FIVE LAYERED MODEL FOR IDENTIFICATION OF SOFTWARE PERFORMANCE REQUIREMENTS

Gopichand.Merugu[1] and AnandaRao.Akepogu[2]

[1] Associate. Professor, CSE, Department, BVRIT,Narasapur ,Andhrapradesh,India
gopi_merugu@yahoo.com
[2] Principal & Professor, JNTUA, Anantapur,Andhrapradesh,India
akepogu@yahoo.co.in

ABSTRACT

Identification of performance requirements is important for successful development and deployment of the software product. The acceptance of the software product by the customer depends on the performance requirements which are incorporated in the software. For this, we need to identify all the performance requirements required by all stakeholders. In the literature not many approaches are available for this purpose. Hence, we have proposed a five layered model for identification of performance requirements. The proposed layered model has many advantages over non-layered approaches. As part of this model some rules are also proposed to be used in performance requirements identification process. The layered model is applied successfully on two case studies. The identified performance requirements are validated using a check list and in addition the completeness of the identified performance requirements is computed using a metric.

KEYWORDS

STAKE HOLDER, GOAL, SUB GOAL, NON-FUNCTIONAL REQUIREMENTS, PERFORMANCE REQUIREMENTS, LAYERED MODEL.

1. INTRODUCTION

Performance is an important property for a software system. If an on-line shopping Web site does not offer responsive experience, a user may lose his patience and shop with its competitor. Even NASA had to delay the launch of a satellite for more than eight months because the Flight Operations Segment software had unacceptable response times for developing satellite schedules. Poor system performance has been called the single most frequent cause of the failure of software projects [19]. The causes of poor performance are inadequately specified performance requirements and poor architectural choices. Performance requirements may be vaguely written, or might not even have been written at all by the time the project is close to completion.

The absence of performance requirements increases the risk that performance will receive inadequate attention during the architectural, development, and functional testing phases of a software project. Performance requirements are drivers of computer and software architecture.

DOI : 10.5121/ijsea.2012.3505

Since performance problems often have their roots in poor performance requirements specification, the early establishment of performance requirements for a new system is crucial to the project's success. Performance requirements describe the expected performance of the software system. Ideally, performance requirements should be specified and validated before detail design starts. Performance requirements are most important Non-Functional requirements to consider in software development. There are few methods in literature for identification of performance requirements. But they are not that much effective and complete in identification of performance requirements. Each method has its own strengths and weaknesses. But if we use these approaches, it is probably difficult to identify the performance requirements of the system completely and correctly.

Due to gaining importance of performance requirements, in recent times, researchers are showing lot of interest in this area. Most of the times, non-compliance of performance requirement by the software is due to improper identification models. In this direction, we proposed layered model for performance requirements identification. This paper consists of five sections. Following the introduction, section 2 briefly describes the existing methods for performance requirements identification and their strengths and weaknesses. The proposed five layered model for identification of performance requirements is discussed in section 3. Section 4 discusses how the layered model has been applied on Library management system and online shopping. Section 5 provides conclusion.

2. RELATED WORK

A few performance requirements identification methods have been proposed in the literature. However, some factors that can affect the system performance may not be available during early phases of software development. For example, performance models, such as queuing network models, are widely used for performance prediction, and the output of the models can be used to validate the feasibility of Performance Requirements. The earliest decisions that can constrain the performance of a software system are made during requirements analysis, when the sequence of operations is developed. There is an approach which considers the stage of developing the system Use Cases into scenarios to be executed, with sequences of responsibilities and with data operations. The specification uses Use Case Maps, which are then analyzed for performance. A second opportunity for performance analysis arises a little later, after the architectural design stage.

We can see very few approaches have been taken to early performance analysis, for both of these stages. The first approach is a qualitative analysis in terms of possible impacts of system aspects on product qualities such as performance. For example, Chung et al. have developed a general approach for analyzing non-functional requirements that they call NFR [6], which has been applied for performance by their co-author Nixon . A similar qualitative analysis applied to the slightly later stage, of system architectural design, is described by Bass, Clements, and Kazman [2]. The second approach is quantitative, using models. Smith developed an approach to building a queuing model based on scenario like "execution graphs" [17] [19] that are specially built for performance analysis. Balsamo, Inverardi and Mangano described an architecture modelling technique [14], and a general framework for modelling called "resource architecture" is described in [21].

The difficulties posed by early analysis, some of which are: Incompleteness of the specification, because it is so abstract. This includes open choices of design approaches, algorithms and components to be used, and lack of definition of the final execution environment lack of knowledge of the computational effort required other aspects such as ignorance of the workload intensity. There is an approach for gathering performance requirements which bases on three views like deployment View to capture the hardware and software environment, constraints and service-level agreements for the deployment environment, Operational Workload View to capture the different types of workloads that the system is subjected to, the pattern and intensity of the requests during the different workloads intervals, the desired responsiveness and the growth projections; and Persistent Data View to capture the current data size and the data growth over a period of time. These views capture the key attributes from business and technological standpoint. But these three views will not present all the performance requirements as many other views are there which are still need's to be considered.

3. PERFORMANCE REQUIREMENTS IDENTIFICATION

As the performance requirements identification is an important activity in software development, in this paper we have proposed a layered model for identification of performance requirements. This model includes five layers like stakeholder, goals, sub-goals, performance parameters, and performance requirements. The first layer is about stakeholders. The term stakeholder is used to refer to any person or group who will be affected by the system directly or indirectly. Single view forces us to look at the requirements only from a particular perspective. In order to identify the performance requirements completely multiple views needs to be considered to meet all stakeholder expectations. Second layer is goals. In this layer all the goals of stakeholders will be identified. Third layer is sub goals. In this layer the goals identified in second layer can be decomposed into sub goals. Fourth layer is performance requirements. In this layer all the performance requirements for corresponding sub goals will be identified. Fifth layer is performance parameters. After identifying the performance requirements, the respective performance parameters for performance requirements will be identified. This is a Five Layered approach to performance requirements identification. This approach includes some rules and layered model for performance requirements identification. Using this approach we can identify the goals, sub goals and finally performance requirements. This is used to identify the all performance requirements from multiple views of different stakeholder. The main objective of this approach is to find out performance requirements which are very important to consider in any system. The five layered model is shown in figure 1.

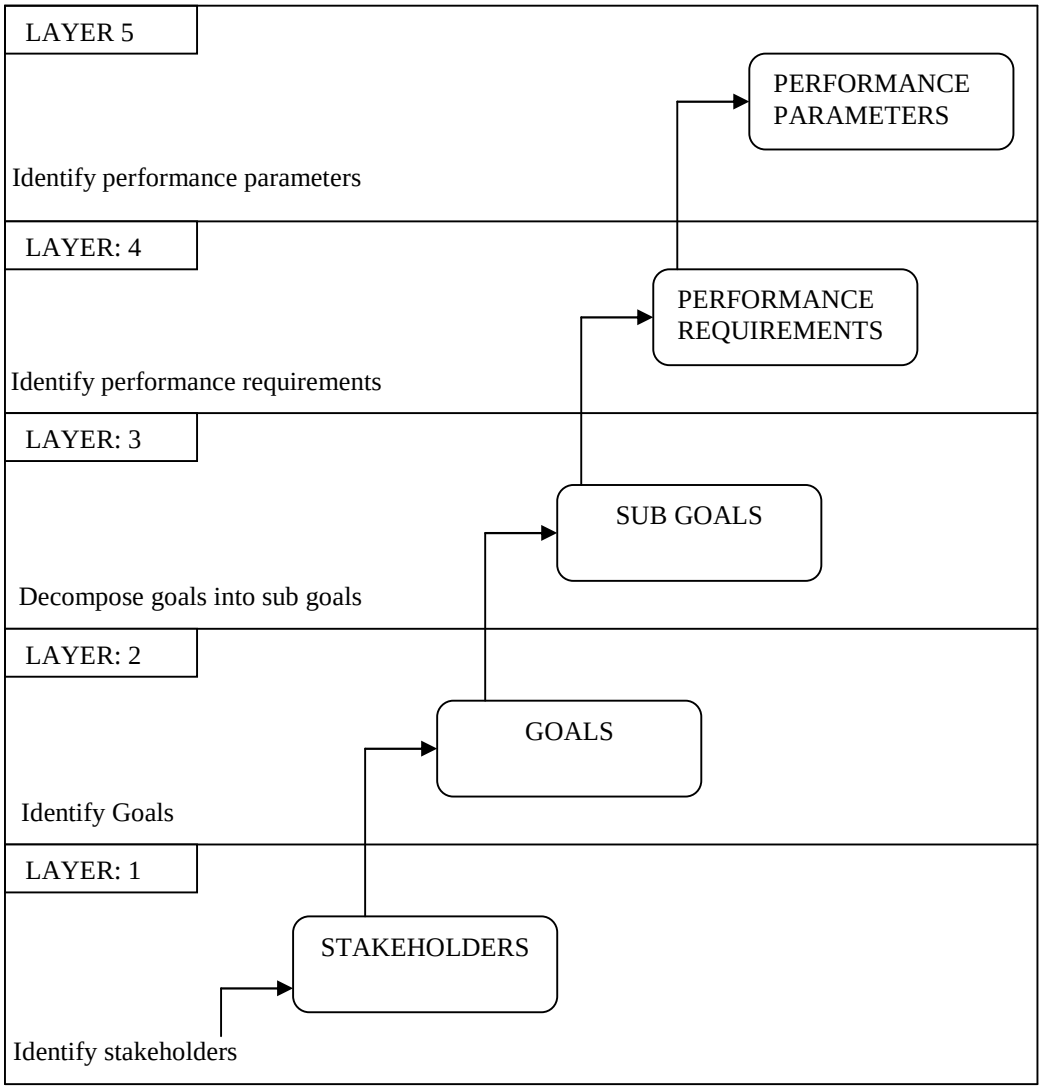

Fig 1: General Architecture for Five layered model for Performance requirements identification

The layered approach offers many advantages, and they are listed below.

- Scalability : A layered approach scales better when compared to horizontal approach
- Better Flexibility: Layered approach improves flexibility in terms of options and choices.
- Cost Effective: The layered approach is the most economical way of developing and implementing any system. In this context developing a system means, identifying performance requirements for the system development depends on how well we identify the requirements.
- Task segmentation: Breaking large complex systems into manageable subcomponents allows for easier development and implementation of any system.
- Enhanced Understanding: Layering allows for examination in isolation of subcomponents (processes/procedures) applicable for each layer and as well the whole.

- Rapid Application Development: Using the layered Approach requirements can be identified in less amount of time which leads fast application development.

With the five layered model we have proposed some rules which are used for identification of goals, sub goals and performance requirements.

Any software system requirements can be put broadly in the following two statements.

Each <system> provides <services> to <user>
Each <service>must satisfy some <constraints> in order to meet customer needs.

Functional requirements state <what> the system supposed to do, and Non functional requirements state <how> the system supposed to be.

The following rules are used in the above process.
The rules are:
<Who> are the stakeholders?
<What> are the services (goals)
<What > are the sub goals of each service?
<How> the sub goals are achieved under performance constraints.

The above rules can be visualized in the following rule.

<Who> is stakeholder <What> are the services (goals) that system should provide to the stakeholder, <What> are the sub goals for the goals, <How> the system will perform sub goals under constraints

Key words like <who>, <what>, <how> are variables which can be assigned for specific words in the given context.

Key words used in the rules are:
Who: Stakeholders
What: Functional requirements/services
How: Performance requirements/constraints

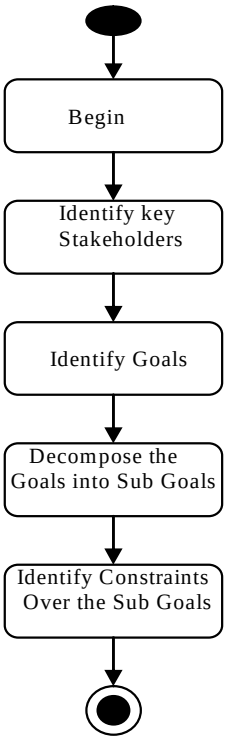

Fig. 2. Performance requirements identification process

In this paper the mains focus is on performance requirements (constraints). The performance requirements identification process can be divided into the four steps as follows and shown in activity diagram. (Fig. 2).
Step 1: Identify the key stakeholders of the system.
Step 2: Generate the goals from Stakeholders based on developers' Knowledge and experience.
Step 3: Decompose the goal into sub goals.
Step 4: Identify performance requirements for each sub goal.

3.1 Metrics for performance requirements specification

Metrics proposed by Davis[24] for identifying and measuring the quality in software requirements specification is used in this paper.
To determine completeness of overall requirements the metric used is:
$MCR = n_c / [n_c + n_{nv}]$
Where n_c is number of requirements has been validated as correct and n_{nv} is number of requirements that have not yet been validated. In this paper we have used this equation only for finding the completeness of the performance requirements.

3.2 Performance Requirements Validation

The following check list is used for validating software performance requirements [23]:

- Does each requirement have source.

- Is each requirement achievable in the technical environment that will house the system?
- Each requirement is testable once implemented.
- Is each requirement bounded and unambiguous?
- Do any requirements conflict with other requirements?
- Is the requirement traceable to goals of the system?
- Is the requirement is bounded in quantitative terms.
- Are requirements stated clearly? Can they be misinterpreted?

To validate our approach we have used the above mentioned check list which includes important questions.
We collected the information in the form of answers for questions from various people both from Library System and online shopping system about performance requirements.

3.3 Finding critical performance parameters.

We can find critical performance parameters based on performance requirements. The traceability matrix is used for finding critical performance parameters.

Table1. Traceability Matrix for Finding Critical Performance parameters

	G1	G2	G3	G4	Gm
PP1			x		
PP2		x	x		x
PP3				x	
PPn			x		

In the above table the PP1, PP2, PP3, PPn represents performance parameters and G1 to Gm represents Goals. From the table we can say that PP2 is critical, since many goals require PP2.

4. CASE STUDY

In this section we have shown how the proposed model helps to identify the performance requirements. We considered two case studies these are online library management system and online shopping system. These systems are used by different stakeholders having different goals covering different perspectives. The proposed rules as part of this approach are used in identifying the performance requirements in two case studies.

4.1 Library management system

The library system is used by many stakeholders like borrower, librarian and having their own goals and constraints. The constraints are identified using the rules.

Rule 1: <who> are the stakeholders?
The identified stakeholders are given in table 3.
Rule 2: <What> are the services (goals)
The identified services (goals) for all the stakeholders are given in table 3.
Rule 3: <What > are the sub goals of each service

The identified sub-goals for all the stakeholders are given in table 3, and sub goals for stakeholder "Member" are shown in figure 3.

Rule 4: <How> the sub goals are achieved under performance constraints. The identified performance requirements for all the stakeholders are given in table 3, and performance requirements for stakeholders "Member" and "Librarian" are shown in figure 3.

The metric MCR is computed for Library management system. The computed value of MCR for library management system is :

$MCR = n_c / [n_c + n_{nv}]$
$MCR = 21 / [21+0] = 1$

The closer the value of MCR to 1 the maximum is the completeness of the requirements. In this paper we considered this equation for performance requirements only. The requirements are validated against the checklist given in section3.2. It is found that for all 8 questions the answer is yes. The validation metric is 8/8 = 1, Hence validated. Critical performances Requirements for Library management system are identified using traceability matrix given in table 2.

Table 2: Traceability Matrix for Finding Critical Performance parameters for Library management system

	G1	G2	G3	G4	G5	G6	G7	G8
R	X	X	X	X			X	X
T					X	X		
RU							X	X
E								

Here we made an effort for identifying critical performance parameters. The above traceability matrix is for finding the critical performance parameters for library management system. The notations represented in rows R, T, RU and E are response time, throughput, Resource utilization and Execution demand. The corresponding columns represent goals like login, Search book, Borrow book, Return book, Add Member, Add Item, Issue Book, and Receive Book which are from Figure 3. So from the above table we can say that response time and throughput are more critical than other performance parameters. The critical performance parameters which are identified must be implemented in the system. The acceptance of the system by the user is more likely if critical performance parameters are implemented in the system

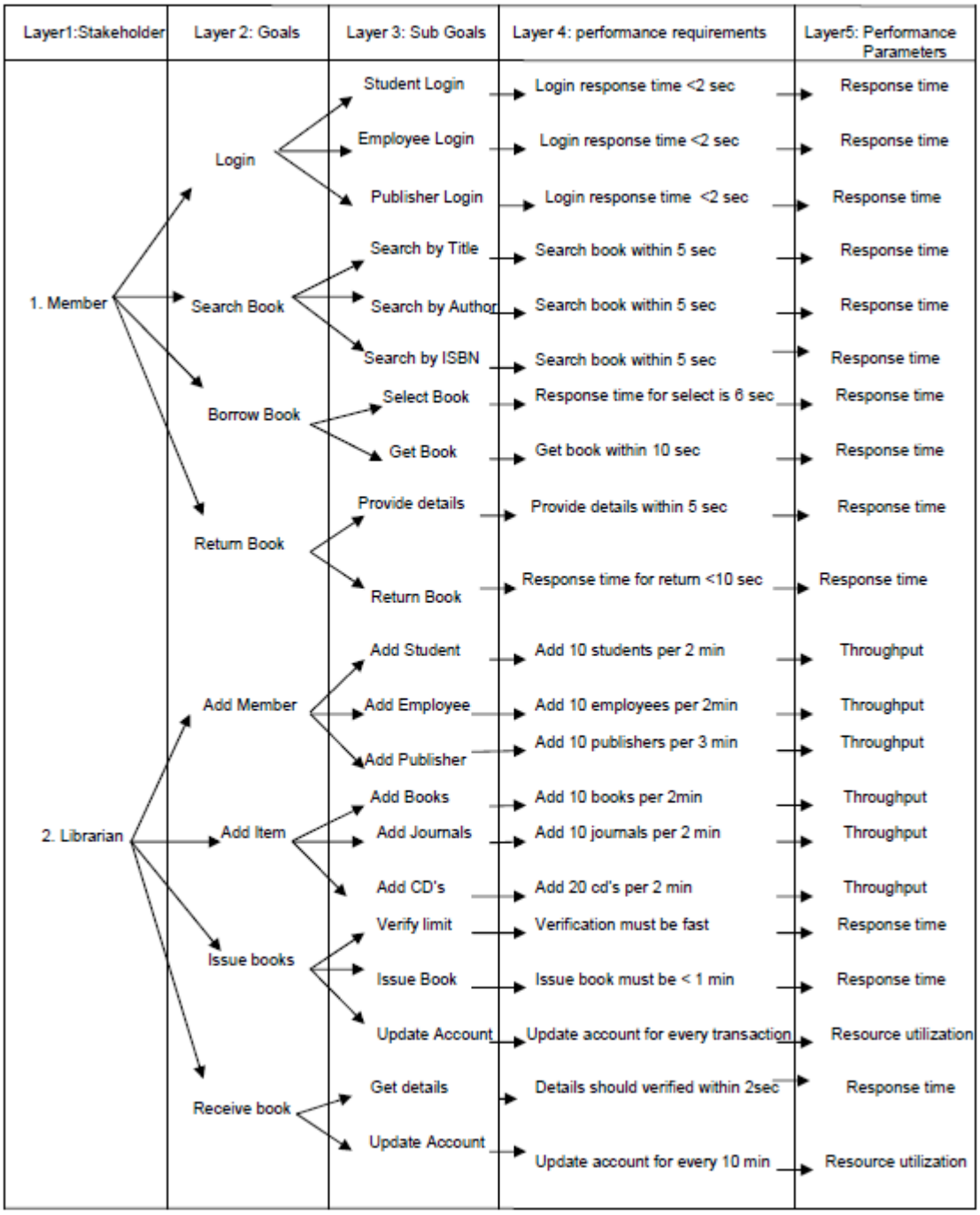

Fig.3 Five layered performance requirements identification sample for Library Management System

Table 3: Performance Requirements identification sample for Library Management System

System	Library management system
Stakeholders	Member, System Administrator, Librarian
All Goals for all stakeholders	Login, view catalogue ,Search book, Reserve Book Barrow book, Return book, pay fine Add item, update database, Register Member, Check Reports, Issue Book
All Sub goals for all goals	Student login ,Employee login, Publisher login , view by subject, view by course, view by publisher , Search book by author , Search book by title ,Search book by ISBN, Request for book, Add book, Add journal, Add Cd's, Register student, Register Faculty, Register Publisher, Update book information, update barrower information, view the report, Edit report, Get book
All Performance requirements	Login response time must <2 sec, Search book within 5 sec, Response time for select is 6 sec, Get book within 10 sec, Provide details within 5 sec, Response time for return <10 sec, Add 10 students per 2 min,, Add 10 employees per 2min, Add 10 publishers per 3 min, Add 10 books per 2min, Add 10 journals per 2 min, Add 20 CD's per 2 min, Verification must be fast, Issue book must be < 1 min, Update account for every transaction, Details should verified within 2sec, Update account for every 10 min
Performance parameters	Response time, Throughput, Resource utilization, Execution demand

4.2 Online shopping System

In online shopping system main stakeholders are like customer and system. Each stakeholder will have their own goals and constraints. The constraints are identified by using the rules.
Rule 1: <who> are the stakeholders?

The identified stakeholders are given in table 5.

Rule 2: <What> are the services (goals)

The identified services (goals) for all the stakeholders are given in table 5.

Rule 3: <What > are the sub goals of each service

The identified sub-goals for all the stakeholders are given in table 5, and sub goals for stakeholder "Customer" are shown in figure 5.

Rule 4: <How> the sub goals are achieved under constraints.

The identified performance requirements for all the stakeholders are given in table 5, and Performance requirements for stakeholder "Customer" are shown in figure 4.

The Performance constraints identified for the stakeholder "customer" are given in figure 4 and all the PRs for all stakeholders are given in table 5. The layered analysis for stakeholder "Customer'' in first layer is given in figure 4.

The metrics MCR is computed for online shopping system.
The computed value of MCR for online shopping system is :

$MCR = n_c / [n_c + n_{nv}]$
$MCR = 16 / [16+0] = 1$

The closer is the value of MCR to 1 the maximum is the completeness of the requirements.
The requirements are validated against the checklist given in section3.2. It is found that for all 8 questions the answer is yes. The validation metric is the 8/8 = 1. Hence identified PRs are validated.

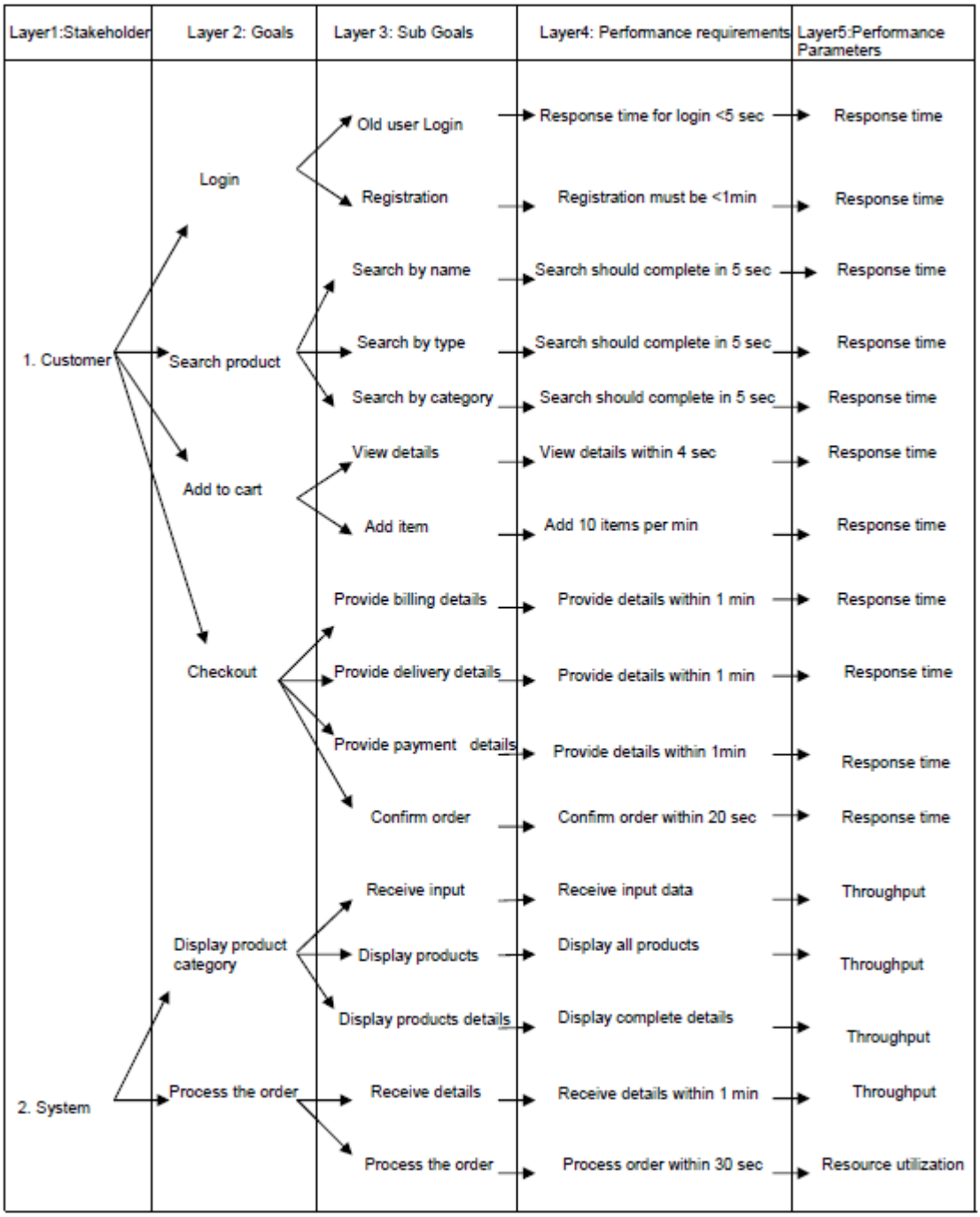

Fig 4: Five layered performance requirements identification sample for online shopping system

Table 4: Traceability Matrix for Finding Critical Performance parameters for online shopping system

	G1	G2	G3	G4	G5	G6
R	X	X	X	X		
T					X	X
RU						X
E						

Here we made an effort for identifying critical performance parameters. The above traceability matrix is for finding the critical performance parameters for online shopping system. But here we considered only one stakeholder i.e. customer. The notations represented in rows R, T, RU, E are Response time, Throughput, Resource utilization, and Execution demand. The corresponding columns represent Goals like login Goals like login, search product, add to cart, checkout, display product category, and process the order, these are from figure 4. So from the above table we can say that Response time and Throughput are more critical than other parameters. In the above figure we have not considered complete Goals for the system. The critical performance parameters which are identified must be implemented in the system. The acceptance of the system by the user is more likely if critical performance parameters are implemented in the system.

Table 5: Performance requirements identification sample for online shopping system.

System	Online shopping system
Stakeholders	Customer, System
All Goals for all stakeholders	Login, Select product category , Select product, Add to cart, Check shopping cart, Checkout, Confirm order
Sub goals for all goals	Old user login, New user registration, Search product by name, search product by type, search product by category, View details, Select item, Provide bill details, Provide delivery details, Provide payment details, Confirm order, Receive input, Display products , Display product details , Receive the details, Process the order.
All performance requirements	Login response time must be < 2 sec, search product should be completed within 5 sec, time for add item is 3 sec, checkout should process large amount data, response time process order must be <7, payment must be completed within 30 sec. execution time for checkout 5 sec,
All Performance parameters	Response time, Throughput, Resource utilization, Execution demand

5. CONCLUSIONS

The acceptance of any software product by the customer depends on how well we identify the Performance requirements and incorporates in the software. In this paper we have proposed a five layered model for identifying Performance requirements. As part of this approach we have proposed some rules to be used in the identification process, metrics for performance requirements completeness and Check list for requirements validation. By this model and rules we can identify all the Performance requirements required by all stakeholders. The proposed model is applied successfully on two case studies.

6. REFERENCES

[1] Ian Somerville software engineering seventh edition pages 142-189.

[2] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice. Addison-Wesley 1998.

[3] Boehm, Barry and In, Hoh. "Identifying Quality-Requirement Conflicts". IEEE Software, March 1996, pp. 25-35

[4] Breitman, K. K, Leite J.C.S.P. and Finkelstein Anthony. The World's Stage: A Survey on Requirements Engineering Using a Real-Life Case Study. Journal of the Brazilian Computer Society No 1 Vol. 6 Jul. 1999 pp:13:37.

[5] Chung L., "Representing and Using Non-Functional Requirements: A Process Oriented Approach" Ph.D. Thesis, Dept. of Comp.. Science. University of Toronto, June 1993. Also tech. Rep. DKBS-TR-91-1.

[6] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, Non-Functional Requirements in Software Engineering.Boston: Kluwer, 2000.

[7] Chung, L., Nixon, B., Yu, E. and Mylopoulos,J. "Non-Functional Requirements in Software Engineering" Kluwer Academic Publishers 2000.

[8] Cysneiros, L.M. and Leite, J.C.S.P. "Integrating Non-Functional Requirements into data model" 4th International Symposium on Requirements Engineering – Ireland June 1999.

[9] Cysneiros,L.M., Leite, J.C.S.P. and Neto, J.S.M. "A Framework for Integrating Non- Functional Requirements into Conceptual Models" Requirements Engineering Journal — Vol 6 , Issue 2 Apr. 2001, pp:97-115.

[10] Cysneiros,L.M. and Leite, J.C.S.P. "Using UML to Reflect Non-Functional Requirements" Proceedigns of the 11th CASCON, IBM Canada, Toronto Nov 2001 pp:202-216

[11] Dardenne, A.. van Lamsweerde A, Fickas, S.. "Goal Directed Requirements Acquisition". Science of Computer Programming, Vol. 20 pp: 3-50, Apr. 1993.

[12] Ebert, C. "Dealing with Nonfunctional in Large Software System"s. Annals of Software Engineering, 3, 1997, pp. 367-395.

[13] Fenton, N.E. and Pfleeger, S.L. "Software Metrics: A Rigorous and Practical Approach" 2nd ed., International Thomson Computer Press, 1997

[14] S. Balsamo, P. Inverardi, and C. Mangano, "An Approach to Performance Evaluation of Software Architectures," in Proc. of First International Workshop on Software and Performance (WOSP98), October 1998, pp. 178-190.

[15]K eller, S.E. et al "Specifying Software Quality Requirements with Metrics" in Tutorial System and Software Requirements Engineering IEEE Computer Society Press 1990 pp:145-163

[16] Kirner T.G. , Davis A .M. , "Nonfunctional Requirements of Real-Time Systems", Advances in Computers, Vol 42 pp 1-37 1996.

[17] C. U. Smith, "Performance Engineering of Software Systems". Addison-Wesley, 1990.

[18] Sutcliffe,A.G. and shailey Mincha,"Scenario-based requirement analysis", Non-Functional Requirements. 1998.

[19] C. U. Smith, L.G. Williams, "Performance Solutions", Addison-Wesley, 2001.

[20] Liu, L. and Yu E. "Designing Web-Based Systems in Social Context: A Goal and Scenario Based Approach" 14th International Conference on Advanced Information Systems Engineering (CAiSE'02), Toronto, May 27-31, 2002. LNCS 2348 Springer Verlag. pp. 37-51.

[21] C. M. Woodside, "Software Resource Architecture", Int. Journal on Software Engineering and Knowledge Engineering (IJSEKE), vol. 11, no. 4 pp. 407-429, 2001

[22] VanLamsweerde,A.Goal-Oriented Requirements Engineering : A Guided Tour" Proc of the 5th IEEE Int. Symp. on Requirements Engineering, pp:249-262, 2001.

[23] R.S Pressman , Software engineering Luiz Marcio Cysneiros, Julio César Sampaio do Prado Leit Using the Language Extended Lexicon to support Non Functional requirements elicitation Proceedings of WER'2001. pp.139~153

[24] A. Davis, S. Overmyer, K. Jordan, J. Caruso, F. Dandashi, A. Dinh, G. Kincaid, G. Ledeboer, P. Reynolds, P. Sitaram, A. Ta, M. Theofanos "Identifying and measuring quality in a software requirements specification" Software Metrics Symposium, 1993. Proceedings., First International In Software Metrics Symposium, 1993. Proceedings., First International (1993), pp. 141-152. doi:10.1109/METRIC.1993

[25] Luiz Marcio Cysneiros, Julio César Sampaio do Prado Leite "Using the Language Extended Lexicon to Support Non-Functional Requirements Elicitation" WER 2001 pp 139-153

[26] Ho, C.-W., M. J. Johnson, E. M. Maximilien, and L. Williams, "On Agile Performance Requirements Specification and Testing", in Proceedings of Agile 2006 International Conference,pp. 47-52, Minneapolis, MN, Jul 2006.
[27] Ho, C.-W. and L. Williams, "Deriving Performance Requirements and Test Cases with the Performance Refinement and Evolution Model (PREM)", Department of Computer Science, North Carolina State University Technical Report No. TR-2006-30, Nov 2006.
[28] Ho, C.-W. and L. Williams, "An Introduction to Performance Testing", Department of Computer Science, North Carolina State University Technical Report No. TR-2006-25, Aug 2006.
[29] Ho, C.-W. and L. Williams, "Managing Software Performance Engineering Activities with Performance Refinement and Evolution Model (PREM)", Department of Computer Science, North Carolina State University Technical Report No. TR-2006-28, September 2006.

[30] Ho, C.-W. and L. Williams, "Developing Software Performance with the Performance Refinement and Evolution Model", in Proceedings of the 6th International Workshop on Software and Performance, pp. 133-136, Buenos Aires, Argentina, Feb 2007.
[31] Ho, C.-W., L. Williams, and A. I. Antón, "Improving Performance Requirements Specifications from Field Failure Reports", in Proceedings of the 15th International Requirements Engineering Conference, pp. 79-88, New Delhi, India, Oct 2007.

Authors

Prof. Ananda Rao Akepogu received B.Sc. (M.P.C) degree from Silver Jubilee Govt. College, SV University, Andhra Pradesh, India. He received B.Tech. degree in Computer Science & Engineering and M.Tech. degree in A.I & Robotics from University of Hyderabad, Andhra Pradesh, India. He received Ph.D. from Indian Institute of Technology, Madras, India. He is Professor of Computer Science & Engineering and Principal of JNTU College of Engineering, Anantapur, India. Prof. Ananda Rao published more than fifty research papers in international journals, conferences and authored three books. His main research interest includes software engineering and data mining.

GopiChand.Merugu is pursuing Ph.D. in Computer Science & Engineering from JNTUA, Anantapur, India and he received his M.Tech. in Software Engineering from the same university. He received B.E. degree in Information Technology from Nagarjuna University, India. He is associate professor of computer science & Engineering, BVRIT, JNTUH, Hyderabad. He is a member of IEEE,CSI and IAENG.