AN ENHANCED STRATERGY FOR DERIVING THE QUALITY REQUIREMENTS IN QoS BASED WEB SERVICES WITH LOCATION RECOMMENDER SYSTEM

T. Priyaradhikadevi[1*], R. M. S. Parvathi[2], K.Lakshmi Narayanan[3]
[1]Research Scholar & Associate Professor
Department of Computer Science and Engineering
Mailam Engineering College, Mailam – 604304, Tamil Nadu, India.
[*]E-mail: priyaradhikadevi@gmail.com
[2]Principal & Professor, Department of Computer Science and Engineering
Sengunthar College of Engineering, Tiruchengode – 637 205, Tamilnadu, India
E-mail: drrmsparvathi@gmail.com
[3]Assistant Professor, Department of Computer Science and Engineering
Mailam Engineering College, Mailam-604304, Tamil Nadu, India
E-mail: lakshmi1076@gmail.com

Abstract- In this paper we propose to translate the customer's need for a set of defined capabilities into a working service. The complete information necessary for a product or service is achieved through Requirement Engineering. We provide a better QoS in terms of requirements which helps the customer to differentiate the best service among different services. A novel framework called G-RIA SMART is proposed, which provides quality adaption for a service based system to develop a quality strategy for requirements by mapping into service based systems using the novel framework. Here we propose the Priority selection algorithm which provides how to filter unwanted and unrelated strings from data storage and also to provide high priority to the required input within requested portion of the string. The result based on priority selection algorithm provides efficient outputs from real time environment. And also we implement the advanced location Recommender system for web service, with efficient query processing method. This location recommender system uses for location-based ratings, rank and quality based recommendations and Qos Information. Existence recommender systems do not consider locality properties of users nor items.

Keywords: AWSLRS, G-RIA SMART, LTR, QoS.

I. INTRODUCTION

Software Engineering is an establishment and the use of principles, in order to obtain systematic approach in design, development and maintenance of software that is reliable and work efficiently on real machines. Software Engineering can be treated as a layered approach with design and implementation as some of its layers. IEEE has defined Software Engineering as, "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software".

A. Motivation of SE

The goal of software engineering is to produce quality software, which is delivered on time, within budget, and which satisfies customer requirements and user needs. A service approach must rest on an organizational commitment to achieve quality. The bedrock that supports software engineering is a quality focus. The initial phase starts with the gathering of requirements based on the needs of a customer or a user. Qualities can be differentiated as internal and external quality. The external qualities are visible to the user such as reliability, efficiency, usability etc, whereas the internal quality concerned with the developers are verifiability, maintainability, adaptability etc. Requirements engineering is a process of discovering the needs of stake holders and documenting them for analysis, communication and implementation.

B. Web Service

Web service is a software system designed to support interoperable machine-to machine interaction over a network. Web services interact with each other, fulfilling tasks and requests that, in turn, carry out parts of complex transactions or workflows. A web service provides programmatic access to a service. The feature of simple integration has made the web services more important. Services run on all kinds of machines, from the desktop to the mainframe. The World Wide Web is like a large library content that enables to share and distribute information. Web services enhance this by enabling the sharing and distribution of services. The development of Universal Description Discovery and Integration (UDDI), Simple Object Access Protocol (SOAP) and Web Service Definition Language (WSDL) standards are the foundation of web services.

C. Recommender System

Recommender systems make use of community opinions to help users identify useful items from a considerably large search space e.g. Amazon inventory, Netflix movies Community opinions are expressed through explicit ratings represented by the triple (user, rating, item) that represents a user providing a numeric rating for an item. Currently, myriad applications can produce location-based ratings that embed user and/or item locations existing recommendation techniques assume ratings are represented by the (user, rating, item) triple, thus are ill-equipped to produce location recommendations. AWSLRS produces recommendations using a taxonomy of *three* types of location-based ratings within a single framework: (1) Locality user with non-Locality items, Location item rating (LTR) represented as a four-tusple (*user*, *ulocation*, *rating*, *item*), where *ulocation* represents a user location, for example, a user located at home rating a book; (2) non-Locality user with Locality.

II. LITRATURE SURVEY

Ruth Malan et al. (2001)[1] has proposed the Use cases which are useful in capturing and communicating functional requirements and they play a primary role in product definition. In 2000, an inventor Davidhas et al. proposed one of the biggest problems the managers face when dealing with a software development project, is that by its very nature, the project is invisible and non-tactile.

M.Bernardo et al. (2003) [5] has addressed this challenge and proposed a goal-oriented approach to architectural design based on the framework for modeling, specifying and analyzing requirements. During the 2005's [4] Lei, Y have proposed a novel approach for web service discovery that combines ontology linking with Latent Semantic Indexing (LSI). The basic idea is to build the service request vector according to the domain ontology have the training set of the LSI classifier based on features extracted from selected WSDL files and finally project the description vectors and the request vector and utilize the cosine measure to determine similarities and to retrieve relevant WSDL service descriptions.

K.Kritikos et al.(2008) [2] has analyze the requirements of a semantically rich QoS-based WSDM and to provide SW and constrained based mechanisms for enriching syntactic QoS-based WS Discovery (WSDi) algorithms.

Kyriakos et al. (2008) [3] has introduced the QoS description models of WSs that dedicated to conduct a research on all QoS-based WS description efforts in order to unveil those features and parts that are necessary for a QoS description model of WSs.

M. Sathya et al. (2011) [6] has introduced this paper has outlined the approach of non functional (QoS) Web service selection based on requirements and specification identified from the thorough study from the literature. This paper reviewed a number of techniques in the context of the QoS based approach and have presented a summary of QoS parameters involved in the techniques identified and also the evaluation metrics that can be applied to obtain and test how the techniques perform against the specification criteria.

B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, [7] has introduced LRS uses item-based Collaborative filtering and Refining (abbr. CFR) as its primary recommendation technique, chosen due to its popularity and widespread adoption in commercial systems. Collaborative filtering and Refining (CFR) assumes a set of n users $U = \{u_1, ..., u_n\}$ and a set of m items $I = \{i_1, ..., i_m\}$. Each user u_j expresses opinions about a set of items $I_{uj} \subseteq I$. Opinions can be a numeric rating (e.g., the Netflix scale of one to five stars), or unary.

III. DERIVING QUALITY REQUIREMENTS

The ample adoption of requirements raises the challenging problem in the service based system. To solve this problem, a methodological technique is described to support the process of mapping of GORE and SBS by utilizing the proposed framework called G-RIA SMART. By adopting this framework, the customer's desire is translated for a set of defined capabilities into a working service and the validation also done by the developer. So, it provides a better quality in terms of requirements which helps the customer to differentiate the best service among different services discovered. Thus leads to increase the profit margins of an organization by using goals.

Requirements Engineering is a process that includes a set of activities such as Requirement Inception, Elicitation and Elaboration etc. Goal Oriented Requirements Engineering takes the view that requirements should initially focus on the why and how questions rather than on the question of what needs to be implemented. A web service is any piece of software that makes it available over the internet and uses a standardized XML messaging system. XML is used to encode all communications to a web service. A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

IV. G-RIA SMART

The main contribution of the proposed work is to translate the customer's need for a set of defined capabilities into a working service. The complete information necessary for a product or service is achieved through Requirement Engineering. The overall requirements are elicited from the customer. These requirements encompass information and control needs, function and behavior, overall performance, design interfacing constraints, and other specific needs. It provides the appropriate mechanism for understanding what the customer wants, analyzing the need, assessing the feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification, and managing the requirements as they are transformed into an operational system.

This research work provides a better QoS in terms of requirements which helps the customer to differentiate the best service among different services discovered thus providing a decision basis for choosing an efficient QoS based service. A novel framework called G-RIA SMART is proposed, which provides quality adaption for a service based system developed where the requirements are transformed into probabilistic values. A generalized view of the proposed work to develop a quality strategy for requirements by mapping into service based systems using the novel framework.

A. Proposed G-RIA Smart Framework

In the emerging trend of web service, QoS has become a major concern. Both service providers and consumers strive for the best service. But the problem is the consumer blindly puts the trust on the provider who is publishing the service. In this case, they cannot guarantee that every provider publishes an efficient service. So there is a need for verification and validation for the services published in the registry, where the quality broker plays its role. In this work, every service is validated before publishing it in the registry so that the consumer can trust the registry for retrieving a qualified service.

B. Applying Q-Factor

First the developer gathers requirements from the customer, classifies it and based on the classification, applies the Q-factor of SPECS (Size, Performance, Effort, Cost and Schedule).The developer put into G-RIA for identification and analysis of goal based requirements. First the requirements are grouped under categories such as normal, exact and exciting. Then the services related to the requirements specified were discovered. Finally mapping is performed which is the match-making phase of the service. Finally an efficient service is selected, based on the QoS in the selection phase. The developer develops a service, which is parsed by a parser, and the requirements are retrieved from the output of the parser.

These are classified and validated based on the SMART criteria before publishing into the registry. If and only if the services satisfy these quality attributes, it is published in the registry which can further be utilized by the consumer or user. If the customer is satisfied, the developer verifies whether the requirements satisfy the SMART properties for validation purpose. Otherwise refinement of G-RIA is preceded. Afterwards, developers render a service according to those requirements and publish the service into a WSDL file. This WSDL file is explored using a WSDL parser and retrieve the requirements what the service provider has stated in the service by the request of user.

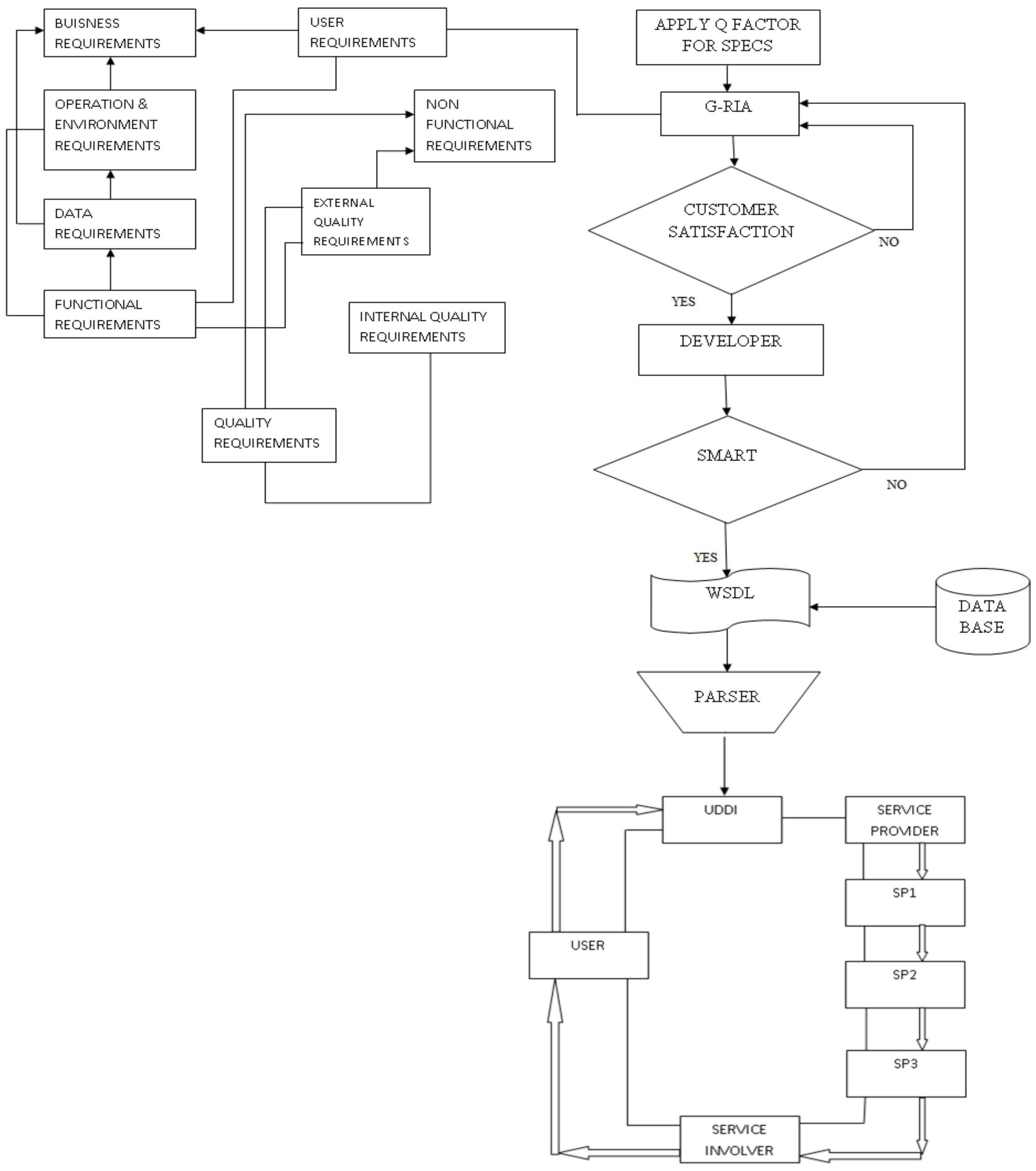


Fig 1. G-RIA SMART Proposed Framework

V.PRIORITY SELECTION ALGORITHM

This algorithm provides how to filter unwanted and unrelated strings from data storage for further process and how to avoid unwanted strings to save user’s time and how to regularize all input methods for accurate results by each filtration method. And also to provide high priority to the required input within requested portion of the string. The result based on priority selection algorithm provides efficient outputs from real time environment.

A. Proposed Priority Selection Algorithm

```
Procedure
Start
//Consider for an application
//Input the element (I) i.e. user request
// Input the selection tool variables
```

```
//Give high priority to the element to be searched
//Sort the element according to algorithm
//Consider low priority element to reserve
For each filtering elements
    I ← (high priority element (n) ← filtering element)
    I ← $i_{i+1}, i_{i+2}, \ldots\ldots\ldots\ldots i_{i+n}$
//separate all filtering elements
//Input p, q, r, & s respectively for sorting element as high, low, very low, medium
if(i< = 0)
{
    //p= higher priority of element
     //q=any string to be searched
    //r, s=sample message strings for input
    p → high priority element
}
q → high priority element
 else if (p == q)
{
p = high priority (q) ←searched element(r)
s = pi+1,pi+2…….pi+n
}
End Procedure
```

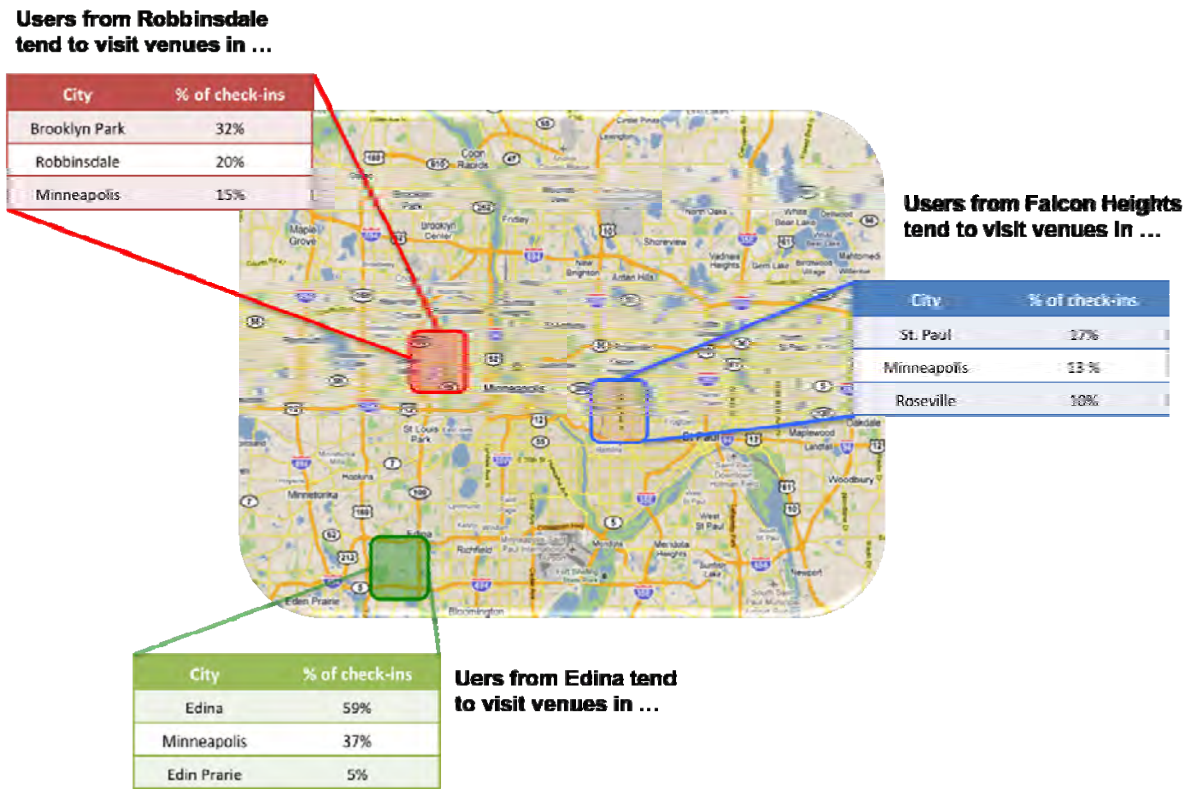
When the proposed model executes its filtration process, immediately it takes request from filtering database and searches for required data string in correct location by its sorting process. Thus it uses higher priority tool for sorting work. As the searching tool does its working from appropriate string identification, it can get its success criteria without any break and stoppage within the specified time. So by proposed algorithm the request to be searched has high receiving result. From this type of process system and its user request can be improved to a high level so that the throughput of system waiting time is maintained well and improved to a great extent.

When the proposed method executes filtration process, immediately it takes request from filtering database and searches for required data string in correct location by its sorting process. Thus it uses higher priority tool for sorting work. As the searching tool it works from appropriate string identification. By the proposed algorithm the request to be searched has high receiving result. From this type of process, the system process and its user request can be improved to a high level so that the throughput of system waiting time is maintained well and improved to such a great extent.

VI.RECOMMENDER SYSTEMS

Here we propose AWSLRS by Hybrid Collaborative filtering and Refining (HCFR), a novel web service based location recommender system built specifically to produce high-quality location-based recommendation in an efficient manner. AWSLRS produces recommendations using a taxonomy of three types of location-based ratings within a single framework: (1) Locality user with non-Locality items, Location item rating (LTR) represented as a four-tuple (user, u location, rating, item), where u location represents a user location, for example, a user located at home rating a book; (2) non-Locality user with Locality.

Location user rating (LUR) represented as a four-tuple (user, rating, item, I location), where ilocation represents an item location, for example, a user with unknown location rating a restaurant; (3) Locality user with Locality items, (LIUR) represented as a five-tuple (user, u location, rating, item, i location), for example, a user at his/her office rating a restaurant visited for lunch. Traditional rating triples can be classified as non-Locality ratings for non-Locality items and do not fit this taxonomy. .**Preference locality**. Preference locality suggests users from a Locality region (e.g., neighbourhood) prefer items (e.g., movies, destinations) that are manifestly different than items preferred by users from other, even adjacent, regions.


Users from Robbinsdale tend to visit venues in …
Users from Falcon Heights tend to visit venues in …
Uers from Edina tend to visit venues in …

City	% of check-ins
Brooklyn Park	32%
Robbinsdale	20%
Minneapolis	15%

City	% of check-ins
St. Paul	17%
Minneapolis	13 %
Roseville	10%

City	% of check-ins
Edina	59%
Minneapolis	37%
Edin Prarie	5%

Fig.2 Preference Locality Model

A.LRS Query Model

Users (or applications) provide LRS with a user id U, numeric limit K, and location L; LRS then returns K recommended items to the user. LRS supports both snapshot (i.e., one-time) queries and continuous queries, whereby a user subscribes to LRS and receives recommendation updates as her location changes. The technique LRS uses to produce recommendations depends on the type of location-based rating available in the system.

B. Item-Based Collaborative Filtering

LRS uses item-based Collaborative filtering and Refining (abbr. CFR) as its primary recommendation technique, chosen due to its popularity and widespread adoption in commercial systems. Collaborative filtering and Refining (CFR) assumes a set of n users U = {u_1, ..., u_n} and a set of m items I = {i_1, ..., i_m}. Each user u_j expresses opinions about a set of items $I_{uj} \subseteq I$. Opinions can be a numeric rating (e.g., the Netflix scale of one to five stars), or unary.

Phase I: Model Building.

Phase II: Recommendation Generation. Given a querying user u, recommendations are produced by computing u's predicted rating $P_{(u,i)}$ for each item i not rated by u:

$$p(u,i) = \frac{\sum_{l \in L} \mathrm{sim(i,l)} * \mathrm{r_{u,l}}}{\sum_{l \in L} |\mathrm{sim(i,}l)|} \qquad (1)$$

Before this computation, we reduce each similarity list L to contain only items *rated* by user u. The prediction is the sum of $r_{u,1}$, a user U's rating for a related item $l \in L$ weighted by *sim(i,l)*, the similarity of l to candidate item i, then normalized by the sum of similarity scores between i and l. The user receives as recommendations the top-k items ranked by $P_{(u,i)}$. **Computing Similarity**. To compute *sim*(i_p, i_q), we repre-sent each item as a vector in the user-rating space of the rating matrix. For instance, Figure 3 depicts vectors for items i_p and i_q from the matrix in Figure 2(a). Many similarity functions have been proposed (e.g., Pearson Correlation, Cosine); we use the Cosine similarity in *LRS* due to its popularity:

$$sim(i_p, i_q) = \frac{\vec{i_p}\,\vec{i_q}}{\|\vec{i_p}\|\|\vec{i_q}\|} \qquad (2)$$

This score is calculated using the vectors' co-rated dimensions, e.g., the Cosine similarity between i_p and i_q For instance, we could easily employ user-based CFR, that uses correlations between users (instead of items).

C. Non-Locality User Ratings for Non-Locality Items

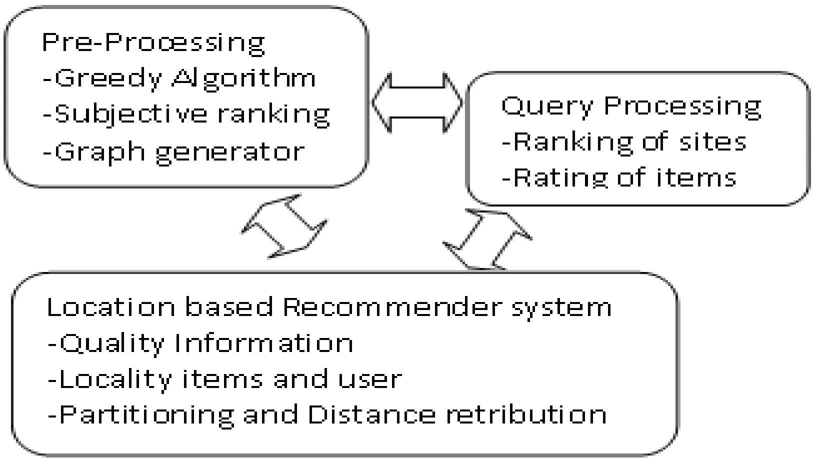


Fig 3. SLRSF (single location recommender system framework)

The traditional item-based Collaborative filtering and Refining (CFR) method is a special case of LRS. CFR takes as input the classical rating triplet (*user*, *rating*, *item*) such that neither the user location nor the item location are specified. In such case, LRS directly employs the traditional model building phase to calculate the similarity scores between all items.

D. Locality User Ratings for Non-Locality Items

This section describes how LRS produces recommendations using Locality ratings for non-Locality items represented by the tuple (user, ulocation, rating, item). The idea is to exploit preference locality, i.e., the observation that user opinions are Locality unique.

E. Pyramid Structure Intuition

An α-Cell requires the highest storage and maintenance overhead because it maintains a CFR model as well as the user/item ratings statistics. On the other hand, an α-Cell (as opposed to β-Cell and γ-Cell) is the only cell that can be leveraged to answer recommendation queries. A pyramid structure that only contains α-Cells achieves the highest recommendation locality, and this is why an α-Cell is considered the highly ranked cell type in LRS. a β-Cell is the secondly ranked cell type as it only maintains statistics about the user/item ratings. The storage and maintenance overhead incurred by a β-Cell is less expensive than an α-Cell. The statistics maintained at a β-Cell determines whether the children of that cell needs to be maintained as α-Cells to serve more localized recommendation. Finally, a γ-Cell (lowest ranked cell type) has the least maintenance cost, as neither a CFR model nor statistics are maintained for that cell. Moreover, a γ-Cell is a leaf cell in the pyramid. LRS upgrades (downgrades) a cell to a higher (lower) cell rank, based on trade-offs between recommendation locality and system scalability.

F. Maintenance Algorithm

Algorithm 1 provides the pseudocode for the LRS maintenance algorithm. The algorithm takes as input a pyramid cell C and level h, and includes three main steps: *Statistics Maintenance*, *Model Rebuild* and *Cell Child Quadrant Maintenance*, explained below.

1) *Step I-Statistics Maintenance*: The first step (line 4) is to maintain the *Items Ratings Statistics Table*. The maintained statistics are necessary for cell type switching decision, especially when new location-based ratings enter the system.
2) *Step II-Model Rebuild:* The second step is to rebuild the item-based Collaborative filtering and Refining (CFR) model for a cell C.
3) *Step III-Cell Child Quadrant Maintenance:* LRS invokes a maintenance step that may decide whether cell C child quadrant need to be switched to a different cell type based on trade-offs between *scalability* and *locality*.

G. Algorithm-1: G-PRM Algorithm

The algorithm first checks if cell C child quadrant q at level h+1 is of type α-Cell. if that case holds, LRS considers quadrant q cells as candidates to be downgraded to β-Cells (calling function *CheckDownGradeToSCells*). We provide details of the *Downgrade* α*-Cells to* β*-Cells* operation in Section 4.5.2. On the other hand, if C have a child quadrant of type γ-Cells at level h + 1 (line 12), LRS considers upgrading cell C four children cells at level h + 1 to β- Cells (calling function *CheckUpGradeToSCells*). The *Updgrade From E to* β*-Cells* operation.

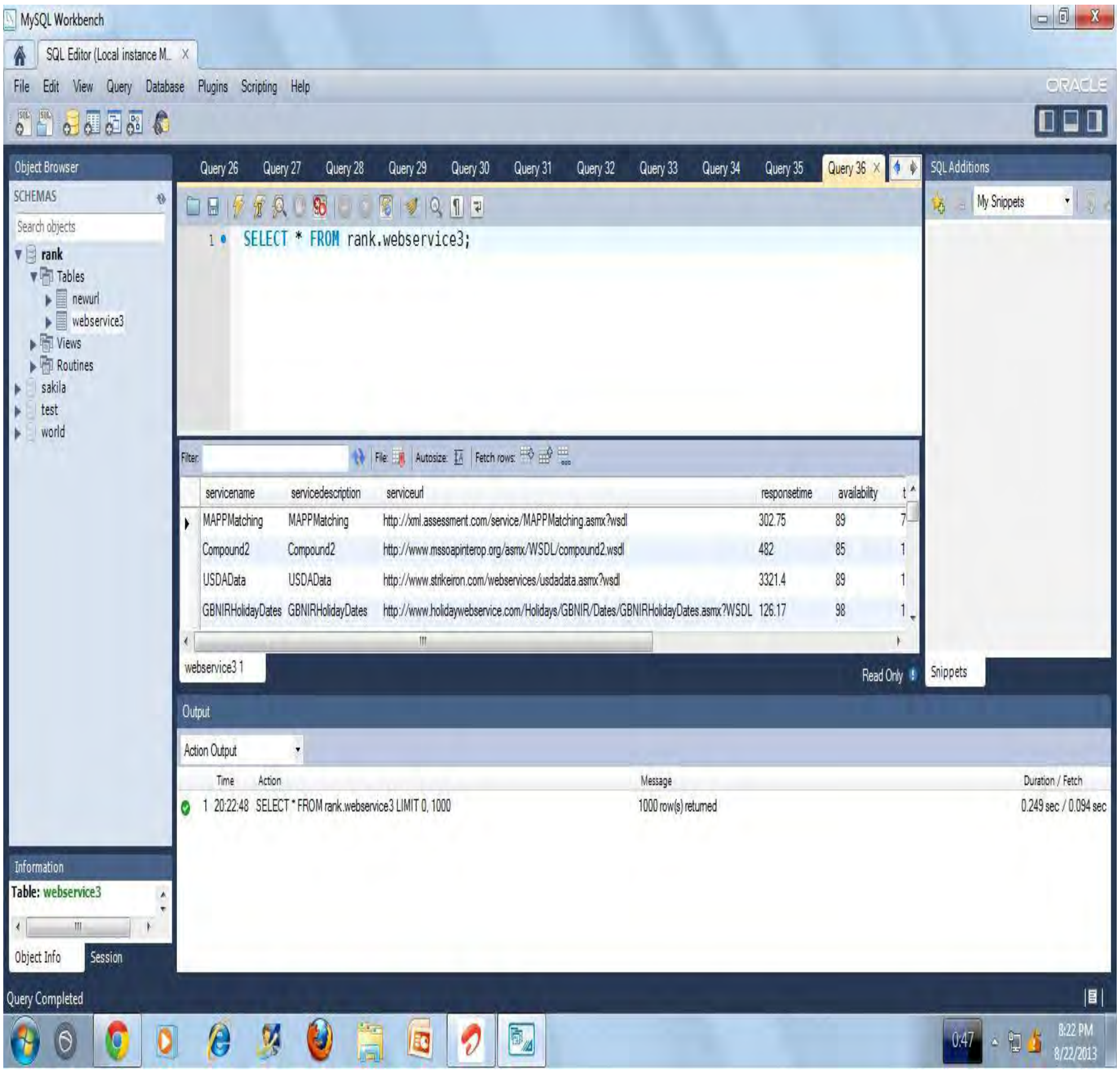
H. G-PRM Algorithm

```
Function PyramidMaintence(Cell C, Level h)
Maintain cell C statistics
If (cell C is an α-Cell) then
        Rebuild item based collaborative filtering and refining model for cell C
End if
If ( Children quadrant q cells are α-Cell) then
CheckDownGradeToCells(q,C)
Else If ( Children quadrant q cells are λ-Cell ) then
CheckUpGradeToCells(q,C)
Else
isSwitchToMCells ← checkUpGradeToMCells(q,C)
if (is SwitchedToMCell is False)then
CheckDownGradeToECells(q,C)
End if
End if
Return
```

If C have a child quadrant of type β-Cells at level h+1, LRS first considers upgrading cell C four children cells at level h + 1 from β-Cells to α-Cells (calling function *CheckUpGradeToMCells*). If the children cells are not switched to α-Cells, LRS then considers downgrading them to γ-Cells (calling function *CheckDownGradeToECells*). Cell Type switching operations are performed completely in quadrants (i.e., four equi-area cells with the same parent). We made this decision for simplicity in maintaining the partial pyramid.

VII. EXPERIMENTAL RESULTS

The following sample collection output explores the best results from the proposed methodology for related searches so that the best services can be given based on Priority algorithm.



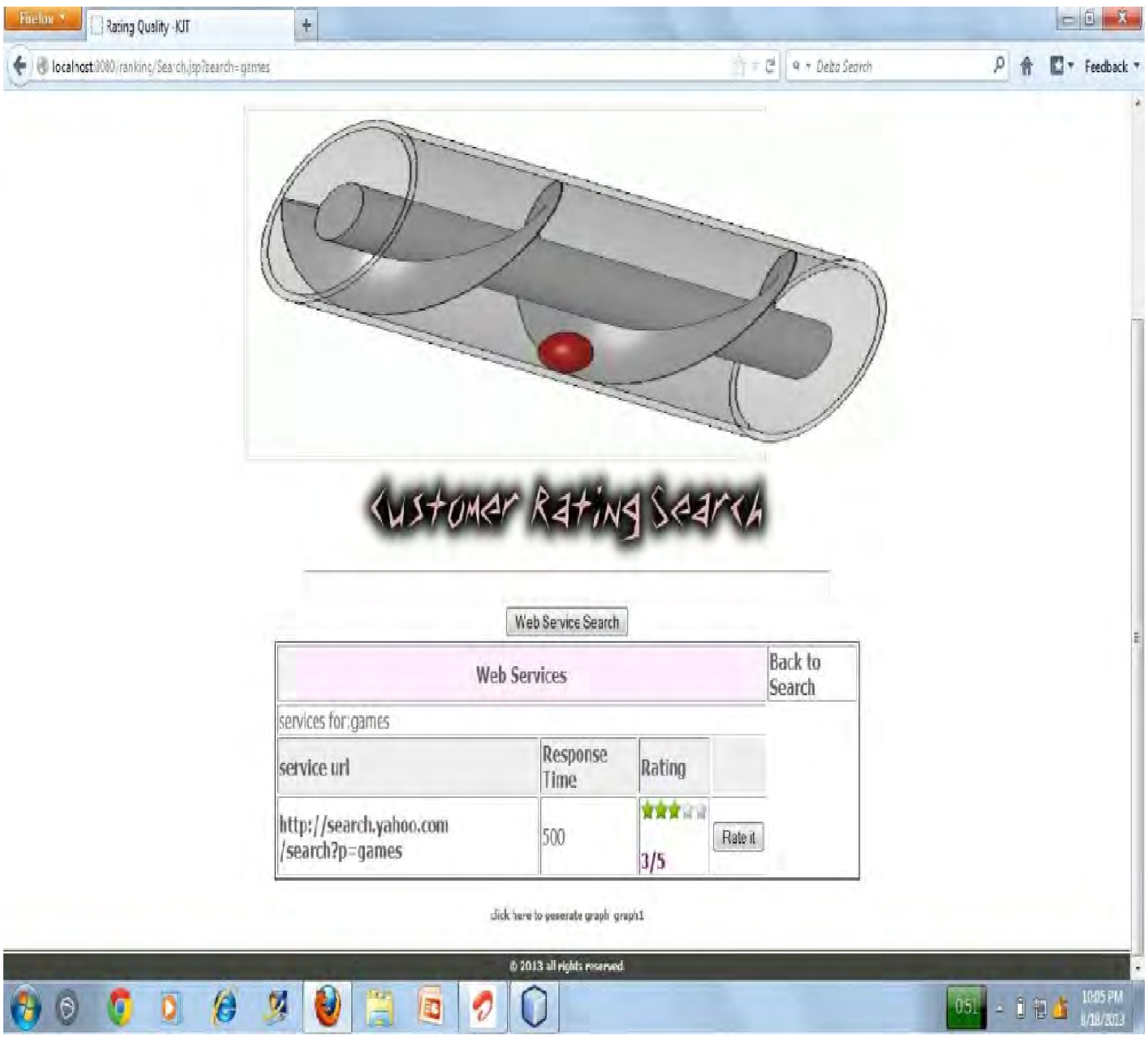
```
SELECT * FROM rank.webservice3;
```

servicename	servicedescription	serviceurl	responsetime	availability	t
MAPPMatching	MAPPMatching	http://xml.assessment.com/service/MAPPMatching.asmx?wsdl	302.75	89	7
Compound2	Compound2	http://www.mssoapinterop.org/asmx/WSDL/compound2.wsdl	482	85	1
USDAData	USDAData	http://www.strikeiron.com/webservices/usdadata.asmx?wsdl	3321.4	89	1
GBNIRHolidayDates	GBNIRHolidayDates	http://www.holidaywebservice.com/Holidays/GBNIR/Dates/GBNIRHolidayDates.asmx?WSDL	126.17	98	1

```
   Time      Action                                        Message                  Duration / Fetch
1  20:22:48  SELECT * FROM rank.webservice3 LIMIT 0, 1000  1000 row(s) returned     0.249 sec / 0.094 sec
```

Fig 4. Database of Web services with limited to 1000 option

This figure shows that the database stores the quality requirements. The real time work is done by collecting sample from a large web based searching for customer elated things from many web sources. Here the limit is set to 1000 websites option for customers testing of satisfactions. In the over 1000 websites, which one is providing user satisfaction and quality of service to customers is to be determined.


Firefox
Rating Quality -KJT
localhost:8080/ranking/Search.jsp?search=games
Delta Search
Feedback
Customer Rating Search
Web Service Search
click here to generate graph graph1
© 2013 all rights reserved

Web Services				Back to Search
services for:games				
service url	Response Time	Rating		
http://search.yahoo.com /search?p=games	500	★★★☆☆ 3/5	Rate it	

Fig 5. Web Services Rating

This figure shows the rating of the web service. When the required strings are typed, then immediately rating the required web service based on our priority selection algorithm is made. First it displays the name of the service which is required. Next it displays the URL of the particular required service. Then it displays the response time of that particular required service and also the rating in both symbolic and numeric. For each service, it can rate out of 5. Here the automatically generated graph can be viewed by clicking graph1. We compare LRS with the standard item-based Collaborative filtering and Refining technique along with several variations of LRS. We also compare LRS to LRS. Experiments are based on three data sets: *Movie Lens:* a real data set consisting of *Locality user ratings for non-Locality items* taken from the popular Movie Lens recommender system. The Foursquare and Movie Lens data are used to test recommendation quality. The Movie Lens data used in our experiments was real movie rating data taken from the popular Movie Lens recommendation system at the University of Minnesota. This data consisted of 87,025 ratings for 1,668 movies from 814 users. Each rating was associated with the zip code of the user who rated the movie, thus giving us a real data set of Locality user ratings for non Locality items.

VIII. CONCLUSION AND FUTURE ENHANCEMENT

As a future work, it is planned to work these environments for Requirements Engineering using advanced web based search process for immediate results of user requests in big cloud based concept to meet current Requirements Engineering process. This work explores the idea of suitable web based result by searching for best filtration output of user or consumer requests. In cloud computing, issues related to the best elicitation are faced with the question of appropriate requirements engineering.

REFERENCES

[1] Ruth.Malan & Dana Bredemeyer 2011, ‘Functional requirements and use cases’, Bredemeyer Consulting, White Paper Society, pp.64-71.

[2] Kritikos,K & Plexousakis,D 2009, ‘Requirements for qos-based web service description and discovery’, IEEE Transactions on Services computing, vol.2, no. 4, pp.320-337.

[3] Kyriakos E. Kritikos 2008, ‘QoS-based web service description and discovery’, Computer Software and Applications Conference, vol.2,pp.467-472.

[4] Y. Lee 2008, ‘Quality context composition for management of SOA quality’, IEEE International Workshop on Semantic Computing and Applications, pp. 117–122.

[5] M. Bernardo & P. Inverardi 2009, ‘Formal methods for software architectures, tutorial book on software architectures and formal methods’, vol.35, no.3, pp.325-346.

[6] Sathya, M, Swarnamugi,M, Dhavachelvan ,P & Sureshkumar, G 2011, ‘Evaluation of qos based web- service selection techniques for service composition’, International Journal of Scientific& Engineering Research(IJSE),vol.1, no.5, pp.73-90.

[7] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-Based Collabo- rative Filtering Recommendation Algorithms,” in WWW, 2001.

[8] Aldeida Aleti, Barbora Buhnova & Lars Grunske 2012, ‘Software architecture optimization methods a systematic literature review’, IEEE Transaction on Software Engineering, pp.1-27.

[9] Chin-Yu Huang, Hareton Leung, Wu-Hon Francis Leung & Osamu Mizuno 2012 ‘Software quality assurance methodologies and techniques’, Hindawi publication on Advances in Software Engineering, vol.12, no.2, pp.1-2.

[10] Christine Strauss, Gabriele Kotsis, Eric Pardede &Fatos Xhafa 2012, ‘Special issue on chalanges with defining and measuring e-services sustainability’, Society of Service Science and Springer, vol.4, pp.169-173.

[11] Artem Katasonov 2012 ‘Ontology-driven software engineering: Beyond model checking and transformations’, International Journal of Semantic Computing, vol. 6, no. 2, pp. 205 – 242.

[12] Lamia Ben Ghezaiel, Chiraz Latiri & Mohamed Ben Ahmed 2012, ‘Ontology enrichment based on generic basis of association rules for conceptual document indexing’, 4th International conference on knowledge engineering and ontology development, vol.4, pp. 53-65.

[13] Nor Shaniza Kamal Bashah, Natalia Kryvinska & Do van Thanh 2012, ‘ Quality-driven service discovery techniques for open environments and their buisness applications’, Society of Service Science and Springer, vol.4, pp.71-77.

[14] Rama Akkiraju, Debarun Bhattacharjya & Sammukh Gupta 2012, ‘Towards effective buisness process availability management’, Society of Service Science and Springer, vol.4, pp.319-351.

[15] Rutviji Mehta, Hongyuan Wang & Lawrence Chung 2012, ‘Dealing with NFRs for smart-phone applications: A goal-oriented approach’, Software Engineering Research Management and application, pp.113-125.