Petri net Workflow Monitor Models for Task Force Automation

Fu-Shiung Hsieh
Department of Computer Science and Information Engineering
Chaoyang University of Technology, Taiwan, R.O.C.
E-mail : fshsieh@cyut.edu.tw

Abstract

Consider the task force assignment problem that takes place from time to time in a company. The staff in the sale office received the query of a potential order. However, the sale staff cannot make the decision alone because there may be other decision makers involved in the decision making processes of the order. Traditionally, the sale staff needs to contact the staffs in other departments for costing, production planning, purchasing of materials and de-livery of the required products in the order. The sale staff must follow-up all the processes before a final decision can be made for the order. To follow-up all the processes, the sale staff may apply a variety of communication tools, including mobile phone, PDA, PC, e-mail and other workflow software, to contact the other staffs relevant to the decision making whenever necessary. The lack of integration among different software, hardware and systems makes it difficult for a sale staff to effectively make decision in collaboration with other staffs. Motivated by the above shortcomings of the existing practices, our research aims to propose an effective workflow model for the above task force assignment problem. We have proposed a methodology to model task force automation application based on Petri nets. To describe a task, we model its workflow information with a Petri net. To model a member's state, we adopt Petri net to model the activities of team member, where a set of places denote the states of a member and a set of transitions represent the change from one state to another. By combining the model workflow Petri nets with the activity Petri net models, we obtain an overall Petri net model to monitor the task execution.

Keywords: Workflow, Petri net, Monitor, Automation.

1. Introduction

Consider the task force assignment problem that takes place from time to time in a company. The staff in the sale office received the query of a potential order. However, the sale staff cannot make the decision alone because there may be other decision makers involved in the decision making processes of the order. Traditionally, the sale staff needs to contact the staffs in other departments for costing, production planning, purchasing of material and delivery of the required products in the order. The sale staff must follow-up all the processes before a final decision can be made for the order. To follow-up all the processes, the sale staff may apply a variety of communication tools, including mobile phone, PDA, PC, e-mail and other workflow software, to contact the other staffs relevant to the decision making whenever necessary. The lack of integration among different software, hardware and systems makes it difficult for a sale staff to effectively make decision in collaboration with other staffs. Motivated by the above shortcomings of the existing practices, our research aims to propose an effective model for the above task force assignment problem based on context-aware computing technologies.

Context-aware computing [1]-[7] is becoming more and more important with the wide deployment of ubiquitous/pervasive computing infrastructure. In this paper, we focus on application of context-aware computing technology in task force automation [8].The task force automation process is driven by the arrival of a potential order. Depending on the requirements of the order, a task workflow will be formed to determine whether the order should be committed. Typical business workflows usually require the formation of a collaborative network to solve the problem or process the task. Although collaborative network formation has been studied in literature, how to achieve effective collaborative network formation by taking the advantage of context-aware computing technologies is an important issue. The goal of this research is to propose a formal model for modeling and verification of the collaborative networks formed in a context-aware computing environment. To describe the context of a task, we model its contextual information with Petri nets [9]. With the proposed Petri net models, we propose a methodology to facilitate the design and analysis of context-aware task force automation applications.

The remainder of this paper is organized as follows. In Section 2, we introduce the task force automation problem. In Section 3, we propose resource activity models and workflow models in Petri nets. In Section 4, we propose a complete workflow monitor Petri net model. We conclude this paper in Section 5.

2. Task Force Automation

Task force automation is concerned with allocation of resources to achieve a certain business objective in a timely and efficient manner. Typically, a variety of resources are involved in the processes to accomplish a task. Fig. 1 shows different steps throughout the lifecycle of handling an order, from sale, accounting, receiving, manufacturing, quality assurance, packaging to shipping. Each step requires distinct resources for processing. For example, in sale step, sale staffs are involved in the negotiation as well as contracting. Costing of the order requires the assistance from the accountants. Therefore, a message is forwarded to the accountants for costing. Once the order contract is established, it will be released to the shop floor for production. In the production process, different operators and machines are assigned to perform the relevant operations. Each final product is then forwarded to the quality control staff for examination. Once the required qualified products are available, the operators package and then ship the products to the customer. The above scenario exhibits a very complicated process flow and intensive messaging and interactions between different workers in the team. How to effectively manage the concurrent workflows as well as messages in the system is a challenge.

Fig. 1 Task Force and Activities

Traditionally, the sale staff needs to contact the staffs in other departments for costing, production planning, purchasing of materials and delivery of the required products in the order. The sale staff must follow-up all the processes before a final decision can be made for the order. To follow-up all the processes, the sale staff may apply a variety of communication tools, including mobile phone, PDA, PC, e-mail and other workflow software, to contact the other staffs relevant to the decision making whenever necessary. The lack of integration among different software, hardware and systems makes it difficult for a sale staff to effectively make decision in collaboration with other staffs. Motivated by the above shortcomings of the existing practices, our research aims to propose an effective model for the above task force assignment problem based on context-aware computing technologies.

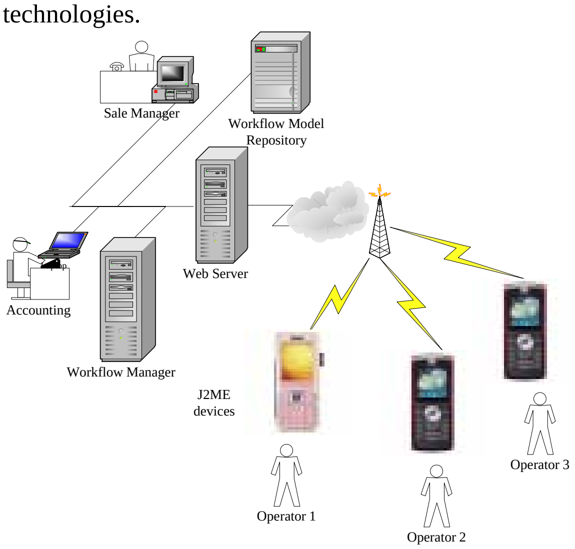

Fig. 2 Architecture for workflow task force automation

We assume each worker is equipped with a mobile phone that is used for communication with other team members. Although mobile phone provides a convenient media for communication, only short messages can be transferred via mobile phone due to its limited computing capabilities and display area. One of the design issues is to deliver the required information only to avoid overloading the mobile phone as well as the users. Another issue is to effectively enact the execution of operations to accomplish the task. Fig. 2 shows our proposed architecture for context-aware task force automation. In Fig. 2, the contextual model repository is used to capture the context of interest to different types of team members and that of task workflows. The information to be delivered to different members in the system varies depending on the role of the member involved in carrying out the task. The workflow manager aims to provide a model to monitor the progress of tasks that are needed to fulfill an order. A team member usually needs to participate in different parts of the task workflows. Depending on the role of a team member, the communication devices accessible vary. How to generate the context-aware information to guide a team member using a variety of different types of communication devices to collaboratively accomplish the task is a significant design issue.

Fig. 3 illustrates the proposed processes to compose a workflow monitoring model for task force automation. The inputs of the processes include resource activity models as well as workflow models. To monitor the progress of each workflow, a workflow monitor model is constructed for each task. Construction of workflow monitor model is obtained by taking into account the interactions between the resource activity models and the relevant workflow models of the tasks. A context composition procedure is needed to combine the resource activity models and the task workflow model into a workflow monitor model. Based on the workflow monitor model, the active user context elicitation procedure identifies the active user context corresponding to the current state of the workflow. Once the active user context has been identified, it can be used to guide the user to perform the next actions to accomplish the goal of the task force.

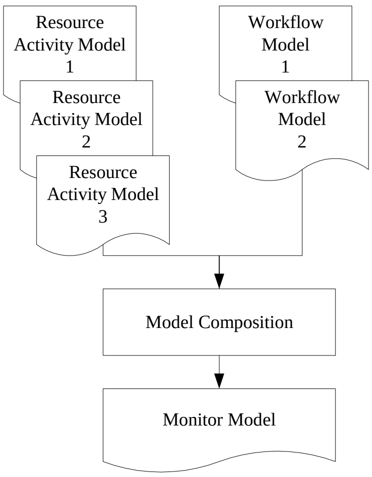

Fig. 3 Workflow monitoring model

3. Resource Activity and Workflow Models

To facilitate automated generation of context-aware guidance information for each team member, an appropriate contextual modeling approach must be adopted in this paper to capture the interactions between the individual contexts of the team members, the contexts of different task workflows as well as the contexts of the communication devices accessible. There are two requirements in selection of models. First, the model must have well-established formal mechanisms for modeling as well as analysis. Second, the model must be accompanied with a standardized interchange format. Petri net [9] is a model that meets these two requirements. Therefore, we adopt Petri nets to model the contexts. The advantages of applying Petri nets formalism to model and analysis of a context-aware system can be summarized as follows. First of all, the graphical nature of Petri nets can visualize sequences of firing via token passing over the net. Second, Petri nets have well-established formal mechanisms for modeling and property checking of systems with concurrent, synchronous and/or asynchronous structures. Third, the mathematical foundation of Petri nets can analyze structural and dynamic behaviors of a system. These advantages make Petri nets a suitable modeling and analysis tool. Furthermore, the emerging Petri Net Markup Language (PNML) [10] is a preliminary proposal of an XML-based interchange format for Petri nets. PNML makes it possible for individual companies to exchange their process models in Petri net. In this paper, PNML is used for sharing and exchanging the process models of participants in a team.

A Petri Net (PN) (Murata, 1989) G is a five-tuple $G = (P, T, I, O, m_0)$, where P is a finite set of places, T is a finite set of transitions, $I \subseteq P \times T$ is a set of transition input arcs, $O \subseteq T \times P$ is a set of transition output arcs, and $m_0 : P \to Z^{|P|}$ is the initial marking of the PN with Z as the set of nonnegative integers. A marking of G is a vector $m \in Z^{|P|}$ that indicates the number of tokens in each place under a state. ${}^{\bullet}t$ denotes the set of input places of transition t. A transition t is enabled and can be fired under m iff $m(p) \geq I(p,t) \quad \forall p \in {}^{\bullet}t$. Firing a transition removes one token from each of its input places and adds one token to each of its output places. A marking m' is reachable from m if there exists a firing sequence s bringing m to m'. The reachability set $R(m_0)$ denotes all the markings reachable from m_0. A Petri net $G = (P, T, I, O, m_0)$ is live if, no matter what marking has been reached from m_0, it is possible to ultimately fire any transition of G by progressing through some further firing sequence.

To model a task workflow in Petri net, we use a place to represent a state in the workflow while a transition to represent an event or operation that brings the workflow from one state to another one. The workflow of a task w_n is modeled by an acyclic sequential marked graph $W_n = (P_n, T_n, I_n, O_n, m_{n0})$, where the set P_n of places denotes the production states whereas the set T_n of transitions denotes the operations.

Definition 3.1: The workflow of w_n is an acyclic sequential marked graph $W_n = (P_n, T_n, I_n, O_n, m_{n0})$. As each transition represents a distinct operation in a task, $T_j \cap T_k = \Phi$ for $j \neq k$. Each W_n has a final transition t_n^f whose firing terminating the task.

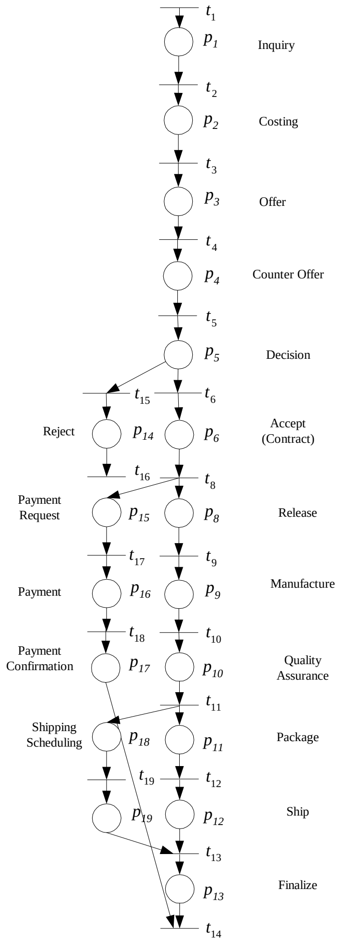

Fig. 4 A typical task force workflow

A workflow only specifies the sequence of operations to complete a task. Each operation in a workflow consumes a number of different types of resources. In task force automation systems, the set of resources includes sale staffs, accountants, operators, etc. In addition to the workflow specified by W_n , the activities for each type of resources must also be specified. Each resource may take part in some of the operations in different workflows. A resource may visit different states throughout the lifetime of its usage in performing various operations in the workflows. For example, a sale staff may be busy with a number of orders. The sale staff will return to idle state after the tasks are finished. For another example, a shipping operator may be in either busy state or idle state. Depending on the location, the shipping operator may be in different busy states. A machine may be either in idle state or in busy state. A quality assurance operator may also be in different states depending on the time, schedule, location and the products to be inspected. To facilitate monitoring and control of resource allocation, a model is required to capture the activities of each type of resources.

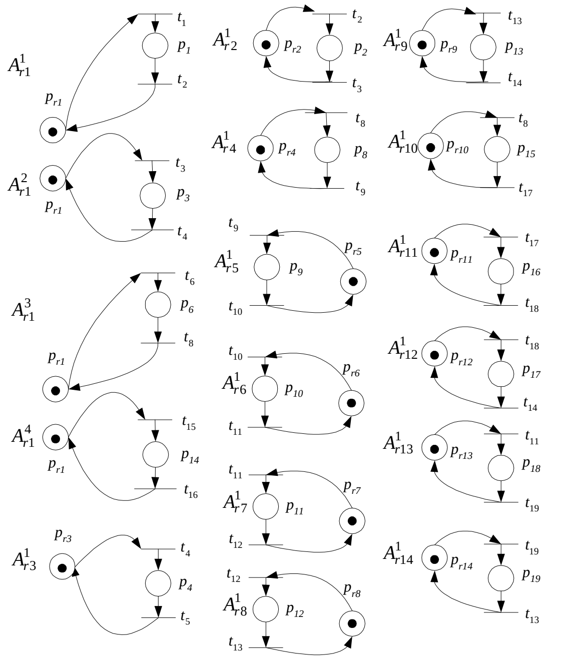

Fig. 5 Activity Petri net models

An activity is a sequence of operations to be performed and states to be visited by a certain type of resources. We use a place in Petri net to represent a state in the resource activity. Each resource has an idle state. Each resource activity starts and ends with an idle state. For example, consider the resource activity $A_{r1}^1 = p_{r1}t_1p_1t_2$ for a sale staff in Fig. 5. Place p_{r1} denotes the idle state of a sale staff. Firing transition t_1 moves the sale staff from the reception of an order inquiry request to the inquiry state. Resource activity $A_{r2}^1 = p_{r2}t_2p_2t_3$ in Fig. 5 represents the activity of an accountant, where place p_{r2} denotes the idle state of an account. Firing transition t_2 moves the accountant from the inquiry state to the costing state. A resource activity is described by a circuit in Petri net. A circuit indicates that the resource activity includes resource allocation and de-allocation. Let $\boldsymbol{R}_n$ denote the set of resource types required to perform the operations in W_n . For the workflow W_1 in Fig. 4,

$\boldsymbol{R}_1 = \{r_1, r_2, r_3, r_4, ..., r_{12}, r_{13}, r_{14}\}$. Fig. 5 shows the resource activities corresponding to the workflow in Fig.4.

The Petri net model for the $k-th$ activity for a type-r resources, where $r \in \boldsymbol{R}_n$, is described by a Petri net A_r^k defined as follows.

Definition 3.2: Petri net $A_r^k = (P_r^k, T_r^k, I_r^k, O_r^k, m_r^k)$ denotes the $k-th$ activity for a type-r resources, where $r \in \boldsymbol{R}_n$. Remark that $T_r^k \cap T_r^{k'} = \Phi$ for $k \neq k'$.

Let K_r be the number of activities of a type-r resources. Let $\Omega_n^r \subseteq \{1,2,3,..., K_r\}$ denote the set of type-r activity IDs in W_n. The initial marking m_r^k is determined based on the set of resource tokens allocated to the $k-th$ activity. More specifically, $m_r^k(p_r)$ is the number of resources allocated to place p_r, where p_r is the idle place of type-r resources.

$$A_{r1} = A_{r1}^1 \| A_{r1}^2 \| A_{r1}^3$$

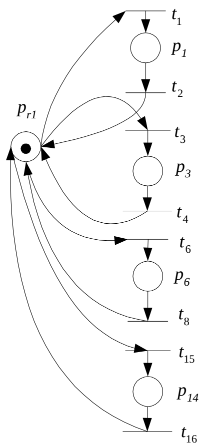

Fig. 6 A Petri net model for resource $r1$

4. Workflow Monitor Model Composition

The complete Petri net model of a process is constructed by combining the resource activity models with the workflow model to take into account the interactions between resources and the workflow. To combine resource activity models with the workflow model, we define the operator "$\|$" as follows to merge two Petri net models with common transitions, places and/or arcs. The operator "$\|$" can be easily implemented based on the incidence matrices of the Petri nets. By applying the "$\|$" operator, resources involved in an operation are synchronized so that the operation can be executed.

Definition 4.1: Given two Petri nets $N_1 = (P_1, T_1, I_1, O_1, m_{10})$ and $N_2 = (P_2, T_2, I_2, O_2, m_{20})$, $N_1 \| N_2 = (P, T, I, O, m_0)$, where $P = P_1 \cup P_2$, $T = T_1 \cup T_2$, $I(p,t) = \begin{cases} I_1(p,t) \text{ if } p \in P_1 \text{ and } t \in T_1 \\ I_2(p,t) \text{ if } p \in P_2 \text{ and } t \in T_2 \end{cases}$, $O(p,t) = \begin{cases} O_1(p,t) \text{ if } p \in P_1 \text{ and } t \in T_1 \\ O_2(p,t) \text{ if } p \in P_2 \text{ and } t \in T_2 \end{cases}$, and $m_0(p) = \begin{cases} m_{10}(p) \text{ if } p \in P_1 \\ m_{20}(p) \text{ if } p \in P_2 \end{cases}$.

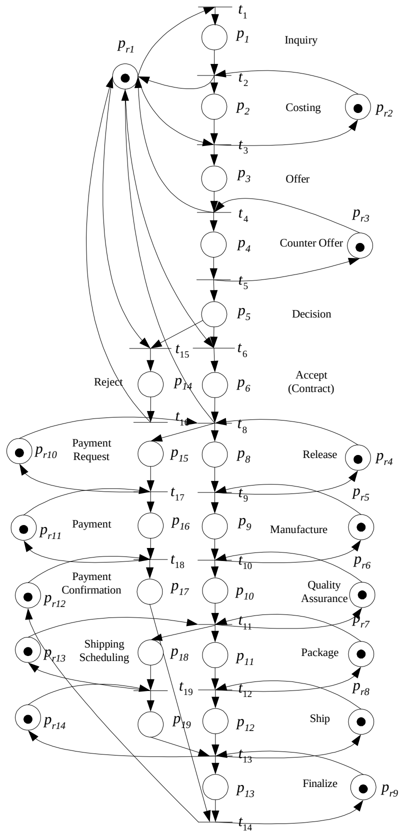

Fig.7 Workflow monitor Petri net

Definition 4.2: The complete Petri net model to process workflow W_n is modeled by $G_n = \|_{r \in R_n} A_r \| W_n$, where $A_r = \|_{k \in \Omega_n^r} A_r^k$.

Based on the composition operation defined in Definition 4.1 and Definition 4.2, we may construct the complete Petri net model. Fig.7 shows $G_n =$

$W_n \| A_{r1}^1 \| A_{r1}^2 \| A_{r1}^3 \| A_{r1}^4 \| A_{r2}^1 \| A_{r3}^1 \| A_{r4}^1 \| A_{r5}^1 \| A_{r6}^1 \| A_{r7}^1 \| A_{r8}^1 \| A_{r9}^1 \| A_{r10}^1$
$\| A_{r11}^1 \| A_{r12}^1 \| A_{r13}^1 \| A_{r14}^1$

With the complete Petri net model, the state of the system can be captured, represented and predicted conveniently. For example, we may apply the reachability tree method to predict the dynamic behaviors of the workflow system.

Given $G = (P, T, I, O, m_0)$ with an initial marking m_0, the reachability graph can be generated with the following algorithm.

Reachability Graph Construction Algorithm $RGC(m)$

Input: G, m

Output: $RG = (N, A)$

Step 0: $N \leftarrow \Phi$
        $A \leftarrow \Phi$
        Find the incidence matrix K of G
Step 1: If $m \notin N$
            $N \leftarrow N \cup \{m\}$
            For each $t_j \in T_n$
                If $m'(i) = K(i, j) + m(i) \geq 0 \forall i$
                    Create arc $a(m, m')$
                    $A \leftarrow A \cup \{a(m, m')\}$
                    $RGC(m')$
                End If
            End For
        End If

Generally, the reachability graph may either contain safe markings as well as unsafe markings. Evolution to unsafe markings is undesirable. Therefore, we must identify the subgraph of a reachability graph that contains safe markings only. Construction of the safe subgraph of a reachability graph can be efficiently achieved by applying the strongly connected component from the reachability graph.

5. Conclusion

In this paper, we focus on application of context-aware computing technology in task force automation. Consider the task force assignment problem that takes place from time to time in a company. The task force automation process is driven by the arrival of a potential order. Depending on the requirements of the order, a collaborative network needs to be formed to determine whether the order should be committed. Although collaborative network formation has been studied in literature, how to achieve effective collaborative network formation by taking the advantage of context-aware computing technologies is an important issue. We propose formal Petri nets for modeling individual contexts of the collaborative networks in a context-aware computing environment. Based on composition of the individual context models, workflows can be monitored. Our future research directions include theoretical development of supervisory control algorithms for the class of Petri nets proposed in this paper, implementation of the model composition mechanism and prototype design based on our proposed system architecture to facilitate the generation of user interface for task force automation applications.

6. Acknowledgement

This paper is supported in part by National Science Council of Taiwan, R.O.C. under grant NSC95-2416-H-324-009-MY2

7. References

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [1]. Where appropriate, include the name(s) of editors of referenced books.

[1] K. Cheverst et al., Experiences of developing and deploying a context-aware tourist guide: The GUIDE project, in: 6th International Conference on Mobile Computing and Networking, Boston, pp. 20–31, August 2000.

[2] S.N. Patel, G.D. Abowd, The ContextCam: Automated point of capture video annotation, in: 6th International Conference on Ubiquitous Computing, in: Lecture Notes in Computer Science, vol. 3205, Springer, pp. 301–318, 2004.

[3] Y. Shi et al., The smart classroom: Merging technologies for seamless tele-education, IEEE Pervasive Computing 2 (2), pp. 47–55, 2003.

[4] H. Chen et al., Intelligent agents meet semantic web in a smart meeting room, in: 3rd International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 854–861, July 2004.

[5] S. Helal et al., Enabling location-aware pervasive computing applications for the elderly, in: 1st IEEE Conference on Pervasive Computing and Communications, Fort Worth, March 2003.

[6] F. D. Davis and V. Venkatesh, "Measuring user acceptance of emerging information technologies: An assessment of possible method biases," in Proc. 28th Hawaii Int. Conf. System Sciences, pp. 729–736, 1995.

[7] B. N. Schilit and M. M. Theimer, "Disseminating active map information to mobile hosts," IEEE Network, vol. 8, no. 5, pp. 22–32, 1994.

[8] F. Croci, M. Perona and A. Pozzetti, "Work-force management in automated assembly systems," International Journal of Production Economics, V. 64, Issues 1-3, pp. 243-255, 2000.

[9] Murata T., "Petri Nets: Properties, Analysis and Applications," Proceedings of the IEEE, vol. 77, No. 4, pp. 541-580, 1989.

[10] Michael Weber1, and Ekkart Kindler The Petri Net Markup Language, http://www2.informatik.hu-berlin.de/top/pnml/download/about/ PNML_LNCS.pdf, 2002.