Task Anticipation: a quantitative analysis using workflow process simulation

Igor Steinmacher[1, 2, 3], José Valdeni de Lima[1], Elisa Hatsue M. Huzita[3]
[1]*Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS) – Brasil*
[2]*Companhia de Informática do Paraná (CELEPAR) – Brasil*
[3]*Departamento de Informática – Universidade Estadual de Maringá (UEM) – Brasil*
igor@celepar.pr.gov.br, valdeni@inf.ufrgs.br, elisa@din.uem.br

Abstract

Traditional Workflow Management Systems (WfMS) are too rigid, posing problems when cooperation and human behavior are important factors. Under these circumstances, there is a need for a relaxation of the end-begin dependency. Task anticipation can provide this, relaxing the flow of control and dataflow among tasks, allowing exchange of results among sequential tasks even before the conclusion of the former task. This anticipation promotes a better cooperation and augments the performance in terms of process execution time. This paper presents some ways to use task anticipation in WfMS, simulating its behavior and comparing its performance to traditional process execution. Simulation results show that using anticipation is statistically better than the traditional approach (presenting profits of up to 33%). Taking into account possible additional effort anticipation shows better performance when initiated before 40% of the predecessors' execution.

1. Introduction

Workflow Management Systems have attracted much attention recently because of their simple model for defining, executing and monitoring processes. However, traditional WfMSs are too rigid, posing problems when cooperation and human behavior are important factors.

Aalst and Hee [1] say that the rigid and inflexible character of the early (and some of the contemporary) products scared away many potential users. Hagen and Alonso [7] show the same point of view, saying that the inflexibility turns workflow technologies restricted to a few domains. In this sense, so many researches point out these as the main reason to the resistance of using WfMS in some domains [8], [10], [16], [17], [18].

So, it becomes evident that is necessary to provide ways to relax some restrictions imposed by the actual systems. The restriction tackled here is the task impossibility of produce, supply or use intermediate results. So, in the traditional WfMSs, tasks are black boxes. Relaxing this restriction the tasks could be triggered before the conclusion of their predecessors using other tasks intermediate results: resulting in task anticipation [5], [6]. This task anticipation is the main focus of this paper.

Task anticipation promotes a better cooperation and augments the performance in terms of process execution time. The better cooperation is related to the intermediate results exchange, allowing two or more tasks to work with a same instance of a given data while it is still being produced. The performance is directly related to the level of parallelism created among tasks, as two or more tasks can be executed in parallel even though they were initially defined to execute in a strictly sequential manner.

The goal of this work is to present a study related to flexible workflow execution, focusing on identifying mechanisms to provide task anticipation and quantifying its performance augmentation. These mechanisms were classified according to their related phase (definition/ execution time). As result, two classes of mechanisms were specified: a priori and a posteriori anticipation.

After defining the mechanisms, some simulations were carefully carried, focusing on comparing the performance of the process execution using anticipation and using traditional execution. These simulations generate results related to processes execution time, considering anticipation and some extra efforts related to the extra communication and intermediate results exchange.

The rest of the paper is organized as follows. Section 2 presents a bibliographic revision related to workflow flexibilization. Section 3 introduces the task anticipation and the possible ways to use it. Section 4 presents the simulations carried. Finally, section 5 presents conclusions.

2. Related Researches

Workflow systems flexibility is a topic explored in many current researches. Such researches objective is to make workflow management systems useful to a large number of domains. Necessary flexibility focuses on kill the extremely rigid structure found in the traditional WfMS.

This section presents the researches and approaches related to flexibility. Analyzing the proposals, we have created a classification for the flexibility mechanisms: *a priori* flexibility and *a posteriori* flexibility.

A priori flexibility represents the flexibility offered during process modeling, related to process definition time. They create ways to make process execution less restrictive through new modeling elements or techniques.

A posteriori flexibility presents ways to make WfMSs less rigid through changes in processes execution. These changes are related to dynamic modifications in process and modifications in the way that processes are executed.

2.1 *A priori* flexibility

The first approach to be presented introduces the *Coo-operator* [4], developed to enhance modeling of virtual workflow processes executed by virtual enterprises with special interest for cooperative situations. This operator does not cover all the aspect of coordination and cooperation. Creating a new modeling element increases the effort necessary to model the process, demanding from

the workflow designer an anticipated configuration of the ways the information exchange will be proceeded.

Freeflow [3] proposes using different kinds of relationships between tasks. The proposed relationships are declarative, capable to separate the tasks dependencies and user information. This separation is made through a state-based task model which distinguishes user states and system states. This approach presents new dependencies based on the control flow, but do not treat the data flow.

Raposo *et al.* [15] also presents modification on relationships between tasks to create flexible processes, separating the interdependence and coordination mechanisms. It allows the deployment of different coordination policies in the same collaborative system by only changing the coordination mechanisms. This approach allows a great number of possibilities to define the processes, but, again, increases the modeling effort because of the new dependencies offered.

The approach presented in [9], [10] proposes that the correct definition of task behavior is the base for modeling less restrictive processes and to allow dynamic modification. Then, it is presented a way to specify different kinds of control flow focusing on defining less restrictive workflows. These dependencies are defined by E-C-A (Event-Condition-Action) rules. The flexibility and the degree of freedom offered are the positive points of this approach. However, the same problem of past approaches appears here: the extra complexity inserted by the necessity of creating and using new modeling elements.

2.2 *A posteriori* flexibility

An approach with *a posteriori* flexibility is those proposed by Nickerson [14], which presents the problems that long transactions can cause. In this way an event-based workflow model is proposed. This model allows changing the course of an in-progress long task. This approach relaxes the task isolation restriction through external events. However, it was not presented a way to relax the dataflow, allowing the results exchange.

In ADEPT project [16], [17], a model that offers support for dynamic process changing is proposed. ADEPT is a conceptual workflow model which is base on graphs with a rigid formal base for syntax and semantic. Based on this model, a minimum but complete set of operations that supports changing the structure of workflow processes instances, while they are executed, allied to restrictions that guarantee the consistency and correctness of the process - $ADEPT_{flex}$ [16]. Although allowing dynamic process adjusting result in extra flexibility, the tasks are still executing in an isolated way.

Co-flow project [11] presents an approach with intelligent workflow support. Intelligent agents are qualified to adapt and extend preplanned processes for specific situations, without influencing other cases based on the same definition. Co-flow is another approach that presents solution for the problem of the dynamic process changing. Using this approach would demand high cost and the tasks would continue as blackboxes, executing in isolated way.

Coo-flow [6] is another approach that presents *a posteriori* flexibility, more specifically it proposes the task anticipation. It presents a workflow-based technology to coordinate process with creative interactions. To allow task anticipation in execution time, two new states were proposed and an intermediate results exchanging protocol was used. This approach was the base of our work.

3. Task Anticipation

Anticipation means that a task is allowed to start its execution before the expected time [18]. It is possible to anticipate the start of a task because of the conclusion of its predecessors before the predefined time. But, it is not this kind of anticipation that is presented here. The concept of anticipation used here is starting a task without the conclusion of its direct predecessors. Evidently, this start happens during the execution of the predecessors, and its conclusion depends on the conclusion of the predecessors.

The advantage on using anticipation is the increasing of cooperation between tasks and the decreasing of the process total execution time. This occurs because of the parallelism created by intermediate results exchanging between tasks modeled as strictly sequential tasks. This parallelism is illustrated on Figure 1. Figure 1(a) presents the traditional process flow, conditioning the start of a task to the end of its predecessor. Figure 1(b) presents the same process but using the concept of task anticipation (at “revision” task).

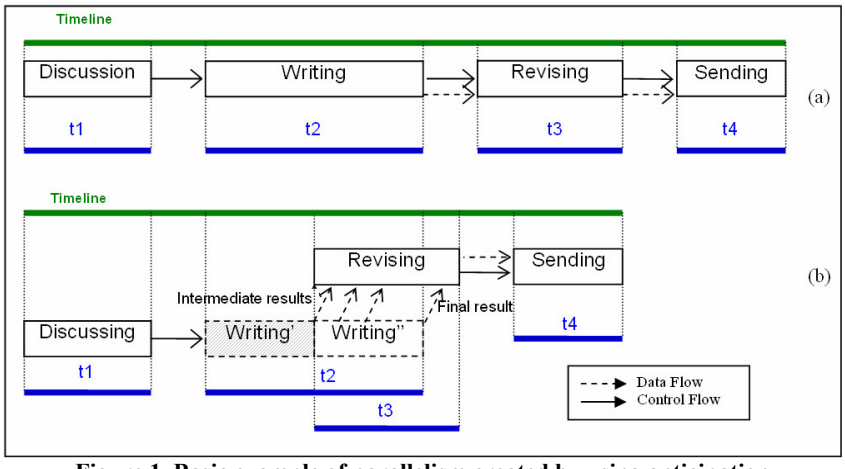

Figure 1. Basic example of parallelism created by using anticipation

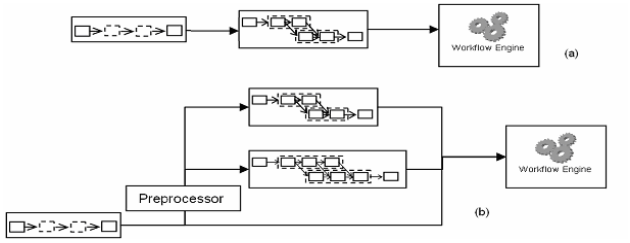

Figure 2. *A priori* anticipation management: manual approach (a) and automatic approach (b).

In the current paper, we separate the anticipation management in two different classes: *a priori* management and *a posteriori* management. Through *a priori* anticipation management, changes are made on process definition time. *A posteriori* anticipation demands some changes on workflow engine, in order to relax task isolation and atomicity restrictions. This approach changes the way that the workflow process is executed.

3.1 *A priori* management

Providing anticipation *a priori* is intuitive. It is called *a priori* because the anticipation is fully managed in process definition time, before the process execution

Anticipation is provided by *a priori* management through modeling the tasks allowed to exchange intermediate results in a lower granularity level. This can be made without using any different element or concept to model an anticipatable task. An anticipatable task and its predecessors are "broken" into smaller tasks. So they become partly sequential and partly parallel. *A priori* management can be provided in two ways:

Manual: Workflow designers have to identify and design all the tasks in the correct granularity level at definition time. An anticipatable task must have its predecessors divided into subtasks the anticipation. Process definition must contain the exact number of intermediate results that can be sent to the anticipatable tasks.

Automatic: Workflow designer just has to identify which tasks are allowed to be anticipated. The changes are inserted into the process definition when it is instantiated, through a preprocessor. If the process is defined in XPDL or BPEL, it is possible to use a XSLT preceding workflow engine, being possible to choose the number of interactions between anticipatable task and its predecessors.

Using manual approach it is not necessary to insert any different information in traditional process definition model. However, the extra effort required is so big, because it is necessary to analyze each task and the number of necessary (and required) results exchange.

Using the automatic approach, the necessary effort is lower than using manual approach. But it is necessary to include a new attribute into the process definition model. This can exclude some existent definition tools. Another turn off of automatic approach is the necessity of creating a preprocessor module before the workflow engine. But this preprocessor has the advantage of transforming one process definition into many different instances, according to the number of results exchanges defined for each anticipatable task. Figure 2 illustrates how *a priori* anticipation works in both cases: manual management (Figure 2(a)) and automatic management (Figure 2(b)).

3.2 *A posteriori* management

A posteriori anticipation management tries to make it easier to model processes with anticipation. In this case task anticipation is controlled at execution time, by workflow engine. To support the anticipation in execution time it is necessary to change the workflow engine. The changes are made in order to allow the tasks to start their execution with a different pre-conditions set. In the case of anticipation, allowing the tasks to be triggered before the conclusion of their predecessors.

Two different ways to provide this kind of management had been identified and are presented in the current paper. These kinds of anticipation will be presented follow.

A posteriori anticipation cannot be provided fully by workflow execution engine. It is necessary that workflow designers identify the tasks allowed to anticipate (may be through an extra task attribute). In other words, it is necessary to the workflow engine to know which tasks can be anticipated and those which can not be.

3.2.1 Modifying tasks state diagram

In this kind of *a posteriori* anticipation management the tasks state diagram is changed. Two new states are created, representing the tasks which have all their anticipation conditions fulfilled (ready to anticipate state) and the tasks that are executing in anticipated way (anticipating state). We have used Grigori approach [5] and WfMC state diagram as base to create the state diagram proposed here. This diagram is presented in Figure 3.

When a process is instantiated all its tasks enter the initial state, in this case the "notReady" state. This state is divided into two sub-states: anticipatable and notAnticipatable. A task is anticipatable if it is identified as a task that is allowed to be anticipated. If the task has not been identified as so, it becomes notAnticipatable. If the workflow process definition was designed with a tool that does not allow the identification, this can be made after the instantiation, in execution time. The bidirectional transition between these sub-states represents this possibility. So, the project manager can define anticipatable tasks in a flexible way, allowing or prohibiting a task to be anticipated according to the instance context. A task becomes ready when its anticipation preconditions are fulfilled. To represent these preconditions the states anticipatable and notAnticipatable have been divided into sub-states. These states concern to:

Anticipation: a task is identified as anticipatable or not (anticipatable/ notAnticipatable);

Dataflow: Input data of the task is available as final result (dataOK), as intermediate result (dataInterm) or are not available yet (dataNotOK);

Control flow: the trigger dependency of the task was fulfilled (controlOK), satisfies the *anticipation condition*[1] (controlAnt) or was not satisfied (controlNotOK)

So, a task becomes ready.toExecute when its predecessors are concluded and its input data are final results. And it becomes ready.toAnticipate when it is anticipatable, its predecessors are running and its input data

[1] The anticipation condition used here is: a task may only be anticipated after its predecessors begin and may only be concluded after its predecessors' conclusion. This kind of dependency is called Medium Dependency [18].

are intermediate results. In both cases the tasks are sent to the worklist of agents able to execute them.

When required by an agent the task enters the state running. If the task required was ready.toAnticipate it becomes anticipating and if it was ready.toExecute it becomes an executing task. When an anticipating task has all its conditions fulfilled (i.e., its predecessors are concluded), it becomes an executing task. The only transition to concluded state has the executing state as origin, so, only the tasks whose predecessors are concluded can be concluded, obeying the *Medium Dependency* [18].

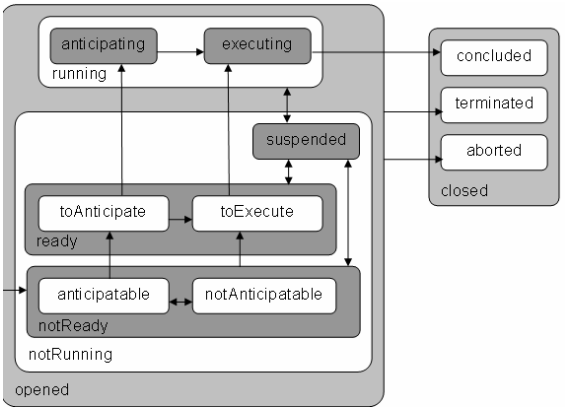

Figure 3. Modified state diagram for tasks, with new states to provide anticipation managed *a posteriori*

3.2.2 Using dynamic process changing

This class of a posteriori anticipation management uses the approach proposed in [17]. Here we have identified that the *task insertion* operation is useful and can be used to provide anticipation in execution time.

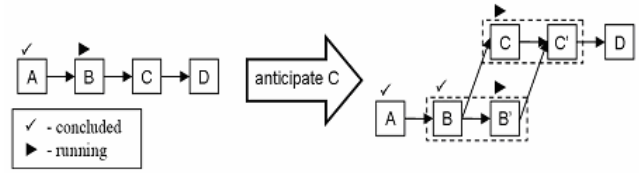

Figure 4. *A posteriori* anticipation management using dynamic process changing.

Using this approach, the workflow engine is the responsible for allowing the tasks to anticipate their start when the anticipation conditions are fulfilled. When task conditions are satisfied and some agent start its anticipation, workflow engine "break" the anticipated task and its predecessors. This operation generates a parallel flow between sequential tasks during process execution.

Figure 4 illustrates the *a posteriori* anticipation using dynamic process changing. Task *C* requested anticipation during the execution of task *B*. So, the system immediately inserted new tasks (*B'* and *C'*) representing the final part of *B* and the initial part of *C*. When *B* is concluded, *C* is also marked as concluded, and *C'* is triggered.

4. Simulating Anticipation

Computer simulation is an efficient way to study the processes and preview their performance [12], [19]. Through simulation it is possible to analyze some aspects of a process in a quantitative and dynamic way [19]. Aiello [2] said that through simulations the developers can execute a process in different scenarios while varying values and input distributions. At the end of simulations, tools may provide valuable information about the process.

Focusing the performance analysis of using task anticipation, some simulations were carefully carried. The results show, in a quantitative way, a comparative between the traditional execution of a process and an execution using anticipation. The results are based on two data analysis: the mean process execution time values and a statistical analysis of the results based on the pairwise t-test (testing which values are considered statistically different from traditional execution values with 95% confidence).

4.1 Test Bed and Scenarios

The test bed used in our simulations is a basic process of creating and submitting a scientific paper. We used this process because it sounds familiar and contains the desirable characteristics to use anticipation: it is a predominantly intellectual process; there is data exchange between tasks. The process used as test bed is presented in Figure 5. In the Figure shaded tasks can be anticipated; values over tasks are the minimum and maximum execution time for each task (in hours).

The results obtained are based on analysis of total process execution time varying some values:

Task execution time: The time expended by a task to be concluded. When set to variable, the values are generated through a Beta PERT distribution;

Anticipated start: Represents the moment when anticipated tasks are started. This value is relative to the task predecessors' execution. For example, if the anticipated start is 0.5, then, the anticipatable tasks will be started after 50% of the predecessors' task execution time. When set to variable, the values are generated through a linear distribution in the interval [0.1, 1];

Additional effort: This variable simulates the human behavior, representing the effort relative to anticipation. This effort may include extra communication, results exchanging, new results waiting and conflicts resolution.

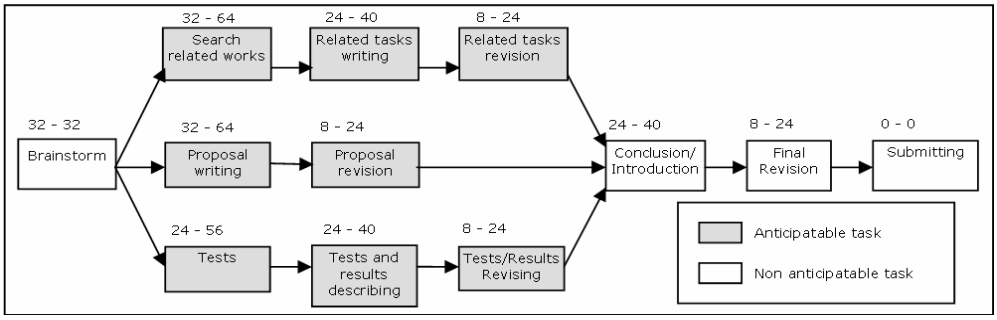

Figure 5. Test bed used in simulations: a basic paper writing and submitting process

Two simulation scenarios were created. For each simulation scenario 500 cases were generated. The use of randomized values tries to express human behavior. The scenarios and their results are presented following.

First Scenario. The first simulation scenario confronts the total process execution time using traditional execution and anticipation. In this scenario the cases were generated varying the *task execution time* and applied to 10 different values of *anticipated started* (from 0.1 to 1). In this scenario we do not consider any *additional effort*.

The results of this scenario are presented in Table 1 and show that task anticipation makes the process execution time decrease, if compared to traditional execution. The numbers show that the better results appear when the anticipation is started sooner.

Table 1. Process execution time varying anticipation start

Anticipation Started	*Mean Process Execution Time*
10%	119,89
20%	123,32
30%	126,87
40%	132,68
50%	140,52
60%	149,34
70%	156,28
80%	164,46
90%	171,83
Randomized	147,38
Traditional Execution	178,67

Comparing anticipation started at 10% of predecessors' task execution time and traditional execution, there is a process execution time decrease close to 30%. Comparing an intermediate value of *anticipation started* (50%) and the traditional execution, the decrease is next to 20%. When the *anticipation started* values are randomized, the profit is close to that obtained with *anticipation started* at 50% (decreasing the process execution time 20%).

Second Scenario. The second scenario uses the *additional effort* variable. The cases were the same used in the first scenario. But in this scenario we simulate the cases with each pair of *additional effort* (from 0 to 1) and *anticipated start* (from 0.1 to 1) values.

This scenario results are presented in Table 2. Table 3 presents the results combining Table 2 values and pairwise t-test with 95% confidence. In this table there are three possible values: *same*, *no* and *yes*. *Same* indicates that the value achieved by traditional execution and the compared value are statistically not different; *yes* indicates that the compared value represents better performance than traditional execution, and *no* indicates that it presents worse performance than traditional execution.

Obtained results focus in producing a behavior pattern linking necessary effort to anticipate tasks and anticipation start. With the data and graphic it is possible to verify the values where task anticipation is advantageous (in terms of execution time). Confronting these values with real cases historical data, manager can free or block the anticipation of a task, for example.

Analyzing the numbers, anticipation best results occur when the tasks are anticipated at the beginning of the predecessors (10% and 20%), when the advantage appears in all simulated cases. In the worst case the additional effort is 100% (duplicating execution time) and the obtained result is better than using traditional execution.

When the anticipation is started between 30% and 40% of predecessors' execution time, the advantage appears to additional effort values lesser than 80%. For anticipation starting at 50%, the advantage appears for additional effort from 0% to 50%. Setting the anticipation start after 60% of execution of predecessors the advantage only appears when the effort is not greater than 20%.

It is explicit that the delayed begin of anticipation is not advantageous for the process. When the tasks are anticipated after 80% of their predecessors, results are better than traditional execution only when there is no additional effort to anticipate.

Table 2. Process execution time varying the anticipation start and the additional effort

Effort / Start	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Randomized
10%	119,89	124,39	130,57	135,64	140,53	144,59	148,66	155,46	159,16	165,10	169,15	148,51
20%	123,32	129,08	133,35	138,30	143,71	147,91	152,30	158,20	163,21	168,09	173,62	153,46
30%	126,87	133,52	138,03	142,84	148,66	155,64	158,14	164,41	168,41	173,51	178,27	158,12
40%	132,68	138,96	144,53	150,04	155,22	159,49	165,16	170,98	175,81	180,91	186,94	164,39
50%	140,52	147,92	153,54	159,37	164,50	169,78	175,84	181,56	187,01	193,33	198,69	173,70
60%	149,34	155,72	161,91	168,59	174,51	180,04	186,61	193,75	199,09	205,29	210,97	183,26
70%	156,28	163,44	170,61	178,14	184,36	190,10	196,48	204,40	210,75	216,78	224,06	194,44
80%	164,46	172,08	179,63	186,13	192,86	199,50	207,23	215,86	221,85	230,54	236,68	204,09
90%	171,83	181,21	188,67	195,90	203,97	211,04	219,52	225,84	233,34	240,86	247,33	215,29
Randomized	147,38	153,27	158,86	166,02	172,11	176,74	183,06	189,04	197,40	203,00	207,73	179,20
Traditional Execution	178,67	178,67	178,67	178,67	178,67	178,67	178,67	178,67	178,67	178,67	178,67	178,67

Table 3. Better than Traditional Execution with 95% confidence

Effort / Start	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Randomized
10%	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
20%	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Same	Yes
30%	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Same	Same	Yes
40%	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Same	Same	Same	Same	Yes
50%	Yes	Yes	Yes	Yes	Yes	Yes	Same	Same	Same	No	No	Same
60%	Yes	Yes	Yes	Yes	Same	Same	Same	No	No	No	No	Same
70%	Yes	Yes	Same	Same	Same	No	No	No	No	No	No	No
80%	Yes	Same	Same	Same	No	No	No	No	No	No	No	No
90%	Yes	Same	No	No	No	No	No	No	No	No	No	No
Randomized	Yes	Yes	Yes	Yes	Same	Same	Same	No	No	No	No	Same

When randomizing start values, the anticipation is better with additional effort until 40%. When randomizing both variables the result was good, presenting a value similar to the mean execution time obtained in traditional execution. We said it is a good result because it shows that a process execution using anticipation with additional effort has the same performance as the same process using traditional execution flow. So, we can have the cooperation advantage introduced by anticipation without losing more time than using traditional execution. Increasing cooperation, we may have better quality products. This motivates us to apply anticipation to real cases to evaluate possible cooperation increasing and final product's quality.

5. Conclusions

The goal of the current paper is to present the task anticipation as a way of relax some restrictions of current Workflows Management Systems, focusing on its performance analysis. Anticipation can increase the performance of cooperative processes through the parallelism created among tasks. To evaluate the performance of task anticipation confronting traditional process execution some measurements had been carried. The results of the simulations had allowed identifying to significant profits in favor of the use of the anticipation and behavior patterns that can be used in future tests.

The results demonstrate that it is necessary to start the anticipation before the half of the execution of tasks predecessors to anticipation become more efficient. In our test bed, anticipation presented profits of up to 33% (tasks starting anticipation with 10% of their predecessors' execution time). In the average case, the profits with relation to the traditional execution had been close to 18%. Also with additional effort, anticipation had been advantageous in most part of the cases. When the anticipation was initiated before 40% it presented better performance. Anticipating tasks next to the end to its predecessors can bring losses, if the necessary extra effort for such anticipation exists. For anticipation started after 70% of the beginning of the predecessors these losses appear when the effort is superior 30%.

Besides the profits, simulations had presented some behavior patterns linking anticipation start, additional effort and process execution time. These patterns can be used to enable/disable anticipation according to historical analysis of additional effort inserted by anticipation. For example, the anticipation of a task can be disabled if its predecessors already exceeded 60% of its mean execution time.

When using randomized values for *additional effort* and *anticipation start* the mean process execution time was the same as using traditional execution. So, we can have more cooperation and better final products without losing time. In our simulations the possible profits related to final product quality and to cooperation increasing had not been analyzed. To make it possible it is necessary to apply anticipation to real cases, comparing results of processes with and without using anticipation approach. Also effort and time of execution had not been analyzed costs relating.

Other results of this research: the workflow simulator used to carry the simulations presented here, a workflow editor and a workflow engine implementing anticipation concepts.

6. References

[1] Aalst, W.; Hee, K. Workflow Management: Models, Methods and Systems. The MIT Press – Massachussets Institute of Technology, Massachussets, 2002.

[2] Aiello, R. Workflow Performance Evaluation, (PhD. Thesis) – Università di Salerno, Salerno, 2004.

[3] Dourish, P.; Holmes, J.; Maclean, A.; Zbyslaw, A;.Marqvarsen, P. Freeflow: Mediating Between Representation and Action in Workflow Systems. In: ACM conference on computer supported cooperative work, 1996, Boston, EUA. Nova York: p. 190-198.

[4] Godart, C.; Perrin, O.; Skaf, H. Coo: a Workflow Operator to Improve the Cooperation Modeling in Virtual Processes. In: International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises, Australia, 1999.

[5] Grigori, D. Eléments de flexibilité des systèmes de workflow pour la définition et l'exécution de procédés coopératifs. 2001. (PhD Thesis) – Université Henri Poincaré, Nancy 1. 2001.

[6] Grigori, D.; Charoy, F.; Godart, C. Coo-Flow: A process technology to support cooperative processes. International Journal of Software Engineering and Knowledge Engineering, Jan 2004.

[7] Hagen, C.; Alonso, G. Beyond the black box: Event-based inter-process communication in process support systems. In: International Conference on distributed computing systems, 1999

[8] Joeris, G.; Herzog, O. Towards Object-Oriented Modeling and Enacting of Processes. TZI-Report 07/98, Universidade de Bremen, 1998.

[9] Joeris, G. Defining Flexible Workflow Execution Behaviors. In: Enterprise-Wide and Crossenterprise Workflow Management: Concepts, Systems, Applications, Germany. 1999.

[10] Joeris, G.; Herzog, O. Towards Flexible and High-Level Modeling and Enacting of Processes. In: international conference on advanced information systems engineering, 11., Heidelberg, Germany, 1999. p. 88-102.

[11] Kramler, G.; Retschitzegger, W. Towards Intelligent Support of Workflow. In: Americas Conference on Information Systems, Long Beach. 2000. p. 581-585.

[12] Laguna, M.; Marklund, J. Business Process Modeling, Simulation, and Design. Prentice Hall, 2004.

[13] Lima, J.; Quint, V.; Layaida, N.; Edelweiss, N.; Zeve, C.; Pinheiro, M.; Telecken, T. The Conception of Cooperative Environment for Editing Multimedia Documents with Workflow Technology (CEMT). In: PROTEM-CC, 2001.

[14] Nickerson, J.V. Event-based Workflow and the Management Interface. In: Hawaii International Conference on System Sciences (HICSS), 36., 2003, Big Island, Hawaii. Proceedings… Washington: IEEE Computer Society, 2003.

[15] Raposo, A.; Magalhães, L.; Ricarte, I.; Fuks, H. Coordination of Collaborative Activities: A Framework for Definition of Tasks Interdependencies. In: International Workshop on Groupware (CRIWG), 7., 2001, Germany. 2001.

[16] Reichert, M.; Dadam, P. Adeptflex – Supporting Dynamic Changes of Workflow Without Loosing Control. Journal of Intelligent Information System: Special Issue on Workflow Management Systems, v.10, n.2, p.93-129, 1998.

[17] Reichert, M.; Rinderle, S.; Dadam, P. ADEPT Workflow Management System: Flexible Support for Enterprise-Wide Business Processes. In: Business process management international conference (BPM), 2003, Netherlands. 2003

[18] Steinmacher, I. Estudo e Análise de Desempenho da Antecipação de Tarefas em Sistemas Gerenciadores de Workflow.(Master Thesis), Universidade Federal do Rio Grande do Sul, Brazil. 2005.

[19] Tramontina, G., Wainer, J., Ellis, C. Applying scheduling techniques to minimize the number of late jobs in workflow systems. In Proceedings of the 2004 ACM Symposium on Applied Computing, USA, 2004. ACM Press.