Method for Constructing Performance Annotation Model Based on Architecture Design of Information Systems

Hui Du[1], Renchu Gan[2], Kecheng Liu[2, 3], Zhenji Zhang[1] and Darren Booy[3]

[1] School of Economics and Management, Beijing Jiaotong University, Beijing 100044, P.R. China hdu@bjtu.edu.cn zhjzhang@bjtu.edu.cn
[2] School of Management and Economics, Beijing Institute of Technology,
Beijing 100081, P.R. China ganrc@bit.edu.cn
[3] Informatics Research Centre, University of Reading, Reading RG6 6AY, United Kingdom
k.liu@reading.ac.uk d.a.booy@reading.ac.uk

Abstract. To forecast performance of architecture design of information system, we used a performance analysis model to directly get performance indexes, as well as performance annotation model to get performance parameters and facilitate construction of performance analysis model that is indispensable. At present, when constructing performance annotation model, UML Profile for Schedulability, Performance, and Time specification has been de facto standard used to attach performance parameters to UML diagrams. However, UML use case diagram and UML deployment diagram annotated are not powerful enough to model complicated interactions among actors and information system in the organization layer and among hardware resources in the hardware layer respectively. Moreover, based on the performance annotation model using UML diagrams, Coloured Petri Nets (CPN) has not yet been used to construct performance analysis model. Therefore, no steps for facilitating construction of CPN based performance analysis model are included in currently processes for constructing performance annotation model. For solving both deficiencies, in this paper, UML activity diagram is proposed to be annotated in both layers. Moreover, performance annotation model construction process including steps for facilitating construction of CPN based performance analysis model is also devised. Furthermore, the application of the proposed method in the initial architecture design of an online bookshop system is briefly specified.

Keywords: *Information system, Performance, Annotation model, Architecture design, UML*

l. INTRODUCTION

“The performance of computer-based information system is the response time and throughput seen by their users, as well as observable quantities associated with the corresponding computers, i.e. resource utilizations and length of resource queues in the operating system.” [1]

It is the fact that severe performance problems have always emerged in implemented information systems. However, at that moment, it was usually very difficult for them to be rectified. To avoid this dilemma, architecture-oriented modeling and analysis of information systems performance was proposed especially in [1-3].

In [3], through analysis and synthesis of the model architectures respectively proposed in [1, 2, 4, 5], a new model architecture for architecture-oriented modeling and analysis of information systems performance was proposed, which is shown in table 1.

Table 1. The New Model Architecture

	Annotation view	Analysis view
Organization layer	Organization annotation model	Organization analysis model
Software layer	Software annotation model	Software analysis model
Hardware layer	Hardware annotation model	Hardware analysis model

Just as shown in table 1, the new model architecture consists of two parts, which are annotation view in the second column and analysis view in the third column. Both views are divided into three same layers, which are organization layer, software layer and hardware layer from the top down. Six models exist in the intersections of views and layers. Three models in the annotation view, in other words performance annotation model, are used to get performance parameters and facilitate construction of corresponding three models in the analysis view, in other words performance analysis model, which are used directly to get performance indexes through analysis or simulation. Each two models in each layer from the top down are used to model interactions respectively among actors and information system, among software processes and among hardware resources.

Since UML Profile for Schedulability, Performance, and Time specification (in short SPT Profile) was adopted by OMG in 2002, it has been de facto standard used to attach performance parameters to UML diagrams in most performance engineering studies. For example, in [4, 6, 7], UML use case diagrams were annotated by the performance extensions defined in SPT Profile [8] to specify workload related parameters in the organization layer. UML activity diagrams were annotated to specify scenario related parameters in the software layer. UML deployment diagrams were annotated to specify hardware resource related parameters in the hardware layer. In addition, in [9], UML sequence diagrams were also annotated to specify scenario related parameters in the software layer. As to construction of performance analysis model, some different languages have been used. For example, multi-class queuing network was used in [4, 7] and stochastic Petri nets was used in [6, 9].

Although SPT Profile has been de facto standard used to attach performance parameters to UML diagrams, deficiencies still exist in UML diagrams annotated. As we know, UML use case diagram is mainly used to get functional requirements of information system, it is too simple to model complicated interactions among actors and information system in the organization layer. UML deployment diagram is in fact a kind of static diagram, which is mainly used to specify distributions of software processes or components and hardware resources of information system. Therefore, it is really unsuitable to use it to model complicated interactions among hardware resources in the hardware layer.

Based on the performance annotation model using UML diagrams, although different languages have been used to construct performance analysis model, CPN [10] has not yet been used. In fact, compared with all kinds of queuing networks, CPN can model synchronization at ease. Compared with stochastic Petri nets, there is less restrictions on CPN transition. Furthermore, based on the performance analysis model using CPN, Design/CPN, a computer tool, can be used to get performance indexes through simulation. To sum up, CPN is indeed a suitable language to be used to construct performance analysis model. However, since CPN has not yet been used, no performance annotation model construction process including steps for facilitating construction of CPN based performance analysis model has been proposed.

In order to solve both deficiencies discussed above, in this paper, UML activity diagram is proposed to be annotated both in the organization layer and in the hardware layer. Moreover, performance annotation model construction process including steps for facilitating construction of CPN based performance analysis model is devised. Furthermore, application of the proposed method based on the initial architecture design of an online bookshop system is briefly specified.

The paper is organized as follows: in section 2, the method for constructing performance annotation model based on architecture design of information system is presented. In section 3, application of the method based on the initial architecture design of an online bookshop system is briefly specified. In section 4, conclusion and future work are provided.

2. METHOD FOR CONSTRUCTING PERFORMANCE ANNOTATION MODEL

The method proposed in this paper consists of two parts, which are UML diagram for constructing performance annotation model and process for constructing performance annotation model.

2.1 UML Diagram for Constructing Performance Annotation Model

As stated in section 1, since it is unsuitable to use UML use case diagram and UML deployment diagram to model complicated interactions in the organization layer and in the hardware layer respectively, new UML diagram has to be chosen.

In SPT Profile, two kinds of UML diagrams, which are UML activity diagram and UML sequence diagram, are used to be annotated in the software layer. Compared with UML use case diagram and UML deployment diagram, though they can both model complicated interactions, “when it comes to model complex hierarchical scenarios, UML activity diagram has some significant advantages due to both its conceptual base and also to its notational convenience” [8]. According to table 1, since performance annotation model has three layers, UML activity diagram is chosen to be annotated in all three layers in this paper.

2.2 Process for Constructing Performance Annotation Model

Since UML activity diagram is chosen to construct performance annotation model and CPN is chosen to construct performance analysis model, new performance annotation model construction process including steps for facilitating construction of CPN based performance analysis model should be proposed. Moreover, according to table 1, since performance annotation model consists of three sub-models, its construction process should also consist of three sub-processes and each sub-model corresponds to each sub-process. Furthermore, since three sub-models have fixed order from the top down, the using sequence of the three sub-processes should also comply with the same order.

2.2.1 Sub-process for Constructing Organization Annotation Model

Starting from the UML activity diagrams modeling key performance scenarios describing interactions among actors and information system in the organization layer, the sub-process for constructing organization annotation model consists of two steps, which are “attaching performance parameters to the UML activity diagrams” and “transforming transitions into object flows”. According to [5], “The key performance scenarios are those that are executed frequently or those that are critical to the perceived performance of the system”.

The aim of the first step is to get organization related performance parameters. The activity based approach defined in [8] is used in this step to attach parameters to the UML activity diagrams.

The aim of the second step is to facilitate construction of CPN based organization analysis model. Three operations are included in this step, which are: “transforming transitions between action states in actor swimlanes and activity states in information system swimlane into actor request object flows”, “assigning state ‘to be dealt with by processes’ to actor request objects in the object flows from action state to activity state”, and “assigning state ‘has been dealt with by processes’ to actor request objects in the object flows from activity state to action state”. When constructing CPN based organization analysis model, all actor request objects will be directly transformed into two socket places according to their states.

2.2.2 Sub-process for Constructing Software Annotation Model

Starting from the UML activity diagrams modeling key performance scenarios describing interactions among processes in the software layer and referring to the organization annotation model constructed, the sub-process for constructing software annotation model consists of three steps, which are “attaching performance parameters to the UML activity diagrams”, “supplementing actor request objects” and “supplementing process request objects”. In fact, each UML activity diagram annotated is the decomposition of the corresponding activity state in the organization annotation model.

The aim of the first step is to get software related performance parameters. The activity based approach defined in [8] is also used in this step to attach parameters to the UML activity diagrams.

The aim of both the second step and the third step is to facilitate construction of CPN based software analysis model. Three operations are included in the second step, which are “supplementing the same actor request object depending on an activity state in the organization annotation model to depend on the first activity state of the corresponding UML activity diagram”, “supplementing the same actor request object depended on by the activity state in the organization annotation model to be depended on by the last activity state of the UML activity diagram”, and “repeating operation one and two until all UML activity diagrams have been dealt with in the same way”. When constructing CPN based software analysis model, all actor request objects supplemented will be directly transformed into two port places according to their states. In CPN, it is hierarchical relationship that between socket place and port place.

Four operations are included in the third step, which are: “supplementing one process request object, which is depended on by the first activity state of an UML activity diagram and has the same name with the activity state and has state ‘to be dealt with by processor’”, “supplementing one process request object, which depends on the last activity state of the UML activity diagram and has the same name with the latest activity state and has state ‘has been dealt with by network’”, and “supplementing two process request objects for each other activity state of the UML activity diagram, of which one having the same name with the activity state and having state ‘to be dealt with by processor’ is depended on by the activity state and another having the same name with the latest activity state and having state ‘has been dealt with by network’ depends on the activity state”, and “repeating operation one to three until all UML activity diagrams have been dealt with in the same way”. When constructing CPN based software analysis model, all process request objects will be directly transformed into two socket places according to their states.

2.2.3 Sub-process for Constructing Hardware Annotation Model

In order to use UML activity diagram to construct hardware annotation model, referring to the queuing network based system execution model of a web based application in [5], a UML activity diagram based hardware resource interaction template shown in figure 1 is firstly proposed.

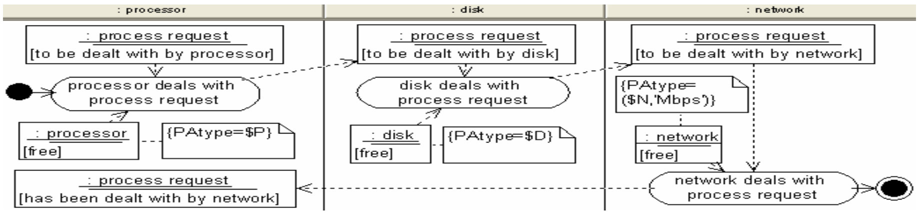
: processor
: disk
: network
: process request
[to be dealt with by processor]
: process request
[to be dealt with by disk]
: process request
[to be dealt with by network]
processor deals with process request
disk deals with process request
{PAtype=($N,'Mbps')}
: processor
[free]
{PAtype=$P}
: disk
[free]
{PAtype=$D}
: network
[free]
: process request
[has been dealt with by network]
network deals with process request

Figure 1. The Hardware Resource Interaction Template

Just as shown in figure 1, when a process request object sent by an activity state in the software annotation model enters the hardware layer, firstly, it waits to be dealt with by a processor. As soon as the processor is free, it is dealt with. Then, it waits to be dealt with by a disk. As soon as the disk is free, it is dealt with. Finally, it waits to be dealt with by a network. As soon as the network is free, it is dealt with. Moreover, three UML notes elements are used to specify types of processor, disk and network.

Starting from the UML deployment diagram modeling distributions of software processes or components and hardware resources of information system and referring to the hardware resource interaction template as well as the software annotation model constructed, the sub-process for constructing hardware annotation model includes one step, which is "getting hardware resource interaction diagrams". In fact, each hardware resource interaction diagram gotten is the decomposition of the corresponding activity state in the software annotation model.

Five operations are included in the step, which are: "assigning the same name of a process request object depended on by an activity state in the software annotation model to all process request objects in the hardware resource interaction template", "specifying names of all hardware resource objects dealing with the process request objects", and "assigning the same name of the hardware resource object to the swimlane including the object", and "assigning values to variables included in the three notes elements", and "repeating operation one to four until all activity states in the software annotation model have been dealt with in the same way".

In the step, the aim of operation one to three is to facilitate construction of CPN based hardware analysis model and the aim of operation four is to get hardware related performance parameters. When constructing CPN based hardware analysis model, all process request objects with state "to be dealt with by processor" or "has been dealt with by network" will be directly transformed into two port places according to their states. All process request objects with state "to be dealt with by disk" or "to be dealt with by network" will be directly transformed into two places according to their states. All hardware resource objects will be directly transformed into other three places according to their classes. All action states will be directly transformed into three CPN transitions according to their names.

3. METHOD APPLICATION

In order to validate the method proposed, it was used to construct performance annotation models of an online bookshop system based on its different architecture designs. In this section, only the application of the method based on the initial architecture design of the system will be briefly specified.

3.1 The Initial Architecture Design of the Online Bookshop System

According to the method, since only the UML activity diagrams modeling key performance scenarios and the UML deployment diagram modeling distributions of software processes or components and hardware resources are necessary, therefore, only these diagrams of the initial architecture design of the online bookshop system will be specified below.

In figure 2, the UML activity diagram modeling the initial key performance scenario describing the interactions between customer and the system is illustrated.

In figure 3, the UML activity diagram modeling the initial key performance scenario describing the interactions among processes iexplore.exe, inetinfo.exe and sqlservr.exe is illustrated. In fact, figure 3 is the decomposition of the activity state "Deal with catalog and booklist request" in figure 2. Because of the page limitation, other two UML activity diagrams, which are the decompositions of other two activity states in figure 2 are omitted here.

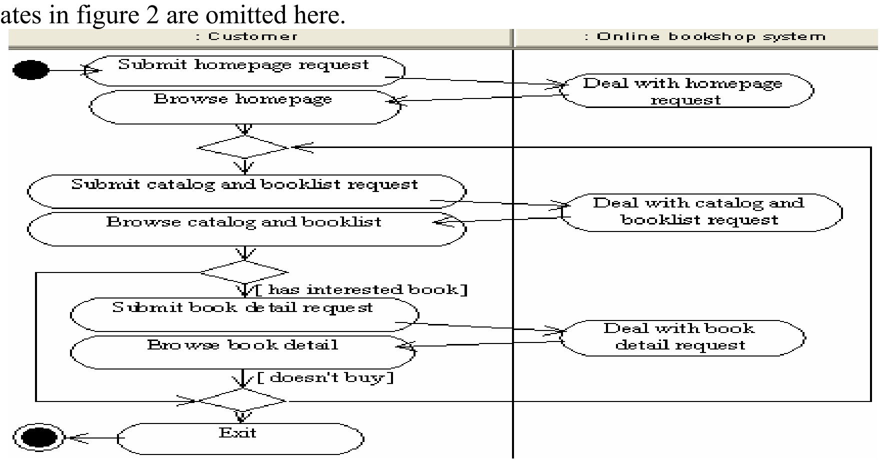
: Customer
: Online bookshop system
Submit homepage request
Deal with homepage request
Browse homepage
Submit catalog and booklist request
Deal with catalog and booklist request
Browse catalog and booklist
[has interested book]
Submit book detail request
Deal with book detail request
Browse book detail
[doesn't buy]
Exit

Figure 2. The Interactions between Customer and the System

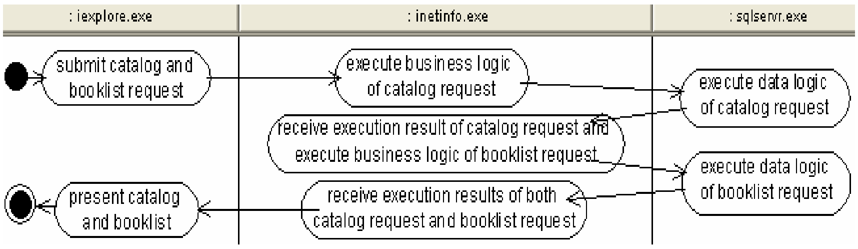
: iexplore.exe
: inetinfo.exe
: sqlservr.exe
submit catalog and booklist request
execute business logic of catalog request
execute data logic of catalog request
receive execution result of catalog request and execute business logic of booklist request
execute data logic of booklist request
present catalog and booklist
receive execution results of both catalog request and booklist request

Figure 3. The Interactions among Processes

In figure 4, the initial UML deployment diagram of the system is illustrated.

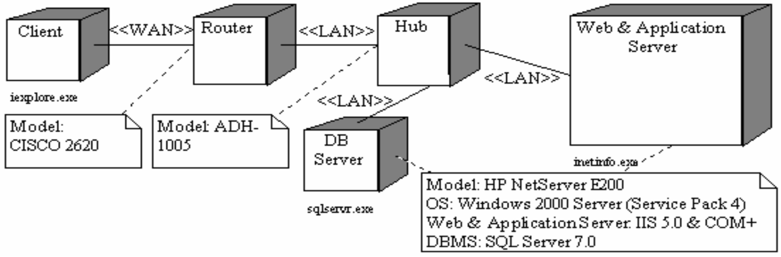

Figure 4. The Initial UML Deployment Diagram of the System

3.2 Construct Initial Performance Annotation Model

According to the method proposed, the initial performance annotation model based on the initial architecture design of the system was constructed.

3.2.1 Construct Initial Organization Annotation Model

According to the sub-process for constructing organization annotation model, starting from figure 2, the corresponding initial organization annotation model of the system was constructed and is illustrated in figure 5.

3.2.2 Construct Initial Software Annotation Model

According to the sub-process for constructing software annotation model, starting from figure 3 and referring to figure 5, the corresponding part of initial software annotation model of the system was constructed and is illustrated in figure 6.

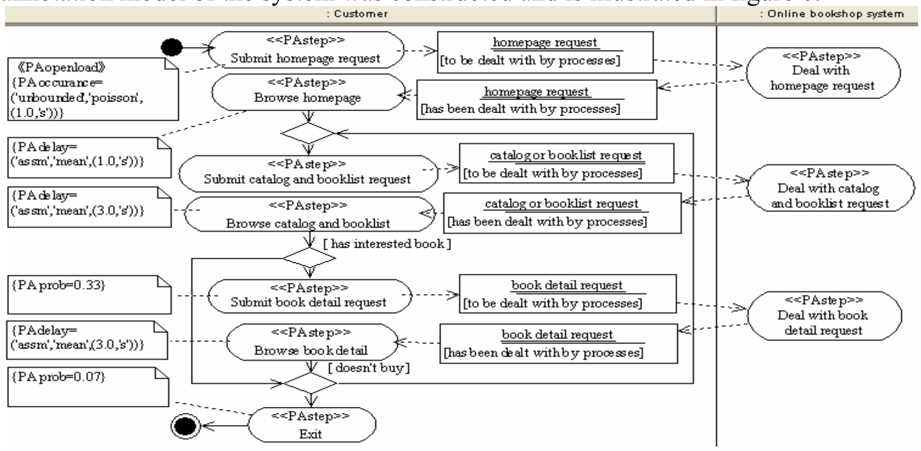
: Customer
: Online bookshop system
<<PAstep>> Submit homepage request
homepage request [to be dealt with by processes]
《PAopenload》 {PAoccurance= ('unbounded','poisson', (1.0,'s'))}
<<PAstep>> Deal with homepage request
<<PAstep>> Browse homepage
homepage request [has been dealt with by processes]
{PAdelay= ('assm','mean',(1.0,'s'))}
<<PAstep>> Submit catalog and booklist request
catalog or booklist request [to be dealt with by processes]
<<PAstep>> Deal with catalog and booklist request
{PAdelay= ('assm','mean',(3.0,'s'))}
<<PAstep>> Browse catalog and booklist
catalog or booklist request [has been dealt with by processes]
[has interested book]
{PAprob=0.33}
<<PAstep>> Submit book detail request
book detail request [to be dealt with by processes]
<<PAstep>> Deal with book detail request
{PAdelay= ('assm','mean',(3.0,'s'))}
<<PAstep>> Browse book detail
book detail request [has been dealt with by processes]
[doesn't buy]
{PAprob=0.07}
<<PAstep>> Exit

Figure 5. The Corresponding Initial Organization Annotation Model of the System

3.2.3 Construct Initial Hardware Annotation Model

According to the sub-process for constructing hardware annotation model, starting from figure 4 and referring to both figure 1 and figure 6, one hardware resource interaction diagram, which is the decomposition of the corresponding activity state “execute business logic of catalog request” in figure 6, was gotten and is illustrated in figure 7.

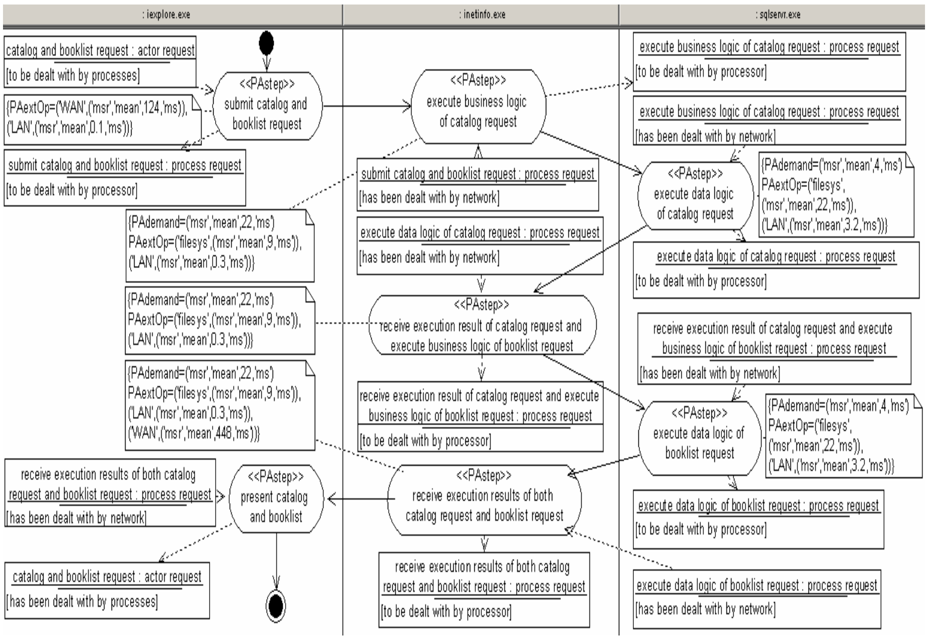
: iexplore.exe
: inetinfo.exe
: sqlservr.exe
catalog and booklist request : actor request
[to be dealt with by processes]
{PAextOp=('WAN',('msr','mean',124,'ms')),
('LAN',('msr','mean',0.1,'ms'))}
<<PAstep>>
submit catalog and booklist request
submit catalog and booklist request : process request
[to be dealt with by processor]
{PAdemand=('msr','mean',22,'ms')
PAextOp=('filesys',('msr','mean',9,'ms')),
('LAN',('msr','mean',0.3,'ms'))}
{PAdemand=('msr','mean',22,'ms')
PAextOp=('filesys',('msr','mean',9,'ms')),
('LAN',('msr','mean',0.3,'ms'))}
{PAdemand=('msr','mean',22,'ms')
PAextOp=('filesys',('msr','mean',9,'ms')),
('LAN',('msr','mean',0.3,'ms')),
('WAN',('msr','mean',448,'ms'))}
receive execution results of both catalog request and booklist request : process request
[has been dealt with by network]
<<PAstep>>
present catalog and booklist
catalog and booklist request : actor request
[has been dealt with by processes]
<<PAstep>>
execute business logic of catalog request
submit catalog and booklist request : process request
[has been dealt with by network]
execute data logic of catalog request : process request
[has been dealt with by network]
<<PAstep>>
receive execution result of catalog request and execute business logic of booklist request
receive execution result of catalog request and execute business logic of booklist request : process request
[to be dealt with by processor]
<<PAstep>>
receive execution results of both catalog request and booklist request
receive execution results of both catalog request and booklist request : process request
[to be dealt with by processor]
execute business logic of catalog request : process request
[to be dealt with by processor]
execute business logic of catalog request : process request
[has been dealt with by network]
<<PAstep>>
execute data logic of catalog request
{PAdemand=('msr','mean',4,'ms')
PAextOp=('filesys',
('msr','mean',22,'ms')),
('LAN',('msr','mean',3.2,'ms'))}
execute data logic of catalog request : process request
[to be dealt with by processor]
receive execution result of catalog request and execute business logic of booklist request : process request
[has been dealt with by network]
<<PAstep>>
execute data logic of booklist request
{PAdemand=('msr','mean',4,'ms')
PAextOp=('filesys',
('msr','mean',22,'ms')),
('LAN',('msr','mean',3.2,'ms'))}
execute data logic of booklist request : process request
[to be dealt with by processor]
execute data logic of booklist request : process request
[has been dealt with by network]

Figure 6. The Corresponding Part of Initial Software Annotation Model of the System

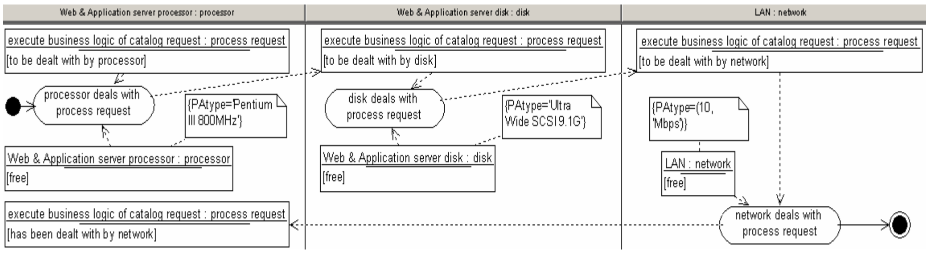
Web & Application server processor : processor
Web & Application server disk : disk
LAN : network
execute business logic of catalog request : process request
[to be dealt with by processor]
processor deals with process request
{PAtype=Pentium III 800MHz'}
Web & Application server processor : processor
[free]
execute business logic of catalog request : process request
[has been dealt with by network]
execute business logic of catalog request : process request
[to be dealt with by disk]
disk deals with process request
{PAtype='Ultra Wide SCSI 9.1G'}
Web & Application server disk : disk
[free]
execute business logic of catalog request : process request
[to be dealt with by network]
{PAtype=(10, 'Mbps')}
LAN : network
[free]
network deals with process request

Figure 7. The Hardware Resource Interaction Diagram

4. CONCLUSION AND FUTURE WORK

Since UML diagrams annotated and processes proposed are two primary deficiencies existing in current methods for constructing performance annotation model based on architecture design of information system, in this paper, UML activity diagram is proposed to be annotated both in the organization layer and in the hardware layer. Moreover, performance annotation model construction process including steps for facilitating construction of CPN based performance analysis model is devised. Furthermore, application of the proposed method based on the initial architecture design of an online bookshop system is briefly specified.

Based on the initial performance annotation model constructed, the corresponding CPN based initial performance analysis model was constructed. Figure 8 illustrates the initial organization analysis model of the system, which was transformed from figure 5.

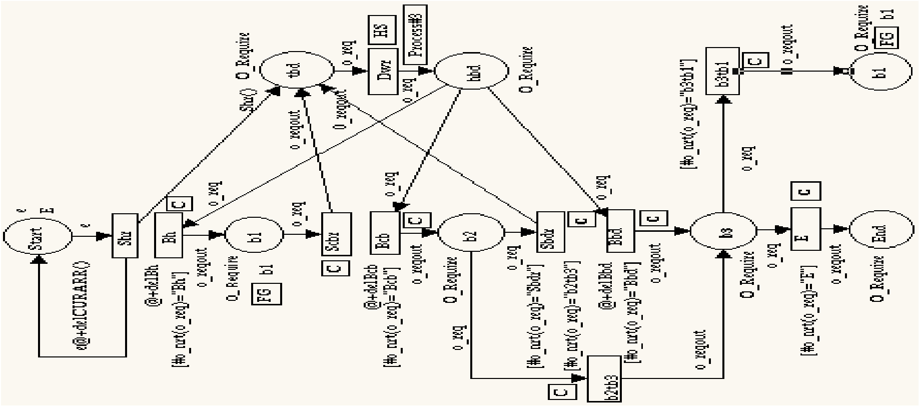

Figure 8. The Initial Organization Analysis Model of the System

In figure 8, the socket place "tbd" was transformed directly from all actor request objects with state "to be dealt with by processes" in figure 5. The socket place "hbd" was transformed directly from all actor request objects with state "has been dealt with by processes" in figure 5.

Based on the initial performance analysis model, simulation was run in the Design/CPN. The performance indexes of the initial architecture design gotten from the simulation are illustrated in figure 9.

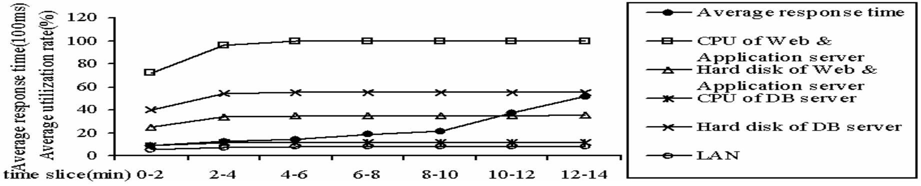

Figure 9. The Performance Indexes of the Initial Architecture Design

Following the same process, performance indexes of different architecture designs of the online bookshop system were also gotten and the architecture design meeting performance objectives was finally identified and chosen. However, all performance indexes gotten from simulations have not yet been validated by a working system. It is the urgent work to be done in the near future.

REFERENCES

1. A.L. Opdahl, *Performance Engineering during Information System Development* (Institutt for Datateknikk of Telmatikk: Trondheim, 1992).
2. Z. Yan, *Dynamic Performance Modeling and Analysis of Information System Based on Colored Petri Net*. Ph.D Thesis, Beijing Institute of Technology (2001).
3. H. Du, R. Gan, K. Liu, and Z. Zhang, A Framework for Architecture-Oriented Modeling and Analysis of Information Systems Performance, in *the Proc. of the 3rd International Conference on Wireless Communications, Networking and Mobile Computing, IEEE* (Forthcoming, 2007).
4. S. Balsamo and M. Marzolla, Performance Evaluation of UML Software Architectures with Multiclass Queuing Network Models, in *Proc. of the 5th International Workshop on Software and Performance* (ACM Press: New York, 2005), pp.37-42.
5. C.U. Smith and L.G. Williams, *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software* (Addison Wesley: Boston, 2001).
6. S. Distefano, M. Scarpa, and A. Puliafito, Software Performance Analysis in UML Models, in *Proc. of the 2005 Workshop on Techniques, Methodologies and Tools for Performance Evaluation of Complex Systems* (IEEE Computer Society: Washington, DC, 2005), pp.115-125.
7. S. Balsamo and M. Marzolla, Efficient Performance Models in Component-Based Software Engineering, in *Proc. of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications* (IEEE Computer Society: Washington, DC, 2006), pp.64-71.
8. Anonymous, *UML Profile for Schedulability, Performance, and Time specification,* Object Management Organization (2005). http://www.omg.org/docs/formal/05-01-02.pdf (Accessed July 14, 2007).
9. M. Woodside, D.C. Petriu, D.B. Petriu, H. Shen, T. Israr, and J. Merseguer, Performance by unified model analysis (PUMA), in *Proc. of the 5th international workshop on Software and performance* (ACM Press: New York, 2005), pp.1-12.
10. K. Jensen, *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use Volume 1* (Springer: Berlin, 1992).