

# Temporal Workflow Management in a Claim Handling System

**J. Leon Zhao**

School of Business and Management  
Hong Kong Univ. of Science & Technology  
Clear Water Bay, Kowloon, Hong Kong  
[zhao@ust.hk](mailto:zhao@ust.hk)

**Edward A. Stohr**

Stern School of Business  
New York University  
New York, NY 10012, USA  
[estohr@stern.nyu.edu](mailto:estohr@stern.nyu.edu)

## ABSTRACT

Temporal workflow management is important for processes that are time-driven. Claim handling, which requires the documentation, diagnosis, and resolution of customer claims due to faulty products or unsatisfactory services, is an example of such a process because fast turnaround is critical for customer satisfaction. However, little research has been reported in this area, especially at the policy level. In this paper, we develop a framework for temporal workflow management, which includes issues such as turnaround time predication, time allocation, and task prioritization. We propose also the use of reward functions to guide workers' behavior with the goal of increasing efficiency while allowing flexibility.

## 1. INTRODUCTION

A workflow system is a piece of software that helps coordinate and manage work processes (WFMC, 1994; Sheth, 1996). Workflow management refers to managerial activities either manual or automatic such as modeling the work processes, allocating tasks to roles that are assumed by workers, controlling the routing of tasks, monitoring work processes, and alerting workers and managers of special and exceptional events in the system (Bussler and Jablonski, 1994; Kappel, Rausch-Schott, and Retschitzegger, 1998; Kumar and Zhao, 1998; Stohr and Zhao, 1997).

Temporal workflow management refers to the aspects of workflow management that involve the time dimension (Chinn and Madey, 1997). Examples are the management of turnaround time of work, the allocation of time to various workflow steps, and the prioritization of tasks in the task queues in the workflow system. Although temporal workflow management has been recognized as an important area (McDermott and Mulvihill, 1996; Jasper, Zukunft, and Behrends, 1996; Haimowitz et al., 1996), most previous work has focused on transaction management issues. In this paper, we explore business level concepts and policy issues (Bussler, 1994).

The focus of our study is temporal workflow management in the context of claim handling. Claim handling occurs in most companies that sell products or provide services to their customers. It is a necessary business function because some products will inevitably fail and some customers will

not be satisfied with the services provided. We chose claim handling as the business context because it is a relatively complex process where the types of products/services provided by the company are numerous.

High quality claim handling is important for keeping customers satisfied and for maintaining a solid market base. Furthermore, claim handling also has important implications to manufacturing and engineering because it can reveal design and production problems in the company. Although claim handling has a special meaning in insurance companies (filing accident reports and authorizing compensation), in this paper, we use the term to refer to the handling of customer complaints in any type of company.

Currently, the state of arts in the management of claim handling is very rudimentary. Typically, claim handling processes are managed via first-come-first-serve queues. Some claims may be **stamped** as "Urgent" so that they are given high priority. In exceptional cases, the manager may **push** certain claims personally to get them through faster. There is not a more systematic way for the manager to control the claim handling workflows beyond "stamping" and "pushing" for special cases. Furthermore, the management has limited capability of predicting and controlling the turnaround times. With the increased level of workflow automation, it is possible to develop a solution for this problem. In this paper, we propose a research framework towards better management of temporal workflows.

Although temporal workflow management resembles job sequencing in scheduling, planning, and project management (Cao and Sanderson, 1995; Daniels and Mazzola, 1994; Homem de Mello and Sanderson, 1991; Olender and Osterweil, 1990; Rajan and Nof, 1996), the issues we address in this paper are unique due to the humanistic nature of workflow systems. In a workflow system, more uncertainty exists, and exact optimization is not possible because humans cannot be programmed like numerically controlled machines (Marshak, 1993; Berman, Larson, and Pinker, 1997).

Several research problems are tackled in this paper:

- (1) Currently, a typical claim handling system uses a workflow policy based on a first-come-first-serve (FCFS) protocol so that claims are processed mainly according to their arrival time. As a result, claims going through a simple process are completed faster than claims requiring more complex processes. As a result, some claims are done on the same day while

To appear in *Proceedings of the International Joint Conference on Work Activities Coordination and Collaboration*, February 22-25, 1999, San Francisco, California, USA.

others may take several weeks to resolve. Many customers may complain that their claims took too long to settle. To improve customer satisfaction, it is desirable to reduce the process time for those claims requiring more complex processes. This goal can be achieved through more advanced workflow policies as outlined below.

- (2) One approach for managing the turnaround time is to assign *expected processing time* (queue time plus work time) for each task, given an *expected turnaround time* (or sojourn time) for a claim. That is, given a turnaround time and the processing nodes, the total time is allocated to each node according to some algorithm. The tasks at a node will be prioritized according to some policy that depends on the expected processing times. This problem is referred to as *time allocation*.

Time allocation in claim handling is not a straightforward problem because the exact process for a claim is usually not known, but is instantiated during the workflow. Further, the process steps of a claim workflow are difficult to predict. Note that the *processing time* of a process step includes the queuing time and the work time, and the *turnaround time* is the sum of process times for all steps. The *queuing time* is the elapsed time from the moment the job is placed into the queue till the job is being taken by a worker, and the *work time* is the time taken to complete the job, assuming that work is always done continuously without preemption. Our objective is to adjust the queuing times of claims by appropriate workflow policies so that the average turnaround time for certain types of claims is minimized.

- (3) Reducing the queuing times for certain claims by rearranging the queue is referred to as *task prioritization*. We propose prioritization policies based on the time value of jobs in order to take into account factors other than merely the arrival time of tasks. Further, priorities of tasks may vary as the workflow processes are instantiated. That is, when the claim goes through the workflow system, the predicted number of process steps may need to be modified based on the workflow trace. At this point, the total suggested process time of the claim changes, as do the allocated times for the remaining steps. A number of task prioritization policies are discussed later in the paper.

Our goal in this paper is to develop a framework of temporal workflow management and to investigate the associated policies and algorithms for time allocation and task prioritization. This framework enables the reduction of weighted “tardiness” of workflow processes and the management of temporal workflow efficiency. Section 2 introduces the basic concepts of claim handling workflow, an example of the workflow model, and the temporal workflow management issues in the context of claim handling. Section 3 develops the algorithms and policies of temporal workflow management, including prediction of turnaround times, allocation of process times to tasks, and

analysis of various prioritization policies. Section 4 delineates the necessary components of a temporal workflow engine and the reward functions needed to accompany the temporal workflow policies. Finally, Section 5 concludes the paper and points out future research directions.

## 2. PROBLEM STATEMENT

### 2.1. A Claim Handling System

A claim handling system is usually designed to provide a single entry point for product returns and repairs, customer complaints, and related business disputes. The general claim handling process involves many possible steps such as:

- Documentation: Customer complaints received through various means, including telephones, emails, Web pages, or walk-ins, are registered and formally documented.
- Classification and dispatching: Complaints are then classified according to company rules and dispatched to various groups and departments within the company to resolve the problems.
- Diagnosis: Problems are then studied by various technical experts to ascertain the reasons, responsibilities, solutions, and relevant business implications of the claim.
- Repair or replacement: Product problems must be fixed through repair or replacement. Certain repairs are done within the claim handling workshop, and other repairs are done either at the customers’ sites for high value and heavy equipment or at the company’s premises.
- Testing and certification: Complex and mission-critical problems must be tested after repair, and high cost problems must be tested and certified by a special team.
- Settlement: The costs of repair and replacement must be debited to either the customer or the company according to the company regulation and the warrantee in effect.
- Dispute resolution: Many times, a dispute may arise as to who is at fault. In this case, a higher level of company authority or even lawyers may get involved to handle the dispute and to reach an agreement.
- Monitoring and control: All processes must be monitored and controlled by maintaining detailed records, alerting proper authorities when necessary, and following proper company rules and approval channels.

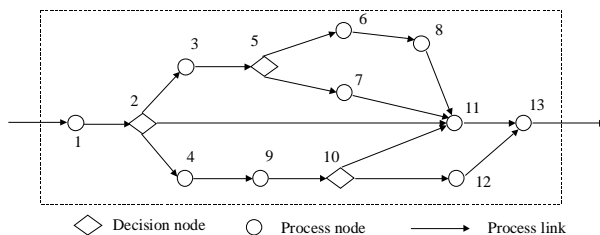
Note that different claim instances take different routes through the workflow system and require different workflow steps. For instance, certain problems may require a return and refund, and the resulting workflow is simple and fast. Others may require a replacement. This is a bit more complex than return and refund because some products may be out of stock and require a special production order; alternatively, they might be replaced with a slightly different product. Problems necessitating a repair involve more workflow steps. Simple repair work may be

done at the workshop of the claim handling department, but a more extensive repair may need to be sent back to a factory.

As a result of different workflow steps, all claims do not have the same turnaround times. Claims requiring an in-factory repair may need many days to complete. Often, customers become impatient with the time taken to complete the claim handling process. Furthermore, customers are disturbed when the receptionists cannot predict reasonably accurately how long the process might take. As mentioned previously, this is because the traditional workflow processes are usually based on a FCFS policy due to a lack of temporal workflow functions in a manual workflow system. Workflow automation has resulted in new opportunities for more complex time allocation and task prioritization policies as outlined below.

## 2.2. A Process Model for Claim Handling

Next, we give a process model for claim handling, which was extracted from analysis of a business case involving a large manufacturer in Hong Kong. A complete workflow model includes also information on data flow, organizational structure, and behavioral constraints. We omit information on these three aspects as they are not essential for our purposes in this paper. However, we use “process model” and “workflow model” interchangeably at times.



**Figure 1. A process model for claim handling.**

Figure 1 illustrates an example process model for claim handling, which includes three decision nodes and ten process nodes. In our definition, a *process node* is a point in the workflow where tasks are performed. A *decision node* is a special process node that leads to two or more branches. Although not explicitly shown, each node (either process node or decision node) contains a work list (or task queue) from which a worker can pick up tasks. The dashed box indicates the boundary of the workflow system.

The workflow tasks corresponding to the nodes in Figure 1 are listed below:

- (1) Reception desk
- (2) Work classification (repair, refund, or field work)
- (3) Inspection and diagnosis before repair
- (4) Field inspection
- (5) Deciding how to repair
- (6) Factory repair
- (7) Workshop repair
- (8) Factory inspection after repair
- (9) Field testing

- (10) Decision on payment
- (11) Accounting
- (12) Inspection by a manager
- (13) Final inspection and delivery

Needless to say, the process model in Figure 1 is a simplified one, and the real world processes can be much more complex. However, the given process model is sufficient for defining the various concepts and algorithms in the paper.

## 2.3. Temporal Workflow Management Issues

We assume that customer satisfaction depends on fast turnaround. In this paper, we concentrate on the reduction of turnaround time for claims requiring more complex processes. We also assume that a workflow system is already in place so that many aspects of claim handling are done electronically:

- A modeling tool is used to specify the workflow model that links all possible tasks in claim handling.
- Each worker has a work list, which contains a queue of tasks and their relevant documents.
- Workflow routing is done electronically based on the workflow model. That is, when a task is completed, the workflow system will forward the information on subsequent tasks automatically to the worklists following the current task.
- The workflow contains a number of decision nodes where decisions are made either by software agents or by human agents. This results in different paths for various workflow instances.
- The workflow processes are monitored and recorded automatically, and the workflow system has the capability of determining special cases and alerting the managers.

For a given set of system parameters (including the number of workers, their role assignments, and the average work time for each task), the turnaround times for claims may be managed in the following manner:

- (1) The expected turnaround time of a claim can be predicted based on system parameters such as the process model, the probabilities of branching at each decision node, the queue length at each node, and the mean work time for each task.
- (2) The expected process times for all tasks in the workflow can be allocated based on the expected turnaround time. Because the decision nodes in the process model can lead to different paths, the number of tasks in each workflow instance are unknown, and therefore, techniques need to be devised to allocate times in the presence of uncertainty.
- (3) Policies need to be developed for task prioritization for each work list. The prioritization may be done based on certain attributes of the tasks including the expected completion time, the customer value factor, the urgency value of the claim, and the potential complexity of the workflow process.

(4) Because workflow systems are rarely automated completely and we assume that workers are able to decide which task to process next, task prioritization cannot be enforced mechanistically as in machine scheduling. Therefore, we suggest a reward policy that encourages workers to follow the given priorities, but still allows them some autonomy. The importance of flexibility in workflow systems has been addressed in (Stohr and Zhao, 1997).

Because claim process instances can take various paths with different outcomes, there is uncertainty in the workflow system. Consequently, there is a need to estimate the expected turnaround time and allocate the total time to various tasks under uncertainty. The forces at play in temporal workflow management include: (1) Customers want to get results quickly and want to be informed of the delivery date right away, (2) Managers must balance the quality and urgency of processes, and (3) Workers need to know the expected deadlines for task prioritization.

Besides time allocation and task prioritization, the reduction of turnaround times can also be achieved potentially by adding new workers, reassigning workers' roles to balance the workload, and employing better technology to reduce the task process times. However, these issues require a separate study, and in this paper, we assume that these factors are fixed.

Although workload in the system can be very high and the system can become saturated, we do not consider overloading problems in this paper. We assume that the system is properly designed to handle the workload without becoming saturated.

#### 2.4. Summary

Given a claim with certain weight factors as discussed previously, we first estimate the turnaround time based on certain algorithms. We then allocate the estimated turnaround time to the processing nodes and prioritize the tasks at each node based on some priority factors. When a task passes a decision node, more information becomes known on the actual route of the claim, the estimated turnaround time is then recomputed and reallocated to the remaining processing nodes.

We adopt and develop various time allocation policies and task prioritization policies in the following section. The effectiveness of various policies is determined by their impact on the overall tardiness of the claim handling system. In this paper, we focus on detailing a research framework, and the analysis of various policies will be done in an extended paper.

### 3. ALGORITHMS AND POLICIES OF TEMPORAL WORKFLOW MANAGEMENT

This section describes the algorithms and policies of temporal workflow management discussed previously. We assume that an automated tool for task prioritization is built into the workflow management system, but that workers are free to deviate from the suggested task prioritization when they deem it desirable. As a result, a reward policy is needed to guide the behavior of workers. The combination

of workflow automation and worker flexibility is necessary for improving system performance without losing flexibility.

We also make a couple of additional assumptions to simplify the analysis in this paper:

- Each node only processes a single type of task. Although the work time of each task may vary in reality due to various reasons, we assume that the average work time is known for the task.
- The workflow model does not contain parallel routes so that at any given time, each claim can have only one task in process in the system.
- The worker at a workflow node does not know the whole routing process of each task. Therefore, it is important for the system to help the worker prioritize the task queue based on some prioritization policy.

No doubt, these assumptions make the model and the results of analysis less general; we will relax some of the assumptions in future research.

We now define a list of frequently used symbols; they will be explained further where they are used for the first time. Let

$Q_{ij}$  = Queue time allocated for claim  $i$ , task  $j$

$a_{ij}$  = Actual time claim  $i$  spent at node  $j$

$f_i$  = Priority factor for claim  $i$

$\Phi_i^k$  = The  $i^{\text{th}}$  workflow path of degree  $k$

$N$  = The highest degree in all paths

$n_j$  = The number of tasks in the queue at node  $j$

$p_{lj}$  = Probability for a task to go through link  $lj$  after reaching node  $l$

$E_l[T_{\text{priority}}^i]$  = Expected turnaround time for claim  $i$  to pass cluster  $l$  when the claim is assigned to a special queue priority, either highest (full queues), or lowest (empty queues).

$t_{ij}$  = Anticipated time to complete task  $j$  of claim  $i$

$w_j$  = Expected work time for a task at node  $j$

#### 3.1. Prediction of Turnaround Times

Time allocation requires the prediction of the turnaround time. We develop an algorithm for this purpose since a closed formula is difficult to find. To simplify the algorithm, we make the *steady state assumption* that the mean length of the task queues in the system remains stable. This assumption is both important and realistic as explained later.

We define a *cluster of paths* as all paths between a decision node  $i$  and a process node  $j$ , where all paths originating from node  $i$  end with node  $j$ . For instance, there are three clusters of paths in Figure 1, including those paths from node 5 to node 11, from node 10 to 13, and from node 2 to node 13. On the other hand, the paths from node 2 to node 11 do not constitute a cluster because some of the paths end with node 13, not node 11. The degree of a path indicates the level of

convolution of path clusters. A path of degree 1 means a path containing one decision node, a path of degree 2 contains two levels of decision nodes, and so on.

Paths of degree 1 (simple paths consisting of single nodes):

$$\Phi_1^1 = \{5, 6, 8, 11\}; \Phi_2^1 = \{5, 7, 11\}; \Phi_3^1 = \{2, 11, 13\}; \\ \Phi_4^1 = \{10, 11, 13\}; \Phi_5^1 = \{10, 12, 13\};$$

Paths of degree 2 (containing paths of degree 1):

$$\Phi_1^2 = \{\{2, 3, \{\Phi_1^1, \Phi_2^1\}, 13\}; \Phi_3^1; \{2, 4, 9, \{\Phi_3^1, \Phi_4^1\}\}\};$$

Paths of degree 3:

$$\Phi_1^3 = \{1, \Phi_1^2\};$$

Note that a valid path of any degree must start from a decision node (or the entry node of the graph) and stop with a node where all paths, originating from the same decision node, merge. For instance, path  $\Phi_3^1 = \{2, 11, 13\}$  stops with node 13 rather than 11 because node 11 is not the end node for all paths originating from node 2. Note that Node 11 in Figure 1 is shared by four paths:  $\Phi_1^1 = \{5, 6, 8, 11\}$ ;  $\Phi_2^1 = \{5, 7, 11\}$ ;  $\Phi_3^1 = \{2, 11, 13\}$ ;  $\Phi_4^1 = \{10, 11, 13\}$ .

For each claim, we compute two expected turnaround times,  $E[T_{full}^i]$ , under full queues, i.e., the highest priority, and  $E[T_{empty}^i]$  under empty queues, i.e., the lowest priority. In both cases, we assume that the claim experiences a theoretical path that is given probabilistically. These two expected turnaround times are computed using the following algorithm:

#### Procedure Expected\_Turnaround\_Times {

For  $k = 1$  to  $N$  {

For each cluster  $\Phi_l^k$  {

$E_l[T_{full}^i] = 0; E_l[T_{empty}^i] = 0;$  For each step in cluster  $\Phi_l^k$  {

If the step involves a single node  $j$  {

(1)  $E_l[T_{full}^i] = E_l[T_{full}^i] + n_j \times w_j;$  // assuming the average path and full queues

(2)  $E_l[T_{empty}^i] = E_l[T_{empty}^i] + w_j;$  // assuming the average path and empty queues

}

If the step involves a cluster {

For each path  $\Phi_j^{k-1}$  {

(3)  $E_l[T_{full}^i] = E_l[T_{full}^i] + p_{\bullet j} \times E_j[T_{full}^i];$  //  $p_{\bullet j}$  is the probability leading to path  $\Phi_j^{k-1}$

(4)  $E_l[T_{empty}^i] = E_l[T_{empty}^i] + p_{\bullet j} \times E_j[T_{empty}^i];$

}

}

}

}

}

#### Figure 2. A prediction algorithm for turnaround times.

Equations (1) and (3) compute the expected turnaround time for all clusters, assuming that the claim has the lowest priority and will go through the average route. Equation (1) computes the expected process time for a single node step. Since the given claim has the lowest priority, it will experience a full queue length. Equation (3) computes the expected process time when the subsequent nodes are in a cluster of paths by taking the expectation of the times in all possible paths. Equations (2) and (4) compute the turnaround time by assuming that the given claim has the highest priority and will go through the average path. In general, we have the relation of  $E_l[T_{full}^i] \geq E_l[T_{empty}^i]$  for all clusters in the workflow model. We use  $E[T_{full}^i] \geq E[T_{empty}^i]$  to indicate the expected turnaround times for the overall workflow model.

The actual turnaround time may vary with the lengths of queues at all nodes of the workflow system. The steady state assumption assumes that the lengths of queues do not vary, in order to make the prediction algorithm tractable. We believe that this is a viable assumption because we are making a conservative estimation. The predicated turnaround times can provide valuable information on the target performance of the workflow system that can be used to calibrate the workflow system and manage customer expectations. For the latter purpose, a more conservative predication of turnaround time is prudent when customers request information on the anticipated time of completion of their claims.

#### 3.2. Recomputation of Turnaround Times

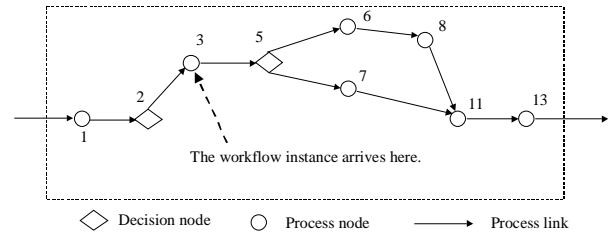


Figure 3. Partial workflow model after decision node 2.

When a claim is completed at a decision node, the next step of the claim becomes known. At this point, the expected turnaround time needs to be recalculated by taking into account the paths the claim has taken. The procedure Expected\_Turnaround\_Times in Figure 2 can still be used for the recomputation of partially instantiated paths, except that the workflow graph is the partial graph after the given decision node. For instance, after a claim leaves the decision node 2 and reaches node 3, Figure 3 will be used for the recomputation of turnaround times. The computation can be done similarly to the algorithm in Figure 2, and the details are therefore omitted here.

#### 3.3. Time Allocation Policies

Given the expected turnaround time, the queue lengths, and the work times for the claim at each node, the anticipated completion time for each step can be computed. This process is referred to as *time allocation*. The prediction of

turnaround times can become more accurate as the claim passes through the decision nodes of the workflow system. Therefore, reallocation of the expected turnaround time becomes necessary. We now present policies for time allocation and reallocation.

In order to prioritize the tasks queued at each node, we need to estimate the amount of time allowed for each task. That way, workers can first process tasks that are more urgent based on the remaining allocated times. That is, our task prioritization policies require the decomposition of expected turnaround times into allocated times for various tasks. The expected process time for the workflow steps of a claim can be initialized or reallocated as follows:

- **Uniform time allocation:** Assume that  $E[T_{full}^i] \gg E[T_{empty}^i]$ . The queuing time allocated to task  $j$  of claim  $i$  is  $Q_{ij} = (E[T_{full}^i] - E[T_{empty}^i])/n$ , where  $n$  is the number of steps on the average path in the remaining workflow graph. The expected completion time,  $t_{ij}$ , of task  $j$  of claim  $i$  can be computed simply by adding all work times  $w_k$  and the allocated queuing times  $Q_{ik}$ , where node  $k$  is on the path from the entry node of the workflow graph to the node  $i$ .
- **Load balanced time allocation:** Assume that the queue lengths ( $n_j$ ) for all nodes are known in the workflow system. The queuing time allocated to task  $j$  of claim  $i$  is  $Q_{ij} = (E[T_{full}^i] - E[T_{empty}^i]) \times (n_j / \sum n_k)$ , where the summation is done for all tasks along the average path in the remaining workflow graph.

Compared to uniform time allocation, load balanced time allocation is more accurate but requires real-time computation since the queue length is time variant.

### 3.4. Task Prioritization Policies

At each workflow node, there is a queue of tasks. Workers take tasks from the queue one by one according to certain priority. Next, we explore a few of the many possible prioritization policies and analyze their relative merits. Further analysis will be done in a full paper.

- **First-Come-First-Serve (FCFS) Policy:** Although pure FCFS policy is rarely used in the real world and is often mixed with some ad hoc priorities, analyzing this policy will be helpful for understanding the benefits of various priority policies studied in this paper. A FCFS queue simply ranks the tasks according to the arrival time  $t_{ij}$  of claim  $i$  at the queue  $j$ . The advantage of the FCFS policy is its simplicity. It is useful when the workflow system has a simple workflow graph and/or contains mainly workflow tasks with relatively uniform workflow processes. However, when a complex workflow graph and variant tasks exist, FCFS tends to result in a wide range of turnaround times since workflow claims that go through a longer path and have more time consuming tasks will experience long turnaround times.
- **Shortest/Longest-Work-Time (SWT/LWT) Policies:** A priority policy can be based on the expected work time, leading to shortest or longest work time policies.

If the reward function is based on the number of claims completed, the worker tends to use the SWT policy; on the other hand, if the reward function is to minimize the maximum turnaround time for all claims, the worker may elect to use the LWT policy. This is because the SWT policy tends to favor simpler claims that can be completed quickly while the LWT policy favors more complex claims. The problem with the SWT policy is that it further increases the spread of turnaround times since the more difficult claims tend to be left unattended for a longer period than under FCFS. While the LWT can lead to shorter turnaround times for complex claims, it may potentially lead to unreasonably long turnaround times for simple claims when the system is fully loaded.

- **Shortest-Remaining-Time (SRT) Policy:** To reduce the turnaround time, a priority policy for tasks at a node can be based on the expected completion time of the workflow step. That is, the task that has the smallest difference between the expected completion time  $t_{ij}$  and the present clock time  $t_{clock}$  is to be processed first. The rest of the tasks are ranked according to this time difference in an increasing order. This way, tasks that have less time remaining are performed earlier. That is, a SRT queue ranks the tasks, in increasing order, according to the value of  $(t_{ij} - t_{clock})$  for claim  $i$  at node  $j$ .

The SRT policy is different from the FCFS, the SWT, and LWT policies since it does not simply consider the arrival time or the process complexity, but takes into account the expected turnaround time (as computed according to the algorithm in the prior section.) The SRT policy tries to reduce the turnaround time for more complex tasks while keeping a balance towards simple tasks as well. More importantly, the SRT policy prioritizes tasks by taking into account the entire workflow system; on the other hand, the other three policies consider only tasks without looking at the whole workflow.

- **Weighted-Remaining-Time (WRT) Policy:** Although the SRT policy helps reduce the turnaround time for claims that requires more complex processes, it does not distinguish the value of the claims to the customers and to the company. Furthermore, it does not consider the urgency of the claims from either the customers' or the company's points of view. To improve on the SRT policy, we propose a weighted-remaining-time (WRT) policy to take into account various factors that are of value to customers and to the company.

A WRT queue ranks the tasks, in increasing order, based on the value of  $f_i \times (t_{ij} - t_{clock})$  for claim  $i$  at the queue  $j$ . The *priority factor*  $f_i$  is claim dependent and can be computed based on *three independent parameters*, the task urgency ( $u_i$ ), the customer value ( $c_i$ ), and the anticipated task complexity ( $m_i$ ). That is, the priority factor can be computed as  $f_i = f(u_i, c_i, m_i)$ .

The task urgency parameter  $u_i$  takes into account the needs of customers regarding the claim since all claims

are not equally urgent. This value is to be determined by an expert at the reception desk based on the description of the claims. The customer value parameter  $c_i$  is determined based on the loyalty and the financial value of the customer to the company. The process complexity  $m_i$  of the claim takes into account the expectation that the claim will go through a longer path. The determination of an appropriate functional form for  $f$  and estimation techniques for  $u_i$ ,  $c_i$ , and  $m_i$ , will be discussed in an extended paper.

A more formal comparison of the prioritization policies outlined in this section will also be given in the extended paper.

#### 4. IMPLEMENTATION OF TEMPORAL WORKFLOW MANAGEMENT

The temporal workflow management approach discussed in this paper must be implemented through a workflow engine that understands the temporal policies and principles. This temporal workflow engine keeps track of the arrival of new claims and provides guidance to the workers in the workflow system. The workflow engine does so by computing various parameters and values needed by the time allocation and task prioritization policies. In the following, we describe a temporal workflow engine and the enactment procedure for temporal workflow management.

As mentioned previously, workflow systems are usually semi-automated, and many tasks in the system are performed either completely or partially by humans. As such, the system cannot be scheduled as in a machine workshop where the subjects of discussion are numerically controlled machines. That is, in a workflow system, the temporal workflow policies can be offered only as suggestions to the workers. Furthermore, since the predicted turnaround times can in no way be perfect and the time allocation schemes are approximate as well, the suggestions may or may not be accurate. Consequently, it is a good idea to allow the workers to have the flexibility of deviating from the suggestions. This is the requirement of system flexibility discussed previously. In this section, we discuss the concept of reward functions that may be used to supplement the temporal workflow policies.

##### 4.1. Enactment of Temporal Workflow Management

To provide guidance to workers on the temporal workflow policies, the workflow engine needs to automate the computation process for the expected turnaround times, the time allocation principles, and the task prioritization policies. The SRT and the WRT policies require relatively sophisticated computations that are difficult to do manually. The general procedure for the temporal workflow engine is as follows:

- (1) At the arrival of a new claim, an initial examination of the claim gives an assessment of the claim. The values of task urgency ( $u_i$ ), customer value ( $c_i$ ), and anticipated task complexity ( $m_i$ ) are to be assigned at this point.
- (2) The expected turnaround time is then computed based on the system parameters such as the workflow model, the queue lengths, and the work times at all nodes.

- (3) Initial time allocation is to be done based on the computed turnaround times and the time allocation policy chosen. More detailed comparison of the two proposed time allocation policies remains to be done in future research.
- (4) The process time and the priority value for each step along the workflow path is then computed based on the time allocation and task priority policies.
- (5) The claim is then inserted into the workflow system and monitored by the workflow engine.
- (6) When a task of the claim passes a decision node in the workflow graph, the workflow engine must recompute the turnaround times, the time allocation, and the priority values of the claim.

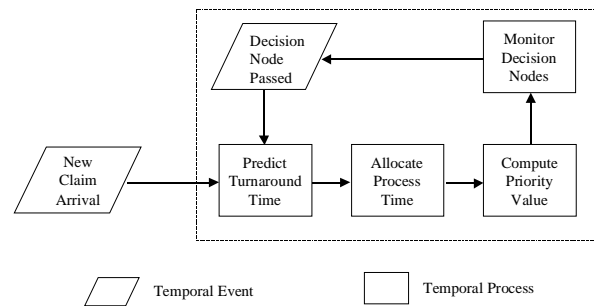


Figure 4. Temporal workflow enactment process.

Figure 4 illustrates the temporal workflow engine functions and the events and processes for enacting temporal workflow management.

##### 4.2. Reward Functions

Claim handling is essentially a support function for the company to provide after sale services, and its main goal is to maintain high customer satisfaction. Consequently, the quality of service in claim handling is a vital part of the workflow process. The motivation of the temporal workflow framework proposed in this paper has been the lack of advanced algorithms in real world claim handling workflow systems. As discussed in the introduction, a simple FCFS policy can result in long turnaround times for complex claims. The temporal workflow management heuristics discussed thus far can provide a remedy for this problem. However, proper reward functions are also important in order to realize the business value of temporal workflow management. In this subsection, we examine two reward functions found in industry and propose a third one. By reward function, we refer to the evaluation criterion used to influence the behavior of workers.

- **Number of claims completed per unit time:** The most common reward function attempts to measure productivity. In the case of claim handling, productivity can be measured by the number of claims completed per unit time. However, this can only be used to measure the productivity of the whole workflow system, not individual workers. This is because workers who work on internal nodes of the workflow model do not deliver outputs directly, and therefore

individual workers cannot be evaluated by the total outputs. Furthermore, the measurement of number of claims completed per unit time does not match the goal of minimizing the turnaround time for more complex workflow processes, and therefore, this reward function is inconsistent with the business goal.

- **Number of tasks completed per unit time:** The productivity of individual workers can be measured by the number of tasks performed per unit time. However, this reward function does not match the business goal of minimizing the spread of turnaround times since the workers would have no incentive to work on more difficult tasks. Furthermore, the number of tasks completed does not provide a metric for comparing workers at different nodes of the workflow system. This is because different paths of the workflow may lead to different complexities of workflow tasks.
- **Total business value delivered per unit time:** We propose an alternative reward function that measures the business value delivered, based on the completion time of tasks. That is, the reward  $r_{ij}$  for the worker who completes task  $j$  of claim  $i$  can be formulated as:

$r_{ij} = p \times f_i \times (Q_{ij} + w_j - a_{ij})$ , where  $a_{ij}$  is the actual time spent by claim  $i$  at node  $j$ ,  $Q_{ij} + w_j - a_{ij}$  is the difference between expected and actual process times, and  $p$  is a monetary conversion factor.

The total reward for a worker at node  $j$  is then equal to:

$$R_j = \sum r_{ij} \text{ for all claims } i \text{ done by the worker at node } j$$

Under this reward function, it is to the worker's advantage to minimize delays in completing high value tasks. To implement such a reward function, it is essential for the workflow engine to compute the relevant parameters and to rank the workflow tasks accordingly.

## 5. CONCLUDING REMARKS

Temporal workflow management is an open research area that has important business implications in the real world (Chinn and Madey, 1997). This paper investigated the basic temporal workflow management issues, focusing on the management of turnaround times. Our preliminary research results are reported in the context of claim handling, which usually involves time-sensitive workflow processes.

We created a conceptual framework for temporal workflow management by developing a procedure for predicating turnaround times that takes into account the workflow model, examining various potential policies and principles for time allocation and task prioritization, and outlining the basic process for enacting temporal workflow policies. The framework first estimates heuristically the expected turnaround times and then use the weighted tardiness of claims to set priorities at task queues. The accuracy of the expected turnaround times can be gradually improved by recomputing the expected values when the claim passes decision nodes. A temporal workflow engine is proposed to execute the recomputation. We also proposed new

prioritization policies for manipulating the workflow process.

Temporal workflow management will enable better control of business processes, especially where the processes are time critical. Through a more systematic prioritization of workflow tasks, temporal workflow management can potentially reduce the need for managers to "push" urgent claims personally. The application of more sophisticated weight factors and reward functions can lead to better quality of service and improve the overall business value of claim handling.

The framework we developed in this paper is unique because workflow automation enables the collection and utilization of more advanced information such as the weight factors we propose in the paper. Another contribution of this paper is that the workflow model is operational and can be implemented in a temporal workflow engine. This distinguishes our work from conventional process optimization models.

We are currently exploring the following issues as future research topics:

- The various time allocation and task prioritization policies will be compared more formally to ascertain their relative merits and suitable business applications.
- The usefulness and impact of the proposed reward function needs to be examined further.
- The proposed temporal workflow framework will be implemented and tested in a prototype system.

## REFERENCES

1. Bussler, C.J. "Policy resolution in workflow management systems." *Digital Technical Journal*, Fall 1994, vol.6, (no.4):26-49.
2. Bussler, C. J. and S. Jablonski, "An approach to integrate workflow modeling and organization modeling in an enterprise," *Proceedings of the 3rd IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Morgantown, WV, USA, 17-19, April 1994.
3. Berman, O., Larson, R.C., and Pinker, E., "Scheduling workforce and workflow in a high volume factory." *Management Science*, Feb. 1997, vol.43, (no.2): 158-72.
4. Cao, T. and A. C. Sanderson, "Task sequence planning using fuzzy Petri Nets", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 25, No. 5, May 1995.
5. Chinn, S.J. and Madey, G.R. "A framework for developing and evaluating expert systems for temporal business applications." *Expert Systems with Applications*, April 1997, vol.12, (no.3): 393-404.
6. Chroust, G. "Interpretable process models for software development and workflow." *Proceedings of 4th European Workshop on Software Process Technology*, Noordwijkerhout, Netherlands, 3-5 April 1995. p. 144-53.

7. Daniels, R.L. and J.B. Mazzola, "Flow shop scheduling with resource flexibility", *Operations Research*, vol.42, no.3 (May-June 1994) p504-22.
8. Haimowitz, I.J., Farley, J., Fields, G.S., and Stillman, J.; and others. "Temporal reasoning for automated workflow in health care enterprises." *Electronic Commerce: Current Research Issues and Applications*, Gaithersburg, MD, USA, 1 Dec. 1994. Springer-Verlag, 1996. p. 87-113.
9. Hall, T. and Shahmehri, N. "An intelligent multi-agent architecture for support of process reuse in a workflow management system." Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology. London, UK, 22-24 April 1996. p. 331-43.
10. Homem de Mello, L. S. and A. C. Sanderson, "Representations for mechanical assembly sequences", *IEEE Transactions on Robotics and Automation*, Vol. 7, No. 2, April 1991.
11. Jasper, H., Zukunft, O., and Behrends, H. "Time issues in advanced workflow management applications of active databases." Proceedings of the First International Workshop on Active and Real-Time Database Systems. Skovde, Sweden, 9-11 June 1995. Springer-Verlag, 1996. p. 65-81.
12. Kappel, G., Rausch-Schott, S., and Retschitzegger, W. "Coordination in workflow management systems-a rule-based approach." *Coordination Technology for Collaborative Applications: Organization, Processes, and Agents*. Springer-Verlag, 1998. p. 99-119.
13. Marshak, R.T. "Scheduling-based workflow." *Workgroup Computing Report*, Aug. 1993, vol.16, (no.8): 3-9.
14. McDermott, G. and Mulvihill, C. "Dynamic workflow analysis in a multiuser task context." *International Journal of Human-Computer Interaction*, Oct.-Dec. 1996, vol.8, (no.4): 433-55.
15. Akhil Kumar and J. Leon Zhao, "Dynamic Routing and Operational Controls in Workflow Management Systems", *Management Science*, 1998 (To Appear).
16. Olender, K. M. and L. J. Osterweil, "Cecil: a sequencing constraint language for automatic static analysis generation", *IEEE Transactions on Software Engineering*, Vol. 16, No. 3, March 1990.
17. Rajan, V. N. and S. Y. Nof, "Minimal precedence constraints for integrated assembly and execution planning", *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 2, April 1996.
18. Sheth, Amit. Editor. *Proceedings of NSF Workshop on Workflow and Process Automation in Information Systems: State of the Art and Future Directions*, Athens, GA, USA, 8-10 May 1996.
19. Stohr, E. A. and J. Leon Zhao, "A technology adaptation model for business process automation", *Proceedings of the 30th Annual Hawaii International Conference on Systems Sciences*, January, 1997.
20. WFMC, "Workflow reference model." *Technical report, Workflow Management Coalition*, Brussels, 1994.