This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier’s archiving and manuscript policies are encouraged to visit:

http://www.elsevier.com/copyright
Dwelling time probability density distribution of instances in a workflow model

Liu Sheng\textsuperscript{a,}, Fan Yushun\textsuperscript{a}, Lin Huiping\textsuperscript{b}

\textsuperscript{a} CIMS Center, Department of Automation, Tsinghua University, Beijing, China
\textsuperscript{b} School of Software and Microelectronic, Peking University, Beijing, China

\textbf{Abstract}

This paper presents a method to judge whether a business process is successful or not. A business process is deemed successful if a large enough proportion of instances dwell in a workflow (wait and be executed) for less than given period. By analyzing instances’ dwelling time distribution in a workflow, the proportion of instances which dwell in the workflow for less than any given period will be achieved. The performance analysis of workflow model plays an important role in the research of workflow techniques and efficient implementation of workflow management. It includes the analysis of instances’ dwelling time distribution in a workflow process. Theoretical methods have been discussed in Workflow Management Coalition (1997), Hofstede and Orlowska (1999). The objective of resource dependency logic verification is to prove correctness of the static resource dependency logic, and information dependency logic (Li et al., 2004; van der Aalst et al., 2000). The correctness analysis of process control logic aims to avoid the deadlocks or structural conflicts in the execution of a workflow model caused by the errors in its process control. Some verification and conflict detection methods have been discussed in Workflow Management Coalition (1998), van der Aalst and ter Hofstede (2000), Han, Himmighofer, Schaaf, and Wikarski (1996). The rationality and correctness analysis should be carried out from four aspects that are relevant for workflow modeling and execution: process control logic, timing constraint logic, resource dependency logic, and information dependency logic (van der Aalst, 2003). The objective of resource dependency logic verification is to prove correctness of the static resource allocation rules and consistence with the process control logic. The information dependency logic cares about the internal consistence of a workflow-related data and the correctness of temporary relation among different workflow applica-
tion data. The timing constraint verification and analysis deal with the temporal aspects of a workflow model such as deadlines (Panagos & Rabinovich, 1997; Pozevaunig, Eder, & Liebhart, 1997; van der Aalst et al., 2007), time scales (Marjanovic, 2000; Marjanovic & Orlowska, 1999a, 1999b; Qu, Lin, & Wang, 2002; Sadiq, Marjanovic, & Orlowska, 2000; van der Aalst et al., 2000; Zhuge, Cheung, & Pung, 2001) schedulability analysis (van der Aalst, 1996), and boundedness verification (Li, Fan, & Zhou, 2003) and time violation handling (Eder, Panagos, Pozevaunig, & Rabinovich, 1999; Eder, Panagos, & Rabinovich, 1999; Sadiq, Orlowska, Sadiq, & Schulz, 2005). Quality of Service in Flexible Workflows is discussed in Sadiq, Orlowska, Lin, and Sadiq (2006), Sadiq, Orlowska, Sadiq, and Lin (2005). The above analysis can ensure only the functionally working workflow (correctness) but not its operational efficiency. The performance level (Eder, Panagos, Pozevaunig, et al., 1999; Eder, Panagos, Rabinovich, 1999; Ferscha, 1994a, 1994b; Kevin, Ng, Ghanmi, Lam, & Mitchell, 2002; Li et al., 2004; Lin, Qu, Ren, & Marinescu, 2002; Schomig & Rau, 1995; Son & Kim, 2001; van Hee & et al., 2000; Yamaguchi, Qi-Wei, & Tanaka, 2000). On the other hand, aims to evaluate the ability of the workflow to satisfy requirements concerning some key performance indicators such as, maximal parallelism, throughput, service levels, and sensitivity. The analysis of resource availability and utilization, and average turnaround time is performed at this level. Performance analysis of workflows has not get enough attention of researchers commensurate with its importance until now (Salimifard & Wright, 2001). The performance analysis of a workflow model (business process) is different from that of WIMS architecture (Gillmann, Weissfels, Weikum, & Kraiss, 2000; Kim & Ellis, 2001). The performance analysis can be conducted only after the rationality and correctness analysis has been carried out. So it is assumed that there are no temporal and logical errors in the considered workflow models at the performance analysis stage.

PNs are the only formal techniques able to be used for structural modeling and a wide range of qualitative and quantitative analysis (Salimifard & Wright, 2001). PN-based workflow management systems are widely used because of formal semantics, local state-based system description, and abundant analysis techniques (van der Aalst & et al., 1998, chap. 10). So PNs are a naturally selected mathematical foundation for the formal performance analysis of workflows. Many researchers use PN techniques to study workflows. Many researchers study PN techniques to study workflows (Adamin et al., 1998; Ferscha, 1994a, 1994b; Han et al., 1996; Lin et al., 2003; Li et al., 2003; Lin et al., 2002; Panagos & Rabinovich, 1997; Schomig & Rau, 1995; van der Aalst, 1998; van der Aalst & ter Hofstede, 2000; van Hee et al., 2000) since Zisman used PN to model workflow processes (Zisman, 1977).

A petri net (PN) is a graphical and mathematical modeling tool. It consists of places, transitions, and arcs that connect them. Input arcs connect places with transitions, while output arcs start at a transition and end at a place. There are other types of arcs, e.g., inhibitor arcs. Places can contain tokens, the current state of the modeled system (the marking) is given by the number and type of tokens if the tokens are distinguishable of tokens in each place. Transitions are active components. They model activities which can occur (the transition fires), thus changing the state of the system (the marking of the petri net). Transitions are only allowed to fire if they are enabled, which means that all the preconditions for the activity must be fulfilled (there are enough tokens available in the input places). When the transition fires, it removes tokens from its input places and adds some at all of its output places. PN which model workflow process definition are called WF-nets (van der Aalst, 1998; van der Aalst & van Hee, 1996). WF-nets are extended to MWF-nets with time, role, and resource information (Li et al., 2004). Methods are discussed to compute the workload that arrival transaction instances generate for the various resource pools and the lower bound of average turnaround time of transaction instances (Li et al., 2004). This paper adopts MWF-nets (Li et al., 2004) as a base mechanism to represent a performance analysis oriented workflow model.

2. Related works

A high-level stochastic PN (SPN) is used to model the routing constructs of a workflow, and then a method to compute throughput of the process is presented (van Hee & et al., 2000). Based on four performance equivalent formulae, the performance of a workflow is approximately analyzed in Lin et al. (2002). These two techniques both aim at calculating instances’ execution time and ignoring waiting time. The probability density of execution time is not taken into account, and cannot be applied to a workflow process of which the resources have stochastic service time. All the control structures are mapped into a Generalized stochastic PN (SPN) (Ferscha, 1994a, 1994b), and then a method based on a CMT to obtain lower bounds of the execution performance is discussed. A so-called load equivalence aggregation model derived from GSPN has been developed in Schomig and Rau (1995), and then some performance-related measures of human resources in a workflow by obtained by simulating the model. By defining change time, a performance evaluation model for the dynamic workflow changes is brought forward in Yamaguchi et al. (2000). However, the technique can be used for only acyclic time WF-net in which the arrival intervals of transaction instances are constant. A queuing network is used to model the workflow (Kevin et al., 2002; Son & Kim, 2001). A method is yielded to identify the critical path of a workflow model and determine the minimum number of servers for the critical activity (Son & Kim, 2001). Some approximate approaches are employed in Kevin et al. (2002) for workflow configuration, and then the corresponding network is analyzed. But these techniques are not immediately applicable since they both assume that dedicated servers exist for an activity’s execution. Existing modeling and analysis techniques used to resolve different aspects of workflow performance-related problems have mainly two shortcomings, which restrict their application in practice. One is that almost no information about instances’ dwelling time (sum of waiting time and execution time) is considered. The other is that no accurate approaches are used to analyze probability density distribution of the instances’ dwelling time in the whole workflow. When obtaining the arrival interval and service time are stochastic. Although some literatures take instances’ dwelling time into account in their workflow models, only some simple disciplines, such as constant arrival interval (Son & Kim, 2001; Yamaguchi et al., 2000) mean execution time (Lin et al., 2002; van Hee et al., 2000) are supposed. They are not necessarily true in actual workflow systems.

In this paper we discuss a method to judge whether a business process is successful or not. In order to get the proportion of instances which dwell in the workflow for less than any given period, instances’ dwelling time distribution in a workflow must be analyzed. There are still no formal papers that focus on this problem so far. This paper will discuss this question.

Section 3 introduces some relevant Queuing Theory and queuing model in the workflow model. Section 4 provides an algorithm computing the instances’ dwelling time probability density at a transaction (activity). Section 5 discusses the method to compute the instances’ dwelling time probability density at four control structures of workflow models. Section 6 discusses the method to compute the instances’ dwelling time probability density at a workflow model. Section 7 presents the method to shorten the instances’ dwelling time by increase number of resources. Section 8 presents an example.
3. Queuing models in workflow models

It is assumed in this paper that the firing of a transition (execution of the corresponding activity) needs the support of a specific role. The situation that one transition is projected to several roles can be transformed to this mapping relation by redefining roles and organization structure (Li et al., 2004).

In the framework of MWF-nets (Li et al., 2004), a resource pool is a class of individual resource agents that have the same skills and capability and performs the same set of roles. \( R = \{ c_1, c_2, \ldots, c_l \} \) which is called resource state of the MWF-net means that there are \( c_j \) individual resource agents in each resource pool \( r_j \). Each role can be allocated to support several transitions’ firing and is performed by an individual resource agent of one of the appointed resource pools. Each resource pool may be appointed to undertake many roles. In the enactment environment, the service request (processing of a transaction instance) generated by the firing of a transition is projected (using role as the medium) to one of the individual resource agents that have the probability to be appointed as the performer.

According to queuing theory, the processes of transaction instances form a queuing model in which the transaction instances act as customers and the resources act as servers. Each instance must queue for the service. Its service time is specified by the firing delay of the corresponding transition (executing time of the corresponding activity). Its dwelling time is the sum of its waiting time and service time.

Many business processes have time constraints called deadlines in their corresponding workflows. Hence, the instances’ dwelling time in a workflow model should satisfy the deadline requirement. Sometimes their service time and arrival intervals in a workflow are stochastically distributed. In order to evaluate a business process a method to compute the instances’ dwelling time probability density in a workflow model is discussed in the following.

A workflow model is an activity network where activities are interconnected by four workflow control structures, i.e., sequence, concurrency (AND), alternative (OR) and iteration (LOOP) (Adam et al., 1998).

Let \( df(t), fw(t) \) and \( sf(t) \) denote the probability density function of dwelling, waiting and service time probability density distribution, respectively, then \( df(t) \) at a workflow will be calculated by three steps:

- calculate \( df(t) \) in each activity;
- calculate \( df(t) \) in each control structure;
- calculate \( df(t) \) in the whole workflow.

4. Instances’ dwelling time probability density in an activity

It is assumed that each queuing system has infinite capacity. The service time of each resource agent is exponentially distributed with an average service rate \( \mu_j \) (instances per hour). The instances arrive with exponentially distributed inter-arrival times at an average rate of \( \lambda_j \) (instances per hour). Therefore the workflow model can be modeled as an M/M/C queuing network where each activity is an independent M/M/C queuing system. It is supposed that there are \( c_j \) resource agents for activity \( j \) and \( m_j \) instances in activity \( j \). Each resource agent serves for one queue, and each instance will enter the shortest queue when it arrives. Let \( l_j = \text{Round}(m_j/c_j) \). When an instance arrives, if \( m_j \geq c_j \), it has to queue. Otherwise, it is processed directly. Let \( P_{m_j} \) denote the probability with which the length of the shortest queue is \( l_j \). Let \( sf/l_j \) denote probability density of the last instance’s dwelling time probability density in the queue whose length is \( l_j \). As is known, its dwelling time distribution is an Erlang distribution. Then we have

\[ sf(t) = \mu_je^{-\mu_j t} \]  
\[ P_0 = \sum_{l_j=0}^{c_j} \left( \frac{\lambda_j \mu_j}{l_j!} \right)^{l_j} \frac{1}{l_j!} \]  
\[ P_{m_j} = \left\{ \begin{array}{ll}
P_{m_j} \mu_j \frac{\lambda_j \mu_j}{(m_j - c_j)!} & (m_j \leq c_j) \\
P_{m_j} \mu_j \frac{\lambda_j \mu_j}{(m_j - c_j)!} & (m_j > c_j)
\end{array} \right. \]

When \( l_j \geq 1, \) \( sf(t) \) of activity \( j \) can be stated as

\[ sf(t) = \sum_{l_j=0}^{\infty} \sum_{m_j=c_j}^{\infty} \left( \frac{\lambda_j \mu_j}{l_j!} \right)^{l_j} \frac{1}{l_j!} e^{-\mu_j t} \left( \frac{\lambda_j \mu_j}{(m_j - c_j)!} \right) e^{-\mu_j t} \]

Also it can be illustrated as

\[ sf(t) = \sum_{l_j=0}^{\infty} \left( \frac{\lambda_j \mu_j}{l_j!} \right)^{l_j} \frac{1}{l_j!} e^{-\mu_j t} \left( \frac{\lambda_j \mu_j}{(m_j - c_j)!} \right) e^{-\mu_j t} \]

Let \( A_l = \frac{\left( \sum_{i=0}^{m_j} \frac{\lambda_i \mu_i}{(i-\lambda_i)!(m_j-i)!} \right)}{\sum_{i=0}^{m_j} \lambda_i \mu_i} \) \( B_{l_j} = \left( \frac{\lambda_j \mu_j}{m_j-c_j} \right) e^{-\mu_j t} \), then Eq. (5) can be illustrated as

\[ sf(t) = \sum_{l_j=0}^{\infty} \left( \frac{\lambda_j \mu_j}{l_j!} \right)^{l_j} \frac{1}{l_j!} e^{-\mu_j t} \left( \frac{\lambda_j \mu_j}{(m_j - c_j)!} \right) e^{-\mu_j t} \]

When an instance arrives, there are a probability that it must queue (\( l_j > 0 \)) and a probability that it is processed directly (\( l_j = 0 \)). It can be expressed as

\[ \int_0^t df_j(t) dt = \int_0^t fw_j(t) dw_j dt + \int_0^t sf_j(t) dt \]

The differential equation about \( x \) of Eq. (7) is stated as

\[ df_j(x) = \mu_je^{x} df_j(t)e^{\mu_j t} dt + \sum_{m_j=0}^{l_j} \sum_{m_j=0}^{l_j} P_{m_j} \mu_je^{x} \]

\[ = \mu_je^{x} \int_0^x A_Be^{x} dt + \sum_{m_j=0}^{l_j} \sum_{m_j=0}^{l_j} P_{m_j} \mu_je^{x} \]

\[ = ABe^{(x-1)x} \]

When there is only one resource agent for activity \( j \), we have

\[ sf_j(x) = (\mu_j - \lambda_j) e^{(\mu_j - \lambda_j)x} \]

Thus the instances’\( sf(t) \) in an activity is figured out.
5. Instances’ dwelling time probability density in four basic control structures

Let \( f_{d_{ij}}(t) \) denote the instances’ \( fd(t) \) at a control structure interconnected by activity 1, activity 2, …, and activity \( n \). It is assumed that the instances’ dwelling time in each activity is independent.

5.1. Instances’ \( fd(t) \) in concurrent control structure

The instances’ dwelling time in a concurrent control structure is the longest dwelling time in its branches. An instance in a concurrent control structure with \( n \) branches can be divided into \( n \) independent sub-instances which may run in parallel. Suppose activity \( i \) and activity \( j \) in Fig. 1 are interconnected by a concurrent control structure in Fig. 1. Thus we have

\[
\int_0^x t * f_{d_{ij}}(t) dt = \int_0^x u * f_{d_i}(u) \int_0^u f_{d_j}(v) dv du + \int_0^x u * f_{d_j}(u) \int_0^u f_{d_i}(v) dv du
\]

The differential equation about \( x \) of Eq. (10) is stated as

\[
f_{d_{ij}}(x) = f_{d_i}(x) \int_0^x f_{d_j}(v) dv + f_{d_j}(x) \int_0^x f_{d_i}(v) dv
\]

When a concurrent control structure is composed of activity 1, activity 2, …, and activity \( n \), Eq. (11) can also be illustrated as

\[
f_{d_{12..n}}(x) = \sum_{i=1}^n \left( \int_0^x f_{d_i}(v) dv \prod_{k=1}^{i-1} \int_0^x f_{d_k}(t) dt \right)
\]

5.2. Instances’ \( fd(t) \) in sequential control structure

The instances’ dwelling time in a sequential control structure is the sum of instances’ dwelling time at all of its branches. Let \( u_1, u_2, \ldots, u_n \) be the instances’ dwelling time in activity 1, activity 2, …, and activity \( n \) interconnected by a sequential control structure. Let \( t \) be the instances’ dwelling time in the structure. We know \( t = u_1 + u_2 + \cdots + u_n \). Thus the instances’ \( fd(t) \) in a sequential control structure can be expressed as

\[
\int_0^x f_{d_{12..n}}(t) dt = \int_0^x \int_0^{u_1 + u_2 + \cdots + u_n} \cdots \int_0^{u_1} \prod_{k=1}^n f_{d_k}(u_k) du_1 du_2 \cdots du_n
\]

5.3. Instances’ \( fd(t) \) in alternative control structure

Each instance in an alternative control structure is exclusively served at branch \( i \) with probability \( P_i \). When an alternative control structure consists of activity 1, activity 2, …, and activity \( n \), the instances’ \( fd(t) \) at alternative control structure is shown as

\[
f_{d_{12..n}}(t) = \sum_{i=1}^n P_i f_{d_i}(t)
\]

5.4. Instances’ \( fd(t) \) in loop control structure

In a loop control structure the instances return to be served again in certain activity with probability \( q \) after they pass it. For example, activity \( i \) and activity \( j \) in Fig. 2 are interconnected by a loop control structure. The instances return to be served again at activity \( i \) via activity \( j \) with probability \( q \) after they pass it. Then the instances will be served for \( n \) times by the resource agents with probability \( P_n = q^{(n-1)}(1-q) \). The loop control structure shown in Fig. 1a is equivalent to the alternative structure shown in Fig. 1b.

In fact the probability with which the instances are served for three or more times in loop control structure is so small that it may be ignored. Then the loop control structure is regarded as an alternative structure with three branches. Let \( f_{d_i}(t), f_{d_{ii}}(t) \) and \( f_{d_{iii}} \) imply the instances’ \( fd(t) \) in the first three branches of equivalent structure in Fig. 1b. According to Eq. (14), the instances’ \( fd(t) \) in a loop structure is stated as

\[
f_{d_{loop}}(t) = (1 - q) * f_{d_i}(t) + q(1 - q) * f_{d_{ii}}(t) + q^2(1 - q) * f_{d_{iii}}(t)
\]

Branch 2 is sequential control structure as well as branch 3 in Fig. 1b. According to Eq. (13), the instances’ \( fd(t) \) in each branch can be obtained as

\[
\int_0^x f_{d_{ii}}(t) dt = \int_0^x \int_0^{u_i} \int_0^{u_{i-1}} \cdots \int_0^{u_1} f_{d_{i}}(w) f_{d_{j}}(v) f_{d_{k}}(u) dw dv du
\]

And

\[
\int_0^x f_{d_{iii}}(t) dt = \int_0^x \int_0^{u_i} \cdots \int_0^{u_{i-1}} \prod_{k=1}^{(n+1)/2} f_{d_k}(u_{2k-1}) \int_0^{u_{2k-1}} f_{d_{ii}}(u_{2k}) du_{2k} \cdots du_n
\]

6. Instances’ \( fd(t) \) in a workflow model

When the instances’ \( fd(t) \) in each activity of a workflow model is calculated, the instances’ \( fd(t) \) in each control structure of a workflow will be figured out according to Eqs. (12)–(15). Then each control structure is considered as an activity with the same instances’ \( fd(t) \). After this step, the activities are reduced and the workflow model is simplified. Then the instances’ \( fd(t) \) in each control structure of the simplified workflow model is computed. And the model is simplified again until there is one activity in the model. The instances’ \( fd(t) \) in the activity is the instances’
7. Balance between workflow performance and resource costs

For a specified workflow model mentioned above, the resource costs will increase with the instances that are accomplished within the deadline when the hired resources increase. And the resources costs will increase with the instances that are accomplished within a deadline. Therefore, P(0 < t < T) and E(t) can be figured out if fd(t) is gained. According to Eqs. (8), (12), (13), (14), (15), (16), and (17), we have

\[
fd(0 < t < T) = \int_0^T fd(t)dt
\]

and

\[
E(t) = \int_0^\infty t * fd(t)dt
\]

8. An example

Suppose that a business process defined in the workflow model can be stated in Fig. 2. The arrival and departure processes of its activities make up of an M/M/C queuing network. The necessary parameters are listed in Table 1. And it should be ensured that 98% of the instances dwell in the workflow model for less than an hour.

Let P(0 < t < T) and E(t) be the proportion of the instances that depart before the deadline T and the mean dwelling time of the instances. Then the relations between P(0 < t < T), E(t) and fd(t) are stated as:

\[
P(0 < t < T) = \int_0^T \int_0^\infty \frac{t * fd(t)}{C_0} dt \]

\[
E(t) = \int_0^\infty \frac{t * fd(t)}{C_0} dt
\]

Table 1

<table>
<thead>
<tr>
<th>Activity</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial resource agents number</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>μ (\text{specified, others deduced})</td>
<td>16</td>
<td>20</td>
<td>20</td>
<td>36</td>
<td>24</td>
<td>40</td>
<td>10</td>
</tr>
<tr>
<td>\text{then} (\text{specified, others deduced})</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>20</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 2. Workflow queueing network.

Fig. 3. The graph of \( fd_{1234567}(t) \) (t \( \in \) [0, 1.2]).

Fig. 4. The graph of \( fd_{1234567}(t) \) (t \( \in \) [0.0180, 0.0185]).

Now it is ensured that 99.8% percent of the instances dwell in the workflow model for less than an hour. Obviously the requirement is satisfied. Otherwise we must add resources for the corresponding activity and compute the \( fd(t) \) again.
9. Conclusion

This paper has presented a theoretical method to calculate the instances’ dwelling time probability density in a workflow where the activities are structured and predictable. By this method the instances’ dwelling time distribution and satisfactory degree based on dwelling time can be analyzed. An example has shown its availability in practice. This paper for the first time considers all the necessary information for the performance-related theoretical analysis of a workflow model. Firstly, an MWF-net is used to model the workflow. Then, it is assumed that the service time of each resource agent is exponentially distributed and the instances arrive with exponentially distributed inter-arrival times. Since activities are the basic units of the workflow, the probability density function of instances’ dwelling time distribution in an activity is calculated firstly. Then with the result the functions in four control structures are computed. The method to obtain the function in a workflow is discussed. At last an approach is conducted to improve the dwelling time distribution of the instances. In this paper the routing of transaction instances in the TWF-net is mapped into the service request arrive rate.

During the discussion of workflow performance-related analysis, it is assumed that service time of each resource agent is exponentially distributed and the instances arrive with exponentially distributed inter-arrival times and the resources do not need to repair. The techniques proposed in this paper need to be extended to deal with the case that service time and arrival interval are normally distributed and the resources have time to repair. This will be left for future exploration.

Acknowledgements

This work was funded by National High Technology Research and Development (863) of China under Grant 2006AA04Z151, National Natural Science Foundation of China (Project No. 60704027).

References


Sadiq, Shazia, Orlowska, Maria E., Sadiq, W., & Lin, Joe (2005). Quality of service in flexible workflows through process constraints. In The international conference on enterprise information systems 2005, March, USA.

Sadiq, Shazia, Orlowska, Maria E., Sadiq, Wasim, Schulz, Karsten (2005). When workflows will not deliver – Case contradicting work practice. In BIS05, April, Poznan, Poland.


