Onunga J, Mbugua S, International Journal of Computer Science and Mobile Applications, Vol.11 Issue. 1, January- 2023, pg. 19-28.

ISSN: 2321-8363
Impact Factor: 6.308

(An Open Accessible, Fully Refereed and Peer Reviewed Journal)

Model Driven Architecture a Review of Current Trends

Jeremiah Onunga[1*], Samuel Mbugua[2]
[1]PhD Candidate, Kibabii University Kenya
[2]Director, Planning & Organization Performance, Kibabii University
E-mail: jerryosida@gmail.com

Received date: 27 December 2022**,** Manuscript No. IJCSMA-22-84758; **Editor assigned:** 8 January 2023**,** Pre QC No IJCSMA-22-84758 (PQ); **Reviewed:** 18 January 2023, QC No. IJCSMA-22-84758 (Q)**;** **Revised:** 23 January 2023**,** Manuscript No. IJCSMA-22-84758 (R); **Published date:** 30 January 2023 doi. 10.5281/zenodo.7824482.

Abstract

The Object Management Group (OMG) adopted the Model Driven Architecture (MDA) approach from among the various Model Driven Engineering (MDE) methods. The MDA technique, which is based on the separation of concerns principle, aims to automate the software development process by using models rather than traditional coding. Model Driven Architecture (MDA) is a new technique to application modeling and creation that has gotten a lot of attention recently. Many organizations are now considering MDA as a way to organize and manage their application solutions, tool vendors are explicitly referring to their capabilities in terms of "MDA compliance," and the MDA lexicon of platform-specific and platform-independent models is now widely referenced in the industry. The OMG produced the second iteration of the MDA guide in June 2014 in an effort to implement the essential ideas and supplement the 2003 original with more precise specifications. Researchers' interpretations of the MDA words vary due to the 11-year difference and their respective perspectives and backgrounds. What causes uncertainty about what falls under and outside the MDA's scope. In this work, we present an overview of the current MDA trends simplifying integration challenges and enhancing business Information Technology (IT) alignment. This paper establishes boundaries around what constitutes MDA, positioning MDA in relation to other model-driven methods. It will also demonstrate its advantages over conventional software development and other model-driven methodologies. This paper highlights problems and difficulties that researchers have identified that affect the MDA. The paper also provides some insight into the MDA trends in research and directions, particularly with regard to the platforms that the MDA development process is aimed at and its automation.

ISSN: 2321-8363
Impact Factor: 6.308
(An Open Accessible, Fully Refereed and Peer Reviewed Journal)

***Keywords**:* MDA; MDE; OMG; Model driven architecture; Trends

1. Introduction

Object Management Group (OMG) War was formed in 1989 and now has a global coalition of over 800 enterprises. The Common Object Request Broker Architecture (CORBA), Unified Modelling Language (UML), Meta Object Facility (MOF), Extensible Markup Language (XML), XML Metadata Interchange (XMI) and the Common Warehouse Metamere (CWM) are just a few of OMG's achievements. All of these standards help to bring the concept of model-driven development to life. OMG introduced a new framework called Model Driven Architecture about 2001. (MDA). Unlike the OMG's other standards, the MDA allows for the usage of models rather than traditional source code. It remains to be seen whether this new approach to software development will gain traction. The OMG rapidly transitioned from its previous Object Management Architecture (OMA) vision [1] by publicly announcing the adoption of the Model Driven Architecture (MDA) approach in 2001. This was done in response to the success in providing technology independent infrastructure standardization [2]. The OMG provided a standardized definition for the Unified Modeling Language (UML), the Meta Object Facility (MOF), XML Metadata Interchange (XMI), and the Common Warehouse Meta models (CWM) in order to enable the MDA technique [1]. These requirements serve as the MDA's fundamental framework [2]. Additionally, it helps in improving system integration and interoperability within systems the first manual for the MDA principles was published by the [1]. The manual offered a thorough explanation of MDA terminology and served as a technical reference for MDA practitioners. OMG launched a new version in June 2014 that was significantly less technical in details than the prior one [1]. Its main objective appears to be to present the business case for MDA. According to the researchers' backgrounds, eleven years of OMG silence forced them to include new ideas and concepts from different MDE techniques into the context of MDA. As a result, future MDA adopters regard these expressions and ideas as legitimate MDA tenets.

It's never been easy to create enterprise-scale software solutions. The difficulty of comprehending very complex business domains are frequently compounded by the difficulties of coordinating a development effort involving big teams of engineers over several phases of a project that can last months. The time-to-market pressures that many of today's product development efforts are subjected to only exacerbate the issues. The original notions of the MDA are constrained by this publication. We also compared the MDA to other traditional MDE methods and traditional software development techniques. In addition to that, this is a review of recent MDA literature. There is no literature for another model-driven method included in the work.

This paper's major objective is to provide a thorough examination of the original MDA principles laid out by the OMG. This paper lays out the path for researchers or software developers who desire to pursue the MDA.

2. Review of MDAs

MDA has become more sophisticated as Enterprise Architecture, tools, and people. The Model Driven Architecture was approved by OMG as a method for utilizing models in software development. Through the use of architectural

ISSN: 2321-8363
Impact Factor: 6.308
(An Open Accessible, Fully Refereed and Peer Reviewed Journal)

separation of concerns, its three main objectives are portability, interoperability, and reusability. New platforms and technologies are continually being developed, MDA speeds up the process of creating new specifications that make use of them. MDA offers a thorough, organized solution for application compatibility and portability in the future in this way. The additional benefit of accurately capturing the solution domain's intellectual property in a technology-neutral manner is provided by Unified Modeling Language (UML) modeling of the problem domain. It is used drive IT deployment, modify business processes, and harmonize processes to enable IT implementation.

3. Model driven architecture

Deriving value from models that help us reduce the complexity and interdependence of complex systems is a key goal of MDA. Three architectural layers are used to implement the separation of concerns concept: The business and domain vocabulary generally provided by subject matter experts was captured by the Computational Independent Model (CIM) at the first layer. This layer is bridging the gap that currently exists between system implementers and domain specialists. The Platform Independent Model (PIM), the second layer, formalizes the data in the (CIM) model without regard to any particular technology platform. The third layer, the Platform Specific Model (PSM), focuses on the technical and platform implementation specifics. Despite its reservations, it is still believed that the (PSM) is too abstract to be implemented on a computer platform. In order to support the PSM and serve as a technical manual for the intended platform, the Platform Model (PM) was put in place. The PM is present in model transformation rules implicitly. The scope of the model transformation is constrained to address a single supposed platform due to the implicit use of the PM during the transformation [3]. The prospect of applying this model mapping for platforms other than the one for which it was built is what causes uncertainty. However, it has been made explicit to drive the shift to a run time environment in [4].

One of the key MDA operations is the transformation or mapping of the model. With the Platform Model (PM), high-level models Personal Information Management (PIM) can be converted to low-level models Process Safety Management (PSM), or vice versa. The translation can also occur at the same level of abstraction (from PIM to PIM) or (PSM-to-PSM). These mapping situations are referred to as Model-to-Model (M2M) mapping or Model-to-Text mapping (M2T). Both are backed by several tools that are provided to address each scenario and mapping and fall under the MDA umbrella [5]. The MDA models are differentiated from the other Model Driven approaches by the automation of the transformation process. This is because methods like Model Driven Engineering MDE, Model Based Engineering, and Model Driven Development (MDD) directly create code from models instead of using them as a communication tool between analysts and programmers for the system. Skipping the MDA architectural levels and lacking development process. Positioning the MDA in relation to other model-driven approaches is shown in the Figure 1 below.

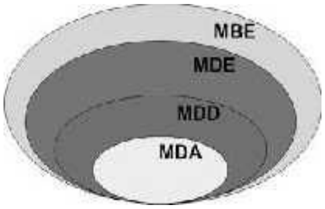

Figure 1. Positioning the Model Driven Architecture (MDA) among other approaches

The MDA has an advantage in terms of software productivity and cost savings thanks to automation as well. Practically, any automation that is conceivable happens from the coding level down. As a result, updates to the top layers above the coding need a significant amount of recoding labor, whereas coding changes do not reflect the top levels. This is due to the fact that the text and images in the aforementioned layers are frequently utilized for documentation and communication. The MDA procedure which shows a level to level automated closed loop. where, with a minimum of effort, time, and expense, changes in the abstract top layers automatically propagated to the lower ones. On the other hand, by reversing the translation from lower level of abstractions code/artifacts to the upper layers with higher level of abstraction models, changes in the lower abstract levels can be reflected in the higher ones [6].

4. MDA to facilitate process harmonization and IT implementation

The Enterprise System's (ES) successful transformation of the logistics and financial operations is a key component of the MDA strategy. The (ES), which was launched in 2003, was the first sizable system in which substantial business process models were constructed to study and determine business requirements. The method was essential in making it easier to analyse, harmonize, and integrate the various business processes found across the various services and Lines of Business (LOB). It would have been very impossible to envision and comprehend how complex the business activities were without these business models. In total, more than 90% of the 600 procedures were standardized and harmonized by the initiative. (ES) Logs were implemented over time using these business process blueprints as a guide. Reusing procedures also reduced the cost of implementing systems by millions of dollars. It also changed how financial and logistical operations are conducted today [7]. Further advantages resulted from expanding this endeavour to include the fields of platforms, buildings, and infrastructure, medical logistics, IT, and Research and Development (R&D).

5. MDA for business process transformation

Business transformation is facilitated and driven by Business Process Management (BPM). It is in charge of creating the Business Maps, facilitating enterprise integration, and collaborating with business owners to lead business transformation programs. Since the business transformation endeavour involves the entire organization, strong commitment and backing are essential to its success. The IT departments of many organizations analyse and approve plans for business transformation, and they also play a crucial role in locating and resolving ownership issues for business areas where there is a lack of clear ownership [7].

Additionally, developed and used was a four-phase Integrated Methodology for Business Transformation. In this method, the conceptual solution is developed after the goal business architecture has been developed and the business functions that will be prioritized and chosen for process mapping have been selected. Implementing the solution is the final stage. This method has now been used to successfully undertake a number of business process transformation projects with improved management skills in sectors like human resources, construction and infrastructure, transportation, and ammunition.

6. MDA to support IT growth

It is now possible to translate model needs into applications due to the market's mature applications. Across organizations worldwide, MDA adoption has been on the rise as a means of advancing IT. The MDA technique makes use of SAP's Application Lifecycle Management (ALM) function for Commercial off the Shelf (COTS) solutions like the System Applications and Products (SAP) Enterprise Resource Planning (ERP) System, where models are synced into a SAP ALM tool for application development. The MDA technique uses BPM Suite (BPMS) to convert business process models into executable applications for customized non COTS systems [8].

7. Problems with MDA

Before the MDA was formally adopted by the OMG, the work in presented the vision and benefits of the MDA, but they also identified the diversity and dynamicity of technology as the key problem that makes system integration and interoperability challenging. Which confronts all methods and phases of software development, including MDA. The need to limit the MDA models' level of abstraction in order to make them executable on the intended platform is what drives this. To handle the platform environment, a chain of model transformations was therefore used, a new PSM generation, in other words. It's not simple to manage these new models with their corresponding transformation rules and methodology. Corporate mergers and new acquisitions, does not a guarantee that businesses would continue with a single middleware, such as CORBA.

The main MDA activity is model transformation. Addressing model transformation, various methods and tools are provided. The transformation models and technologies that facilitate model to model transformation were the topic of a survey study by [9]. Model to Text Transformations are encouraged by [10]. The MOF Script language, which was submitted to the OMG as a model for a text transformation language, is evaluated as they do so. It might be challenging for MDA practitioners to determine which tools support which transformation methodology because there are various transformation methodologies that are supported by some of the transformation technologies available on the market.

A chain of model transformations from a high level of abstraction to a lower level is instead caused by the distinct levels of abstraction in MDA. The model transformation made an implicit assumption about the PM. The model transformation may not be applicable to platforms other than the one for which it was designed. The PSM is directed at a certain platform. It is still regarded as being too speculative to be used on the platform, though. On the other side, there isn't another platform like it. Java itself slowed down to technology diversity (Java 2 Platform, Standard

Edition (J2SE), Java 2 Platform, Enterprise Edition (J2EE), Java Platform, and Micro Edition (J2ME). A suitable framework that can drive the transition with little change and additional information about the execution environment are required. When defining the platform elements using the ontology addressed these challenges. However, because there are so many classes and instances, processing significant ontology cannot be automated. This is in addition to the fact that manual ontology construction takes more time when platform diversity and data volume rapidly rise. This is then reflected in the quantity of ontologies needed to handle this technological volatility [11].

Another issue that the MDA is all about is the automation of the development lifecycle and software productivity. Developers need automated techniques that are scalable, general, and extensible. However, the MDA process lacks proper tools that can support the automation. The work in proposes a framework to supports the MDA automation directions. Beside their presentation to the platform component, they provide a benchmark to the tools that support the MDA automation vision [12].

8. Actions by MDA

New directions encourage the omission of some MDA layers or the addition of new ones in light of the aforementioned problems. The research in encourages the removal of the PSM and direct PIM addressing of the targeted platform. It is comparable to the agile approach used by [11, 12]. While moving straight for the UML 2.0 profiles, which offer some flexibility in expressing and addressing platform's elements, some agilest, like, are chasing the ideal of having an executable model [5].

An already-existing idea called Agile MDA is growing in popularity with software companies as the model's focus shifts from documentation to actual execution and deployment. Despite its claims, the MDA is seen as a cumbersome procedure that may result in the expensive and delayed delivery of the incorrect system. The idea that the code and model are operationally the same is driven by the agile methodology. Where an executable model can be built, executed, tested, and updated iteratively in brief cycles [5].

On the other side, methods like Arch MDE, this aims to provide a new standard that permits software architecture to be created from an analytical model. Because they want to expand the MDA concept by including a new layer (AIM). According to some studies, using CASE tools could be a hybrid approach that combines MDD and MDA [13]. The original CASE Tools were created using a conventional database model. Where models are kept in a repository that can be accessed. We now have the freedom to represent models in a textual format that can be converted into an XML format thanks to the work in [13]. We did, however, stay inside the MDA limits. The MDA specifications allow the text format, which provides more control over the model constraints and dependency management.

Similar to this, the work of offers a methodology for an end-to-end transformation from PIM to PSM under the MDA's purview to support the software product line. Additionally, the study of showed a strong dedication to the MDA tenets [14, 15].

9. The Process of Review

This literature was reviewed using a search of the OMG MDA database and its specifications section. MDA and other model-driven methodologies were also searched for in academic databases like Google Scholar, Springer's, IEEE Explorer, and ACM. In order to compare the MDA approach with other model-driven techniques, we go through an operational definition process to establish what is inside the MDA scope using the OMG version 1 and guide as a foundation. The principle of separation of concerns was adopted, the conformance of the MDA architectural layers, Methodology for transformation, Managing Platform Diversity versus Development Lifecycle Automation.

This review effort developed a hypothesis for changing the existing MDA architectural layers by examining a relevant type concepts and research in the domain of model-based software development. The high level architecture along with the two MDA guides published by the OMG, served as the basis for the evaluation criteria for the evaluated literature.

10. Trends

It is clear from the literature cited above that Model Driven Approaches (MDAs), including process intelligence, model-driven testing, model-driven documentation and development, and support for IT expansion, are increasingly being used in software development. However, MDA-based applications are growing quickly. MDAs' portability, interoperability, reusability, business process transformation, process harmonization, and IT deployment are its key selling points for software application companies. The systems created from MDA must be simple to use at anytime and anywhere if MDA applications are to be adopted by customers. This is the "MDA experience," which is crucial for gaining the support of clients and organizations.

The objectives of the MDA's first phase of adoption have been achieved. The technique will gradually be spread throughout Corporate IT (CIT) systems in the following phases. To do this, the current MDA methodology will be examined in order to improve its effectiveness and efficiency in accommodating various kinds of IT systems. The continued improvement of AVATAR's connection with SAP ERP systems will also be a main priority. This might be accomplished by further automating the management of the application lifecycle and improving the caliber of the models already recorded by AVATAR.

The MDA method assures business continuity in the event that it becomes necessary to re-platform the SAP ERP system, in addition to the goals of enhancing business agility and cutting the time between application creation and implementation. This overview includes many contemporary research trends, which are grouped into two categories: The MDA infrastructure and the tools that support its concepts are the focus of the first trend. If researchers follow this trend strictly and obtain OMG clearance for their methods or tools, they will only have a little amount of area to work with. This is due to the OMG's extremely low productivity in releasing improved or new specifications to MDA adopters. As a result, various tools emerged that do numerous MDA fundamentals. However, it may not always adhere to OMG requirements.

For this circumstance, the Model-to-Text is an excellent illustration. The criterion is there, but there are very few tools and studies that adhere to these requirements. As a result, new MDA members are in dire need of tools that offer a respectable degree of documentation and support. The MDA applications are the subject of the second research trend. This trend is developing quickly and has reached an advanced stage of maturity. The results and success stories of various MDA applications and studies in the fields of cloud computing, software testing, and embedded systems are quite intriguing.

Based on research using MDA models the three MDA views are represented by three default models of a system. Since a series of models may be built within each of these three levels, each of which corresponds to a more specialized viewpoint of the system (user interface, information, engineering, and architecture), the author seems to think that these models can be considered of as layers of abstraction. According to this review, common MDA models include the following: Platform Independent Model (PIM), which demonstrates a sufficient level of independence to enable its mapping to one or more platforms, Computation Independent Model (CIM), which primarily serves as a business or domain model, and Platform Specific Model (PSM), which combines the PIM's specifications with the information needed to specify how a system uses a specific type of platform. The author quotes the literature in saying that in an MDA specification, the PIM and PSM components that implement the CIM requirements should be able to be traced back to them, and vice versa.

According to the assessment, the Model Driven Architecture's objective is to make it easier to create machine-readable models with the aim of achieving long-term flexibility in terms of: Technology obsolescence: Existing designs can more easily support and incorporate new implementation infrastructure; Portability: As required by business requirements, current functionality can be more quickly moved into new settings and platforms; Productivity and time to market: By automating a number of time consuming development processes, architects and developers are able to concentrate on the fundamental logic of the system; Quality: This approach's formal separation of concerns as well as the created artifacts consistency and dependability all help to improve the overall system's quality. Integration: Significantly easier to create integration bridges with legacy and/or external systems; Maintenance: By making the design machine-readable, analysts, developers, and testers have direct access to the system's specification, which makes their maintenance tasks simpler; Models can be immediately evaluated against requirements and tested against different infrastructures through testing and simulation. They can also be used to mimic how the system under design would behave; Return on investment: Companies can get more value out of their tool investments.

11. Discussion

There are also model-centric techniques, such as MDE and MDD that strive to design and construct software products that can run on numerous platforms with minimal engineering. These strategies' top priorities are improving software quality, productivity, interoperability, cost, and time to market. In actuality, both strategies use several levels of operation to accomplish the same objectives. In order to develop software, the MDA used models at higher, independently concerned abstraction layers, avoiding hard coding in an end-to-end automation. The other model-driven approaches, however, weren't just concerned with process automation or concern separation. Thought: Models have occasionally been utilized for communication or documentation purposes.

Many researchers have developed their own initiatives from various model-driven techniques based on their unique perspectives and backgrounds as a result of the lack of OMG guidance in implementing the MDA concepts. As a result, a state of confusion predominates in the research findings, and the MDA fundamental principles are undermined.

In this review, the Agile MDA is a topic that is highly intriguing. Particularly in the model merging section. Since both the PSM and the PM are models and may be represented graphically or textually, they can be combined. Lack of a tool prevents the removal of the PSM component or the addition of a new layer to the MDA process. Such activities cannot be managed using the MDA standard tools. MDA emphasizes model-to-model transformation as well. We must adopt the MOF paradigm to text if we are to have executable code. This means that we will only be able to use a few languages, like EMF or ATL.

12. **Conclusion**

Model Driven Architecture (MDA) usage is still uncommon. The traditional method of development is altered by software development based on models and mapping operations. To develop the system, one will choose models rather than building systems repeatedly as technology advances. MDA improves quality, lowers costs, and boosts software durability while enabling reuse at the domain level.

In order to explicitly control many tasks and contexts (Hardware/Software activities), computer systems combine a number of features and components that are often exclusive to one technological stream. The quality and productivity of the software development process are being greatly improved by the capacity to produce software out of models to support many platforms for various technologies at various degrees of abstraction. The MDA principles, in particular how it differs from other model-centric methods, can positively impact software quality and productivity in general. This work's significance lies in the fact that it can be utilized to comprehend the MDA's present stance and how to adhere to its values. Additionally, this evaluation can be used to position and preserve the work of scholars who wish to pursue other methodologies. This paper has evaluated and discussed the problems that the MDA progress is now facing. It's interesting to note that same problems affect all other model driven approaches in one way or another.

The study demonstrates that MDA may be embraced and used to improve interoperability, portability, and usability in software systems. The MDA specifies a model architecture that successfully divides issues pertaining to integration, interoperability, and portability. MDA enhances productivity by automating the mapping process, quality by reusing tried-and-true techniques in the mapping, and maintainability by fostering better concern separation and consistency between models and code.

Despite all these benefits, a review of the literature reveals that MDA is now dealing with some problems and limitations. Organizations interested in employing MDA, however, find that there are problems and restrictions with it.

ISSN: 2321-8363
Impact Factor: 6.308

(An Open Accessible, Fully Refereed and Peer Reviewed Journal)

References

[1] Agner, L. T. W., et al. "Model refinement in the model driven architecture context." (2012).

[2] Almeida, J. P. A., et al. "The role of the RM-ODP Computational Viewpoint Concepts in the MDA approach." *Workshop on ODP for Enterp Comput WODPEC* (2004).

[3] Dobing, B. & Jeffrey, P. "Understanding the role of Use Cases in UML: A review and research agenda." *J Database Manag* 11.4 (2000): 28-36.

[4] Elleuch, N., Khalfallah, A., and Ahmed, S., B., et al. "Archmde approach for the development of embedded real time systems." Reliable Software Technologies–Ada Europe 2007: 12th Ada-Europe International Conference on Reliable Software Technologies, Geneva, Switzerland, (2007), 25-29. Proceedings 12. Springer Berlin Heidelberg, (2007).

[5] Gerber, A. & Gabriel, M. J. M. Aquatic invertebrates of South African rivers field guide. *Inst Water Qual Stud Dep Water Aff For* (2002).

[6] Czarnecki, K. & Simon, H. "Classification of model transformation approaches." *Proc 2nd OOPSLA Workshop Gener Tech Context Model Driven Archit* 45.3 (2003).

[7] Jackson, Ethan, K., et al. "Components, platforms and possibilities: towards generic automation for MDA." *Proceedings of the tenth ACM international conference on Embedded software*. (2010).

[8] Lim, H. C., et al. Driving Business Transformation through a process centric approach. *DSTA Horizons* (2010): 28-43.

[9] McFarlane, L. N., et al. "Development of simulation games to improve the practice of program management." *Syst Inf Eng Des Symp IEEE* (2009).

[10] Mellor, S. J., et al. MDA distilled: principles of model-driven architecture. *Addison Wesley Prof* (2004).

[11] Mukhtar, M. A. O., and Abdullah, A. "Mapping of Behavior Model using Model-Driven Architecture." International Journal of Computer Applications 13.8 (2011): 35-39.

[12] Oldevik, J. "Transformation composition modelling framework." *Springer* (2005).

[13] Poole, J. D. "Model-driven architecture: Vision, standards and emerging technologies." *Workshop on Metamodeling and Adapt Object Models ECOOP*. 50 (2001).

[14] Soley, R. "Model driven architecture." *OMG White P* 308 (2000): 5.

[15] Dennis, W., and Straeten, R. V. D. "Platform ontologies for the model driven architecture." *European Journal o f Information Systems* 16.4 (2007): 362 373.