Embedded System's Performance Analysis with RTC and QT

Fulong Chen[1,2] and Xiaoya Fan[1]

[1] School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China
chenfulong@mail.nwpu.edu.cn
[2] Department of Computer Science and Technology, Anhui Normal University, Wuhu 241000, China
fanxy@nwpu.edu.cn

Abstract. Maximal, minimal and mean response time are several performance metrics of embedded system. Some embedded systems have not real-time constraints, e.g. printer and router, but their mean case performance is more important, and some others have those constraints and need evaluating their utmost performance. In this paper, with real-time calculus and queuing theory, both mean and utmost case performances of embedded multitasking system are analyzed as referred target for design, verification, decision and optimization.

Keywords: embedded system, performance analysis, real-time calculus, queuing theory.

1 Introduction

With the development of computer theory and technology, more and more smart and embedded devices are applied in many areas, e.g. automobiles electronics, handheld game systems, industrial environments, telecommunication or fabrication equipments, aircraft electronics, medical systems, military applications, authentication systems, consumer electronics, smart buildings, robotics and so on. Embedded systems can be defined as information processing systems embedded into these enclosing products [1]. Hence, embedded computer has also been called the disappearing computer, which form the basis of the so-called pervasive computing era.

As shown Fig.1, Sensors receive signals or information from the physical environment, and send them to A/D converters or sample-and-holders, which convert continuous sequences of analog values to discrete sequences of digital values. The kernel of the whole embedded system-information processor processes those digital values and generates some digital results. These results can be sent to display device, which displays these results, and also be used to control the external environment through actuators, which convert digital signals to analog ones.

During developing of embedded system, it is very important to evaluate its performance. By evaluating memory requirements, timing properties, processor speed, bus utilization, etc., developers can find the bottleneck components so as to select appropriate hardware or software resources that meet the system's functional and non-functional requirements and cost low. Furthermore, by evaluating performance of actual embedded application system, testers can verify whether the performance meets the design requirements and constraints those are presented by customers.

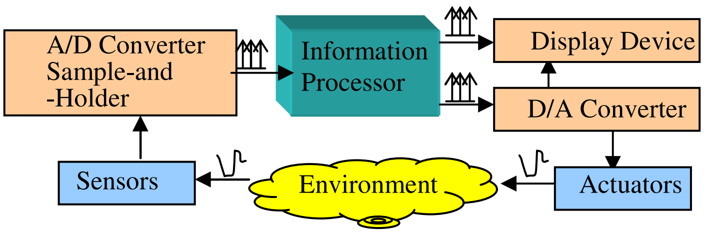

Fig. 1. Architecture of Embedded System

There are three solutions for evaluating performance of embedded system:

- *Measurement*-by executing some test benchmarks or instances, testers can get Execution Time (ET) and other performance parameters for evaluating the actual system. In the past years, many universities, institutes and corporations have been devoted to studying the technologies of benchmark performance for embedded system. Some performance evaluation tools and benchmarks, e.g. Dhrystone, EEMBC, NullStone, MediaBench, MiBench, etc., have been developed and released. With these tools and benchmarks, some performances of hardware and software of embedded system can be evaluated. However, many system functions and attributes are hard to be obtained by testing due to the variety of the architectures of embedded system.
- *Simulation*-by executing some programs for simulating the evaluated system and its load, monitoring members can get more precise performance parameters. But this solution's cost is higher when its model is constructed. At present, for embedded system, there are many simulation tools such as Carosse-Perf, SES Workbench [2], STRESS, schedalyzer, ARTS, PERTS [3], RTC Toolbox [8] and so on.
- *Analysis*-by making use of some mathematical theories and methods, analyzers can study and describe the relation of performance and system before design, and get loose performance metrics. According to this approach, its cost for modeling is the lowest. Real-Time Calculus (RTC) and Queuing Theory (QT) are two kinds of system level analysis tools for evaluating performance of real-time embedded system. Performance of embedded system includes best, worst and mean ones. RTC is fit for analyzing best performance and worst performance, and QT is more propitious to analyzing mean performance.

In section 2, Real-Time Calculus is introduced, and performance of embedded system in utmost state is analyzed. In section 3, basic queuing theory is introduced, and by making use of this basic theory and method, performance of embedded system in steady state is also analyzed.

2 Real-Time Calculus (RTC) and Performance Analysis of Embedded System in Utmost State

2.1 RTC

RTC is on basis of Linear System Theory, Calculus for Network, Adversarial Queuing Theory [7], and composed of input/output event streams model, resource (service) model and processing model (Fig. 2)

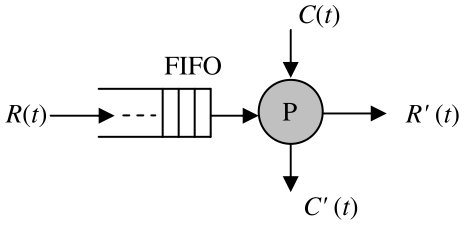

Fig. 2. Modeling of Real-Time Calculus System

- $R(t)$:number of events that arrived in time interval [0, t),represents input event stream function;
- $R'(t)$:number of events that left in time interval [0, t), represents output event stream function;
- $C(t)$:maximal number of events that could have been processed by processor P in time interval [0,t], represents available resources or services function;
- $C'(t)$: maximal number of events that processor P could still process by in time interval [0,t], represents remaining resources or services function.

If the description of resource/service and event streams is available, then $C(t)$ and $R(t)$ are determinate. So

$$R'(t) = \inf_{0 \le u \le t} \{R(u) + C(t) - C(u)\} \tag{1}$$

$$C'(t) = C(t) - R'(t) \tag{2}$$

2.2 Performance Analysis of Embedded System in Utmost State

Provided that, in the interval $[0, \Delta]$, $\alpha^u(\Delta)$ represents maximum number of arriving events, $\alpha^l(\Delta)$ represents minimum number of arriving events, $\beta^u(\Delta)$ represents maximum number of available services (resources), and $\beta^l(\Delta)$ represents minimum number of available services (resources), according to RTC, the following processing model can be got:

$$\alpha'^u(\Delta) = f_\alpha{}^u(\alpha^u, \alpha^l, \beta^u, \beta^l) \tag{3}$$

$$\alpha'^{l}(\Delta) = f_{\alpha}{}^{l}(\alpha^{u}, \alpha^{l}, \beta^{u}, \beta^{l}) \tag{4}$$

$$\beta'^{l}(\Delta) = f_{\beta}{}^{l}(\alpha^{u}, \alpha^{l}, \beta^{u}, \beta^{l}) \tag{5}$$

$$\beta'^{u}(\Delta) = f_{\beta}{}^{u}(\alpha^{u}, \alpha^{l}, \beta^{u}, \beta^{l}) \tag{6}$$

Here, $f_{\alpha}{}^{u}$, $f_{\alpha}{}^{l}$, $f_{\beta}{}^{u}$ and $f_{\beta}{}^{l}$ depend on real processing model. For example, in greedy shaper component model, $\beta^{u}(\Delta) = \beta^{l}(\Delta) = \sigma$,so $\alpha'^{u}(\Delta) = \inf_{0 \le \lambda \le \Delta}(\alpha^{u}(\Delta - \lambda) + \sigma)$ [9,10]. Through this method, with RTC Toolbox [8], for a known processing model, its arrival curves [α^{u}, α^{l}], leave curves [α'^{u} , α'^{l}], service curves [β^{u}, β^{l}] and availability curves [β'^{u} , β'^{l}] are obtained. If the performance doesn't meet the requirements of actual embedded application system, designers need adjust input events stream so as to change arrival curves [α^{u} , α^{l}] and reconfigure resources so as to change service curves [β^{u} , β^{l}], or, reconstruct new processing model so as to change $f_{\alpha}{}^{u}$, $f_{\alpha}{}^{l}$, $f_{\beta}{}^{u}$ and $f_{\beta}{}^{l}$.

3 Queuing Theory and Performance Analysis of Embedded System in Steady State

3.1 Basic Queuing Theory

In Queuing theory, a random service system is taken into account. By studying the probability parameters of customers waiting in queue, analyzers can get the performance of queuing system in steady state. A queuing system can be described as Fig.2. Any customer that wants to pass through the system need finish the following processes: arriving, waiting, served and leaving. The interval between two customers arriving orderly satisfy Determinate Length Distribution (DLD), Negative Exponential Distribution (NED, called Poisson stream), Erlang Distribution (ED) or general random distribution. Customers wait in the queue according to wait or loss rule. In wait rule, customers are served according to First Come First Serve (FCFS), Last Come First Serve (LCFS), Priority Serve (PS), general random serve, etc. There is one server or multi-servers in the system. The former is called Singer Server Queuing System (SSQS), and the latter goes by the name of Multiple Servers Queuing System (MSQS). The served time length of each customer may be subject to DLD (*D*), NED (*M*), k ranks ED (E_k) or general random distribution (*G*). In 1950s,D.G.Kendall introduced *A/B/C/n* queue model- *A* stands for input stream, *B* stands for served time, *C* stands for number of servers, n stands for queue length. For instance, for *M/M/1/∞* queue, the first M means that customers arrive in Poisson stream, the second *M* stands for Markovian or memoryless (exponential) served time distribution, 1 stands for only one server, and ∞ is indicated that the length corresponding to the queue is limitless. In Queuing system, mean queue length L, mean steady queue length L_q, mean stay (or response) time T and mean wait time T_q are four performance metrics in common use for performance analysis. Queuing system has been used for multi-process controlling [11], VoIP QoS analysis [12], Evaluation for UML [13], multiple access method [14], simulator [15], dispatching strategy [16], railway computer interlocking system performance evaluation [17], estimation of the storage space [18], Evaluation of anonymity providing techniques [19], staff software maintenance centers [20], etc.

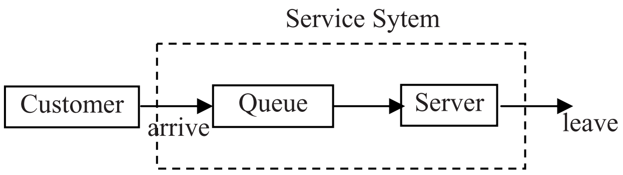

Fig. 3. Specification for Service System

3.2 Embedded Multitasking Sequential Processing Model

An embedded system and some tasks can be considered as a Markovian queuing process, or a birth and death process. The embedded system is the server, and tasks, which are waiting in a queue to get services, are customers. Each task, located in the memory or external environment and requesting service, is considered as a customer arrival in queue. The rate of arrival of the tasks is the birth rate and the rate of their leaving the system after finishing their services is the death rate. It is assumed that the death rate can be influenced by some queuing control strategies but the birth rate cannot. It is the goal to change the death rate, or the users' service rate, in a way that the number of served tasks would be maximized after some finite time.

Assumed that tasks arrival rate is subject to Poisson distribution, the processor processes tasks according to FCFS, the spent time is subject to NED, and then an *M/M/1/∞* queue model can be constructed (Fig.4).

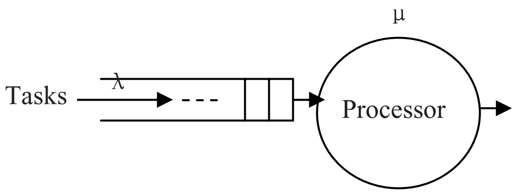

Fig. 4. Embedded Multitasking Sequential Processing Model

In this model, the following parameters are defined:

- λ—arrival rate stands for number of tasks arriving in unit time;
- μ—service rate or departure rate stand for number of tasks processed in unit time;

- ρ—server utilization or service intensity , ρ=λ/μ.

λandμof embedded system can be solved, see [21]. Then, mean response time T of tasks can be got through Little formula [22]:

$$T = \frac{1}{\mu(1-\rho)} = \frac{1}{\mu - \lambda} \tag{7}$$

3.3 Embedded Multitasking Non-preemptive Model

In Fig.4, if there are n ranks of different priorities for tasks, each rank of tasks are Poisson stream, arrival rate is λ_i, $1 \le i \le n$, processing time is not determinate, but mean processing time $1/\mu_i = E[s_i]$ and $\gamma_i = E[s_i^2]$ can be acquired, kindred tasks are processed according to FCFS, different kinds of tasks are processed according to non-preemptive rule, then that model becomes a Non-Preemptive *M/G/1/∞* Model.

Given $\lambda = \sum_{i=1}^{n} \lambda_i$, $1/\mu = \sum_{i=1}^{n} \frac{\lambda_i}{\lambda \mu_i}$, $\gamma = \sum_{i=1}^{n} \frac{\lambda_i \gamma_i}{\lambda}$, $u_j = \sum_{i=1}^{n} \frac{\lambda_i}{\mu_i}$, then mean wait time of j-th rank of tasks:

$$T_q(j) = \frac{\lambda \gamma}{2(1-u_{j-1})(1-u_j)} \tag{8}$$

And mean response time:

$$T(j) = T_q(j) + \frac{1}{\mu_j} \tag{9}$$

Mean wait time of whole system:

$$T_q = \sum_{j=1}^{n} \frac{\lambda_j T_q(j)}{\lambda} \tag{10}$$

And mean response time of whole system:

$$T = \sum_{j=1}^{n} \frac{\lambda_j T(j)}{\lambda} \tag{11}$$

3.4 Embedded Multitasking Preemptive and Non-blocked Model

In Preemptive and Non-blocked Model, high rank of tasks can snatch the processor from low rank of tasks, which will go back to the queue and wait for next process. Then the system becomes a feedback Jackson Queuing Network [22](Fig.4 (a)). Provided γ is arrival rate of whole system, q is the probability of preemption, μ is processor's service rate, then tasks arrival rate λ of processor queue can be obtained according to Equation (12):

$$\lambda = \gamma + q\lambda \Rightarrow \lambda = \frac{\gamma}{1-q} \tag{12}$$

The processor utilization is

$$\rho = \frac{\lambda}{\mu} = \frac{\gamma}{\mu(1-q)} \tag{13}$$

So mean number of tasks of whole system is:

$$L = \frac{\rho}{1-\rho} = \frac{\gamma}{\mu(1-q)-\gamma} \tag{14}$$

According to Little formula, mean respone time of whole system is:

$$T = \frac{L}{\gamma} = \frac{1}{\mu(1-q)-\gamma} \tag{15}$$

It is obvious that relation between *T* and *q* is showed by Fig. 4(b), and while *q*=0,this model becomes that one of Fig.3; while $q \rightarrow 1- \gamma/\mu$,$T \rightarrow \infty$.

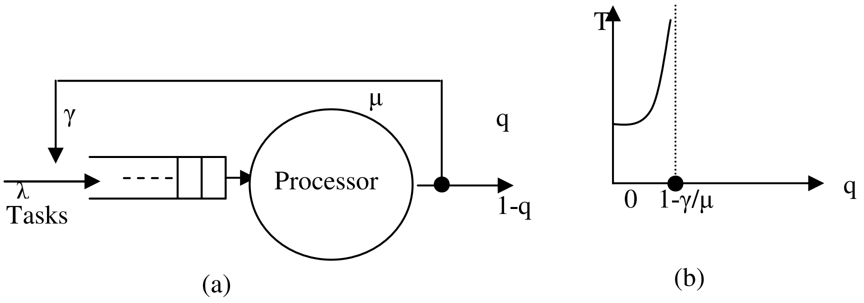

Fig. 5. Embedded Multitasking Preemptive and Non-blocked Model Model

3.5 Embedded Multitasking Non-preemptive and Blocked Model

Now see non-preemptive and blocked model (Fig.5). Tasks arrive at the system in arrival rate γ. Processor processes them in service μ_1.Tasks can be blocked by so-called block task during processing as a result of some I/O operations or synchronization. Supposed that blocked probability is q, and rate of release from block is μ_2, blocked tasks will go back to processor queue and wait for next process after completing their I/O operations or synchronization, then real arrival rateλ_1 of processor queue and real arrival rateλ_2 of block queue have the following relation:

$$\lambda_1 = \gamma + \lambda_2 \tag{16}$$

$$\lambda_2 = q\lambda_1 \tag{17}$$

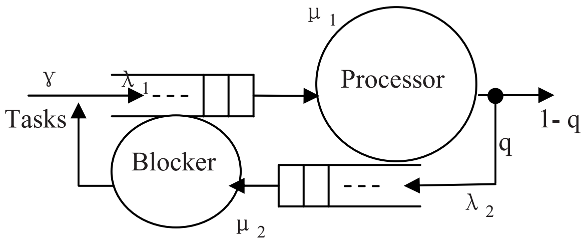

Fig. 6. Embedded Multitasking Non-preemptive and Blocked Model

So,

$$\lambda_1 = \frac{\gamma}{1-q} \tag{18}$$

$$\lambda_2 = \frac{q\gamma}{1-q} \tag{19}$$

Accordingly, processor service intensity ρ_1 and block service intensity ρ_2 are:

$$\rho_1 = \frac{\lambda_1}{\mu_1} = \frac{\gamma}{\mu_1(1-q)} \tag{20}$$

$$\rho_2 = \frac{\lambda_2}{\mu_2} = \frac{q\gamma}{\mu_2(1-q)} \tag{21}$$

Mean length of processor queue L_1 is:

$$L_1 = \frac{\rho_1}{1-\rho_1} = \frac{\gamma}{\mu_1(1-q)-q\gamma} \tag{22}$$

Mean length of blocker queue L_2 is:

$$L_2 = \frac{\rho_2}{1-\rho_2} = \frac{q\gamma}{\mu_2(1-q)-\gamma} \tag{23}$$

Mean length of whole system queue is:

$$L = L_1 + L_2 \quad = \frac{\gamma}{\mu_1(1-q)-q\gamma} + \frac{q\gamma}{\mu_2(1-q)-\gamma} \tag{24}$$

So, mean response time of whole system is:

$$T = \frac{L}{\gamma} = \frac{1}{\mu_1(1-q) - q\gamma} + \frac{q}{\mu_2(1-q) - \gamma} \tag{25}$$

Range of q is:

$$0 \le q < \min(\frac{\mu_1}{\mu_1 + \gamma}, 1 - \frac{\gamma}{\mu_2}) \tag{26}$$

3.6 Embedded Multitasking Preemptive and Blocked Model

Both preemption and block are considered, see Fig.6.Tasks arrive at the system in γ. Processor processes them in μ_1.Low rank of tasks can be preempted by high ones, and also be blocked due to some I/O operations or synchronization. On the assumption that preempted probability is q_1, blocked probability is q_2, and rate of release from block is μ_2.After those interruptions, tasks will go back to processor queue and wait for next process. Then, real arrival rateλ_1 of processor queue and real arrival rate of block queueλ_2 have the following relation:

$$\lambda_1 = \gamma + \lambda_1 q_1 + \lambda_2 \tag{27}$$

$$\lambda_2 = \lambda_1 q_2 \tag{28}$$

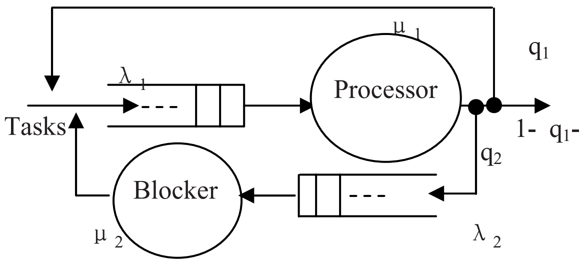

Fig. 7. Embedded Multitasking Preemptive and Blocked Model

So,

$$\lambda_1 = \frac{\gamma}{1 - q_1 - q_2} \tag{29}$$

$$\lambda_2 = \frac{q_2\gamma}{1 - q_1 - q_2} \tag{30}$$

Mean response time of whole system is:

$$T = \frac{L}{\gamma} = \frac{L_1 + L_2}{\gamma} = \frac{\frac{\rho_1}{1-\rho_1} + \frac{\rho_2}{1-\rho_2}}{\gamma} = \frac{\frac{\frac{\lambda_1}{\mu_1}}{1-\frac{\lambda_1}{\mu_1}} + \frac{\frac{\lambda_2}{\mu_2}}{1-\frac{\lambda_2}{\mu_2}}}{\gamma} \tag{31}$$
$$= \frac{1}{\mu_1(1-q_1-q_2)-\gamma} + \frac{q_2}{\mu_2(1-q_1-q_2)-q_2\gamma}$$

To all appearances, non-preemptive and blocked model ($q_1 \to 0$), preemptive and non-blocked model ($q_2 \to 0$) and sequential processing model ($q_1 \to 0, q_2 \to 0$) are three special cases of preemptive and blocked model.

4 Conclusions and Future Work

With RTC and QT, utmost and mean performance of embedded system can be assured loosely and quickly so as to select appropriate design scheme, verify and optimize system. RTC tool has been offered [8].However QT tool for embedded system is not available.In the future, QT tool of different models will be submitted for evaluating performance of embedded system.

Acknowledgments. This paper is supported by the National Technology Research and Development 863 Program of China (No. 2005AA1Z1193),the National Natural Science Foundation of China (No. 60573143) ,the Soft Science of Anhui Province of China(No.06035021) and Youth Fund of Anhui Normal University(No. 2006xqn54).

References

1. Marwedel, P.: Embedded System Design, pp. 1–7. Springer, Heidelberg (2006)
2. Castelpietra, P., Song, Y.Q., Lion, F.S., et al.: Analysis and Simulation Methods for Performance Evaluation of a Multiple Networked Embedded Architecture. IEEE Transactions on Industrial Electronics 49(6), 1251–1264 (2002)
3. Audsley, N.C., Burns, A., Richardson, M.F., et al.: STRESS:a Simulator for Hard Real-time Systems. Software-Practice and Experience 24(6), 543–564 (1994)
4. Li, Y.T.S., Malik, S.: Performance Analysis of Embedded Software Using Implicit Path Enumeration. IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems 16(12), 1477–1487 (1997)
5. Malik, S., Martonosi, M., Li, Y.T.S.: Static Timing Analysis Of Embedded Software. In: 34th Design Automation Conference, pp. 147–152 (1997)
6. Lee, J.Y., Park, I.C.: Timed Compiled-Code Functional Simulation of Embedded Software for Performance Analysis of SOC Design. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 22(1), 1–149 (2003)

7. Thiele, L., Chakraborty, S., Naedele, M.: Real-time calculus for scheduling hard real-time systems. In: 2000 IEEE International Symposium on Circuits and Systems, pp. 101–104. IEEE Computer Society Press, Los Alamitos (2000)
8. ETH Zurich.Real-Time Calculus(RTC) Toolbox, http://www.mpa.ethz.ch/rtctoolbox,2007-4-1
9. Thiele, L.: Performance Analysis of Distributed Embedded System. Technical report,ARTIST 27 UNU-IIST Spring School in Xi'an,China (April 3-25, 2006)
10. Thiele, L.: Introduction to Real-Time Calculus. Technical report,ARTIST 27 UNU-IIST Spring School in Xi'an,China (April 3-25, 2006)
11. Egerstedt, M., Wardi, Y.: Multi-Process Control Using Queuing Theory. 41st IEEE Conference on Decision and Control 2, 1991–1996 (2002)
12. Pitts, J.M., Wang, X., Yang, Q., Schormans, J.A.: Excess-rate queuing theory for M/M/1/RED with application to VoIP QoS. Electronics Letter 42(20), 1188–1189 (2006)
13. Perdos, A., Chatzigeorgiou, A., Stephanides, G.: Evaluation of a Queuing Theory and Systems Modeling Course Based on UML. In: 6th International Conference on Advanced Learning Technologie, pp. 507–509 (2006)
14. Jamshidifar, A.A., Khorram, E., Afshar, A.: Applying queuing theory to multiple access method for store and forward satellites. In: 2nd International Conference on Recent Advances in Space Technologies, pp. 416–419 (2005)
15. Granados, D., Garcia, A.J.: A Web based simulator on elementary queuing theory. In: 5th International Conference on Information Technology Based Higher Education and Training, pp. 83–86 (2004)
16. Zong, Q., Xing, G., Chen, D., Ya, S.: The queuing theory based research of dispatching strategy for elevator group control system during up-peak. 5th World Congress on Intelligent Control and Automation 6, 5307–5311 (2004)
17. Guo, J., Zhu, C., Yang, Y.: Performance evaluation of railway computer interlocking system based on queuing theory. In: 4th International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 420–423 (2003)
18. Yang, C.-W., Huang, K.-S., Yu, G., Jan, D.-Y.: Using queuing theory to estimate the storage space of stocker in automated material handling systems. In: Semiconductor Manufacturing Technology Workshop, pp. 102–104 (2002)
19. Kesdogan, D.: Evaluation of anonymity providing techniques using queuing theory. In: 26th Annual IEEE Conference on Local Computer Networks, pp. 316–322. IEEE Computer Society Press, Los Alamitos (2001)
20. Antoniol, G., Casazza, G., Di Lucca, G.A., Di Penta, M., Rago, F.: A queue theory-based approach to staff software maintenance centers. In: 2001 IEEE International Conference on Software Maintenance, pp. 510–519 (2001)
21. Luo, G., Zhao, Z., Zhao, H.: Testing and Evaluation of Network Capability with Queueing Theory in a Embedded System. Journal of Shenyang Normal University (Natural Science), China 23(1), 54–56 (2005)
22. Lin, C.: Computer Nerwork and Computer System Perforance Evaluation, pp. 224–260. TsingHua University Press, China (2001)