IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-11, Nov 2018 E-ISSN: 2347-2693

Model to Model Transformation for Declarative Models

Smita Agarwal[1], S. Dixit[2], Alok Aggarwal[3*]

[1]Research Scholar, Department of Computer Science, Mewar University, Chittorgarh (Raj), India
[2]Department of Computer Science, Mewar University, Chittorgarh (Raj), India
[3]School of Computer Science, University of Petroleum & Energy Studies, Dehradun, India

[*]*Corresponding Author: alok289@yahoo.com, Tel.: 7906230838*

Available online at: www.ijcseonline.org

Accepted: 15/Nov/2018, Published: 30/Nov/2018

***Abstract*—**In Model Driven Architecture (MDA), model and meta-model are the primary artifacts. In this work, a detailed analysis of the existing meta-model-based transformation tools is done for the declarative model using an exhaustive criterion. The evaluation of the eleven chosen tools, which are open source and has download page available using search engine like Google Scholar and Github, is analyzed; like UML–RSDS, Tefkat, JTL, PTL etc. Analysis is performed over fourteen different parameters like language, model query, type of transformation, compatibility, cardinality etc. Results show that all selected tools produce platform specific target model which mostly transform PSM to PSM and none produces platform independent target model transforming a PSM into PIM.

***Keywords*—**Model Driven Re-engineering, Model Transformation, Declarative User Interface, Transformation tools

I. INTRODUCTION

This decade of 21st century has witnessed the rapid growth and use of mobile devices. This has led to rise in demand for the access of the web application from the mobile devices. This involved re-engineering user interface of web application for mobile application. In classical software re-engineering processes, graphical user interfaces cannot be reused across development platforms.

The software re-engineering based on Model Driven Architecture defined by OMG (Object Management Group) is Model Driven Re-Engineering is emerging area of interest for researchers. The Re-engineering process involves Model Driven Reverse Engineering the source code for declarative user interface to obtain meta-model [1]. The Model Transformation is applied to transform meta-model into platform specific models for a given platform (known as PSMs). The techniques used are essentially modeling techniques and model transformation techniques. Over the last few years, a large number of Model Transformation Tools are available in the market.

In Model to Model Transformation, models and Meta – Models serve as primary artifacts. These models are Platform Specific Models (PSMs). These models can be transformed either into another platform specific model (PSMs) or into Platform Independent Models (PIMs). These tools can be used to transform, merge, compare, and verify models and meta-models.

In this work a detailed analysis of the existing meta-model-based transformation tools is done for the declarative model using an exhaustive criterion. The evaluation of the eleven chosen tools, which are open source and has download page available using search engine like Google Scholar and Github, is analyzed; like UML–RSDS, Tefkat, JTL, PTL etc. Analysis is performed over fourteen different parameters like language, model query, type of transformation, compatibility, cardinality etc. Results show that all selected tools produce domain specific target model which mostly transform PIM to PSM and none produces domain independent target model transforming a PSM into PIM. Rest of the paper is organized as follows. Section 2 outlines the research methodology, selection of tools, and detailed study of tools capabilities and identification of comparison parameters. Section 3 deals in evaluative and comparative study by applying comparison parameters on the selected tools. Section 4 discusses the conclusion and future scope of work.

II. RESEARCH METHODOLOGY

The main objective of this paper to identify the gap in the existing in the meta-model-based transformation tools for the Model to Model transformation for the declarative models. The research methodology starts with the selection of tools to be considered for the purpose of study, identification of comprehensive and extensive criterion for evaluation and comparison, and detailed analysis and discussion to identify the gap in the current tools and finally conclusion.

1. Selection of Tools

For the purpose of our study, we identified mostly those tools which are open source and has download page available using search engine like Google Scholar and Github. The literature and published surveys on these tools were studied and reviewed. We identified and finalized 11 such tools for the study as follows:

1. UML–RSDS(UML Reactive System Development Support)- Lano *et al*.[2] describes relation between the two models by expressing when a change in a model, what changes are required to be made in another model to preserve the truth of the constraints.
2. Tefkat[3] - The tool is declarative, logic-based, and implements transformation that written in Tefkat language where the output is XML Schema-based model.
3. JTL (Janus Transformation Language)[4] -It is a bidirectional model transformation language, embedded in the Eclipse platform, specifically designed to support non-bijective transformations and change propagation.
4. PTL(Prolog-based Transformation)- Almendros-Jiménez et al. [5] proposed hybrid Language in which ATL-style rules are joined with logic rules for defining transformations. ATL-style rules are utilized to characterize mappings from source models to target models while logic rules are utilized as helpers.
5. ModTransf – Bonde et al.[6] developed a Model Transformation engine to apply on UML profiles to generate System Transaction Models for Intensive Signal Processing (ISP) on System on Chip(SoC) Platforms.
6. Echo –Macedo et al.[7] proposed a tool that simplifies the task of keeping all models in a software project consistent, both with their meta-models and among themselves by automating inconsistency detection and repair using a solver based engine.
7. QVTR-XSLT - QVTR –XSLT tool [8] that checks the formalized semantics of QVT Relations, a standard language to specify bidirectional model transformations proposed by the OMG.
8. ModelMorf- Reddy et al.[9] proposed a tool ModelMorf that fully supports the QVTr language and uses OCL to specify templates, and when and where conditions are in relations.
9. MediniQVT-is a tool that implements the Query/ View/ Transformation (QVT) Relations specification shown the textual concrete syntax of the Relations language defined by OMG for model-to model transformations.
10. PETE (Eclipse Prolog EMF Transformation Engine) [10] is a tool based on the EMF Ecore framework that supports the model transformations using a declarative, rule-based description of transformation operations.
11. TXL–TXL is a general-purpose language tool for implementing efficient, scalable model transformations.

2. Comparison Parameters

1. Language – This parameter consider the programming language used to develop the tool to identify the most popular language deemed fit for developing the tool.
2. Modeling Language –This parameter probes into the modeling language [11] supported by the tool. This could be either domain specific (DSM) or general (GPM).
3. Meta-Modeling Language – This parameter considers the meta-modeling language [11] that is adhered by input and output models in a transformation. The Meta-Object Facility (MOF) is the OMG standard for defining meta-model. Other includes Ecore from Eclipse Modeling Frame work and Kernel Meta-Model (KM3) is an extension of Ecore for textual representation.
4. Model Query –It's an essential parameter for choosing and fetching the model elements from the models [12].
5. Compatibility with Standards – For the interoperability and migration of Models between the tools, the tools must support the well known standards and languages such as XMI, QVT Language by OMG and OCL.
6. Model Transformation Language Syntax – This feature of the tool provides the information about the syntax of the modeling language whether it is textual or graphical or both.
7. Target Model –The target model of M2M transformation tool could be conservative or destructive. In conservative transformation, the source model is preserved while producing the new target model while in the destructive transformation; the source model is modified to give the target model. Conservative tools are more suitable for endogenous transformation and destructive tools are suitable for exogenous transformation [11].
8. Cardinality –It defines the number of models that can be managed at input and output of the tool [11]. Tools can support 1-1, 1-N, N-1 and N-N model.
9. Type of Transformation - This parameter looks into whether the source and target model belong to same meta-model i.e. endogenous transformation or different meta-model i.e. exogenous transformation [11].
10. Directions-The Model Transformations can be unidirectional, bi- directional and multi –directional.
11. Verification of correctness and completeness of the transformations – The transformation model must clearly define the termination condition [13] and the output obtained from the transformation should be unique.
12. Traceability – This parameter establishes the relation between the elements of source model and target model which is instrumental in evaluating the transformations. This can auto generated by the tool
13. Creating/Retrieving/Updating/Deleting transformations (CRUD) - Any transformation tool must have ability to

create, retrieve, modify and drop rules for transformation.

14. Level of automation –This parameter identifies the level of automation involved in the transformation [14]. Although the manual intervention during transformation gives the maximum control but requires in-depth understanding of transformation.
15. Re-usability the transformation Model – Using inheritance mechanism of object oriented language like composition and decomposition, one can reutilize the transformation rules and functions defined for one transformation from source platform to target platform to other set of source and target platform.

Table 1. Summary of Comparison Parameters

S.No.	Parameter	Value
1.	Language	Java
		ASP
		Turing
2.	Modeling Language	Domain Specific
		General
3	Meta-Modeling Language	Meta-Object Facility
		Ecore
		Kernel Meta-Model
		Other
4.	Model Query	Tool has modeling query feature
		Tool do not have modeling query feature
5.	Compatibility with Standards	Tool supports XMI
		Query/View/Transformation (QVT) Language
		Tool supports OCL expression
6	Model Transformation Language Syntax	Textual syntax is used in tool
		Graphical syntax is used in tool
7.	Target Model	Constructive
		Destructive
8.	Cardinality	1 to1
		1 to N
		N to 1
		N to N
9.	Type of Transformation	Exogenous
		Endogenous
10.	Directions	Multi Directional Transformation
		Bi- Directional Transformation
		Uni- Directional Transformation
11.	Verification –	Syntactic Correctness
		Semantic Correctness
		Completeness
		Robustness
12.	Traceability	Automatic
		User-defined
13.	CRUD	Retrieve transformation
		Create transformation
		Update transformation
		Delete transformation
14.	Level of Automation	Manual
		Semi-Automatic
		Automatic
		Decomposition

III. EVALUATIVE AND COMPARATIVE STUDY

In this section, we assessed the selected tools based on above identified criterion and present a comprehensive and comparative result of the selected tools.

1. Language - The selected tools were written mostly in Java (82%) and rest used either ASP or Turing. It is shown in Table 2 and figure 1.

Table2. Language Percentage

Parameter Value	No of tools	Percentage
Java	9	82
ASP	1	9
Turing	1	9

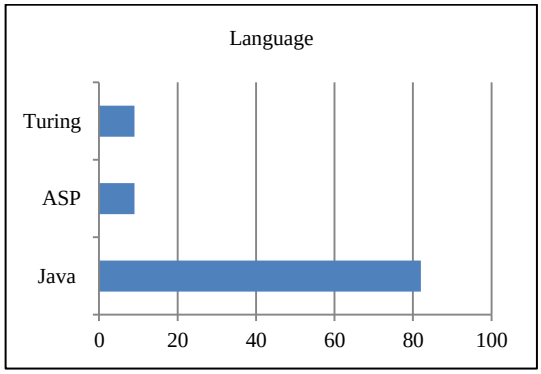

Figure 1. Language Percentage

2. Modeling Language - Among the general purpose language, UML is the preferred modeling language. Most of the tools don't any modeling language and specify their own modeling language. It is shown in Table 3 and figure 2.

Table 3.Modeling Language Percentage

Parameter Value	Number of tools	Percentage
Domain Specific	1	9
General	0	0
NA	10	91

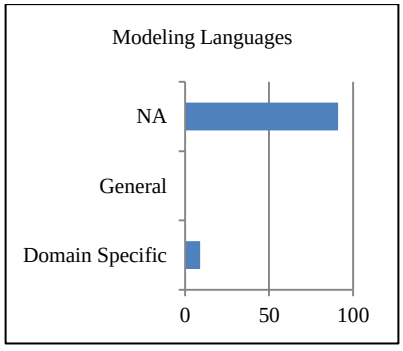

Figure 2Modeling Language Percentage

3. Meta-Modeling Language- The Meta Modeling Language Ecore is supported in 55% of the tools like

Tefkat, JTL, PTL, Echo, MediniQVT, PETE whereas Tools like UML -RSDS supports Meta-Object Facility. It is shown in Table 4 and figure 3.

Table 4. Meta-Modeling Language Percentage

Parameter Value	Number of tools	Percentage
Meta-Object Facility	4	36
Ecore	6	55
Kernel Meta-Model	0	0
Others	2	18

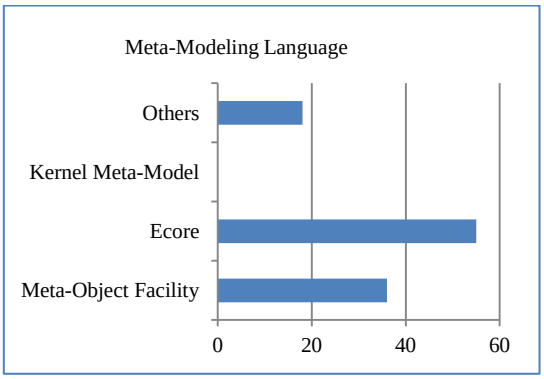

Figure 3. Meta-Modeling Language Percentage

4. Model Query -Querying is required for analysis, evaluation and reporting about Model. 82%of the tools don't have support for querying. It is shown in Table 5 and figure 4.

Table 5.Model Query Percentage

Parameter Value	Number of tools	Percentage
Yes	2	18
No	9	82

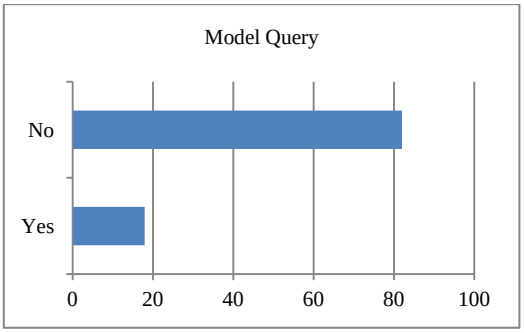

Figure 4. Model Query Percentage

5. Compatibility with Standards -All the selected tools supports XMI and only 8% supports OCL. It is shown in Table 6 and Figure 5.

Table 6.Compatibility with Standards Percentage

Parameter Value	Number of tools	Percentage
XMI	11	100
OCL	8	73

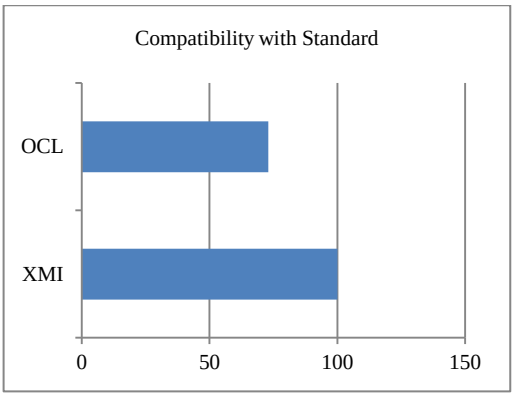

Figure 5. Compatibility with Standards Percentage

6. Model Transformation Language Syntax -73% tools such as Tefkat, PTL, ModTransf, and Echo provide textual syntax for their modeling language whereas QVTR-XSLT provides only graphical syntax and tools like UML-RSDS and JTL provide both textual and graphical syntax. It is shown in Table 7 and Figure 6.

Table 7.Model Transformation Language Syntax Percentage

Parameter Value	Number of tools	Percentage
Graphical	1	9
Textual	8	73
Both	2	18

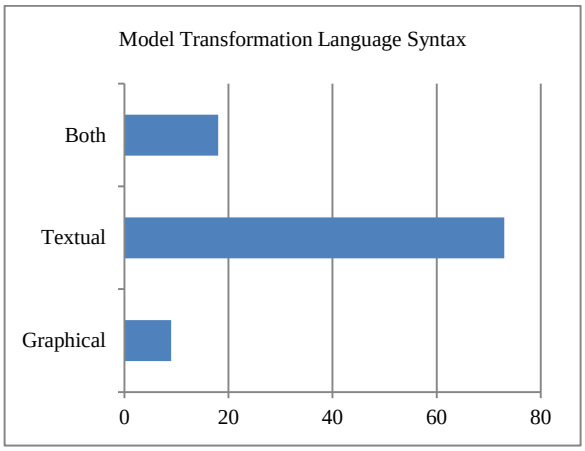

Figure 6. Model Transformation Language Syntax Percentage

7. Target Model -45 % of the tools can produce target model that can be both conservative as well as destructive whereas 36% of the tools can produce only conservative target model. It is shown in Table 8 and figure 7.

Table 8.Target Mode lPercentage

Parameter Value	Number of tools	Percentage
Destructive	1	9
Conservative	4	36
NA	1	9
Both	5	45

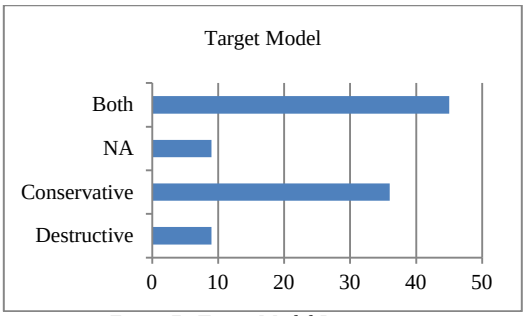

Figure 7. Target Model Percentage

8. Cardinality -82% of the tools supports 1-1 cardinality i.e. they take one model as source model and gives out only one model as target model. Tools like UML-RSDS, Tefkat, JTL, PTL and TXL supports all the cardinalities. It is shown in Table 9 and figure 8.

Table 9.Cardinality Percentage

Parameter Value	Number of tools	Percentage
1 to1	9	82
1 to N	6	55
N to 1	6	55
N to N	7	63
NA	1	9

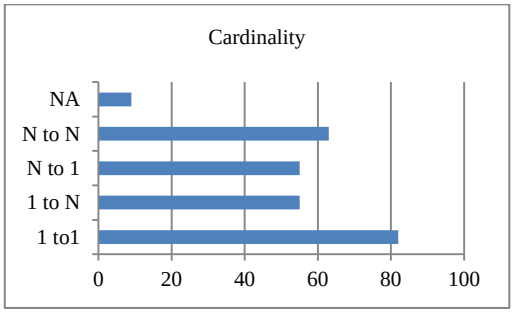

Figure 8. Cardinality Percentage

9. Type of Transformation - The translation of a platform-independent UML model into a platform-specific Java model is exogenous. 91% of the tools selected are exogenous. It is shown in Table 8 and figure 7.

Table 10. Type of Transformation Percentage

Parameter Value	Number of tools	Percentage
Exogenous	10	91
Endogenous	7	64
NA	1	9

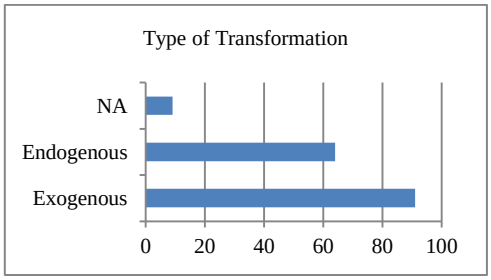

Figure 9 Type of Transformation Percentage

10. Directions -All the selected tool are unidirectional whereas only 45 % are bi directional. It is shown in Table 11 and figure 10.

Table 11. Directions Percentage

Parameter Value	Number of tools	Percentage
Multi Directional	1	9
Bi- Directional	5	45
Uni- Directional	11	100

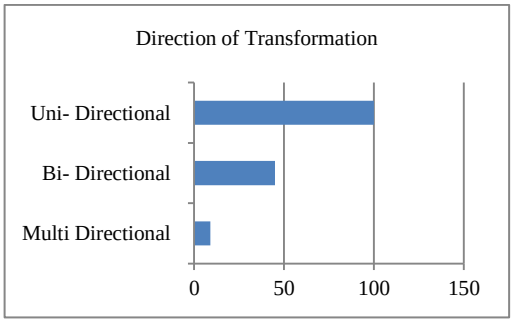

Figure 10. Directions Percentage

11. Verification of correctness and completeness of the transformations -Only 55% of the tools support for syntactic correctness where as 27% of the tools support for semantic correctness. It is shown in Table 12 and figure 11.

Table 12. Verification Percentage

Parameter Value	Number of tools	Percentage
Syntactic	6	55
Semantic	3	27
Completeness	1	9
Robustness	0	0
NA	1	9
No Support	4	36

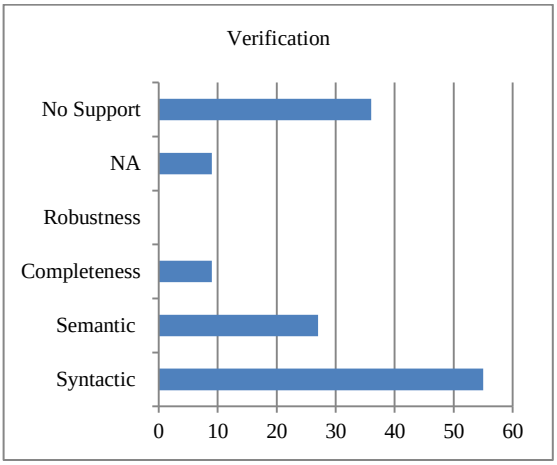

Figure 11. Verification Percentage

12. Traceability -55% automatically generated traceability information whereas 36 % traceability links can also be defined and generated by the user. 27% of the tools

don't have support for traceability information. It is shown in Table 13 and figure 12.

Table 13. Traceability Percentage

Parameter Value	Number of tools	Percentage
Automatic	6	55
User Defined	4	36
No Support	3	27

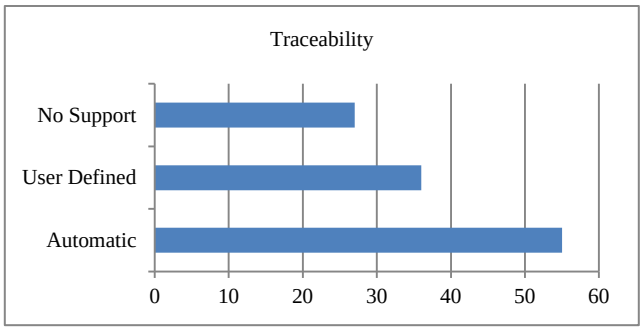

Figure 12. Traceability Percentage

13. Creating/Retrieving/Updating/Deleting transformations (CRUD) - All the selected tools support Creating, Retrieving, Updating and Deleting transformations. It is shown in Table 14 and figure 13.

Table 14. CRUD Percentage

Parameter Value	Number of tools	Percentage
Retrieve	11	100
Create	11	100
Update	11	100
Delete	11	100

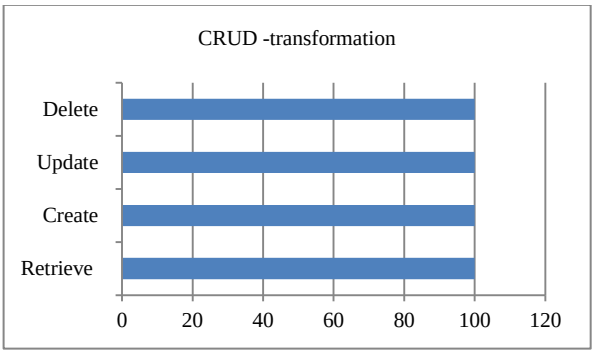

Figure 13. CRUD Percentage

14. Level of automation -All the selected tools are semi - automatic i.e. having partial manual control and partial automatic. It is shown in Table 15 and figure 14.

Table 15. Language Percentage

Parameter Value	Number of tools	Percentage
Manual	0	0
Semi-Automatic	11	100
Automatic	0	0

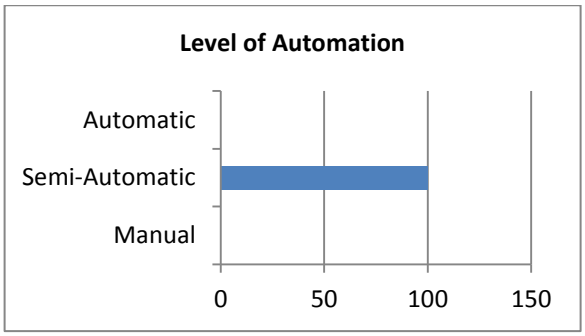

Figure 14. CRUD Percentage

IV. CONCLUSION AND FUTURE SCOPE

In this research paper, we selected, studied and analyzed 11 model transformation tools for Declarative Model meticulously with respect to exhaustive and extensive criteria. Our study and analysis shows that Java is the preferred programming language and UML is the preferred modeling language for the development of model transformation tools for declarative models. Most of the tools are exogenous i.e. they generate Platform Specific Model. The tools though don't support Querying Model but has full compatibly with standards like XMI. Most of the tools take single model as source model and gives out only one model as target model. All of them can create, retrieve, modify and drop transformations. The tools are semi-automatic i.e. they do have some kind of manual intervention while generating target model. Most of the tools have capability of generating traceability information automatically. All of the tools produce platform specific target model which mostly transform PSM to PSM. None of the tool produces domain independent target model transforming a PSM to PIM. Our future work will explore the possibility of tool to produce domain independent target model transforming a PSM into PIM.

REFERENCES

[1] S. Agarwal and A. Agarwal. "*Model driven reverse engineering of user interface — A comparative study of static and dynamic model generation tools*."in International Conference on Parallel, Distributed and Grid Computing.pp 268 – 273, 2014.

[2] K. Lano, S. Kolahdouz-Rahimi. "*Specification and verification of model transformations using UML-RSDS."*International Conference on Integrated Formal Methods (pp. 199-214). Berlin, Heidelberg: Springer., 2010.

[3] M. Lawley, J. Steel. (2005). "Practical declarative model transformation with Tefkat". International Conference on Model Driven Engineering Languages and Systems (pp. 139-150). Berlin, Heidelberg.: Springer. 2005.

[4] D. Cicchetti, D. Di Ruscio, R. Eramo, & A. Pierantonio. " *JTL: a bidirectional and change propagating transformation language*". International Conference on Software Language Engineering (pp. 183-202). Berlin: Springer. 2010

[5] J.M. Almendros-Jiménez, , L. Iribarne, J. López-Fernández, and Á. Mora-Segura. "*PTL: A model transformation language*

based on logic programming." Journal of Logical and Algebraic Methods in Programming *85*, 332-366. 2016

[6] L.Bondé, C. Dumoulin, and J.L. Dekeyser. "*Metamodels and MDA transformations for embedded systems*." Advances in design and specification languages for SoCs, Springer, 89-105.2005.

[7] N. Macedo, T. Guimaraes, A. Cunha. "Model repair and transformation with Echo." 28th IEEE/ACM International Conference on Automated Software Engineering. IEEE Press, 694-697, 2013.

[8] D. Li,X. Li, V. Stolz. "QVT-based model transformation using XSLT." ACM SIGSOFT Software Engineering Notes 36, no. 1, 1-8, 2011.

[9] S. Reddy, R. Venkatesh, A. Zahid. "A relational approach to model transformation using QVT Relations." TATA Research Development and Design Centre, 1-15, 2006.

[10] B. Schätz,"Formalization and rule-based transformation of EMF Ecore-based models." International Conference on Software Language Engineering,. Berlin, Heidelberg: Springer, 227-244.2008.

[11] M.Brambilla, J. Cabot, and M.Wimmer. "*Model-driven software engineering in practice.*" 1-182. Synthesis Lectures on Software Engineering 1, no. 1, 2012.

[12] L. Lúcio. "Model transformation intents and their properties." Software & systems modeling 15, no. 3 , 647-684.2016.

[13] Prince Singha, Aditya, Kunal Dubey, Jagadeeswararao Palli, "Toolkit for Web Development Based on Web Based Information System," Isroset-Journal (IJSRCSE), 6, no. 5, pp.1-5. 2018..

[14] Shubham, Deepak Chahal, LatikaKharb, "Security for Digital Payments: An Update," Journal (IJSRNSC), 6, no. 5 , pp. 51-54. 2018.

Authors Profile

Smita Agarwal has earned Bachelor's degree of Electronics & and Master's degree of Information Technology in 1998 &2001 respectively from University of Delhi. She is currently pursuing Ph.D. in Computer Science & Engineering.She has seven years of industry experience.

Sarvottam Dixit did his Ph.D. in Physics (Material Science) from Dr. B.R. Ambedkar University Agra in 1990 and completed Post-Doctorate work from Tata institute of fundamental research (TIFR) Mumbai funded by DST in 1996 and M.E. in CSE. Current he is working as advisor to Chancellor and Professor in Faculty of Engineering in Mewar University. Earlier he was Pro-VC and acting Vice Chancellor Shri Venkateshwara University Gajurala (UP) and Venkateshwara Open University Arunachal Pradesh.

Alok Aggarwal received his bachelors' and masters' degrees in Computer Science& Engineering in 1995 and 2001 respectively and his PhD degree in Engineering from IITRoorkee, Roorkee, India in 2010. He has academic experience of 18 years, industry experience of 4 years and research experience of 5 years. He has contributed more than 150 research contributions in different journals and conference proceedings. Currently he is working with University of Petroleum & Energy Studies, Dehradun, India as Professor in CSE department

