Requirement Validation Model for Virtual Distributed System

Tayyaba Kiran[1], Saima Farhan1, Huma Tauseef[1] and Muhammad Abuzar Fahiem[1]

[1]Department of Computer Science, Lahore College for Women University, Lahore, Pakistan

***Abstract*— Requirement engineering plays very important role in almost every field of computing and development. Many different techniques or models have been proposed for requirement engineering process. All these models follow different techniques to solve the requirement engineering issues. There are a number of general activities common to all processes which are requirement elicitation, requirement analysis, requirement specification, requirement validation and requirement management. Currently many researches have been proposed for various requirement validation techniques. All these techniques proposed by researchers majorly focus on centralized projects. In this research paper, a requirement validation framework has been proposed for distributed virtual projects. Main purpose of this framework is to enhance the quality of the distributed virtual systems by providing easy and systemic way for validating the requirements of such systems. Distributed Virtual Environments are those systems which are designed for users who are located at different geographical areas and uses different networks. These are designed without any requirement specification from end users. Our proposed framework provides a systematic way to requirement engineers through which they could analyze the requirements that come from user end on the basis of some factors like completeness, correctness etc. For developing proposed framework, three approaches have been followed; Prototyping, Review and Test Based. The successful implementation of proposed requirement on distributed virtual environment can have a good influence for building quality software in future.**

***Keywords*— Distributed Virtual Environment Systems, Requirement Engineering and Requirement Validation Techniques**

I. INTRODUCTION

Requirement engineering is the most complex task of software engineering process. It plays very vital role in ensuring the overall quality and success of software projects [1]. Many researchers have proposed different models for different applications of distributed projects [2], [3]. Requirement engineering has always been a challenging task for developing any system or project especially when it comes in distributed system. Distributed system involves multiple independent computers that communicate through a computer network. The computers interact with each other in order to achieve a common goal. Distributed virtual systems are developing mechanisms that facilitate the users to organize, integrate, and utilize the growing number of computing and information resources that are available on large networks, without having the need for a central authority to impose structure on the resources. To develop these systems, it is very important that the requirements for these systems must be clearly defined, well understood or verified [4], [5]. This is because if the requirements do not meet the clients' need and expectation, the system is considered fail despite of its proper working. So in order to meet client needs, prevent failures, enhance the system quality, check completeness, consistency and understandability in distributed virtual systems, there should be some organized framework or guidelines which could verify the overall requirement validation issues in a more efficient and manner able way. Major focus of this research work will be on resolving these issues by proposing a framework for distributed virtual environment systems.

Requirement Validation (RV) is a process in which consistency, completeness and accuracy of software requirements are analyzed. RV is done through several techniques in order to create quality products and systems and analyze whether defined requirements are right and applicable. Some major requirement validation techniques are Prototype, Testing Based, Writing User Manual, Formal Specification, Review, Preview, Model Based, View Point Oriented, Tracing Approach, Simulation and Functional Test Design [6].

The paper is divided into 5 sections. Section II describes the literature review which is followed by our proposed work in section III. Discussion is provided in section IV. Paper ends with section V as conclusion and future directions.

II. LITERATURE REVIEW

Distributed Virtual Environment systems (DVEs) are complex systems that comprises of graphics, physical simulation and network state synchronization. Performance of a system depends on all these components including the maintenance of load, Application Programming Interface (APIs) and network resources [7]. Many DVE systems are developed. Researchers have done a lot of work on DVEs and proposed different models for DVEs. Some models are related to security, networking and quality etc. User collaboration and interaction does not exist in the development of these systems because of complex architecture. Requirements are always developer specified. Development of DVEs involved large

number of resources and provides a platform for multiple users on different networks. User input in requirement specification is not involved in DVEs. Requirements Validation is a process to verify the development of right product; system develops according to the stakeholder's requirement. Requirement Validation Model (RVM) also assures that software fulfills the system goals. Some authors have proposed different requirement validation models for different systems or environment but no validation model is built for DVEs because of its complexity.

Some researchers have defined different Requirements Validation Techniques (RVTs) like requirements reviews, prototyping, testing based, writing user manual, formal specification, preview, model based, view point oriented, tracing approach, simulation, functional test design, throwaway, evolutionary, model checking, theorem proving, commenting, Fagan's inspection process, test case driven inspection, walk through, reading, ad hoc based, check list based, perspective, defect based scenario based, pattern based, data flow models, compositional models, classified models, stimulus response models, process models and simulation models. These techniques are helpful to build a validation model according to different system and environment needs.

Table 1 represents a comparison of different requirement validation techniques, their features and relationship between them. Preview technique make availability of different features such as save cost, time, reworking, checking of defects and standards against a single person or a team, phase functionality that is helpful in removing defects in initial phase of System Development Life Cycle (SDLC), at system level and at final phase of a system. Review technique provides different features such as Check Defects and Standard team wise, Phase Functionality to remove defects in early stage of SDLC, Clarified missing requirements, Correctness, completeness, inconsistences and informal form. Writing user manuals techniques have features like Completeness, ambiguity, usability. Functional test design techniques have features like Phase functionality on System level, Completeness and Ambiguous requirement. Simulation techniques covers features like Scenario based implementation on graphically demonstration (formulation, structured, .model) at Abstract Level. Traceability technique follows Traceability. View point Oriented features are Interactive system, completeness, inconsistence and correctness. Model Base involved phase functionality at final phase, graphically demonstration (formulation, structured, model) and inconsistence.

TABLE 1
COMPARISON OF REQUIREMENT VALIDATION TECHNIQUES

Features		Techniques										
		Prototype [6,8,9,10,11]	Testing Based [6,9,10,11]	Writing User Manual [10,11]	Formal Specification [10,11]	Review [6,8,10,11,12]	Preview [6,10,11]	Model Based [6,10,11]	View point Oriented [6,9,10,11]	Tracing Approach [10,11]	Simulation [10,11]	Functional Test Design [10,11]
Save cost, Time, Reworking							√					√
Check defects and standard	Single person											
	Team					√						
Phase functionality	Remove defects in early stage of SDLC					√						
	System level			√								√
	Final phase		√					√				
Clarified missing requirement						√						
From sketching or picture to high level fourth language		√										
Graphical								√				

Features		Techniques										
		Prototype [6,8,9,10,11]	Testing Based [6,9,10,11]	Writing User Manual [10,11]	Formal Specification [10,11]	Review [6,8,10,11,12]	Preview [6,10,11]	Model Based [6,10,11]	View point Oriented [6,9,10,11]	Tracing Approach [10,11]	Simulation [10,11]	Functional Test Design [10,11]
demonstration (formulation, structured, model)	Abstract model										√	
	Test case design		√									
Testable requirement		√										
Correctness			√		√	√			√			
Inconsistence			√		√	√		√	√			
Completeness		√	√	√	√	√			√			√
Interactive system									√			
Traceability matrix										√		
Unambiguous requirement				√								√
Usability				√								
Desirable properties					√							
Check all execution paths					√							
Informal						√						
Scenario based											√	

A. Prototype Sub Techniques

Prototype Technique is further divided into two sub techniques that are Throwaway and Evolutionary techniques as shown in Table 2. Throwaway technique consists of various features like limited functionality, poorly understandable requirements and initial to experiment phase. Evolutionary technique involves features like high priority requirement, deliver working system, final state quality attributes and understandable requirement for customers.

TABLE 2
COMPARISON OF PROTOTYPING SUB TECHNIQUES

Features	Prototyping Technique	
	Throwaway [6,9,10,11]	Evolutionary [6,9,10,11]
Limited functionality	√	
Poorly understandable requirements	√	
Initial to experiment phase	√	
High priority requirement		√
Deliver working system		√
Final state quality attributes		√
Understandable requirement for customers		√

B. Formal Specification Sub Techniques

Formal Specification technique is further divided into two sub techniques that are Model Checking and Theorem Proving, depicted in table 3. Model checking consists of various features like finite state concurrent system, satisfy desirable property, and check correctness of model and formal language with state transition. Theorem Proving technique involves features like formal language with state transition, check reachability, conformance and equivalence.

TABLE 3
COMPARISON OF FORMAL SPECIFICATION SUB TECHNIQUES

Features	Formal Specification Technique	
	Model Checking [10,11]	Theorem Proving [10,11]
Finite state concurrent system	√	
Satisfy desirable property	√	
Check correctness of model	√	
Formal language with state transition	√	√
Check reachability, conformance and equivalence		√

C. Review Sub Techniques

Review technique is further divided into four sub techniques that are Fagan's inspection process, Walk through, Reading and Commenting, shown in table 4. Fagan's inspection process consists of various features like informal, find defects of product, remove errors in early stage, less complex and save reworking. Walk through technique involves features like informal, developer technique, improve quality and find defects. Reading techniques involves features like series of steps for reading and specific methods. Commenting technique involves features like find defects of product.

TABLE 4
COMPARISON OF REVIEW SUB TECHNIQUES

Features	Techniques			
	Fagan's Inspection process [6,8,9,10,11,12]	Walk Through [10,11,12]	Reading [6,9,10,11]	Commenting [10,11]
Informal	√	√		
Find defects of product	√			√
Remove errors in early stage	√			
Less complex	√			
Save reworking	√			
Series of steps for reading			√	
Specific methods			√	
Developer technique		√		
Improve quality		√		
Find defects		√		

Fagan's Inspection process Sub Techniques

Fagan's inspection process Technique is further divided into sub technique that is Test case driven inspection. Test case driven inspection consists of various features like pre project phase, complete requirement for product and planning, minimize cost, market driven environment with large requirement, and minimize resources and reusable.

Reading Sub Techniques

Reading Technique is further divided into sub six techniques that are Ad hoc based, Check list based, Perspective, Defect based, Scenario based and Pattern based. Ad hoc based consists of various features like inspect without guidelines and reviewer experience and knowledge based. Check list based consists of various features like standard checking and quality question for inspector. Perspective based consists of various features like assigned specific perspective, detect specific faults, structured manner working, scenario based and Represent actual requirement. Defects based technique used to identify data objects. Scenarios based consist on different scenarios. Pattern based consist on various features like scenario, patterns and formal validation frame.

D. Model Based Sub Techniques

Reading Technique is further divided into sub six techniques that are Data flow, Compositional, Classified, Stimulus response, Process and Simulation, presented in Table 5. Data flow consists of various features like DFD, functional decomposition and top down activity. Compositional consists of various features like ERD and crow's foot notation. Object diagram is built in classified technique. Stimulus response consists of various features like STD, system reaction on internal or external events. Process model is built in process technique. Simulation consists of various features like Simulation model, reduce failures and easy to detect errors and performance factors.

TABLE 5
COMPARISON OF MODEL BASED SUB TECHNIQUES

Features	Techniques					
	Data Flow [6, 10, 11]	Compositional [6, 10, 11]	Classified [6, 10, 11]	Stimulus Response [6,10,11]	Process [6,10,11]	Simulation [6,10,11]
DFD	√					
Functional decomposition	√					
Top down activity	√					
ERD		√				
Crow's foot notation		√				
Object diagram			√			
STD				√		
System reaction on internal or external events				√		
Process model					√	
Simulation model						√
Reduce failure						√
Easy to detect errors and performance factors						√

E. Advantages and Disadvantages of Requirement Validation Technique

Table 6 represents advantages and disadvantages of different Requirement Validation techniques and relationship between them.

TABLE 6
ADVANTAGES AND DISADVANTAGES OF VARIOUS REQUIREMENT VALIDATION TECHNIQUES

Techniques		Advantages	Disadvantages
Prototyping		Understandable requirement. Convert from sketches or pictures to high level fourth generation languages.	Used for specific perspective. Used for requirement elicitation and validation phase.
	Throwaway	Both customers and developers agreed on the requirements than prototype remove.	Initial phase to experiment phase and later on it is discarded. Develop requirements that are difficult for customers to understand.
	Evolutionary	Easy to understand requirement for customers. Workable system for end users. High priority requirement.	Best when developers have minimum contact with the stakeholders. Performance, design quality, and maintainability are quality factors which effect on it. Steps must be clearly defined. Cannot develop for AI and user interface systems.
Testing Based		It contains requirements integrity, consistency, data's sharp definition, requirement ambiguity, and testability. Systematic approach for test case design. System according to the requirement. All requirements must be tested. Functional requirements of specification document are analyzed, check and define test.	Difficult to develop test cases for specific problems. Missing requirements. Requirements are not clearly described. Only for validate requirement not for system validation. Complete information for test cases.
Writing User Manual		Expose problems earlier. Detail focus on requirements. Explanation of the functionality and implementation. Concepts to remove errors. System installation and usage. Requirement related to usability.	Done at some levels. Beneficial if the application is rich in user interfaces or usability requirements.
Formal Specification		Having desirable properties (completeness, consistency, mutual exclusion). Checking all execution paths of specification.	
	Model Checking (for FSMs and Temporal logic)	Automatic technique for proving finite state concurrent system. Confirm system which satisfies specified property. Correctness model are verify.	Expressed in formal language with state transition semantics.
	Theorem Proving (More General for any Formal Specification)	State machine base. Check reachable, handle, deadlock, conformance, equivalence.	
Reviews		Involve readers on both sides (clients and developers). It helps customers and developers to resolve problems at early stages of SDLC. Widely used techniques. Misplaced requirement are cleared. Conflicts, omissions, inconsistencies and errors are found out. Check verifiability, Comprehensibility, Traceability, Adaptability.	Time consuming. Independent review teams usually produce good quality systems.
	Commenting		
	Fagan's Inspection process	Less complex. Effective way For Error Detections of software product. Remove errors in early stage. Find 50% to 90% defects. Save reworking.	Costly. Time consuming. Large amount of software artifacts are required to analyzed, Searched and Sorted.

Techniques				Advantages	Disadvantages
			Test Case Driven Inspection	Remove defects before project start. Ensure requirements are sufficient for product and planning activities. Less costly and effective. Used in market driven environment (Large number of requirement). Minimize resources. Reusable artifacts.	
	Walk Through			Developer technique. Used to improve quality. Emphasis on finding errors.	
	Reading			Define a series of steps to review or inspect the software artifact.	Specific methods. Inspection team required.
			Ad Hoc Based	Artifacts are given to the reviewers for inspection without any guidelines.	Little reading support. Error exposure is depending upon the knowledge and experience of reviewer/inspector. Inspection team required.
			Check List Based	Usually used techniques. Contains conventional questions which help the reviewer/inspector. Question related to quality.	Standard reading technique. Questions are more general. Missing Instructions. Designed question which are previously detected defect type. Inspection team required.
			Perspective	Specific perspective is helpful to find specific type of error or defects. Reviewers are more focus. Working in a structured manner and read actively. Signifies customer, tester and the designer for scenario based reading.	Inspectors are assigned specific perspective. Inspection team required.
			Defect based	Identify data objects declared in views (hardware component, application variables). In which section data objects declare (Input or Output).	For every functional requirements data objects identify. Any data object appearing in the external interface multiple requirements needed.
			Scenario based (Desk Testing Requirement)	Set of tangible developments provides. Sequence of event walk through in every scenario. Inspect presence and correctness of requirement document. Reviewers prepare scenarios.	Information is not written by inspector only in mind.
			Pattern based	Used Patterns to review against scenario. Machine function pattern. Collect first objective last pattern.	
Pre-Review Checking				Early error detection by single person. Save cost and time from full requirements reviews.	Errors which can be easily detected without full review are discovered. One person is dedicated to check the documents.
Model Based				Formulation, structured and modeling requirement. Systematic approach for document, analysis and validation. Used variety of modeling techniques.	No single perfect requirement method offered.
	Data Flow Models			Data flow diagrams. Functional decomposition (top-down activity).	
	Compositional Models			Entity relation diagrams are used. Four components entity, relationship, cardinality and attributes.	

Techniques		Advantages	Disadvantages
	Classified Models	Object/inheritance diagrams are used. Fundamental concepts of object oriented modelling are object, classes, methods, messages, encapsulation and inheritance.	
	Stimulus Response Models	State transition diagram. Establish to check the reaction on internal and external events.	
	Process Models	Process Model. Principal and deliverables that are involved in carrying out process.	
	Simulation Models	Simulation Model used. Performance evolution. Reduce failure. Having internal and external consistency. Reflect stakeholder's requirements. Easy to detect errors, inconsistencies and incompleteness.	Difficult for non-technical persons to understand for instance data flow diagrams, event diagrams, or object models. Suggest natural language.
Viewpoint Oriented		Encapsulation of partial information about system requirement. Interactive systems (viewpoints) based on user matters and organization apprehensions. Also use class system. Identifies that a single perspective is not enough to get all requirement. Viewpoint impartial to identify problem related to correctness, completeness, and inconsistency.	System expert describes the Faults.
Tracing Approaches		Checks are done by tracing. Elicitation notes are covered. Goal against requirement, features and task are checking. Developing traceability matrix.	
Simulation		Abstract model (requirement or design solutions). Scenario based.	
Functional Test Design		Some software test before programming (Agile methods). At system level (Sooner or later). Each test case design to its requirement.	Non-functional or exclusive are not testing. Missing or ambiguous create problem. Designing functional test design expose errors in the specification (before designing and complete system).

In this research work, we take three validation techniques for DVEs requirement validation model that covers all scenarios and complexity features of DVEs. As a common practice no input and user interaction is involved in creation of DVEs. In this model, we have tried to facilitate users by getting their suggestions and inputs as a requirement specification for developing new DVEs or a module in existing DVE systems. User can also report solutions against any problem facing by him during the use of DVE system. This opens new horizon in DVEs, a combination of RVM techniques and DVE is used to facilitate not only user but to reduce some major issues in DVEs such as load balancing and minimize throughput of whole system. Various RVMs are used but we prefer three techniques that fully clarify requirement validation model in this research work.

III. PROPOSED MODEL

Requirement Validation Model involves end user feedback in a proposed system. In Distributed Virtual Environment (DVE), Requirement Validation Model (RVM) is not used as a common practice. This research depicts the role of RVM in DVEs and design RVM for DVEs by implementing Validation Specific Technique. The proposed model is focused on facilitating the users by solving their concerns and improves the quality of existing system through user's input.

A. Requirement Validation Model for DVEs

In this section, proposed Requirement Validation model has been discussed. The proposed model is shown in Figure 1. For the validation of proposed model, three case studies named Skype, Facebook and a game UNO has been picked. Brief description of each case study along with the implication of proposed model is provided in subsequent sections.

B. CASE STUDY: Skype

Skype is the most widely used software application that

allows users for Instant Messaging (IM), make Voice and Video calls over the Internet. This software also facilitates users with some additional features by allowing them to:

- Share a story
- Celebrate a birthday
- Hold a meeting, work with colleagues etc.

According to the user review, considering a situation where the users have given suggestion for providing facility to privatize their profile by allowing them to set the privacy on who can or cannot access their account. The requirement (suggestion) given by users is forwarded to requirement engineer.

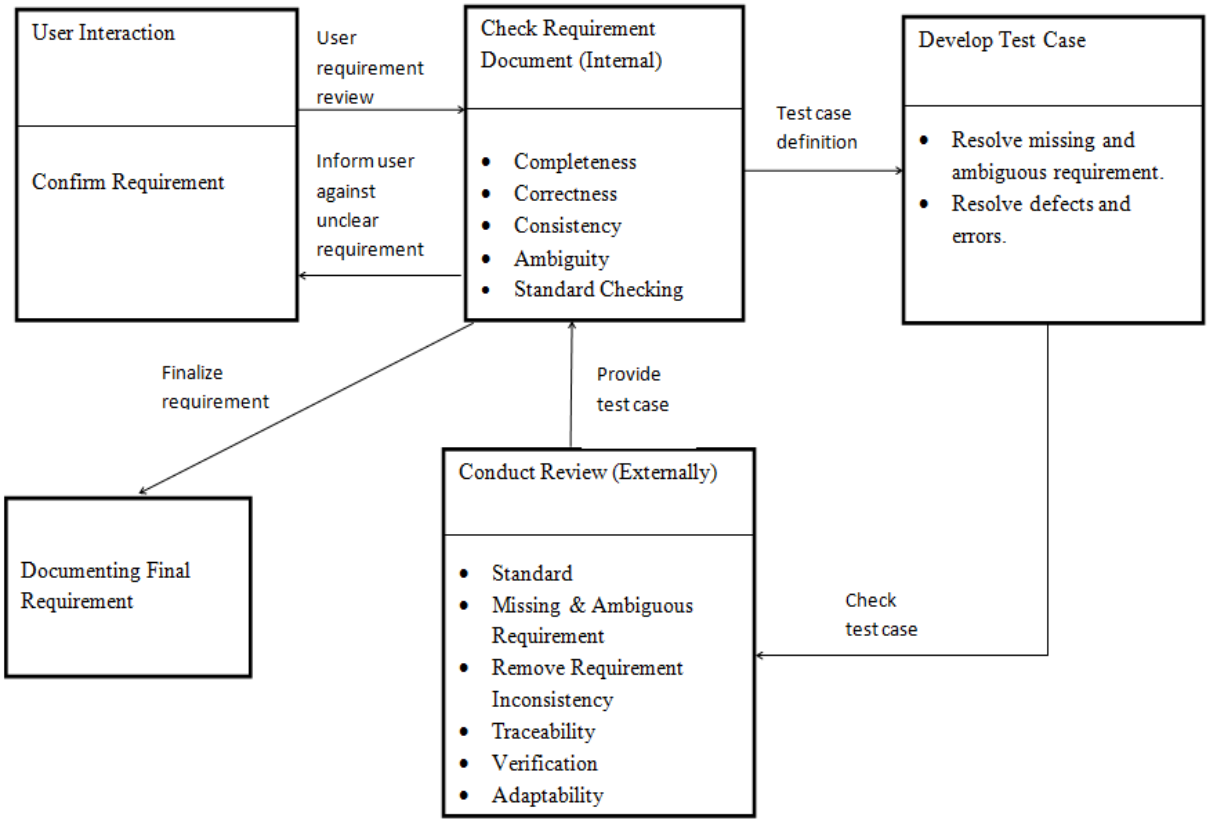

Fig.1: Requirement Validation Model for Distributed Virtual Environment system

Requirement engineer is responsible for ensuring that the given requirement is clear and can be incorporated easily without disturbing the functionality of other modules. After confirming the requirement, the requirement engineer will check in the requirement document whether the confirmed requirement:

- Can be easily incorporated
- Clearly understandable
- Easily implementable
- Is consistent with other modules
- Can fulfill the standard followed by the software developer.

If the requirement is unclear, ambiguous and inconsistent, the requirement engineer then informs the user against inappropriate requirement.

Once the requirement engineer has completely verified the requirement, he then develops test case for that requirement with purpose of resolving the missing, erroneous, defected and ambiguous requirement if any. After developing test cases, the requirement engineer conducts a review session with experts. The experts are making sure that the test cases are:

- Correctly developed
- According to standard
- Clear and understandable
- Adjustable in specified environment

On completion of review session, the test cases along with reviewer comments are sent to the requirement engineer. The requirement engineer critically analyzes reviewer's comments and will then incorporate the finalized requirement into the requirement document.

C. CASE STUDY: Facebook

Facebook is considered to be one of the most widely used social networking website being used by number of users these days. It allows users to:

- Send Instant Messages (IM)

- Share thoughts
- Share pictures and videos
- Create events
- Advertising etc.

In accordance with the user review, considering a situation where the users have given suggestion that there should be clear guidelines about how to adjust the size of profile photos. The given user's requirement (suggestion) is forwarded to requirement engineer. Requirement engineer is liable for ensuring that the given requirement is clear and can be incorporated easily without disturbing the functionality of other modules. Once the requirement is confirmed, the requirement engineer checks the requirement document to see whether the confirmed requirement is:

- Easy to incorporate
- Clearly understandable
- Easily implementable
- Consistent with other software modules
- According to the defined standards

If the requirement engineer finds that the requirement is unclear, ambiguous and inconsistent, he notifies the user against inappropriate requirement.

Once the requirement engineer has completed requirement verification process, he then develops test case for the confirmed requirement with the aim of resolving the missing, erroneous, defected and ambiguous requirement if any. After developing test cases, the requirement engineer conducts a review session with domain experts. The experts are responsible ensuring that the test cases are:

- Correctly developed
- According to standard
- Clearly understandable
- Flexible in specified environment

Once the review session is completed, these test cases along with reviewer comments are forwarded to the requirement engineer. The requirement engineer is the responsible for critically analyze reviewer's comments and incorporating the finalized requirement into the requirement document.

D. CASE STUDY: UNO Game

UNO is an online card game in which players try to get rid of all the cards which they are holding in their hands. The players take their turn by following the card placed during start of the game. If the player fails to follow the placed card, they must draw card until they can play again. The game continues until some player runs out of card. In this game, players must say "UNO" when they have one card left. If any other player notices that any player having one card has not said "UNO", then that player will draw two cards from the pile as penalty.

Consider a situation where the user has given requirement (suggestion) to increase the time limit of every player's turn. The requirement (suggestion) given by users is forwarded to requirement engineer. Requirement engineer is responsible for ensuring that the given requirement is clear and can be incorporated easily without disturbing the functionality of other modules. After confirming the requirement, the requirement engineer will check in the requirement document whether the confirmed requirement:

- Can be easily incorporated
- Clearly understandable
- Easily implementable
- Is consistent with other modules
- Can fulfill the standard followed by the software developer.

In this case, the requirement engineer finds it difficult to incorporate this requirement because by exceeding the time limit, overall game's functionality is disturbed. So in this situation, the requirement engineer considers this requirement to be unclear, unambiguous and inconsistent and will inform the user against inappropriate requirement.

IV. DISCUSSION

After implementing a proposed model on different case studies as mentioned above it has been found that the proposed model can facilitate requirement engineer to gather appropriate requirement before going to develop a new system/application or modify existing systems/applications that are for distributed virtual environment. This is so because this model gives major importance to users by allowing them to provide their own requirement which they wish should be present in the system.

During implementation, it has also been found that the proposed model only validate those requirements which could be incorporated. All those requirements that are useless and couldn't be incorporated are discarded by the requirement engineer after getting comments from experts during review process.

Therefore, we can say that this model can help all those software development companies who are developing various systems or applications for distributed environment.

V. CONCLUSION & FUTURE DIRECTION

In this research paper, we have presented a framework that develops Distributed Environment after the validation of requirements provided by end users. The method makes easier the development of DVE after requirement specification phase while "validating" the formal description of the requirements. It is based on three main processes. The first process defines the submission of requirement from user. The second is an evaluation process based on correctness and completeness of requirement in developer perspective. After validating and analyzing the requirements, mapping of these requirements in real time on DVE system comes next. The third is a review phase from a third party to analyze requirement validations.

The framework is used to validate all requirements that may be use in system development. The main contributions of our work are: present a set of observation patterns which covers all the discussed techniques and some examples have been

exposed in this paper. Our framework allow user to express the requirements and suggestions for existing and new DVEs systems. A better quality products and systems can develop by using this approach that provides a feel of satisfaction and contentment to the end user. Currently this framework is proposed only for DVES while in future it can be used in agile developmental approach.

REFERENCES

[1] Jiang, L., & Eberlein, A. (2007, March). Selecting Requirements Engineering Techniques Based on Project Attributes--A Case Study. In Engineering of Computer-Based Systems, 2007. ECBS'07. 14th Annual IEEE International Conference and Workshops on the (pp. 269-278). IEEE.

[2] Kuijpers, N., & Jense, H. (1997, March). Collaborative engineering in distributed virtual environments. In Proceedings of the 1997 Spring Simulator Interoperability Workshop, Orlando, FL.

[3] Wang, T., Wang, C. L., & Lau, F. C. (2006). An architecture to support scalable distributed virtual environment systems on grid. The Journal of Supercomputing, 36(3), 249-264.

[4] Haumer, P., Pohl, K., & Weidenhaupt, K. (1998). Requirements elicitation and validation with real world scenes. Software Engineering, IEEE Transactions on, 24(12), 1036-1054.

[5] Petersen, S. A., Lillehagen, F. M., & Anastasiou, M. (2006, May). Interoperablity Requirements Elicitation, Validation and Solutions Modelling. InICEIS (3) (pp. 152-159).

[6] Saqi, S. B., & Ahmed, S. (2008). Requirements Validation Techniques practiced in industry: Studies of six companies. Blekinge Institute of Technology, Sweden.

[7] Singh, H. L., & Gracanin, D. (2012, March). An approach to Distributed Virtual Environment performance modeling: Addressing system complexity and user behavior. In Virtual Reality Workshops (VR), 2012 IEEE (pp. 71-72). IEEE.

[8] Yousuf, F., Zaman, Z., & Ikram, N. (2008, December). Requirements validation techniques in GSD: A survey. In Multitopic Conference, 2008. INMIC 2008. IEEE International (pp. 553-557). IEEE.

[9] Raja, U. A. (2009, February). Empirical studies of requirements validation techniques. In Computer, Control and Communication, 2009. IC4 2009. 2nd International Conference on (pp. 1-9). IEEE.

[10] Pohl, K. (2010). Requirements engineering: fundamentals, principals and techniques. Springer Publishing Company, Incorporated.

[11] MacKenzie, S. B., Podsakoff, P. M., & Podsakoff, N. P. (2011). Construct measurement and validation procedures in MIS and behavioral research: integrating new and existing techniques. MIS quarterly, 35(2), 293-334.

[12] Balci, O. (1994). Validation, verification, and testing techniques throughout the life cycle of a simulation study. Annals of operations research, 53(1), 121-173.