2009 International Symposium on Computing, Communication, and Control (ISCCC 2009)
Proc .of CSIT vol.1 (2011) © (2011) IACSIT Press, Singapore

Response Time Estimation: a Study of Hospital Information Management System

Suneeta H. Angadi [1], Narasimha H Ayachit [2] and Prakash. R.Patil [3 +]

[1] CSE Department, R N S Institute of Technology, Bangalore, Karnataka 560060, India
[2] Physics Department, B V B College of Engineering & Technology, Hubli, Karnataka 580031, India
[3] MCA Department, B V B College of Engineering & Technology, Hubli, Karnataka 580031, India,

Abstract. Software Performance Engineering (SPE) is a systematic and quantitative approach to constructing software systems that meet performance objectives. Performance is an important quality attribute of software. Most systems fail to meet performance objectives when they are initially constructed. Performance failures result in cost overruns due to tuning or redesign, damaged customer relations. Most performance failures are due to lack of performance consideration early in the development process. Response time estimation is basic factor to validate performance objective. This paper describes steps required for estimating response time of an application. A simple case study illustrates the process through Hospital Information Management System.

Keywords: Software Performance Engineering; Response Time Estimation; Hospital Information Management System (HIMS)

1. Introduction

Software Performance Engineering (SPE) deals with quantitative approach in constructing software systems. The process is iterative, performed keeping in mind performance objectives [1-4]. The creation of model for software requirements and designs is evaluated to check whether predicted performance metrics meet the specified goals. The process continues through the detailed design, coding, and testing stages to develop more precise models of the software and its predicted performance. It is an engineering approach to performance, avoiding the extremes of performance-driven development and "fix-it-later" approach. SPE does predictions using a model and evaluate trade-offs in software functions, hardware size, quality of results, and resource requirements [5-8]. Performance refers to Response time/responsiveness and scalability are the parameters through which performance are measured. The response time is the time required to respond to a request while scalability refers to how fast a system can respond to increasing workload. Problems associated with software are most often due to fundamental architecture or design factors rather than inefficient coding which affects the system performance. Required quality attributes are largely determined by the time the architecture is chosen [6, 9-11]. This means that performance problems are introduced early in the development process. The principles of SPE helps in creating responsive software, the data required for evaluation, procedures for obtaining performance specifications, and guidelines for the types of evaluation to be conducted at each development stage. It incorporates models for representing and predicting performance. When SPE techniques are used during software development, models predict the impact of software architecture and design decisions before coding begins. With SPE, proactive performance management

+ Corresponding author.
E-mail address: (suni_ha@rediffmail.com, narasimha1957@yahoo.co.in, prp.mca.res@gmail.com).

identifies and resolves performance problems early in the process and avoids the negative effects of the “fix-it-later" approach [12, 13].

Modelling is essential to ensure that the software architecture is one that will make it possible to meet performance objectives. Models should be easy to analyze, understand and to get rapid feedback on whether the proposed software is likely to meet performance objectives. Software execution model has these characteristics. Software execution models are sufficient for identifying serious performance problems at the architectural and early design phases [14].

Analyzing the software model which is dynamic provides analysis of the mean, best- and worst case response times. It characterizes the resource requirements of the proposed software alone, in the absence of other workloads, multiple users or delays due to contention for resources. If the predicted performance in the absence of these additional performance-determining factors is unsatisfactory, then there is no need in constructing more sophisticated models. If the software execution model indicates that there are no problems, analysts can proceed to construct and solve the system execution model. At each phase, the models are refined based on the more detailed design and analysis; objectives are revised to reflect the concerns that exist for that phase [15, 16].

In this paper, an existing model for hospital information management is re looked into by taking the parameters discussed. The improved model is presented with the evaluation in terms of response time at each stage.

2. Case Study: Hospital Information Management System

Hospital Information Management System (HIMS) is aimed at hospital & patient management and such work is available in literature [17, 18]. It stores and processes patient data, accounting information, hospital administration and inventory updates. It enables Health care institutions/organizations to automate their workflow and provide them an efficient way of handling business. Major areas in it are Front Office, Finance, Cash Counter, Pharmacy, Medical Information, Laboratory, Inventory and Payroll.

It is a complete solution, automating any hospital or medical institution of any size - quickly, economically and completely. HIMS is a modular and integrated package, designed by medical professionals and hospital managers, thus helping healthcare providers deliver the best healthcare to patients at the lowest cost. It instantly gives a hospital's management the most accurate information about operations. Management can therefore use HIMS to run operations in the most optimal way, serving patients' interests while ensuring the best possible return on investment. HIMS delivers a low cost of ownership because it is easy to use and maintain, and can be entirely supported by the in-house staff of a hospital. HIMS eases all key medical and administrative tasks.

HIMS does effectively utilize hospital facilities and improves inventory control. Provides information required to support patient care. It also Safeguards data integrity, security and accessibility. Captures and analyses clinical data. Captures the progress of treatment of individuals and gauges the response to treatment and drugs for groups of patients. Makes data collected available for research purposes. It is used to generate MIS reports, which help the management in making policy decisions. It used to maintain records necessary for statutory requirements.

Registration module records the registration of each patient during the first visit to the hospital and generates a unique patient identification number. Maintenance of single ID helps in better understanding the progress of the patient. Outpatients are directed to the concerned doctors for consultation. In-patients are directed to the respective wards.

The wards module maintains in-patient records. An in-patient may undergo various tests and avail services such as drug administration, diet administration, surgical procedures, etc. The case sheet is prepared and maintained by this module. During his / her stay in the hospital, a patient may move from one ward to another, ward to Operation Theatre (OT) and back, ward to Intensive Care Unit (ICU) and back, etc. The wards module tracks this movement. The casualty module handles cases of medical emergencies and accidents. The doctor appointment module keeps track of individual doctors' out-patient and inpatient appointments.

The diet module handles the diet requirements of the patient as well as the instructions to the kitchen for preparation of the diet. A distribution list is also prepared. The labs module handles requests for various tests and makes the results available after verification. Reports can also be prepared. The operation theatre scheduling module handles the booking of the operation theatre and its resources. This information is made available to all the locations to facilitate convenient booking of the OT. The billing module prepares the patient's bills, taking into account the tests undergone by the patient and the hospital services availed. The rosters module handles the duty scheduling of the hospital staff. The system administration module aids the EDP department of the hospital in keeping the system running. Registration module is considered for further analysis.

Every patient who visits the hospital has to get registered prior to getting any consultation, treatment or investigations done. Registration of patients involves accepting certain general and demographic information about the patient. The patient is allocated a unique Registration number and a Patient Identification number (ID). The Patient ID will remain same for his all subsequent visits to the hospital whereas he will be allocated a new registration number on every visit. An operator sitting at a terminal in hospital performs registration of patient.

3. Existing Design (Design I)

In this design whole logic is embedded in User Interface (UI) component. For each test order and medicine prescription there is one access to database. *Synchronous processing* of each is done. Software execution model and process overhead matrix for this are presented Fig 1 and Table1. Test orders and medicine prescription loops will be executed 'k' and 'l' times respectively. From the specification usually a patient undergoes **4** tests and given with prescription of **3** medicines. Specification of these values is obtained from observation of working system.

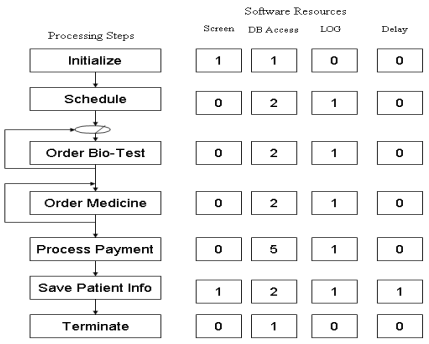

Fig 1 Software Execution Model for Design - I

Device	CPU	Disk	Delay	Display	Network
Quantity	1	1	1	1	1
Service units	Sec	Phy I/O	Units	Screens	Messages
Log	--	1	--	--	--
Screen	0.0001	--	--	1	--
DB Access	0.001	3	--	--	1
Delay	--	--	3	--	--
Service Time (in sec)	1	0.005	1	0.5	0.05

Table 1 Process Overhead Matrix for Design

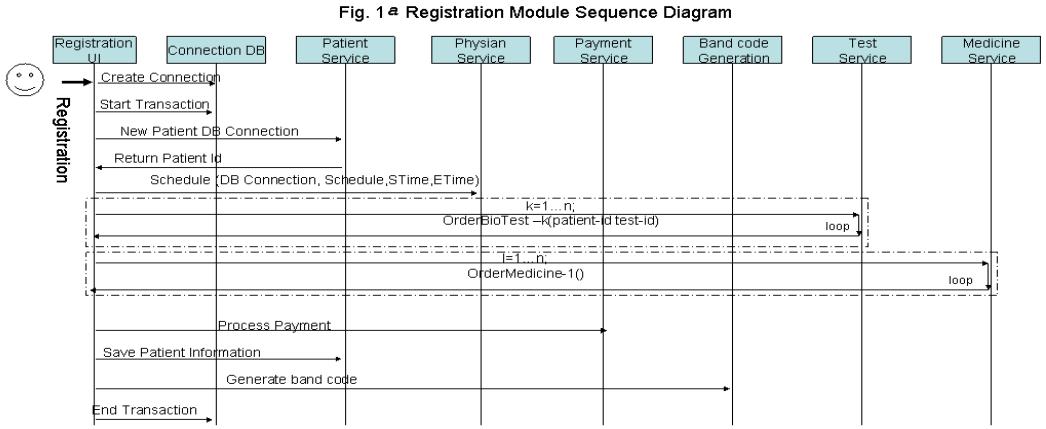

Fig. 1 a Registration Module Sequence Diagram

Fig. 1: (a) Registration Module Sequence Diagram

Response Time {(Screen - 2[0.0001 + 0.5]) + (Delay -3*1) + (Log - 10*0.015) + (DB Access - 25[0.001+0.015+0.05])} + 5.8002 sec +1 = 6.8002 sec

Response time value mentioned in performance objective for this module is 8sec. Design drawn meets this objective since 6.8002 sec < 8 sec. Still we proposed an alternative to know if there is an opportunity for improvement. The registration module sequence diagram is presented in Fig 1a.

4. Proposed Design (Design II)

Software execution model and process overhead matrix for the proposed design are presented Fig 2 and Table2.

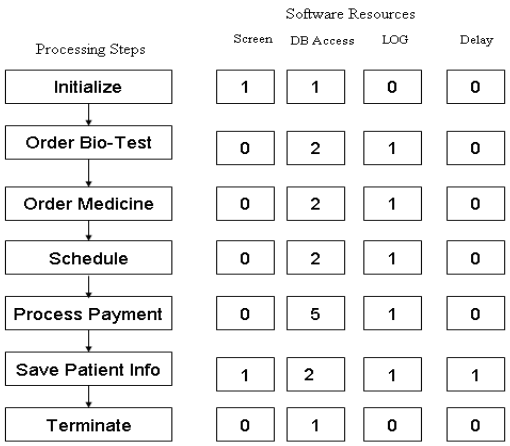

Fig 2 Software Execution Model for Design – II (Proposed)

Table 2 Process Overhead Matrix for Design II

Device	CPU	Disk	Delay	Display	Network
Quantity	1	1	1	1	1
Service units	Sec	Phy I/O	Units	Screens	messages
Log	--	1	--	--	--
Screen	0.0001	--	--	1	--
DB Access	0.001	3	--	--	1
Delay	--	--	3	--	--
Service time (in sec)	1	0.005	1	0.5	0.05

Response Time {(Screen - 2[0.0001 + 0.5]) + (Delay -3*1) + (Log - 5*0.015) + (DB Access - 15 [0.001+0.015+0.05])} + 5. 062 sec +0.5 = 5.5652 sec

The corresponding registration module sequence diagram is presented in Fig 2(a). In the proposed/solution design improvements proposed are,

i. DB-connection pool is employed to establish connection in advance i.e. prior to operator clicks for registration.
ii. Test-orders and medicine prescription are sent in the form of bunch.
iii. Asynchronous processing of tests and medicine orders is performed.
iv. Business logic is embedded in registration service component for better layered architecture.

In the proposed design, still DB-connection overhead is associated in both designs. The above said pre-connected object improvement holds good only for initialization step i.e. when first time registration module is loaded. But once registration for one patient has been done, both the designs take same computational time for processing. Only minor difference can be found with respect to test order's and medicine prescription's individual and list processing. Separation of business logic from UI component makes simple implementation

Fig 2 a Proposed - Registration Module Sequence Diagram

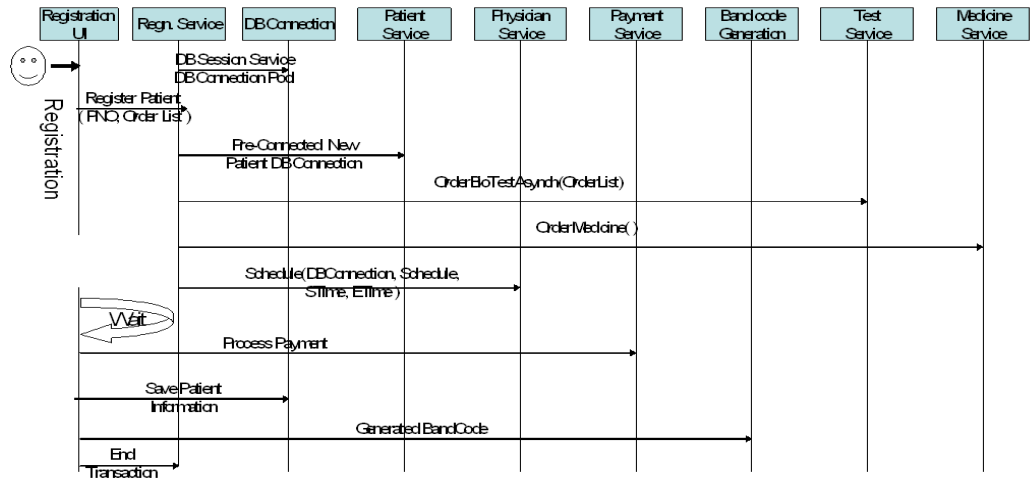

Fig. 2: (a) Registration Module Sequence Diagram

5. Discussion of Results

With this design alternative 1.235sec can be saved. Difference between the two designs is that in first design DB connection is set up after user logs in, where as in the second design DB connection pool is already established. So there response time in second case is reduced by 1.235 sec. However this holds good only during first time loading of registration module, once it is up and running both the designs give nearly same response time. There is a slight difference of 0.235 sec between the two approaches. This improvement is achieved because of list processing of bio tests and medicine order. Asynchronous processing of them is done. There is a wait time of 0.5 sec incorporated with the design II because of asynchronous processing.

Following chart (Fig 3) shows response time variation with respect to number of DB Accesses, since DB Access in this application will have more effect on response time. We have considered this factor.

In Table 3 are presented detailed comparison in each case.

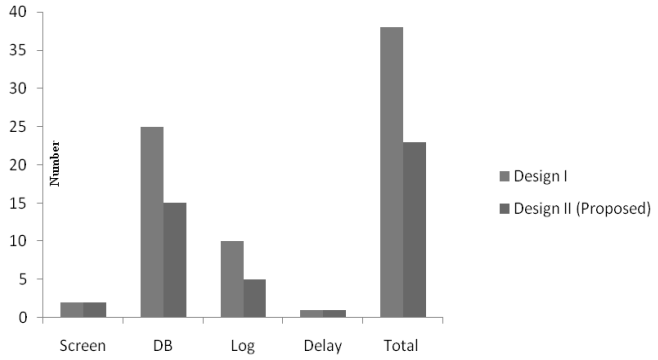

Fig. 3: Comparison Process counts in two designs

	Screen		DB		Log		Delay	
	I	II	I	II	I	II	I	II
Initialization	1	1	1	1	0	0	0	0
Order Bio Test	0	0	2	2	1	1	0	0
Order Medicine	0	0	8	2	4	1	0	0
Schedule	0	0	6	2	3	1	0	0
Process Payment	0	0	5	5	1	1	0	0
Save Patient Info	1	1	2	2	1	1	1	1
Terminate	0	0	1	1	0	0	0	0
Total	2	2	25	15	10	5	1	1

Table 3 Comparison Process counts in two designs

6. Conclusion

SPE is an approach to ensure performance characteristics of evolving software. Its proactive approach avoids negative effects of "fix-it-later" approach. Modeling technique of this approach is powerful. Response time estimation task is preliminary step for architectural and design decisions. It is a baseline for performing comparison between design alternatives with respect to response time. This concept was demonstrated using HIMS domain as a case study.

7. Acknowledgements

The authors (NHA, and PRP) acknowledge the financial help and encouragement shown by the Principal Ashok Shettar and Management of BVB College of Eng and Tech., Hubli, India

8. References

[1] G. Booch, J. Rambaugh, and I. Jacobson, The Unified Modeling Language User Guide, Addison Wesley.2001

[2] C.U. Smith and L. G. Williams, Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software, Boston, MA, Addison-Wesley, 2002.

[3] Elmsari Navathe, Fundamentals of Data Base Systems.Jim Hughes, SES Inc,Performance Modeling Methodology. Addison Wesley.2002

[4] Michael K. Cook, An Enterprise Level Approach to Proactive Performance Engineerig. SES Conference On Performance Modeling - April 13-15, 1999

[5] Connie U. Smith and Lloyd G. Williams, Best Practices for Software Performance Engineering, Performance Engineering Services and Software Engineering Research,1,2003

[6] B. Boehm, "Software Risk Management:Principles and Practice," IEEE Software, vol. 8, no. 1, pp. 32-41, 1991.

[7] P. C. Clements and L.M. Northrop, "Software Architecture: An Executive, Overview," Technical Report No. CMU/SEI-96-TR-003, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1996.

[8] D. J. Reifer, Making the Software Business Case: Improvement by the Numbers, Boston, Addison-Wesley, 2002.

[9] L. G. Williams, C. U. Smith, Craig Hanson, Mary Hesselgrave, Thad Jennings, Panel: “The Economics of Software PerformanceEngineering,” CMG 2002, Reno, December, 2002.

[10] Chih-Wei Ho and Laurie Williams, Developing Software Performance with the Performance Refinement and Evolution Model WOSP 2007, Feb 5–8, 2007, Buenos Aires, Argentina

[11] Balsamo, S., A. D. Marco, and P. Inverardi, "Model-BasedPerformance Prediction in Software Development: A Survey,"IEEE Transactions on Software Engineering, vol. 30,no. 5, pp. 295-310, May 2004.

[12] Ho, C.-W., M. J. Johnson, E. M. Maximilien, and L. Williams,"On Agile Performance Requirements Specificationand Testing," in Proceedings of Agile 2006 International Conference, pp. 47-52, Minneapolis, MN, Jul 2006.

[13] “Performance Evaluation of Software Architectures", Research Report, CS-2000-3, Dipartimento di Informatica Universita Ca' Foscari di Venezia, Italy, March 2000.

[14] Smith, C., “Performance Engineering of Software Systems”, Addison-Wesley, 1990

[15] Hennessy, J., Patterson, D., “Computer Architecture, a Quantitative Approach”, Third Edition, Morgan Kaufmann, 2002.

[16] Avritzer, A., J. Kondek, D. Liu, and E. J. Weyuker, "Software Performance Testing Based on Workload Characterization,"in Proceedings of the 3rd International Workshop on Software and Performance, pp. 17-24,Rome, Italy, Jul 2002.

[17] Brataas, G.; Hughes, P.H.; Solvberg, A Framework for performance engineering of workflows: a blood bankcase study Proceedings of the Thirty-First Hawaii International Conference on System Sciences, 1998

[18] Akin, Gib and David Hopelain, ‘Information Systems for Organizational Productivity,"Computer Personnel, 10(3):4-1 1, April 1986.