while using the MDA approach [14]. We will focus our attention in this study on the Bizagi business process management suite (http://www.bizagi.com).

2. Bizagi

Bizagi business process management suite is comprised of three main tools: Bizagi Modeler, Bizagi Studio and Bizagi Engine that provide the capability to manage the complete lifecycle of business processes from modeling, execution and improvement [2].

Bizagi Modeler: enables business experts to design, document, execute and evolve their process models. It supports intuitive drag and drop functionality, code-free updates and automatic document generation The modeling notation that the Bizagi process modeler supports is BPMN (Business Process Modeling and Notation) [16].

Bizagi Studio: Bizagi implements the MDA principles or the "modeling over programming" philosophy. Bizagi studio provides to business experts everything they need to transform process models into real, running applications and workflows, from defining the data model and User Interface to integrating IT assets and everything in between [2].

Bizagi Engine: Bizagi engine executes and controls the business processes automated by Bizagi Studio. It supports deployment in JEE or .Net and provides a set of useful performance KPIs for process improvement [2].

Business Process Modeling and Notation (BPMN) is a standard for business process modeling that provides a graphical notation for specifying business processes based on a flowcharting technique. The primary goal of BPMN is to support business process management, for both technical users and business users, by providing a notation that is intuitive to business users, yet able to represent complex process semantics [21].

3. Modeling the User Access Management flow

The business process that is taken into consideration is the classical User Access Management (UAM) flow; the process that needs to be followed in order for employees to obtain or revoke access rights in specific applications (CRM, ERP etc) within the organization. As per the MDA principles to create an information system means to create the specific model that represents the domain under study. In Bizagi process management suite the first step is to create the BPMN model using the process modeler [9, 1].

The BPMN model created for the User Access Management flow is represented in Figure 1, containing all the stakeholders involved in the process with the respective tasks/activities that each should perform. The start and end event are also easily distinguished [16, 21]. The model is straightforward. The process starts when an employee enters a permission request for a particular system within the company. After the request has been completed from the employee it goes for approval to the respective line manager. If the line manager rejects the request, the employee is informed for the rejection reason and the process ends. On the other hand if the line manager approves the request flows to the Corporate Security Manager, which is a central entity within the company responsible for the security of the systems and which approves all permission requests.

The Corporate Security manager also makes a choice. In case the request is rejected the employee is notified and the process ends. Only if the request is approved also from the Corporate Security manager it goes for implementation to the respective implementation team within the company. After the request is implemented the Corporate Security analyst is informed and the process ends. The employee is being granted with access in the system asked.

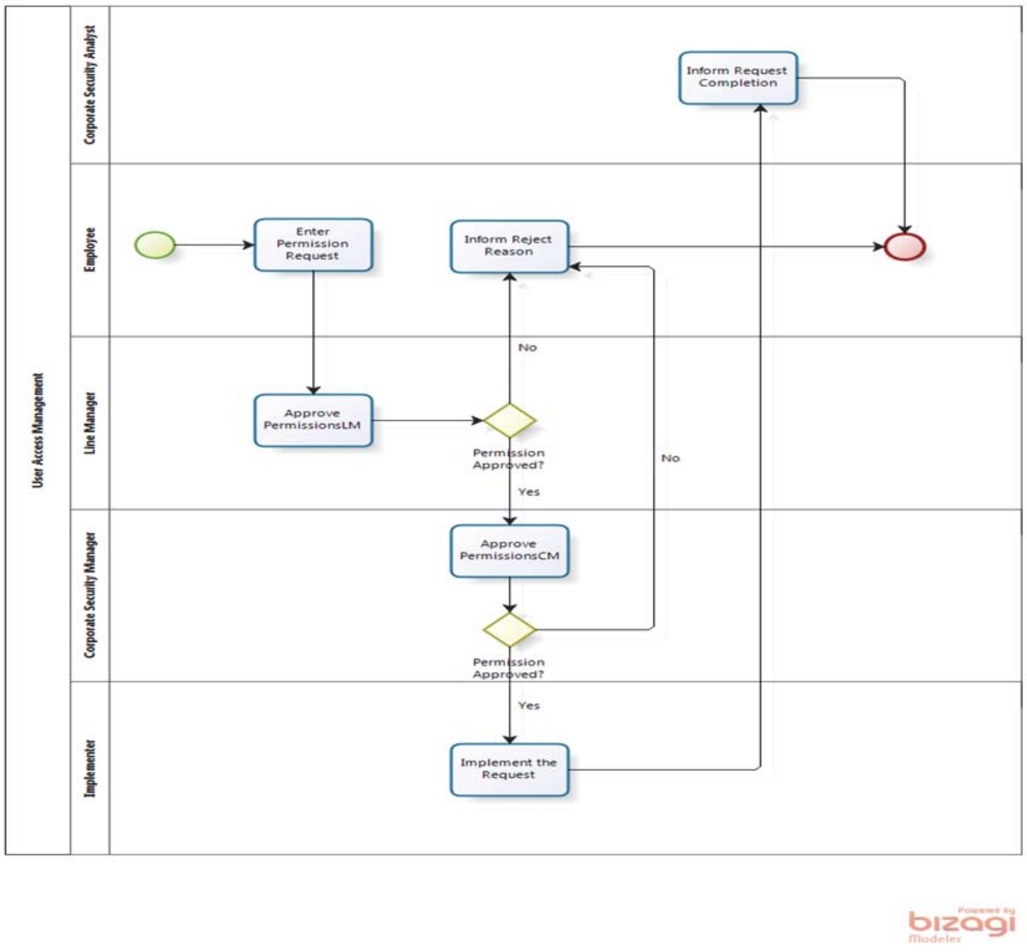


Figure 1: BPMN model of the User Access Management flow.

4. Automating the User Access Management flow

Once the BPMN model is completed it is possible to create automatically the running application by following the process automation steps in Bizagi Studio [1]. The process automation wizard in Bizagi Studio is comprised of several steps that are necessary to enrich and automate the BPMN process produced from the process modeler as shown in Figure 2.

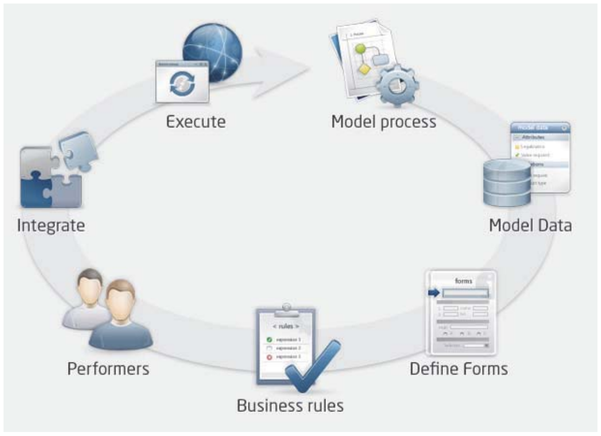


Figure 2: Bizagi Studio process automation wizard.

4.1 Data Modeling

The first step is to define the data that the process requires for its execution. Bizagi allows to structure business data in a graphical and logical way similar to an Entity-Relationship schema, resulting in an easy to understand Data Model. To provide an organized and coherent structure Bizagi provides different types for Entities and different types of Relationships that can be used in order to build the data model [5].

Entities are real or abstract objects (people, places, events, and so on) that can be uniquely identified and are of interest to the business about which information is stored in the system. Entities can be usually thought of as nouns. For example *a Customer, a City, a Company, an Invoice, a Car*. Entities have Attributes. These are the properties that describe each entity. For example a customer has a name, a social security number a gender and an age [5].

Every process in Bizagi has its main Process Entity. This entity provides the starting point to access the rest of the process data, that is, it is the principal entity through which a user accesses the rest of the data model entities [5].

Relationships capture how entities relate to one another and can be thought of as verbs. For example: *a Customer owns a car, a company is located in a city, a customer has a gender* [5]. In Figure 3 is presented the data model created for the User Access Management flow containing all the data needed during the process execution.

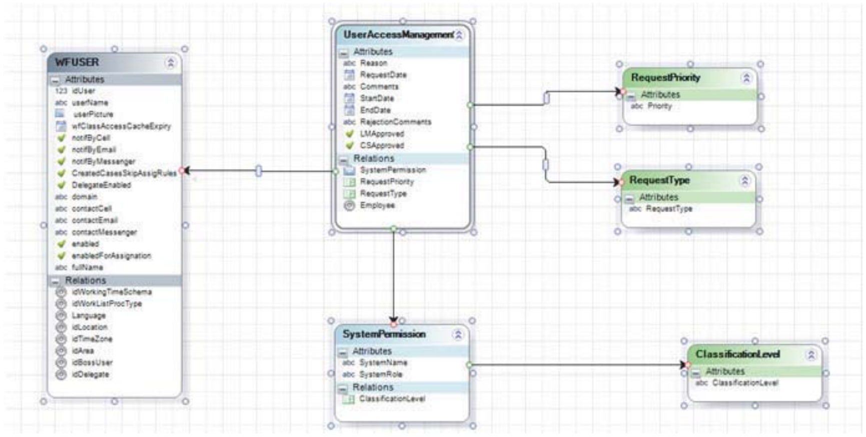
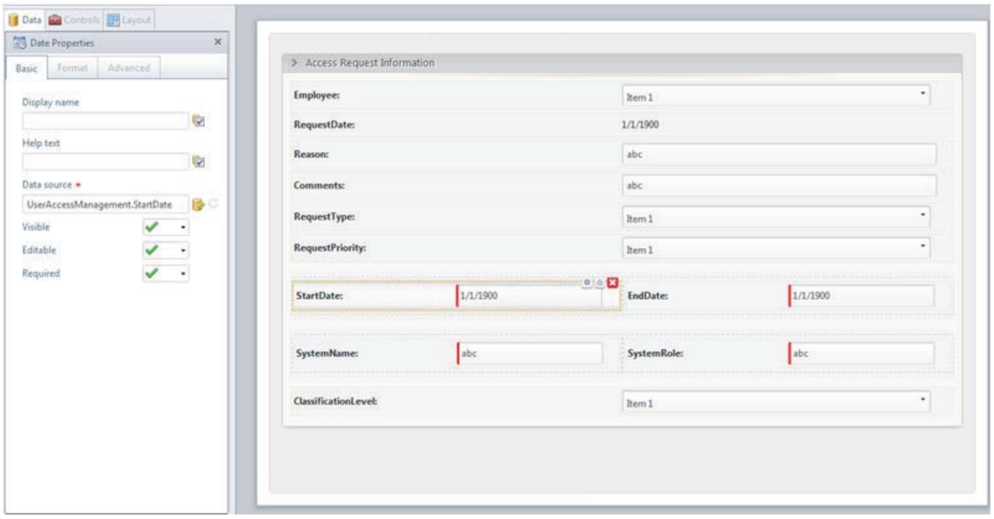


Figure 3: The data model for the User Access Management flow.

Bizagi Studio provides ready for use a set of entities which belong to the Bizagi's internal data model. These are called System entities and the ***WF USER*** entity presented in Figure 3 is one of them. System entities include information concerning the end users, areas, locations etc. These entities are created by default for each project and help in the creation of the data model for the specific process. All is needed is to include and link these entities in the data model created. They represent general entities which are present in any process and thus help to focus our attention in creating the entities which are particular for our process and utilizing System entities for general use cases (like end users in Figure 3). Creating the data model is a continuous process that goes throughout all the steps of process automation until the moment that we are convinced that all the information needed throughout the process execution is present in the data model.

4.2 Forms – User Interface

The second step is to define the user interface for all the activities in the model that require human interaction. In the *User Access Management* process all the activities require human interaction since there are not present automatic activities. To define the user interface for an activity the ***Forms Designer*** is used which provides an intuitive and user friendly structure to drag and drop the data fields onto a form and arrange them accordingly without the need of programing [4]. In Figure 4 is presented the user interface created for the first activity of the *User Access Management* process, *Enter Permission Request*.



Display name

Help text

Data source *
UserAccessManagement.StartDate
Visible ✓
Editable ✓
Required ✓

Access Request Information

Employee: Item 1
RequestDate: 1/1/1900
Reason: abc
Comments: abc
RequestType: Item 1
RequestPriority: Item 1
StartDate: 1/1/1900
EndDate: 1/1/1900
SystemName: abc
SystemRole: abc
ClassificationLevel: Item 1

Figure 4: User interface for the Enter Permission Request activity.

Each item or element included in the design area is known as a *Control* in Bizagi. *Controls* can be added individually by means of drag and drop from the left panel. When a *Control* is added in the design area it does not have a reference in the data model to the specific attribute it represents. In order to link the *Control* with the corresponding attribute in the data model the *Data Source* property of the *Control* should refer the correct attribute in the data model. In Figure 4 is shown how the ***Data Source*** property of the ***StartDate*** control is related with the ***StartDate*** attribute in the data model (*UserAccessManagement.StartDate*) [4].

4.3 Business Rules

Bizagi provides powerful business rules that can be implemented in various places throughout the process [6].The following points clarify the use of business rules.

Perform actions in Activities - calculate and validate: It is very common to perform validations to control the values entered by end users in a field. Bizagi also allows making calculations to manage the data model and assign or delete values to any attribute.

Route Processes - Sequence flow: Business rules are implemented as transition conditions in a process to route the sequence flow. The expressions used for this purpose always return a value of ***True*** or ***False*** (Boolean) in order to tell the process where to go next.

Managing the user interface: it is very important to be able to control what fields are displayed or hidden to the end users, as well as control what fields are mandatory, editable or read-only.

Users allocation: Business rules can be used to control the users that will be allocated to tasks according the business conditions [6].

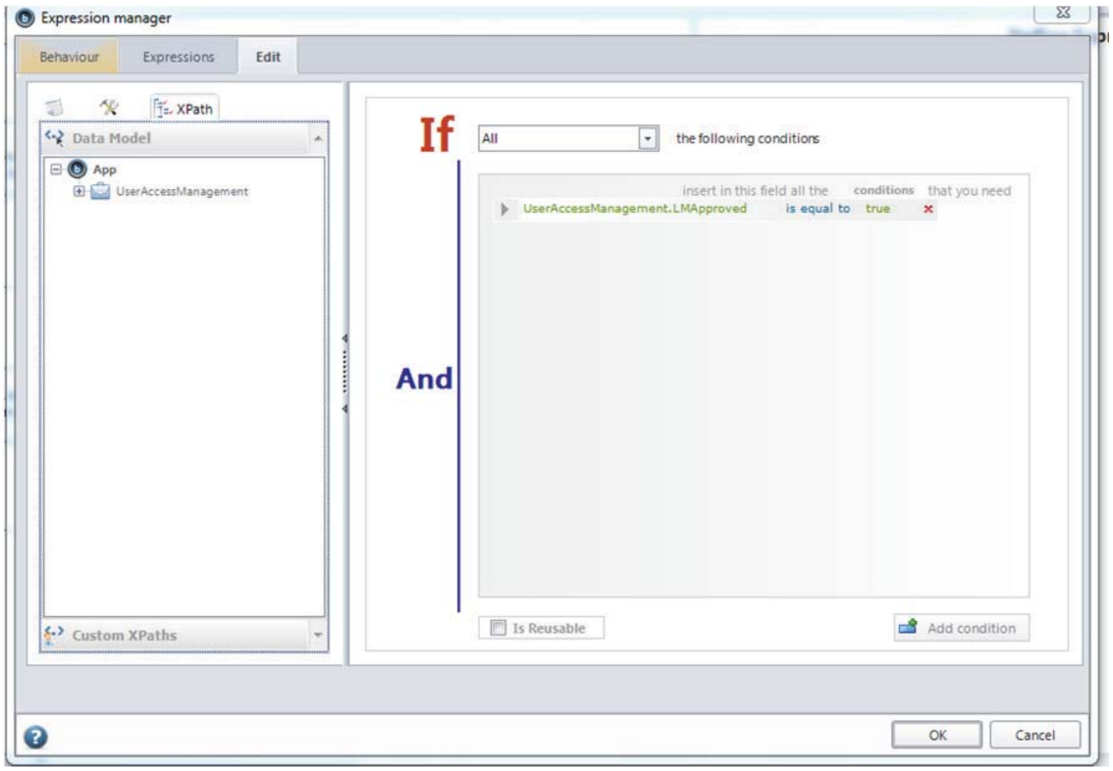


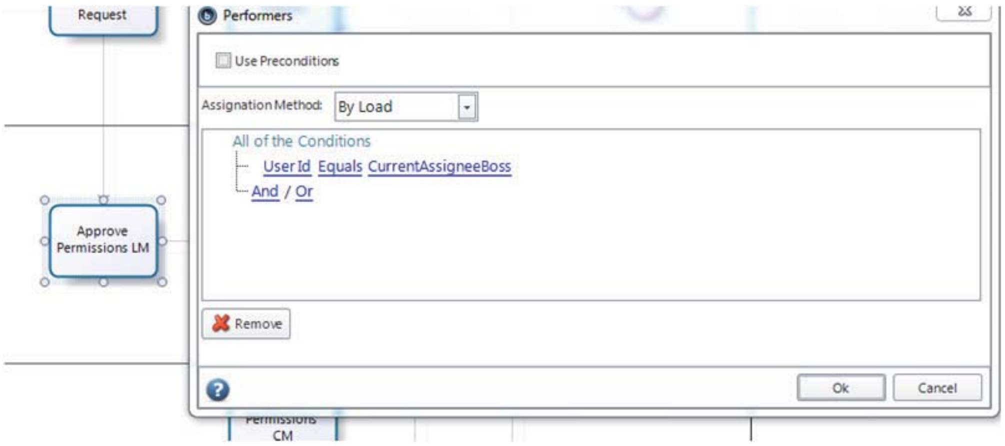
Figure 5: Business rules to route the *User Access Management* process

In Figure 5 is presented how business rules are used in order to route the user access management process. Let's consider the case when the Line Manager has approved the request. This is represented by the following expression (*UserAccessManagement.LMApproved is equal to true*). LMApproved is a boolean attribute present in the data model as shown in Figure 3 that will store the choice performed from the Line Manager. In a similar way are defined the business rules for the other routes of the process.

4.4 Define performers

Work allocation is the next step of the process automation wizard where *Performers* are defined for each activity of the process. *Performers* are the users that have the qualities to be assigned to activities. Each activity created for end user interaction requires definition that will allow Bizagi to allocate the correct users within the organization. Bizagi automatically evaluates the allocation rules defined for each activity and selects one or more users that meet the given conditions from the user's list. Only these users will have access to work on the activity allocated to them [8].

To allocate performers, it is necessary to have a user account created for everyone that intends to work with Bizagi. It is important that each user account must be set up correctly to ensure Bizagi selects appropriately. This powerful feature can be configured based on different criteria like positions, geographical location, skills, roles, among others.



Performers

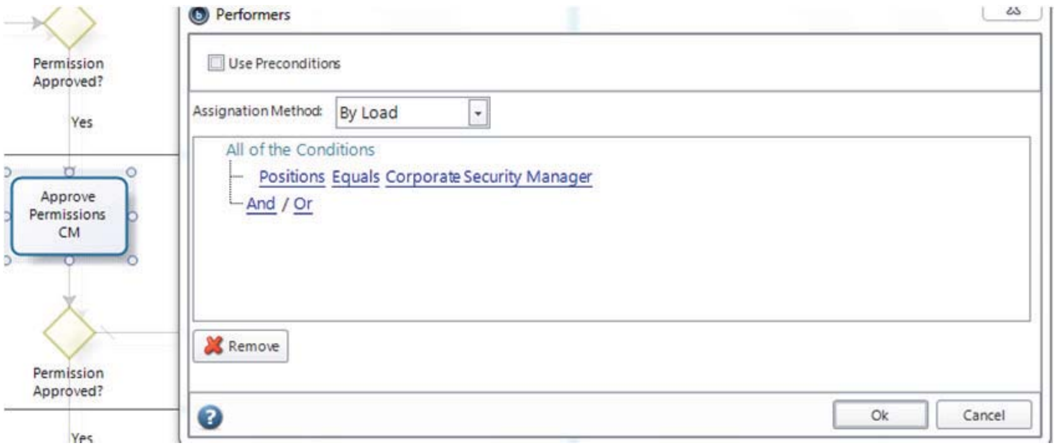
Use Preconditions

Assignation Method: By Load

All of the Conditions
UserId Equals CurrentAssigneeBoss
And / Or

Figure 6: Define performers for the activity *Approve Permissions LM*

In Figure 6 is shown how is defined that the activity *Approve PermissionLM* is always performed from the line manager of the user that has raised the request. The expression used is ***UserId Equals CurrentAssigneeBoss*** that instructs Bizagi that the activity *Approve Permissions LM* should be performed from the line manager of the user that submitted the request in the first place. Similarly in Figure 7 is shown how the activity *Approve Permission CM* is allocated to the Corporate Security Manager through the application of positions present within the company [8].



Performers

Use Preconditions

Assignation Method: By Load

All of the Conditions
Positions Equals Corporate Security Manager
And / Or

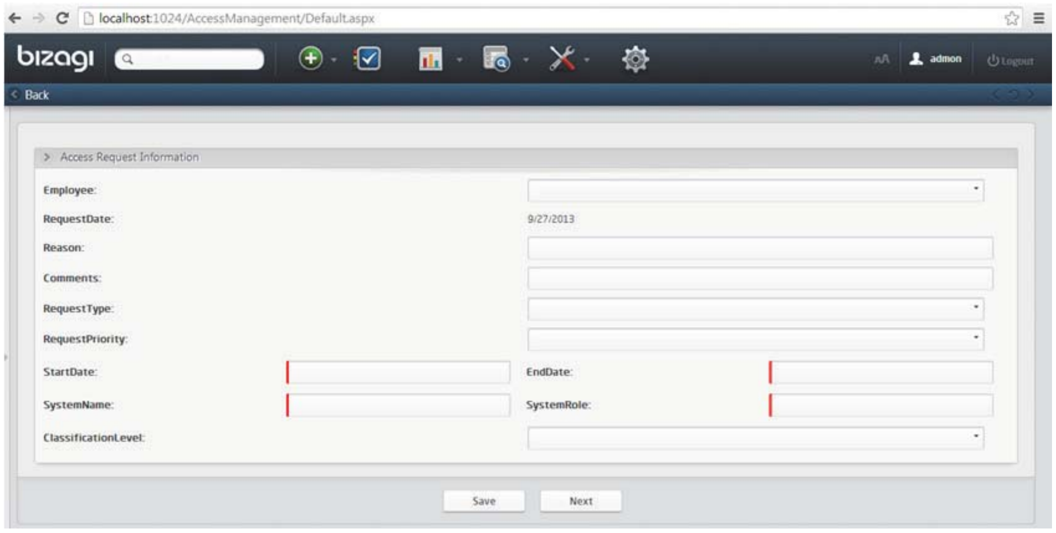
Figure 7: Allocation of the Corporate Security manager to the activity *Approve Permissions CM*.

4.5 Integrations

The next step in the automation wizard provides the capability to integrate the process with other external systems present within the enterprise [7]. In the *User Access Management* process there are considered no integration with external systems.

4.6 Execution

Once the process has performed all the steps of the process automation it is ready to be executed in the Bizagi execution environment which is called Bizagi Engine [3]. An enterprise web application is generated automatically which is ready for execution starting from the BPMN model without the need of programming. The user access management application in execution is presented in Figure 8.



Access Request Information

Employee:
RequestDate: 9/27/2013
Reason:
Comments:
RequestType:
RequestPriority:
StartDate:
EndDate:
SystemName:
SystemRole:
ClassificationLevel:

Figure 8: User Access Management application in execution

5. Limitations

Starting from a BPMN model that represents a particular business process and following the process automation wizard in Bizagi Studio is possible to generate a web application that automates the business process without the need to write any code. Bizagi business process management suite fully supports the MDA principles. Although the following points need to be taken into consideration as limitations:

a) Bizagi is a suite for process automation and is not applicable for general purpose application development.
b) The processes automated with Bizagi can be executed only within the Bizagi environment (Bizagi Engine).
c) Bizagi is a proprietary solution since respective license fees need to be paid for enterprise deployments.

6. Conclusions

In this article the Bizagi business process management suite was presented through the practical development of an application that represents the user access management flow in modern organizations. An enterprise web application starting from a BPMN model was generated without the need to write code confirming the 100% MDA approach of Bizagi business process management suite.

References

[1] Bizagi BPM suite documentation, available http://help.bizagi.com/bpmsuite/en/.
[2] Bizagi BPM suite overview, available http://help.bizagi.com/bpmsuite/en/index.html?overview_what_is_bizagi_bpm_suite.htm
[3] Bizagi Engine, available at http://help.bizagi.com/bpmsuite/en/index.html?process_execution.htm
[4] Bizagi Studio, creating the user interface, available http://help.bizagi.com/bpmsuite/en/index.html?creating_the_user_interface.htm.
[5] Bizagi Studio data modelling, available http://help.bizagi.com/bpmsuite/en/index.html?modeling_data.htm.
[6] Bizagi Studio defining business rules, available http://help.bizagi.com/bpmsuite/en/defining_business_rules.htm
[7] Bizagi Studio integration, available http://help.bizagi.com/bpmsuite/en/index.html?integrating.htm
[8] Bizagi Studio work allocation, available http://help.bizagi.com/bpmsuite/en/index.html?work_allocation.htm
[9] Bizagi Process Modeler documentation, available http://help.bizagi.com/processmodeler/en/.
[10] Forester Consulting, Modernizing Software Development through Model-Driven Development. A commissioned study conducted by Forrester Consulting on behalf of Unisys, August 13, 2008.
[11] Frankel, S. D.: Model Driven Architecture. Applying MDA to Enterprise Computing. Wiley Publishing, Inc. OMG Press 2003.
[12] Guttman M, Parodi J: Real-Life MDA: Solving Business Problems with Model Driven Architecture, (The MK/OMG Press), 2007.
[13] Kleppe, A., Warmer, J., Bast, W.: MDA Explained. The Model Driven Architecture: Practice and Promise. Addison-Wesley, Reading, MA 2003.
[14] MDA Success Stories, available http://www.omg.org/mda/products_success.htm
[15] Object Management Group (OMG), available http://www.omg.org

[16] Object Management Group (OMG), Business Process Model and Notation, available http://www.omg.org/spec/BPMN/
[17] OMG, “MDA Guide Version 1.0.1”, available http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf 12th, June 2003.
[18] OMG Model Driven Architecture, available http://www.omg.org/mda
[19] Papajorgji P., Pardalos, P.: Towards a Model-Centric Approach for Developing Enterprise Information Systems.
[20] Pastor O., Molina, J.C.: Model-Driven Architecture in Practice A Software Production Environment Based on Conceptual Modeling, Springer 2007.
[21] White S.A.: Introduction to BPMN, July 2014.