mROSE: To Determine Tool Selection and to Understand Model-Driven Software Evolution

Madhavi Karanam
Computer Science and Engineering Department, Nalgonda Institute of Technology and Science, Nalgonda, Andhra Pradesh, India

Ananda Rao Akepogu
Computer Science and Engineering Department, JNTUCEA, Jawaharlal Nehru Technological University, Anantapur, Andhra Pradesh, India

ABSTRACT
Growing interest in the model driven approaches has largely increased the number of tools into the model driven development environment. Previous research has shown that the stakeholders often do not use or know all of the tools available in the model evolution environment that they regularly use. The common solution to this problem is to provide a means to search through passive help documents. However, this approach requires a stakeholder to be able to express their desires in a form understood by search engine. So, choosing the right tool for MoDSE tasks has become difficult because of the diverse nature of numerous tools available. To overcome this limitation, this paper aims to present a prototypical recommendation system, named mROSE, to provide timely and useful recommendations to stakeholders. Two empirical studies were conducted to investigate if mROSE helps or hinders the stakeholders in MoDSE, if so under what conditions. First one was longitudinal user study and the second one was a laboratory user study. Performance of mROSE was also evaluated by using some of the existing metrics. These studies confirmed that mROSE can help stakeholders to choose right tools more efficiently and users liked the idea of having a recommendation system for MoDSE environment, like mROSE. These studies also revealed future directions that would improve the functionality of mROSE.

Keywords
Model Driven Approach, Model-driven Software Evolution, MDA Tools, UML Tools, and Recommendation Systems for Software Engineering.

1. INTRODUCTION
Model Driven approaches have become a new software trend in software development process. MDE needs a new paradigm for software evolution which is known as MoDSE [1]. Many CASE tools have evolved due to wide usage of model driven approaches. Tools are used for different activities of model driven evolution such as model transformation, model mapping etc. So, here the question arises “how do you choose the right tool?”

The contradictory experiences with MDA and UML tools appear puzzling and difficult to interpret. Tools do much work in model driven approaches [2]. So, it is very much essential to choose the tools carefully. Basically, an MDA tool is a tool used to develop, interpret, compare, align, measure, verify, transform, models or meta models. In MDA approach we have essentially two kinds of models: *initial models* are created manually by human agents while *derived models* are created automatically by programs. For example, an analyst may create a UML initial model from its observation of some loose business situation while a Java model may be automatically derived from this UML model by a Model transformation operation which can be done with the help of automated tools. These tools perform more than one of the desired functions. For example, some creation tools may also have transformation and test capabilities. There are other tools that are solely for creation, solely for graphical presentation, solely for transformation, etc. There is an increasing need for more disciplined techniques and engineering tools to support a wide range of model evolution activities, including model-driven software evolution, model differencing, model comparison, model refactoring, model consistency, model versioning and merging, and (co-)evolution of models.

The research presented here suggests that by shifting the focus from specific outcome expectations, it may be able to make sense of the apparently inconsistent findings. This paper presents a recommendation system ‘mROSE’ for tool selection which is conceptualized as a form of stakeholder[1] interests and/or concerns. Such a perspective allows users to anticipate, explain, and evaluate different experiences and consequences following the introduction and intention of the tools. Recommendation system is a software application that aims to support users in their decision-making while interacting with large information spaces [3]. They recommend items of interest to users based on preferences they have expressed, either explicitly or implicitly. The ever-expanding volume and increasing complexity of information has therefore made such systems essential tools for users in a variety of information seeking activities. Recommendation system helps to overcome the information overload problem by exposing users to the most interesting items, and by offering novelty, surprise, and relevance.

However, there has been no systematic formulation of MDA tools for stakeholder concerns in MoDSE. The stakeholder concerns in MoDSE are identified as model mapping, model merging, model integration, model transformation, model consistency etc, [1, 2, 4, 5]. To gain the knowledge about the tools stakeholders definitely explore the existing tools in the literature. Much of the literature on these tools has intended to focus on discussion forums, panels, comparison strategies and frameworks. Discussion forum and/or panel are where the users can share their ideas and answering the questions of the audience [7, 8]. Framework determines the comparison strategies for features of tools of same category under a uniform platform [5, 9, 10, 11, 12]. Thus, the proposed system is intended to provide the recommendations by bringing stakeholder concerns and tools together.

The remainder of the paper is structured as follows. Section 2 reviews the related work. Proposing a Recommendation

[1] Stakeholder and user terms are used interchangeably in this paper.

System in Model-Driven Software Evolution Context is the primary contribution of this paper which is discussed in the section 3. Evaluation of mROSE is presented in the section 4. Section 5 closes with conclusions and future work.

2. RELATED WORK

This section outlines the comparison strategies, forums, panels for MDA and UML tools and also about the recommendation systems for software engineering. Many more recommendation systems are available for software engineering. But few recommendation systems are discussed here.

2.1 Tools

Integrated constrain support in MDA tools evaluated in [13]. The different tools were classified in Categories like CASE tools, MDA specific tools, MDD methods and OCL tools. MDA-tools considered in the classification are closest to the MDA standard. Only few tools such as Poseidon, Rational Rose, Magic Draw, Objecteering/UML , Together ,ArcStyler, OptimalJ and AndroMDA etc, have selected for the comparison and evaluation purpose. The support of current tools regarding the automatic generation of the code required to enforce the Integrated Constraints specified in a PIM also surveyed. The main shortcomings encountered are the lack of expressivity and efficiency in integrated constraints.

A short comparison of the three MDA tools was presented in [11]. It was focused on the concepts behind the tools as well as how to use them. The tools considered for comparison are ArchitectureWare, AndroMDA, and openMDX. The comparison strategy presented might be useful for evaluation purposes to find out about the differences in approach, features and concepts which are to be considered as the implementation of OMG's MDA specification.

Computer Aided Software Engineering tool Community is an open access web application. In which the different categories of the CASE tools like MDA, UML, reverse engineering, agile modeling etc. are listed. The key functions, external links, and rating of the tools are provided. modelbased.net forum [7] is dedicated to tools and information related to model-driven system development, aiming at supporting OMG's vision MDA are provided. This website provides the over view and the resource links of MDA oriented tools, UML, MOF, and Model transformation tools.

MDA tools are categorized in three ways in [12]. The first, whether the tool is open source or commercial, will help to choose a tool that is right for the culture of an organization, among other things. The second, whether the tool offers a partial or complete MDA solution, will helps in such considerations as cost, quality, and flexibility. The final category, whether the tool generates code from the model or executes the model. It is not mentioned what are the tools that fits in to the specified categories. IBM Rational software has several products that support MDA and Model Driven Development (MDD) in varying capacities [11]. These tools fall into three basic categories such as general-purpose, domain-specific, and supporting. For example Rational Software Architect is in general-purpose category, IBM Rational Systems Developer in domain-specific category, and IBM WebSphere Business Modeler in the supporting category. The usage of these tools in MDA was described. Here only IBM products are considered for categorization. Above mentioned forum, panel and community are only the information content of the tools which do not provide the auto suggestions about the tools for different activities of MoDSE.

2.2 Recommendation Systems for Software Engineering

The Strathcona system [14] retrieves relevant source code examples to help developers use frameworks effectively. For example, a developer who's trying to figure out how to change the status bar in the Eclipse IDE can highlight the partially complete code (the context) and ask Strathcona for similar examples. Strathcona extracts a set of structural facts from the code fragment. Strathcona uses PostgreSQL queries to search for occurrences of each fact in a code repository. Next, it uses a set of heuristics to decide on the best examples, which it orders according to how many heuristics select them. It returns the top 10 examples, displaying them in two formats - a structural overview diagram and highlighted source code. Developers can also view a rationale for a proposed example. A prototype is also available in.

Dhruv is a Recommendation System [15] which recommends people and artifacts relevant to a bug report. It operates chiefly in the open source community, which interacts heavily via the Web. Using a three-layer model of community (developers, users, and contributors), content (code, bug reports, and forum messages), and interactions between these, Dhruv constructs a according to the similarity between a bug report and the terms contained in the object and its metadata. Finding the right software experts to consult can be difficult, especially when they're geographically distributed. Expertise Browser [16] is a tool that recommends people by detecting past changes to a given code location or document. It assumes that developers who changed a method have expertise in it.

ParseWeb [17] recommends sequences of method calls starting from an available object type and producing a desired object type. ParseWeb analyzes example code found on the Web to identify frequently occurring call patterns that link available object types with desired object types. Developers use the tool by specifying available and desired object types and requesting recommendations.

To date, most Recommendation Systems for software engineering (RSSE) have focused on recommendations related to software development artifacts, particularly source code. RSSEs typically recommend code—to look at, change, or reuse. However, recommendations could address many other aspects of software development such as quality measures, tools, project management, and people could support an ever-widening array of software engineering tasks. Recommendation system might use activity logs to deduce questions developers ask, and then coach them automatically on appropriate, possibly unfamiliar tools or features to answer those questions more efficiently [18].

Thus, the proposed system provides the recommendations for tool selection in the context of Model-Driven Software Evolution. It is different from the above mentioned forums and communities. The proposed system brings tools, MoDSE concerns, and stakeholders together and it is described in the next section.

3. PROPOSED SYSTEM

Many stakeholders spend substantial effort in finding out appropriate tool among the huge number, to perform activities in MoDSE. The major goal of this paper is to bring stakeholder, tools and concerns together and assist in selecting appropriate tool suitable to accomplish concerns of interest. The following sections demonstrate the design issues and different components of the proposed System.

Recommendation System for Software Engineering (RSSE) is defined in as “a software application that provides information items estimated to be valuable for a software engineering task in a given context”. mROSE, the proposed recommendation system, generates useful tool recommendations for users to understand evolution of the models in the Model-Driven Software Evolution. The design dimensions, characteristics and other concepts of RSSE are available in [17] and the design dimensions like nature of the context, recommendation engine, and output modes are considered for the proposed mROSE.

3.1 Components of the mROSE

mROSE consists of four major components: Basic Mode, Expert Mode, Tool Comparison and Knowledge Base. Based on the user profile two modes of interaction are provided. Tool Comparison is responsible for generating recommendations. Knowledge Base is the information space where user can explore detailed information about concerns and tools. mROSE can also be connected to the web for further expansion. Details of each component are provided in the rest of this section.

3.1.1 Basic Mode

This mode of operation is for a beginner who is not well versed with the terminology associated with MoDSE and MDA tools. Basic Mode (Fig.1a) lists all the possible concerns of MoDSE and allows the user to select the interested concerns while providing an option to learn more about these concerns. After selecting interested concerns, user can click the button titled ‘*Get tools for Selected Concerns’* to list all tools that support user’s selection. Further user can select few suggested tools and use the button titled ‘*Compare Selected Tools’* to enable Tool Comparison component that generates comparison summary and recommendations based on the number of concerns selected by the user and number of concerns satisfied by the tool.

3.1.2 Expert Mode

Expert Mode allows the user who has the knowledge about the MoDSE and MDA tools. Interface (Fig.1b) allows the user to search for interested concerns with search option and auto suggestions. It populates the list of the selected concerns. Recommendations are generated for the selected concerns. Comparison summary of the selected recommended tools are presented from which identifying the right tool becomes easy for a user.

3.1.3 Tool Comparison

This component lists the tools which are useful to understand the evolution of the models in MoDSE. Tools are categorized as MDA and UML tools, and also as Non-Commercial and Commercial tools. Tool Comparison generates the comparison summary of similarities and dissimilarities of the selected tools. Tool Comparison component implicitly works as a recommendation engine which generates the recommendations for the user interest. *Comparison* and *Recommendation* buttons (Fig 1c and Fig 1d) are used to view the comparison summary and the recommendations based on the user selected concerns, tools and the number of concerns satisfied by the selected tool.

3.1.4 Knowledge Base

Knowledge Base is an information repository where the complete content of the tools, description of various concerns of MoDSE and other related information about Models, UML diagrams etc. are available. User can capture and gain the knowledge about MoDSE from the search option (Fig. 1e). mROSE can be connected to web thru this component helping user to explore more information and also to overcome the limited information space of the repository. Users can leave feedback about mROSE using *Post Comment* option and (Fig. 1f.)

3.1.5 Algorithm

Recommendation engine algorithm is constructed by determining various tool similarities and algorithm used here is available in the literature as Item-Based schemas [19]. The following symbols are used in the algorithm.

N = number of recommendations that need to be generated for a particular user

n = number of distinct users

m = number of distinct tools in a particular transaction, user selecting the concerns, selecting the appropriate tools considered as a transaction.

k = number of similar tools

Input to this algorithm is the model *M*, an $m\times 1$ vector *U* that stores items that have been selected by the active user. Active user’s information in vector *U is* encoded by setting $Ui = 1$ if the user has selected *i* th tool and zero otherwise. Output of this algorithm is an $m\times 1$ vector *x* whose nonzero entries correspond to the number of items to be recommended (*N*). Weight of these nonzero entries represents a measure of the *recommendation strength* and various recommendations can be ordered in non-increasing recommendation. In most cases *x* will have exactly *N* nonzero entries; however, actual number of recommendations can be less than *N* as it depends on the value of *k* used to build *M* and the number of items that have already been selected by active user.

```
**Algorithm :** RecommndGen ($M$, $U$, $N$)
$x \leftarrow MU$          (1)
**for** $j \leftarrow 1$ **to** $m$      (2)
**do**
**if** $U_i \neq 0$
**then** $x_i \leftarrow 0$
```

Fig 1: mROSE User Interface

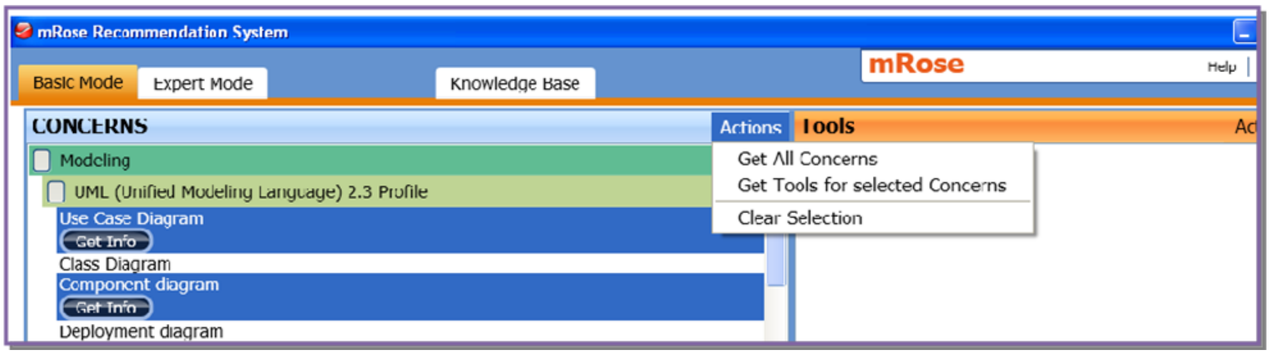


Fig 1a: Basic Mode with List of Selected Concerns and Actions

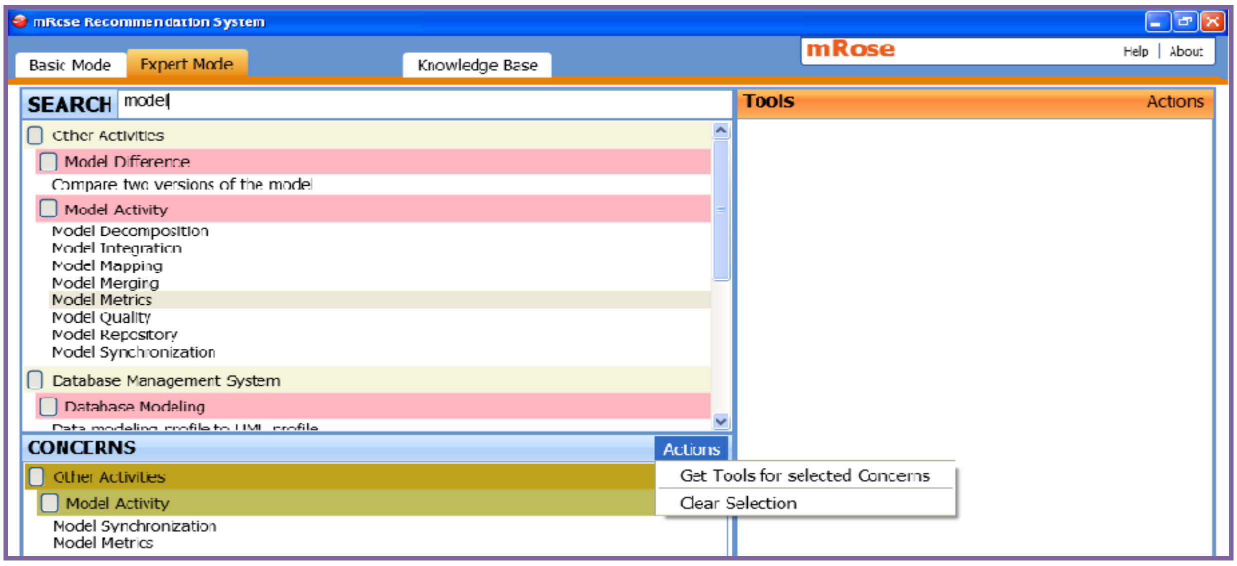
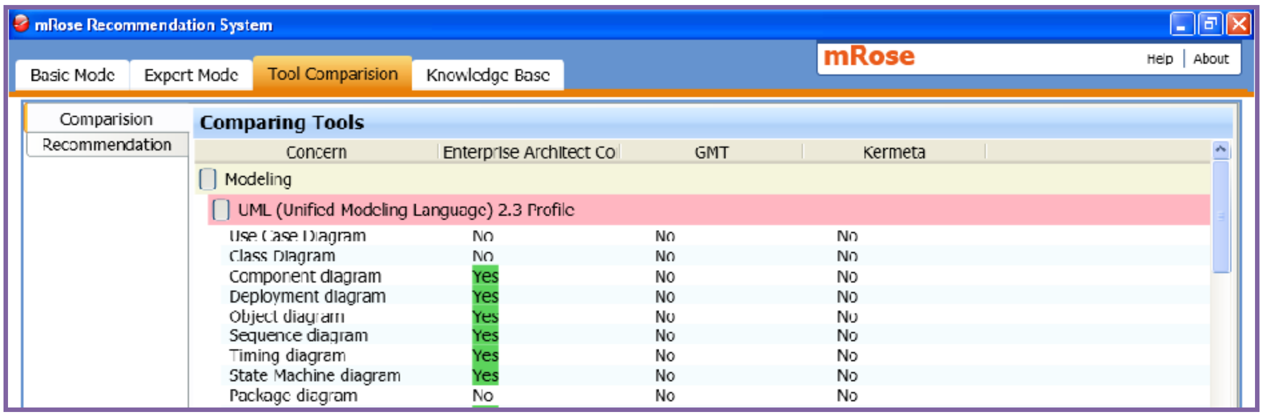
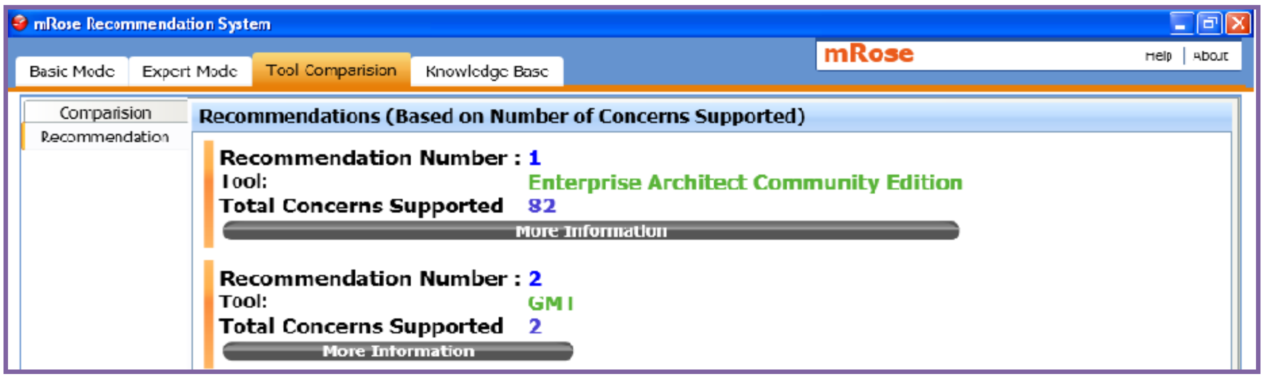


Fig 1b: Expert mode with search and auto suggestions options and Actions



Concern	Enterprise Architect Co	GMT	Kermeta
Use Case Diagram	No	No	No
Class Diagram	No	No	No
Component diagram	Yes	No	No
Deployment diagram	Yes	No	No
Object diagram	Yes	No	No
Sequence diagram	Yes	No	No
Timing diagram	Yes	No	No
State Machine diagram	Yes	No	No
Package diagram	No	No	No

Fig 1c: Comparison summary of selected tools in Tool Comparison

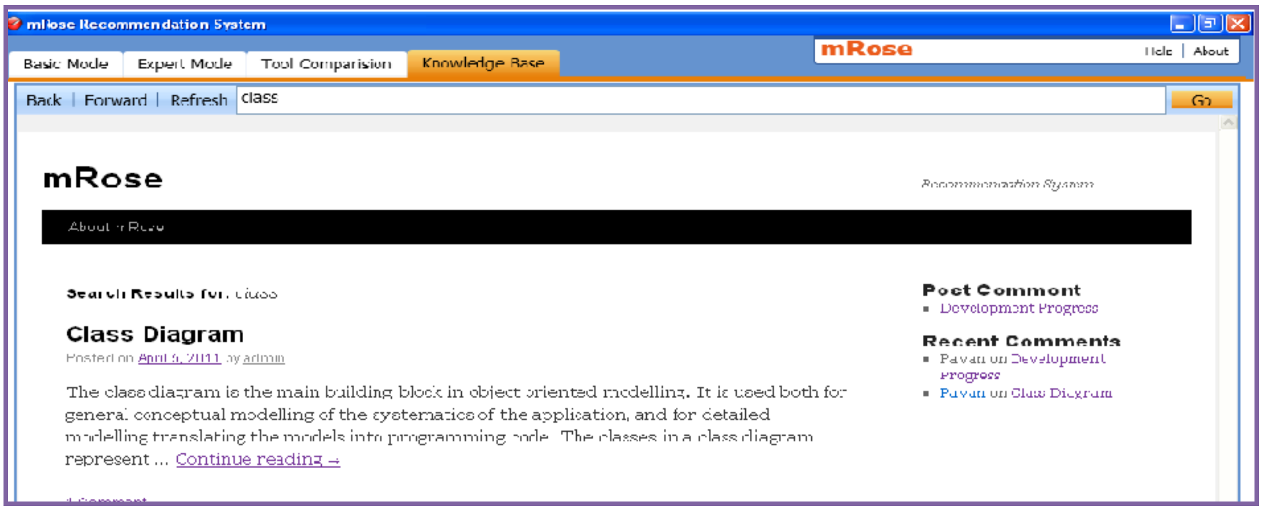


Recommendations (Based on Number of Concerns Supported)

Recommendation Number : 1
Tool: Enterprise Architect Community Edition
Total Concerns Supported 82

Recommendation Number : 2
Tool: GMT
Total Concerns Supported 2

Fig 1d: Recommendations generated in Tool Comparison



Search Results for: class

Class Diagram
Posted on April 5, 2011 by admin

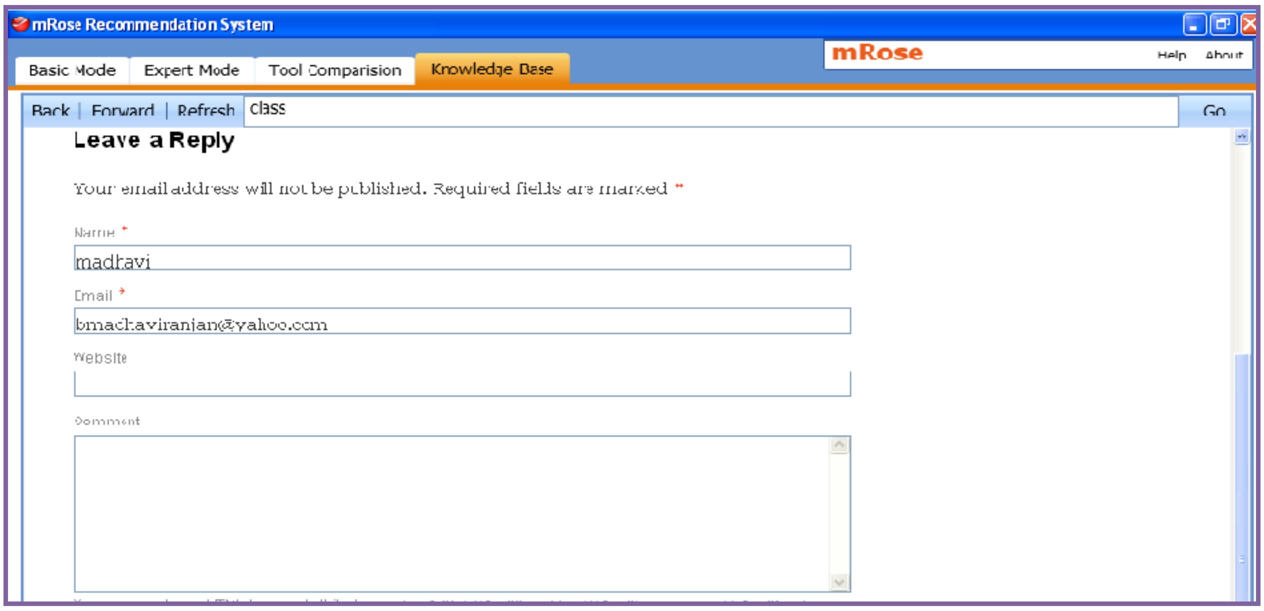
The class diagram is the main building block in object oriented modelling. It is used both for general conceptual modelling of the systematics of the application, and for detailed modelling translating the models into programming code. The classes in a class diagram represent ... Continue reading →

1 Comment

Post Comment
- Development Progress

Recent Comments
- Pavan on Development Progress
- Pavan on Class Diagram

Fig 1e: Information capturing from Knowledge Base



Leave a Reply

Your email address will not be published. Required fields are marked *

Name *	madhavi
Email *	bmadhaviranjan@yahoo.com
Website	
Comment	

Fig 1f: Feedback Form of mROSE in the Knowledge Base

for $j \leftarrow 1$ **to** m (3)

do

if $x_i \neq$ among the N largest values in x

then $x_i \leftarrow 0$

return (x)

Vector *x* is computed in three steps. First, vector *x* is computed by multiplying *M* with *U* (line 1). Note that the nonzero entries of *x* correspond to union of *k* most similar concerns for each tool that has already been selected by an active user, and weight of these entries is nothing more than the sum of these similarities. Second, the entries of *x* that correspond to tools that have already been selected by an active user are set to zero (loop at line 2). Finally, in third step, the algorithm sets to zero and all entries of *x* that have a value smaller than the *N* largest values of *x* (loop at line 3).

Comparison between the tools is computed as an ItemSimilarity by using MapReduce [20,21] framework which shields the programmer from distributed processing issues such as synchronization, data exchange, and load balancing. Existing MapReduce [21] technique is used in this research to compute pair wise similarity between the tools. This technique is selected because of its efficiency in computing pair wise similarity for large collections. In this paper pair wise tool similarity can be expressed as an inner product of concern weights. A tool t is represented as a vector W_t of concern weights $w_{c,t}$, which indicates the importance of each concern c in that tool by considering symmetric similarity measure, which is defined as follows:

$$Sim(t_i, t_j) = \sum_{c \epsilon V} w_{c,ti} \cdot w_{c,tj} \qquad (1)$$

Where $Sim(t_i, t_j)$ similarity between tools ti and tj and V is is the concern set. In this type of similarity measure, a concern will contribute to the similarity between two tools only if it has non-zero weights in both. Therefore, $c \epsilon V$can be replaced with $c \epsilon t_i \cap t_j$ in equation (1). For example, if a concern appears in tools x,y, and z, it contributes only to the similarity scores between (x,y),(x,z) and (y,z). The list of tools that contain a particular concern is exactly retrieved. Thus, processing all retrieves, entire pair wise similarity matrix can compute by summing concern contributions.

Algorithm : Compute Tool Similarity Matrix

```
1: $\forall$i, j : sim[i, j] $\Leftarrow$0
2: for all $c \epsilon V$ do
3: $r_c \Leftarrow$ retrieves(c)
4: for all $t_i, t_j \epsilon$ $r_c$ do
5: sim[i, j] $\Leftarrow$ sim[i, j] + $w_{c,ti} . w_{c,tj}$
```

Algorithm formalizes the idea: retrieves(c) denotes list of tools that contain concern c. In mROSE prototype model, information space of tools and concerns considered is small. So, this algorithm runs efficiently to compute entire similarity matrix in memory. For larger collections disk access optimization is needed, which is provided by the MapReduce [29, 30] runtime, without requiring explicit coordination. In mROSE comparison report is treated as an n×m matrix, where n is the total number of concerns and m is the concerns present in selected tools. If the concern is present in a tool, that is treated as a 'YES' if not 'NO'. Increase in number of n and m, increases the number of iterations which in turn increases computation complexity. This can be reduced by reducing the number of iterations by assuming that number of concerns fixed (n) and number of tools may vary (m).

4. mROSE EVALUATION

This section describes the validation of the proposed mROSE prototype model by longitudinal and laboratory user studies. Standard evaluation techniques that refer directly to the performance metrics [22] related to time, storage requirements and computation complexity is used to evaluate the performance of mROSE.

4.1 Longitudinal User Study

A formal evaluation of mROSE system is needed to determine whether and under what conditions this system help or hinder the users. These evaluations can also help assess whether benefits of mROSE system support the users need or not. To evaluate the value and acceptance of prototype of mROSE, a longitudinal user study has conducted with 15 participants who are academicians and research students from diverse organizations. Longitudinal User Study aimed to answer nine different tasks which consist of questions about MoDSE concerns, activities, and tools. Due to the space limitation nine tasks are not mentioned here. Since mROSE prototype model could only recommend 60 tools at the moment.

To observe the effect, efficiency and usability of mROSE, longitudinal study was divided into two phases: In First Phase, answering the nine tasks without using mROSE system; Second Phase, answering the nine tasks with the help of mROSE. During the first phase, mROSE was disabled to the participants, meaning that participants could not interact with mROSE or receive any information/knowledge to solve nine tasks. Participants have recorded the time taken (man hours) to answer each task as shown in the Table 1 and columns from P1 to P15 represent the participants and rows Tk1 to Tk9 represent task1 to task9. Values in Table 1 are time taken to answer the each task by each participant measured in minutes (man hours). Average time taken to answer each task by 15 users also computed and it is shown in Table 1.

After completion of the first phase participants remarked that answering the tasks required lot of information gathering from distinct sources like communities, forums, social networks, websites, search engines etc. So, it is not only a time consuming and tiresome job but also information gathered may be accurate and may not be. Participants who do not have an idea about MoDSE claimed that for few questions they didn't get the answers and approaching communities and forums did not worked out. From the results in the Table 1, it is clear that answering tasks is time consuming and the participants who do not have knowledge about MoDSE took more time compared with the participants who have knowledge about MoDSE.

After each participant had finished the first phase they were asked to start second phase. During this phase participants were answering the same nine tasks with using mROSE. Time taken to complete each task by each participant is recorded and it is shown in the Table 2. Participants reported that answering the tasks with help of mROSE reduced the time for searching various sources to gather the information further reducing effort and difficulty in getting right information. From the comparison results in Table 1 and Table 2, time taken to complete the tasks has drastically reduced.

Table 1. Results During First Phase of Longitudinal Study (without using mROSE)

Participants	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	Average time (minutes)
Task 1	40	45	35	28	25	38	48	48	35	90	60	90	51	60	60	50.2
Task 2	45	35	90	120	30	100	80	90	90	45	45	35	50	50	90	66.33
Task 3	35	25	30	28	30	100	35	75	25	25	60	90	25	90	90	50.8
Task 4	25	60	69	60	25	25	25	60	35	40	35	45	40	40	35	41.27
Task 5	20	30	38	58	105	35	25	25	25	35	40	50	70	80	90	48.4
Task 6	20	30	30	20	20	25	35	40	35	30	30	35	20	20	20	26.67
Task 7	30	60	60	60	60	60	60	30	30	35	30	30	30	30	30	42.33
Task 8	50	90	100	150	100	120	120	130	140	140	135	135	130	135	150	121.67
Task 9	40	45	45	55	65	40	40	45	65	75	45	65	75	90	90	58.67

Table 2. Results During Second Phase of Longitudinal Study (using mROSE)

Participants	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	Average time (minutes)
Task 1	5	8	9	9	9	5	7	5	7	7.5	9.5	10	10	10	10	8.07
Task 2	7	7	10	10	12	6	7.5	7.5	8	10	12	12	12	15	15	10.07
Task 3	10	10	10	9	15	10	9	8	10	9.5	7.5	9.5	19	15	15	11.1
Task 4	5	5	5	15	10	10	15	9	12	10	10	10	18	19	10	10.87
Task 5	5	5	5	5	10	10	5	8	7	10	10	10	10	10	10	8.0
Task 6	3	3	5	3	5	2	3	5	4	5	5	5	5	5	4	4.13
Task 7	3	3	4	4	4	3	4	5	5	5	5	4	4	4	4	4.35
Task 8	30	30	40	40	35	35	30	45	40	40	35	30	35	30	30	35
Task 9	15	15	20	20	25	15	18	20	25	15	15	20	20	20	20	25.53

4.2 Laboratory User Study

The experimental study is essential to determine user perception towards prototype model of mROSE. Laboratory study reveals many future directions which would help the authors to extend and make mROSE as a complete working model. In this study ten participants were participated, and given the computer in which mROSE is installed and asked them to go through each menu and action buttons. All ten participants were asked to reply the questionnaire (which is not mention because of limited space) prepared to rate usefulness, ease of use, recommendations, user interface, and action buttons, etc. 5-level Likert scale is used to rate mROSE (1-strongly disagree to 5-strongly agree). Table 3 shows opinion of all the participants. During this study there was an interaction session with all participants personally to know their thoughts whether this kind of recommendation system should be available in MoDSE context. All participants agreed that it was a good idea and it would likely help the users to find many MDA and UML tools and also about MoDSE. Laboratory study also highlighted the difficulties of using Basic Mode and Expert Mode components. Tool Comparison component will appear only after selecting the tools for comparison and recommendations also generated. This was not observed by four participants and others have gone through the differentiation. However, many of the participants did not recognize where to provide the feedback about mROSE. Instead of using the facility to leave comments, suggestions and feedback, participants have sent the mails.

Mean, Standard Deviation and Population Standard Deviation also calculated for queries L1 to L10 (questioner consists of queries from L1 to L10 are not mentioned due to limited space). Eight out of ten participants satisfied with the information in mROSE. (L1: mean=4, standard deviation=0.67, population standard deviation=0.63). Eight out of ten participants felt that they gain the knowledge about tools and MoDSE. (L2: mean=4.1, standard deviation=1, population standard deviation=0.94). Nine participants liked the idea of mROSE (L3: mean=3.9, standard deviation=1.2, population standard deviation=1.14). Nine participants trusted the information in mROSE (L4: mean=4.4, standard deviation=0.81, population standard deviation=0.77). Nine participants felt mROSE is useful for users who involved in MoDSE. (L5: mean=3.9, standard deviation=0.57, population standard deviation=0.54). Eight out of ten participants felt that mROSE is not cumbersome to use. (L6: mean= 1.9, standard deviation=2.3, population standard deviation=2.21). All participants reported that learning is easy (L7: mean= 4.7, standard deviation=0.88, population standard deviation=0.83). All participants reported that mROSE was not often frustrating (L8: mean= 1.6, standard deviation=2.58, population standard deviation=2.44). All participants felt that easy to get mROSE to do what they want to do (L9: mean= 4.4, standard deviation=0.67, population standard deviation=0.63). Nine participants claimed that mROSE is easy to remember, to perform the tasks (L10: mean=3.8, standard deviation=0.47, population standard deviation=0.44).

Table 3. Ratings of mROSE by Participants during Laboratory Study

	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10
P1	3	2	1	3	4	3	5	2	4	3
P2	4	4	4	5	4	2	4	2	4	4
P3	4	5	4	4	3	2	5	2	4	4
P4	4	5	5	5	5	2	5	2	5	4
P5	4	4	5	4	4	2	5	2	5	4
P6	5	4	5	4	4	1	5	1	5	4
P7	5	4	5	5	4	1	5	1	4	4
P8	4	5	4	5	4	1	5	1	4	4
P9	4	5	4	5	4	2	4	2	5	4
P10	3	3	4	4	3	3	4	1	4	5

4.3 Metrics Evaluating Performance

4.3.1 Response Time

It is widely used performance metric, which is utilized for various purposes and in different domains. In this case it defines the time that elapsed between a user's stated request and system's response to that request. User may request for a recommendation from the system at time T1. mROSE will accept the request, process the input, and after a successful completion of required task, it will provide a response at time T2, where necessarily T2>T1. Then the response time Tr, is defined as the difference between these two times i.e Tr=T2-T1. To calculate response time of mROSE, eight different requests and their responses have considered and average response time Tr is calculated and is shown in Table 4. Total response time for all eight requests is 1.8 sec and the average is 0.225 sec which is an effective response time.

Table 4. Response time of mROSE

Request	R1	R2	R3	R4	R5	R6	R7	R8
T1sec	0.1	0.2	0.01	0.09	11.5	1.5	2	2.3
T2 sec	0.6	0.4	0.1	0.02	11.5	2	2.3	2.4
Tr=T2-T1 sec	0.5	0.2	0.09	0.11	0	0.5	0.3	0.1

4.3. 2 Storage Requirements

Another way of evaluating mROSE is based on its storage requirements. It is natural to expect from modern recommendation systems to provide services that involve more number of users and items. So, it may be wise to evaluate how mROSE manipulate the space provided. Storage requirements are usually analyzed in two ways: by checking their main memory and secondary storage requirement. Prototype model of mROSE works offline, so the storage space usage including database is 97MB which is very much reasonable.

4.3.3 Computation Complexity

Typically most filtering algorithms can be divided into two separate steps [22]: Fist, there is a model building step, usually executed off-line, followed by a second execution step, which is always executed on-line. Preprocessing, representation, calculation and recommendation, prediction generation, which appear in most discussed filtering algorithms, can be as part of off-line step. Prediction or recommendation generation is an on-line step.

In mROSE preprocessing is interacting with the system thru either Basic Mode or Expert Mode, selecting the concern, searching for concern, get the tools for selected concern, and get information about tools and concerns of MoDSE usually executed off-line. Tool comparison, recommendation generation is on-line step. Tool comparison is crucial performance factor in the recommendation process, since it is based on the number of tools selected for comparison. It is obvious that when number of selected tools increases, comparison summary becomes large and delayed. But in mROSE computational complexity is totally reduced and same time taken to generate comparison summary irrespective of the number of tools. And it is also observed that mROSE takes considerably less time for both the off-line and on-line steps and give faster responses. Thus, it is clear that computation complexity is very less in mROSE.

5. CONCLUSIONS AND FUTURE WORK

mROSE is a recommendation system which is useful for tool selection and for understanding Model-Driven Software Evolution. mROSE consists of four major components: Basic Mode, Expert Mode, Tool Comparison and Knowledge Base. User profile is considered either as a beginner or expert. All possible stakeholder interests and/or concerns in MoDSE are considered in the proposed prototypical system. For each user interest all possible matching MDA tools are considered. Description of the entire content of mROSE is made available in the repository which is called as Knowledge Base. Tool Comparison implicitly works as a recommendation engine. An approach for recommending right tools for the right concerns in MoDSE is also presented. Prototype model of mROSE is also implemented. MROSE is useful for stakeholder to understand the evolution of the models, variant activities of MoDSE and role of the tools. mROSE is evaluated by longitudinal and laboratory user studies which conforms the desire for a system like mROSE. Evaluation of the proposed system reveals many future directions like change in user preferences, prediction generation, and ranking mechanism. Increase in number of concerns and tools rises the scalability which is the major challenge of many existing recommender systems is subject of future work.

6. REFERENCES

[1] Arie van Deursen, Eelco Visser, and Jos Warmer. Model-Driven Software Evolution: A Research Agenda",In Dalila Tamzalit (Eds.). Proceedings 1st International Workshop on Model-Driven Software Evolution, University of Nantes, 2007. pp. 41-49.

[2] Michel Hoste, Jorge Pinna Puissant, Tom mens,2008. Challenges In Model_ Driven Software Evolution, Technical report of BENEVOL Workshop, Technische University Eindhoven.

[3] Martin Robillard, Robert J. Walker, Thomas Zimmermann. Recommendation Systems for Software Engineering. IEEE Software, Vol.27, No.4, pp 80-86, July/Aug 2010.

[4] A. Anand Rao, K.Madhavi, 2010. A Framework for Visualizing Model-Driven Software Evolution- Its Application, International Journal of Computer Science Issues (IJCSI), Vol.7,Issue 1, No.3, pp 47-53.

[5] K.Madhavi, A.Anand Rao,2009.A Framework for Visualizing Model-Driven Software Evolution, Proceedings of IEEE International Advance Computing Conference, Patiala, Punjab, India, pp 1785-1790.

[6] Rational Product Support for MDA, Model Driven Architecture(MDA) Information Center, IBM,

[7] modelbased.net, www. modelbased. Net

[8] Objects by Design Forum, Jelsoft Enterprises Limited. http://forums.objectsbydesign.com

[9] Behrouz H. Far, Mari Ohmori, Takeshi Baba, Yasukiyo Yamasaki, Zenya Koono, 1996. Merging CASE tools with knowkedge-based technology for automatic software design, Decision Support Systems, Volume 18, issue 1, September, pp 73-82.

[10] Mikko Konito, 2005. Architectural manifesto:choosing MDA tools, Three categories for evaluation, Model Driven Architecture(MDA) Information Center, IBM,

[11] Peter Wittmann. Comparison of MDA tools. www.wittmannclan.com

[12] Philip Liew, Kostas Kontogiannis, Tack Tong, 2004. A Framework for Business Model Driven Development, Proceedings of the International Workshop on Software Technology and Engineering Practice.

[13] Jordi Cabot, Ernest Teniente, 2006. Constraint Support in MDA tools: A Survey, Proceedings of 2nd European Conference on Model Driven Architecture, LNCS, pp 256-267.

[14] R. Holmes, R.J. Walker, and G.C. Murphy. Approximate Structural Context Matching: An Approach for Recommending Relevant Examples. IEEE Trans. Software Eng., vol. 32, no. 1, 2006, pp. 952–970.

[15] A. Ankolekar et al. Supporting Online Problem-Solving Communities with the Semantic Web. Proc. Int’l Conf. World Wide Web, ACM Press, 2006, pp. 575–584.

[16] A. Mockus and J.D. Herbsleb,. Expertise Browser: A Quantitative Approach to Identifying Expertise. Proc. Int’l Conf. Software Eng. (ICSE 02), IEEE CS Press, 2002, pp. 503–512

[17] S. Thummalapenta and T. Xie, PARSEWeb: A Programming Assistant for Reusing Open Source Code on the Web. Proc. IEEE/ACM Int’l Conf. Automated Software Eng. (ASE 07), ACM Press, 2007, pp. 204–213. RSSE community website at http://rsse.org.

[18] Emmanouil Vozalis, Konstantinos G. Margaritis,. Analysis of Recommender Systems’ Algorithms. In Proceedings of the Ninth Panhellenic Conference in Informatics,2003.

[19] Mulund Deshpande and Geroge karypis. Item-Based Top-N Recommendation Algorithms, ACM Transactions on information system, Vol.22, No.1, January 2004, Pages 143-177.

[20] Tamer Elsaved, Jimmy Lin and Douglas W.Oard. Pairwise Documents Similarity in Large Collections with MapReduce. Proceedings of ACL-08:HLT, pp 265-268.

[21] Jeffrey Dean and Sanjay Ghemawat, MapReduce: Simplified Data Processing on Large Cluste rs. OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December, 2004.

[22] Marcel Bruch, Thorsten Schafer, and mira Mezini. On Evaluating Recommender Systems for API sages. RSSE ’08, November 10, Atlanta, Georgia, USA pp16-29.