**Model Driven Engineering or Model Driven Development** – A software engineering approach to raise the level of abstraction of developers who create a software with the goal of simplifying the process and tasks that are involved in the software life cycle [1, 3].

**Model Driven Architecture** – Model Driven architecture provides a framework for a software system to be built [2].

**Model to Model Transformation –** Model transformation is a process to convert one model to another model [4].

**Model Transformation Languages -** Model transformation languages are well defined to perform model transformation. Some of the model transformation languages are ATL, QVT, SmartQVT, ETL etc. [4].

**Benefits of Model to Model Transformation**

1. Model transformation can be used for creating, filtering, and modifying models [4].
2. Model transformation can be used in the different phases of the development of software life cycle [4].
3. Model transformation is portable i.e. it can work on various platforms like network of computers, middleware etc. [3, 5].
4. It helps in reducing complexity like bridging the gap between the abstraction and the implementation. [3, 5].
5. Model Transformation can be used with the information what is captured at the first instance [4].
6. Model Driven Development helps in achieving the goals very easy like transforming the source model to the desired target model [6].
7. Model Driven Development helps in standardizing the process from which the automation would be very easy [6].
8. Model Driven Development as a software engineering approach helps to define the architecture that will be followed to build a product [6].
9. Model Driven Development helps a company to increase the return on investment and to improve the productivity [7].
10. Model Driven Development helps the developers to increase the short term and long term productivity of the software product designed [7].

**Benefits of Model Transformation Languages**

**ATL:**

ATL is known as Atlas Transformation Language. It is used in model to model transformation.

1. ATL supports both programming and automating constructs i.e. both simple and complex mappings of the data can be easily achieved [4].
2. The ATL transformation language is composed of transformation rules which will help to create a target model from the source model [4].
3. ATL is integrated in Eclipse Modeling Framework (EMF) so it can also handle the models based on EMF [4].
4. ATL code is compiled and executed by ATL transformation engine [4].
5. The transformation rules for the target model are given in this ATL [4].
6. ATL creates a target model from the given source model directly as the incremental transformation is not supported by ATL [4].

**QVT:**

QVT is known as Query/View/Transformation and it is a standardized language for model transformation [4].

1. QVT has a mechanism from where the external code can be called for the transformation purpose [4].

QVT defines three different transformation languages [4]

1. QVT Relational – Supports functioning in two directions and it is used to check the consistency of the models.
2. QVT Core – Simple, act as a foundation for QVT Relational and the variables of source, target and trace models are treated equally.
3. QVT Operational – Imperative language designed for single directional transformations.

**Limitations of the existing techniques versus Benefits of proposed technique**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No** | **Method** | **Limitations of the Existing Techniques** | **Benefits of the Proposed Technique** |
| 1 | Part of Speech Tagging | *The main challenge in the POS is removing ambiguities, for instance “Flies like a flower” here the POS for the words in a sentence are*  *Flies: noun or verb?*  *like: preposition, adverb, conjunction, noun, or verb?*  *a: article, noun, or preposition?*  *flower: noun or verb?*[[1]](#footnote-1) | Model transformation helps in reducing complexity like bridging the gap between the abstraction and the implementation. [3, 5].  *In software development models play a major role, these models help developers to develop a technology or software to solve a problem.* |
| 2 | Optical Character Recognition | The main limitation in extracting information is because the data is often mixed with text and graphics [11].  Variations in style and shape of the data [11].  Variations because of subscripts and superscripts in the data [11]. | Model Transformation can be used with the information what is captured at the first instance [4].  Model Transformation helps in creating a model from the information captured at the first instance instead of starting from the scratch. |
| 3 | Big Data Analytics | Data extraction and cleaning is one of the limitation in big data analytics.  Data Integration is another limitation as the data will be coming from different databases or web portals. | Model Driven Development helps in standardizing the process from which the automation would be very easy [6]. |
| 4 | Web Crawling in Information Retrieval | Web crawling is very difficult because of its large volume and its rate of change [14].  Large volume states that it can download only certain limit of web pages in each time.  Rate of change implies that the web page either might be added or deleted as there will be lot of changes in the web pages on daily basis. | Model Driven Development as a software engineering approach helps to define the architecture that will be followed to build a product [6]. |
| 5 | Data Mining | Issues may arise with the missing, corrupted, inconsistent data as the information recorded will be in different format from different sources [10]. | Model Transformation can be used with the information what is captured at the first instance [4].  Model Transformation helps in creating a model from the information captured at the first instance instead of starting from the scratch. |
| 6 | Text Analytics | Lot of software programming is needed to extract textual information from different sources [9].  Managing the unstructured data from various sources is complicated [9]. | In MDA, a model to text transformation (M2T) is a transformation definition (set of transformation rules) that transforms an expressed Specific Model to target source code or documentation. M2T transformation tool allows writing transformation definitions, running transformations, and produce texts (target source code or documentation of a system) as outputs [13]. |
| 7 | Probabilistic Model in Traditional Information Retrieval | Probabilistic models are very hard to build and program. | Model Driven Development helps in achieving the goals very easy like transforming the source model to the desired target model [6].  For instance, if the target model should be in specific format or type it can be easily achieved by giving specific instructions during the conversion. |
| 8 | Text Summarization | Summarizing the whole content of the document from various sources in a right way (language, format, etc.) to a specific user [11]. | In MDA, a model to text transformation (M2T) is a transformation definition (set of transformation rules) that transforms an expressed Specific Model to target source code or documentation. M2T transformation tool allows writing transformation definitions, running transformations, and produce texts (target source code or documentation of a system) as outputs [13]. |

**REFERENCES:**

1. Sven Fortuin, Model Driven Engineering: the decomposition of environments, 2016
2. Stephen J. Mellor, Kendall Scott, Axel Uhl, Dirk Weise, Model-Driven Architecture,
3. Robert France, Bernhard Rumpe, Model-driven Development of Complex Software: A Research Roadmap,
4. Matthias Biehl, Literature Study on Model Transformations, July 2010
5. Gabriel Costa Silva, Louis M. Rose, and RaduCalinescu , A Qualitative Study of Model Transformation Development Approaches: Supporting Novice Developers
6. B.Hailpern, P.Tarr, Model Driven Development: The good, the badand the ugly.
7. Colin Atkinson, Thomas Kuhne, Model Driven Development: A Metamodeling Foundation
8. Line Eikvil, Optical Character recognition, December 1993
9. William M. Darling, Michael J. Paul, Fei Song, Unsupervised Part-of-Speech Tagging in Noisy and Esoteric Domains with a Syntactic-Semantic Bayesian HMM
10. HianChye Koh, Gerald Tan, Data Mining Applications in Healthcare
11. Line Eikvil, Optical Character recognition, December 1993
12. Anjali Ganesh Jivani, A comparative study of Stemming Algorithms. November 2011
13. K.M Arif Aziz, Evaluating Model Transformation Technologies, May 2011
14. Namrata H.S Bamrah, B.S. Satpute, Pramod Patil, Web Forum Crawling Techniques, January 2014

1. https://www.cs.umd.edu/~nau/cmsc421/part-of-speech-tagging.pdf [↑](#footnote-ref-1)