IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITB12594

This paper appears in the book, *Emerging Trends and Challenges in Information Technology Management, Volume 1 and Volume 2* edited by Mehdi Khosrow-Pour © 2006, Idea Group Inc.

QoS Driven Dynamic Task Assignment for BPM Systems Using Fuzzy Logic

Cagil Tasdemir and Candemir Toklu, Siemens Corporate Research (SCR), 755 College Rd East, Princeton NJ 08540, USA,
Cagil.Tasdemir@scr.siemens.com, T: 609 734 6500, F: 609 734 6565, Candemir.Toklu@siemens.com

ABSTRACT
Adoption of standards-based business process management systems is picking up. With this trend the need for novel techniques for process optimization is growing. One such technique is managing task assignment to resources at runtime. In this paper, a state-of-the-art task assignment algorithm that considers worker skills, task relationships, worker relationships and worker workload has been extended in two different manners: First, the historical evaluation of the workload of each worker has been added; and second, the quality of service (QoS) criteria has been introduced to be evaluated as the worker skills and workload are done. In addition, the task assignment solution has been designed and implemented on a business process management system.

1 INTRODUCTION
Business process management systems (BPMS) include methods, techniques, and tools to support the design, deployment, execution, run-time optimization and analysis of operational business processes. They can be considered as extension of classical workflow management systems and approaches. While workflow management focuses on managing the control flow of activities, it needs to integrate other technologies to fully manage the lifecycle of a business process. These other technologies may include:

- re-usability of process components for aggregation,
- mobile behavior for better managing process participants [4, 6, 7, 13].

Therefore, these emerging systems are built on a new computing foundation based on process models and pi-calculus to describe, simulate, deploy, execute and optimize the whole processes. With these technologies, the gap between design and execution is narrowing. This is realized by a standards-based approach for describing processes and enabling to go from process design to execution with little if any software development in between. For example, the executable processes are specified using Business Process Execution Language for Web Services, BPEL in short, which is a standardized XML syntax and vocabulary for defining executable processes based on web service orchestration [8]. Another standard is Business Process Modeling Notation (BPMN) [9]. Its aim is to provide a more user-friendly way to describe the process models that can be understood by other business users. It also provides a mapping between the graphics of the notation to the underlying constructs of execution languages, particularly BPEL. Each BPEL process is actually a new service which provides operations through its port types and can be invoked using the resulting composite web service. BPEL is a very powerful language to specify the order of the tasks, either in a sequential or parallel way, express conditional behavior, and define fault handlers, etc. [3]

Shen et al. proposes a task assignment algorithm which mainly relies on fuzzy logic [1]. Fuzzy logic resembles human reasoning in its use of imprecise information to generate decisions. While classical logic requires a deep understanding of a system, fuzzy logic incorporates an alternative way of thinking, which allows modeling complex systems using a higher level of abstraction originating from our knowledge and experience. Fuzzy logic is a good approach to model role resolution since the initial information about the workers and tasks are more suitable to be described in linguistic variables rather than variables which have exact values [5, 10]. For membership function, arithmetic operations, and defuzzification of triangular fuzzy numbers, which have been used in our study, please refer to Shen et al.

In this paper, we concentrate on the techniques and tools for process optimization, in particular the resource assignment to user tasks in business processes in light of the past performance of people for similar tasks. We also attempt to realize our solution within a BPM system using the BPEL standard for process representation. Therefore, the dynamic task assignment algorithm is implemented as a web service, and is integrated into a business process management system as an analytical tool for optimizing the process execution. Since the management of QoS metrics directly impacts the success of enterprises participating in BPMS, the user task assignments are optimized by choosing the most appropriate assignee for the current task at runtime by taking into account the quality of service. The matching between the tasks and the workers is established by applying fuzzy logic concept, which has been used in calculating the worker capabilities, task requirements, worker relationships and workload to these QoS criteria. According to Cardoso, workflow QoS represents the quantitative and qualitative characteristics of a workflow application necessary to achieve a set of initial requirements [2]. We consider only quantitative QoS characteristics which can be evaluated in terms of concrete measures such as time, cost, etc. Initially given the required QoS criteria for the tasks and the workers, the worker who meets the QoS criteria for a given task better than the other workers is chosen.

The rest of the paper is organized as follows: Section 2 presents Shen et al.'s algorithm and our contribution. The basic architecture is covered in Section 3 and a numerical example is demonstrated in Section 4. The conclusions are made in Section 5.

2 TASK ASSIGNMENT ALGORITHM

2.1 Shen et al.'s Algorithm
In [1], first, the relationships between the skill set of a worker and the skills required for a task are considered. Second, social relationships among workers are ranked, and finally, the relationships among tasks are taken into account. The key points of the task assignment algorithm are mentioned below.

2.1.1 Capabilities
Let $\tilde{e}(U_i, C_t)$ be the assessment score of worker U_i on skill C_t, and let $\tilde{w}(J, C_t)$ be the weighting of C_t. For task *J*, considering it requires k skills (C_t, t=1,2,...,k), the total suitability of capability of U_i is evaluated by the equation:

$$\tilde{E}_{CAP}(U_i) = \frac{1}{k} \otimes \sum_{t=1}^{k} [\tilde{e}(U_i, C_t) \otimes \tilde{w}(J, C_t)] \tag{1}$$

$\tilde{w}(J, C_t)$ is a linguistic variable with five possible values: not important, barely important, moderately important, very important and extremely important. The corresponding triangular fuzzy numbers are (0, 0, 0.1), (0, 0.3, 0.5), (0.3, 0.5, 0.7), (0.5, 0.7, 0.9), and (0.7, 0.9, 1) respectively.

$\tilde{e}(U_i, C_t)$ is a linguistic variable with five possible values: no, low, normal, good, and expert. The corresponding triangular fuzzy numbers are (0, 0, 0), (0, 0.2, 0.4), (0.2, 0.4, 0.6), (0.4, 0.6, 0.8), and (0.6, 0.8, 1) respectively.

2.1.2 Social Relationships

Let $\tilde{E}_{IND}(U_i)$ denote the suitability of a worker for performing task *J*. If *J* must be performed by a team of workers, the suitability of a candidate team is calculated by the equation:

$$\tilde{E}_{CAND} = \sum_i^m \tilde{E}_{IND}(U_i) \tag{2}$$

In a team, the relationships between every pair of workers are evaluated and then totaled. The total score of social relationships for a team is obtained by the equation:

$$\tilde{E}_{RL} = \frac{1}{C_2^f} \otimes \sum_{p,q} \tilde{e}_{RL}(Up, Uq) \tag{3}$$

where $C_2^f = f(f-1)/2$, and *f* denotes the number of team members and $\tilde{e}_{RL}(Up, Uq)$ is the evaluation score of social relationships among team members. The possible values for $\tilde{e}_{RL}(Up, Uq)$ are worst, poor, fair, good, and best. The corresponding triangular fuzzy numbers are (0, 0, 0), (0, 0.3, 0.5), (0.3, 0.5, 0.7), (0.5, 0.7, 0.9), and (0.7, 0.9, 1) respectively. The suitability score of a candidate team is assessed using the equation:

$$\tilde{E}_{TEAM} = \tilde{E}_{CAND} \oplus \tilde{E}_{RL} \tag{4}$$

2.1.3 Task Relationships

Let *J* be the task for which the appropriate assignee is being searched. Eq.5 assesses the similarity between *J*and the *j*th task in the worklist of worker U_i with regard to skill C_t. Eq.6 obtains the similarity between J_{ik} and*J*, and Eq.7 gives the total similarity between *J* and all tasks already assigned to worker U_i where *n* is the number of tasks in the worklist of U_i.

$$s(J_{ij}, J, C_t) = \frac{Area[\tilde{w}(J_{ij}, C_t) \cap \tilde{w}(J, C_t)]}{Area[\tilde{w}(J_{ij}, C_t) \cup \tilde{w}(J, C_t)]} \tag{5}$$

$$S(J_{ij}, J) = \frac{1}{k} \sum_{t=1}^k s(J_{ij}, J, C_t) \tag{6}$$

$$\tilde{E}_{SIM}(U_i) = \frac{1}{n} \sum_{j=1}^n S(J_{ij}, J) \tag{7}$$

2.1.4 Worker Workload

To prevent abler workers from being assigned more heavily than those who are less capable, workload should also be considered. By doing so, the fair assignment of tasks is ensured. The total of work hours remaining for all previously assigned tasks is the criterion to measure worker workload. Work hours are represented by fuzzy numbers.

If worker U_i needs T_{ij} hours to finish the *j*th task on his/her worklist, then his/her workload is $\sum_j T_{ij}$, and the worker score for workload is calculated using the following equation:

$$\tilde{E}_{LOAD}(U_i) = (\sum_{i,j} \tilde{T}_{ij} - \sum_j \tilde{T}_{ij}) / (\sum_i (\sum_{i,j} \tilde{T}_{ij} - \sum_j \tilde{T}_{ij})). \tag{8}$$

2.2 Contribution

2.2.1 Worker Workload

In [1], the workload for each worker is calculated using Eq.8. Let's say worker U_1 has 2 hours remaining work while worker U_2 has 4 hours remaining work. Then the workload scores will be calculated as:

$$\tilde{E}_{LOAD}(U_1) = (6-2)/[(6-2)+(6-4)] = 0.67$$

$$\tilde{E}_{LOAD}(U_2) = (6-4)/[(6-2)+(6-4)] = 0.33$$

To obtain the worker suitability scores, we propose to consider the proportion of the finished hours of work of a worker to the total work hours of the same worker in a time period which has been previously defined (i.e. week, month, etc.). If worker U_1 is supposed to work S_{ij} hours in a pre-defined period to finish the *j*th task on his/her worklist, then his/her workload sum is $\sum_j S_{ij}$. If worker U_1 has already worked F_{ij} hours for the task *j* in this period, then his/her finished workload is $\sum_j F_{ij}$. Then the workload score for each worker can be calculated as:

$$\tilde{E}_{LOAD}(U_i) = \frac{(\sum_{i,j} \frac{\tilde{F}_{ij}}{\tilde{S}_{ij}} - \sum_j \frac{\tilde{F}_{ij}}{\tilde{S}_{ij}})}{(\sum_i (\sum_{i,j} \frac{\tilde{F}_{ij}}{\tilde{S}_{ij}} - \sum_j \frac{\tilde{F}_{ij}}{\tilde{S}_{ij}}))} \tag{9}$$

If we say worker U_1 worked 30 hours this week while worker U_2 worked 20 hours, and worker U_1 has 2 hours remaining work while worker U_2 has 4 hours remaining work; then the new workload scores for each worker will be:

$$\tilde{E}_{LOAD}(U_1) = \frac{(30/32 + 20/24 - 30/32)}{[(30/32 + 20/24 - 30/32) + (30/32 + 20/24 - 20/24)]} = 0.47$$

$$\tilde{E}_{LOAD}(U_1) = \frac{(30/32 + 20/24 - 20/24)}{[(30/32 + 20/24 - 20/24) + (30/32 + 20/24 - 30/32)]} = 0.53$$

Note that worker U_1 obtains a lower suitability score when the finished work hours are taken into account.

2.2.2 Quality of Service Criteria

In [1], the skills are considered as both tasks' requirements and workers' capabilities. As an example, "programming" is a skill which is required by task "system modification" at a level of "extremely important", and also it is a capability of worker U_1 with a rating of "normal".

Our quality of service criteria consist of time, cost and reliability. We consider time and reliability as requirements for tasks, and cost as a property of worker. Therefore, we calculate time and reliability ratings of workers as skills, and cost scores as workload. Time and reliability criteria are easier to describe using linguistic variables, whereas cost can be represented using precise numerical quantities.

2.2.2.1 Time and Reliability

Time is a common measure of performance. In our study, from the "task" point of view, it means how important it is to finish the tasks on time. From the "worker" point of view, it means how successful the worker is to finish his/her work on time. Our second criterion is the reliability. In [12], reliability is defined as the ability of a system or component to perform its required functions under stated conditions for a specified period of time. From the task point of view, as a requirement, it means how important it is to be completed without failure. From the worker point of view, it means how successful the worker is in finishing his/her work without failure.

Each task's time and reliability QoS requirements and each worker's ratings on these requirements are assigned by the business designer. We consider the time and reliability as fuzzy variables. The calculation of the capability assessment scores can be evaluated as:

$$\tilde{E}_{CAP}(U_i)=[\frac{1}{k}\otimes\sum_{t=1}^{k}[\tilde{e}(U_i,C_t)\otimes\tilde{w}(J,C_t)]]\oplus[\frac{1}{2}\otimes\sum_{t=1}^{2}[\tilde{e}(U_i,QoS_t)\otimes\tilde{w}(J,QoS_t)]] \quad (10)$$

where QoS_t is the QoS requirement for task *J*. The task similarity is calculated using the equations:

$$s(J_{ij},J,C_t)=\frac{Area[\tilde{w}(J_{ij},C_t)\cap\tilde{w}(J,C_t)]}{Area[\tilde{w}(J_{ij},C_t)\cup\tilde{w}(J,C_t)]}$$

$$s(J_{ij},J,QoS_t)=\frac{Area[\tilde{w}(J_{ij},QoS_t)\cap\tilde{w}(J,QoS_t)]}{Area[\tilde{w}(J_{ij},QoS_t)\cup\tilde{w}(J,QoS_t)]} \quad (11)$$

$$S(J_{ij},J)=\frac{1}{k}\sum_{t=1}^{k}s(J_{ij},J,C_t)+\frac{1}{2}\sum_{t=1}^{2}s(J_{ij},J,QoS_t) \quad (12)$$

$$\tilde{E}_{SIM}(U_i)=\frac{1}{n}\sum_{j=1}^{n}S(J_{ij},J)$$

Here *J* is the task for which we are searching the appropriate assignee. Eq.11 assesses the similarity between *J* and the *j*th task in the worklist of worker U_i with regard to QoS criteria QoS_t. Eq.12 obtains the similarity between J_{ij} and *J*. Eq.7 is used to calculate the total similarity between *J* and all tasks already assigned to worker U_i where *n* is the number of tasks in the worklist of U_i.

2.2.2.2 Cost

The last criterion for the QoS is cost. It represents the cost associated with the workers. It can also be considered as a fuzzy number. We obtain the worker costs using the algorithm which also calculates the worker workloads. Therefore, the worker with the lower cost obtains a higher suitability score. The worker cost is calculated using the following equation where Co_i is the cost of worker U_j per unit time:

$$\tilde{E}_{COST}(U_i)=(\sum_i Co_i-Co_i)/(\sum_i(\sum_i Co_i-Co_i)) \quad (13)$$

2.2.3 Evaluation

The evaluation procedure is as follows:

Step 1: List candidate workers, and eliminate workers already performing conflicting tasks.

Step 2: Evaluate the capability $\tilde{E}_{CAP}(U_i)$ for each candidate worker U_i considering both skills and time and reliability QoS criteria.

Step 3: Evaluate the task similarity $\tilde{E}_{SIM}(U_i)$, workload $\tilde{E}_{LOAD}(U_i)$ (considering both remaining and finished work hours), and cost $\tilde{E}_{COST}(U_i)$ for each candidate worker U_i.

Step 4: The suitability score of each candidate worker U_i is $\tilde{E}_{IND}(U_i)=\tilde{E}_{CAP}(U_i)\oplus\tilde{E}_{SIM}(U_i)\oplus\tilde{E}_{LOAD}(U_i)\oplus\tilde{E}_{COST}(U_i)$.

Step 5: Group the workers into candidate teams. The suitability score of a candidate team is $\tilde{E}_{CAND}(U_i)=\sum_i^m\tilde{E}_{IND}(U_i)$, where m denotes the number of workers required.

Step 6: Evaluate social relationships $\tilde{E}_{RL}$ for each candidate team.

Step 7: The suitability score of a candidate team is $\tilde{E}_{TEAM}=\tilde{E}_{CAND}\oplus\tilde{E}_{RL}$.

Step 8: Rank candidate teams according to their $\tilde{E}_{TEAM}$.

3 ARCHITECTURE

The basic architecture is depicted in Figure 1. A process involving human tasks is designed using the business process designer. Then it is deployed into the process engine. After the process is initiated, the tasks appear in the worklists of the assignees, based on the order defined in the process map. The appropriate assignee is chosen via the calls to the web service which implements the task assignment algorithm. When a user completes a task, it sends a notification to the process engine using the interface supplied for him/her. The logs of the user tasks are stored in the database so that the business analyst can offline analyze them and change the worker capabilities and task requirements if necessary. The web service also uses these logs for the assignment algorithm.

3.1 Business Process Designer

This step consists of the creation of the business process using a BPEL designer and custom developed utilities for transforming the process. We are using XSLT (Extensible Stylesheet Language Transformations) to insert the calls to the task assignment web service as an activity right before each user task in the original business process.

We use Oracle BPEL Designer, which enables BPEL process development using an intuitive graphical editor instead of writing BPEL code by hand. Oracle BPEL Designer also allows the automatic deployment of BPEL processes.

Figure 1. Basic architecture

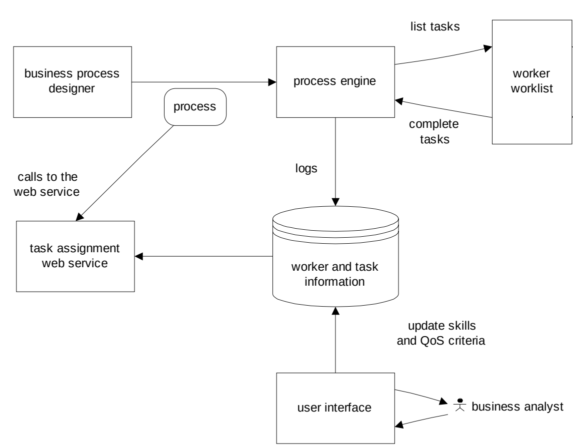


Figure 2.a. Initial process map

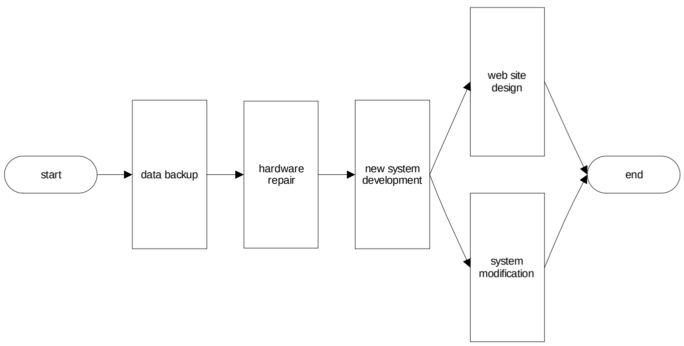
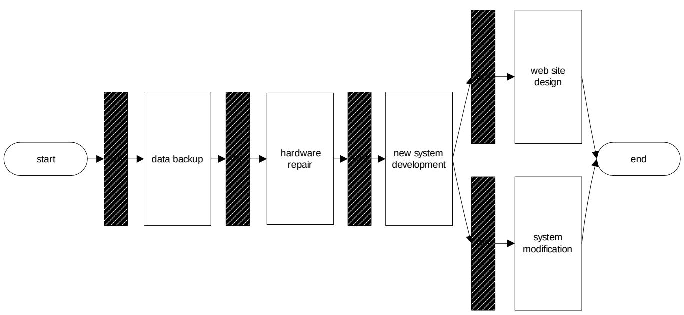


Figure 2.b. Final process map



Our example process is shown in Figure 2.a. The tasks are adopted from Shen et al. The process starts with data backup task. After data backup is completed, hardware repair and new system development tasks take place respectively. Finally, web site design and system modification tasks run in parallel. The process ends after the completion of these two tasks. The invocations for the web service are placed before each task assignment as shown in Figure 2.b where the shaded boxes denote the web service calls.

3.2 Process Engine and Worker Worklist

After the process is designed and deployed, it is executed on the process engine. Our process engine is Oracle BPEL Process Manager. Oracle BPEL Process Manager seems to be one of the most promising BPEL servers. In addition to deploying and running BPEL processes, it offers advanced functionality that makes it one of the most powerful BPEL servers. One of the most important features of Oracle BPEL Process Manager is that the user tasks are integrated into business processes. Oracle offers a built-in task manager service which is an asynchronous service with two interfaces. The first interface is a WSDL interface used by the BPEL process. During the execution of a BPEL process, task manager service is simply invoked as a web service. After the user task is completed, the task manager performs a callback operation to the BPEL process. The second interface is the client API, which provides developers with the functionality of handling user interaction. One can list, display, complete tasks using API functions [3, 11].

3.3 Audit Trails

The process logs are stored in the database. We use MSSQL Server 2000 to store the data.

3.4 Task Assignment Decision Web Service and User Interface

We implemented the task assignment algorithm as a web service. We also implemented an administrative interface which allows users to insert/update the initial scores of the worker capabilities, task requirements, and worker relationships. The web service and the interface are both implemented using C#.NET.

4 NUMERICAL RESULTS

We use Shen et al.'s example with some additional information about QoS criteria [1]. We will demonstrate the assignment for the task system modification. The task requirements, worker capability ratings and social relationship ratings are shown in Table 1.a.

We consider that there may be more than one instance of the same process, so current job allocation and the workload are assumed to appear as follows:

U_1= {hardware repair, data backup}, remaining work $=\tilde{5}$, finished work $=\tilde{40}$;

U_2= {web site design}, remaining work $=\tilde{2}$, finished work $=\tilde{65}$;

U_3= {new system development}, remaining work $=\tilde{20}$, finished work $=\tilde{2}$;

U_4= {new system development, web site design}, remaining work $=\tilde{22}$, finished work $=\tilde{20}$;

U_5= {new system development}, remaining work $=\tilde{15}$, finished work $=\tilde{10}$.

The costs per unit time for each worker are as follows:

$U_1=\tilde{20}$, $U_2=\tilde{25}$, $U_3=\tilde{10}$, $U_4=\tilde{10}$, $U_5=\tilde{20}$.

Table 1.a. Skill and QoS requirements for tasks

Criteria	System modification	New system development	Data backup	Hardware repair	Web site design
programming	e imp	e imp	b imp	not	v imp
system analysis	m imp	e imp	not	not	m imp
hardware	b imp	v imp	v imp	e imp	not
network	not	not	b imp	v imp	v imp
art design	not	not	not	not	v imp
coordination	m imp	e imp	not	not	m imp
leadership	not	v imp	not	not	not
time	e imp	v imp	b imp	m imp	m imp
reliability	v imp	e imp	v imp	m imp	v imp

Table 1.b. Capability ratings for workers

Skills	U_1	U_2	U_3	U_4	U_5
programming	normal	normal	expert	expert	expert
system analysis	low	low	good	expert	normal
hardware	expert	low	normal	good	low
network	normal	normal	good	normal	low
art design	no	expert	no	low	good
coordination	no	low	normal	good	good
leadership	low	no	normal	good	normal
time	normal	low	good	expert	expert
reliability	expert	normal	normal	good	expert

Table 1.c. Social relationship ratings among workers

	U_1	U_2	U_3	U_4	U_5
U_1	-	worst	poor	fair	good
U_2	worst	-	good	best	best
U_3	poor	good	-	good	worst
U_4	fair	best	good	-	poor
U_5	good	best	worst	poor	-

Table 2. Resultant individual suitability scores

Score	U_1	U_2	U_3	U_4	U_5
$\tilde{E}_{CAP}(U_i)$	0.180	0.120	0291	0.269	0.269
$\tilde{E}_{LOAD}(U_i)$	0.172	0.164	0.242	0.207	0.215
$\tilde{E}_{COST}(U_i)$	0.191	0.176	0.221	0.221	0.191
$\tilde{E}_{SIM}(U_i)$	0.220	0.178	0.141	0.320	0.141
$\tilde{E}_{IND}(U_i)$	0.763	0.638	0.895	1.017	0.816

Table 3. Team scores

Candidate Team	$\tilde{E}_{TEAM}$
$\{U_3, U_4\}$	3.391
$\{U_2, U_4\}$	3.366
$\{U_1, U_4\}$	3.183
$\{U_1, U_5\}$	2.893
$\{U_4, U_5\}$	2.880
$\{U_2, U_5\}$	2.876
$\{U_2, U_3\}$	2.787
$\{U_1, U_3\}$	2.537
$\{U_3, U_5\}$	2.201
$\{U_1, U_2\}$	2.079

According to these data, the suitability scores for each worker are calculated.

Then, candidate teams are ranked by $\tilde{E}_{TEAM}$.

After the evaluation step, the most appropriate team for the system modification task consists of workers U_3 and U_4 as shown in Table 3. However, workers U_2 and U_4 would be chosen for the same task without the consideration of the QoS criteria and finished work hours. These results indicate that the QoS criteria should be considered during task assignment as well as worker abilities, social relationships, and task relationships; and the finished work hours of the workers are as important as the remaining work hours.

5 CONCLUSION

In this paper, we propose novel extensions to the role resolution algorithm described in [1] and address how a BPM system can utilize this algorithm for process optimization. We have also shown with a numerical example how the algorithm will perform assignment for a task in a sample process. The improved algorithm with the inclusion of QoS criteria and historical workload information effectively found the most appropriate candidate team for the task in consideration.

In order to demonstrate the feasibility of this idea within a BPM system we have designed a software environment with modular components – the process designer, process engine, web service based realization of our algorithm, and data store. All the components we have used are of the shelf tools and can be easily exchanged with other similar tools.

We are currently developing algorithms for feeding the historical process performance data back into the task assignment algorithm. Later, we will create randomly generated data sets for testing the algorithms and check the performance and the benefit of the feedback.

REFERENCES

1. Minxin Shen, Gwo-Hshiung Tzeng, and Duen-Ren Liu, "Multi-Criteria Task Assignment in Workflow Management Systems", Proceedings of the 36th Hawaii International Conference on System Sciences, 2002.
2. Jorge Cardoso, Amit Sheth and John Miller, "Workflow Quality of Service", Technical report, LSDIS Lab, Computer Science, University of Georgia, Athens, 2002.
3. Matjaz B. Juric, "Business Process Execution Language for Web Services", Packt Publishing, pp.155-216, 2004.
4. Howard Smith and Peter Fingar, "Business Process Management: The Third Wave", Meghan-Kiffer Press, pp. 231-256, 2003.
5. E. H. Mamdani, B. R. Gaines, "Fuzzy Reasoning and Its Applications", Academic Press, pp.34-38 1981.
6. David McGoveran, "An Introduction to BPM and BPMS", Business Integration Journal, 2004.
7. "The Architecture Revolution: Exploring the Network Effect on Infrastructure, Applications and Business Process", The Leading Edge Forum, http://147.29.80.116/foundation/library/publicationdetail01.asp?ID=121
8. "Business Process Execution Language for Web Services Specification V1.1. OASIS Specification", 5 May 2003.
9. "Business Process Modeling Notation Specification", http://bpmi.org
10. Daniel D. Zeng, and J. Leon Zhao, "On Role Resolution Decisions in Workflow Systems", University of Minnesota IDSc Friday Workshop Discussion Paper, November 2002.
11. "Oracle BPEL Process Manager", http://www.oracle.com/technology/products/ias/bpel/index.html
12. "IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries", Institute of Electrical and Electronics Engineers, 1990.
13. "The Workflow Management Coalition", http://www.wfmc.org