Uma Maneira de Antecipar Tarefas em um Sistema Gerenciador de Workflow

Igor Steinmacher, José Valdeni de Lima, Ana Vitória Piaggio

UFRGS – Universidade Federal do Rio Grande do Sul
II – Instituto de Informática
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil
{ifsteinmacher, valdeni, avpfreitas}@inf.ufrgs.br

Resumo

Sistemas Gerenciadores de Workflow (SGWf) têm atraído muita atenção nos últimos anos, por serem baseados em um modelo simples que permite definir, executar e monitorar a execução de processos. Mas este esquema é um tanto quanto rígido e apresenta alguns problemas quando da necessidade de sua utilização no controle de processos com características cooperativas, em que o comportamento humano deve ser levado em conta. Neste tipo de processo deve haver uma certa flexibilidade, de forma a possibilitar a troca de resultados entre tarefas. Este trabalho tem como objetivo apresentar um meio de se melhorar a eficiência dos processos e a cooperação entre agentes durante a execução de tarefas controladas por workflow, através de um mecanismo de antecipação, permitindo a troca de resultados intermediários.

Palavras-chave: *Workflow, antecipação, flexibilidade, resultados intermediários*

1. Introdução

A fim de tornar o trabalho controlável e encorajar a comunicação entre empregados, Sistemas Gerenciadores de Workflow (*SGWf*) foram criados. Estes se tornaram uma nova classe de sistemas de informação. Eles tornam possível construir, de maneira avançada, uma ponte entre o trabalho das pessoas e os sistemas de informação [Aalst e Hee 2002].

Porém, os SGWf tradicionais são, na maioria dos casos, focados em processos de negócio bem estruturados do ponto de vista transacional. Estes SGWFs são utilizados seguindo um rigoroso mecanismo de engenharia e especificação de processos, resultando em processos rígidos e inflexíveis. Esta rigidez e inflexibilidade acabam por tornar as tecnologias de workflow atuais restritas à apenas alguns domínios de aplicação [Hagen e Alonso, 1999].

Os processos administrativos e de produção geralmente são bem coordenados por SGWfs tradicionais [Godart *et al.*, 2000], pois estes domínios apresentam tarefas bem definidas, que podem ser modeladas e executadas como transações ACID. Porém, em certos domínios de aplicação, onde tarefas são predominantemente executadas por recursos humanos, é necessário relaxar algumas das restrições impostas pelos mecanismos tradicionais de forma a possibilitar a interação entre tarefas que possam cooperar.

Uma das restrições impostas pelo modelo tradicional é a impossibilidade das tarefas produzirem, disponibilizarem ou utilizarem resultados intermediários. Isto é, as tarefas em processos de *workflow* funcionam como caixas pretas, interagindo com as outras tarefas apenas no início e ao final de suas execuções.

Relaxando esta restrição, estaria-se provendo, por exemplo, um modo de antecipar algumas tarefas, reduzindo o tempo de execução dos processos. É justamente este o objetivo principal deste trabalho, flexibilizar o modelo de workflow tradicional, de forma a possibilitar

que tarefas seqüenciais possam trocar resultados intermediários, iniciando sua execução de maneira antecipada. Neste artigo, é apresentada então, uma maneira de flexibilizar o modelo de workflow tradicional, de forma a possibilitar que tarefas seqüenciais possam trocar resultados intermediários, iniciando sua execução de maneira antecipada.

2. Antecipação de Tarefas

O conceito de por traz da antecipação de tarefas é intuitivo [Grigori *et al.,* 2004]. Antecipação é a maneira através da qual uma tarefa pode iniciar sua execução mesmo antes de suas condições de execução serem totalmente satisfeitas.

Em processos tradicionais, o *workflow* é modelado de forma que as tarefas sejam caixas pretas e obedeçam uma dependência do tipo *fim-início*. Isto quer dizer que uma tarefa é atômica, deve ser executada de maneira isolada e somente pode ser iniciada quando suas antecessoras forem concluídas e seus dados de entrada (quando houverem) forem resultados finais de tais tarefas.

O foco da antecipação de tarefas é justamente acabar com as restrições impostas acima. Isto é feito flexibilizando o fluxo de dados de forma a permitir que certas tarefas troquem resultados intermediários e o fluxo de controle, alterando as condições de disparo das tarefas e incluindo diferentes tipos de dependência entre as tarefas. Isto permitirá que as tarefas iniciem sua execução antes mesmo que suas condições de início não estejam totalmente satisfeitas ainda. Ao contrário dos processos tradicionais, as tarefas passarão a poder serem iniciadas antes mesmo de suas antecessoras concluírem sua execução e aceitarão como dados de entrada resultados intermediários.

A grande vantagem de se permitir a antecipação está no aumento da eficiência com que um processo pode ser executado. Isto se deve ao paralelismo (parcial) criado entre tarefas que deveriam executar de maneira seqüencial. Tal acréscimo no paralelismo é ilustrado na Figura 1, onde a Figura 1(a) representa o fluxo tradicional que o processo deveria seguir e a Figura 1(b) representa o fluxo do mesmo processo com a antecipação de uma tarefa. Fica evidente o ganho com relação ao tempo de execução do processo, ao aumentar o paralelismo.

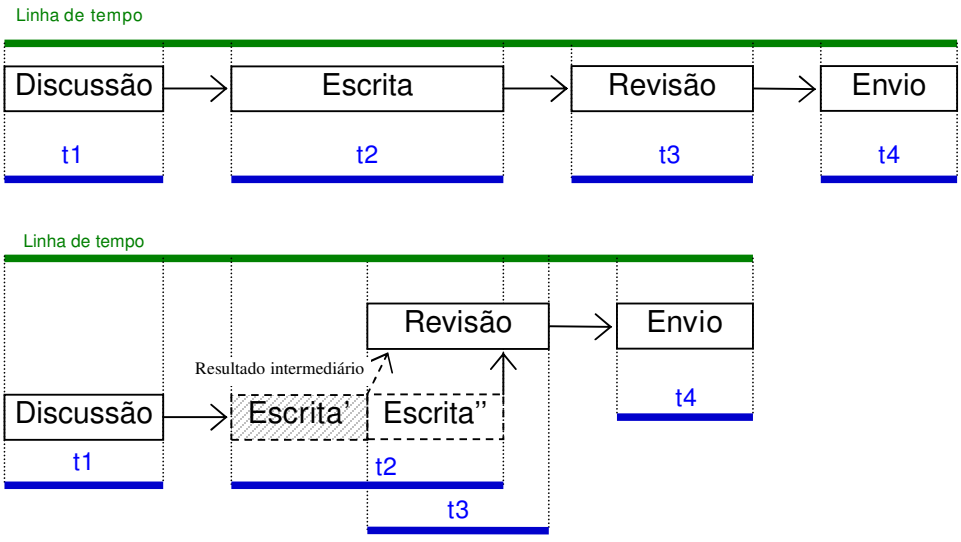

Figura 1. Aumentando o paralelismo através da utilização de antecipação

Nas próximas seções ser alterações necessárias ao sistema gerenciador de *workflow* para que seja possível adotar a antecipação de tarefas. Primeiramente serão apresentadas as novas possibilidades de dependência entre tarefas inseridas ao modelo. Em seguida será apresentada a mudança que deve ser realizada junto à máquina de execução para permitir o disparo diferenciados das tarefas. Por último será apresentada a maneira como é realizada a troca de resultados intermediários de maneira simples.

3. Alterações no Modelo

A antecipação não é aplicável a todos os processos nem a todas as tarefas de um determinado processo. Existem processos e tarefas que devem ser executadas obrigatoriamente de maneira atômica e isolada. Algumas tarefas não executadas por recursos humanos (executadas automaticamente), possivelmente não saberiam como lidar com um resultado intermediário. Outro grupo de tarefas que não podem ser antecipadas são aquelas tarefas que necessitam de um resultado obrigatoriamente final para serem executadas. É o caso da tarefa de envio de um artigo, que não pode ser realizada em partes, através do envio de resultados parciais.

Visto isto, se torna necessário adicionar uma informação às tarefas indicando se uma tarefa pode ou não ser antecipada. Contudo, dizer que uma tarefa será antecipada já na fase de modelagem é um tanto quanto complexo. Porém, dizer quais tarefas podem ou não podem ser antecipadas é possível (e desejável). Com isto, ficariam identificadas para o sistema aquelas tarefas que possuem permissão para executar de maneira antecipada e aquelas que devem seguir sua execução de modo tradicional, necessitando aguardar todas suas condições de disparo serem satisfeitas.

A necessidade de tal identificação porém, esbarra em alguns problemas apontados em outras abordagens, que são o aumento da complexidade na definição das tarefas e a alteração do modelo de processos tradicional. A alternativa encontrada foi a adição de um atributo às tarefas. Tal atributo é de caráter não obrigatório, e deixaria a critério do gerente responsável pela criação do processo a decisão de utilizar a antecipação ou não.

A alternativa encontrada foi a adição de um atributo às tarefas, na tentativa de minimizar a alteração e o aumento da complexidade do modelo tradicional de processos. O atributo proposto é de caráter não obrigatório, deixando a critério do gerente responsável pela criação do processo a decisão de utilizar a antecipação ou não. Qualquer ferramenta de modelagem poderia facilmente se adequar a tal modificação. Além do que, as máquinas de execução que não suportarem antecipação poderiam interpretar os modelos e desconsiderar o atributo estendido.

Para tal atributo será considerada a "*antecipabilidade*" das tarefas e também a dependência desta tarefa com relação às suas antecessoras. Com isto pretende-se adicionar mais flexibilidade ainda ao fluxo de controle de um processo. Portanto, o gerente seria capaz de definir de que maneira as tarefas poderiam ser antecipadas. Muitas outras pesquisas sugerem a criação e a flexibilização da dependência inter-tarefas.

Enquanto tradicionalmente as tarefas se relacionam através de dependência *fim-início*, através dessa abordagem avançada o gerente poderá optar por outras alternativas de tipo de dependência de fluxo de controle. Além da tradicional *fim-início* (tarefas não antecipáveis) é oferecido um conjunto de tipos de dependências baseadas apenas no início e no fim da execução das tarefas. Abaixo são apresentadas algumas das dependências identificadas e propostas para flexibilizar o fluxo de controle:

- *Dependência Forte*: este é o tipo de dependência padrão, isto é, é a dependência *fim-início.* Define a tarefa como *não antecipável,* fazendo-a obedecer o fluxo de controle de maneira tradicional. Uma tarefa condicionada a este tipo de dependência deve aguardar que suas antecessoras sejam concluídas para iniciar sua execução no caso ser antecedida por um *AND-join*; ou uma de suas antecessoras sejam concluídas no caso de um *OR-join*.
- *Dependência Média:* esta é a dependência mais conservadora dentre as alternativas aqui propostas para definir uma tarefa como antecipável. Ela une as restrições da dependência *início-início* àquelas da dependência *fim-fim.* Uma tarefa sujeita a este tipo de dependência somente poderá ser disparada após o início de todas as suas antecessoras, no caso de um *AND-join*. E, no caso de um *OR-join*, após o início de uma de suas antecessoras. Além disso, esta tarefa só poderá ser concluída ao final de suas antecessoras. Quando antecedida de um *and-join* terá de aguardar a conclusão de todas as suas antecessoras. Caso seja um *or-join*, é necessário aguardar que a tarefa que foi utilizada para satisfazer a condição de disparo seja concluída.
- *Dependência fraca*: São definidos dois tipos de dependência fraca distintas, de acordo com a restrição mantida:
    - *Início*: definindo-se uma tarefa com dependência de tipo *início-início* a torna antecipável. Isto flexibiliza o fluxo de controle de maneira a restringir o disparo desta tarefa ao início de suas antecessoras. Fica relaxada a dependência *fim-fim*, ficando a tarefa independente da conclusão de suas antecessoras. No caso desta tarefa ser antecedida por um *OR-join* é necessário que pelo menos uma de suas antecessoras tenha sido iniciada. Já no caso da tarefa ser antecedida de um *AND-join*, é necessário que todas as suas antecessoras tenham sido iniciadas para esta poder ser disparada;
    - *Fim:* Este tipo de dependência não condiciona o disparo da tarefa (flexibiliza a dependência *início-início*). A flexibilidade oferecida por este tipo de dependência permite que uma tarefa seja iniciada a qualquer momento. A única restrição fica por conta da conclusão da tarefa, que fica condicionada ao término de todas as suas antecessoras no caso de um *AND-join* e de pelo menos um de suas antecessoras quando tratar-se de um *OR-join;*
- *Independente do Fluxo de Controle:* Nada restringe o disparo das tarefas. As tarefas podem ser iniciadas a qualquer momento, estando todas disponíveis para execução. No caso deste trabalho, o disparo das tarefas estaria sujeito ainda aos dados de entrada da tarefa. O disparo das mesmas ocorre quando um resultado parcial de um de seus dados de entrada estiver disponível.

Para o restante do trabalho deve ser considerada apenas a "*dependência média*", que oferece maior segurança e limita a antecipação apenas a algumas tarefas, diminuindo a chance de desperdício de trabalho.

4. Flexibilizando a Máquina de Execução

Para permitir que tarefas possam ser disparadas sem que suas condições sejam satisfeitas se fez necessário alterar a máquina de execução de processos.

A fim de prover a flexibilidade quando da execução do processo, foi necessário, então, alterar o diagrama de estados das tarefas da máquina de execução do *SGWf*. Tal alteração é proposta a fim de tornar a antecipação clara e simples. Para a concepção de tal diagrama, foram tomados como base o diagrama de transição de estados proposto pela WfMC [WfMC, 2004] e os estados e comportamento previstos no *Coo-Flow* [Grigori *et al*, 2004], que leva em conta a antecipação de tarefas. O diagrama de transição e seus respectivos estados propostos são apresentados na Figura 2.

Neste novo diagrama foram criados os novos estados `pronta` e `nãoPronta` (sub-estados de `nãoRodando`) e os estados `antecipando` e `executando` (sub-estados de `rodando`). O estado `suspensa` foi re-alocado como um sub-estado de `nãoRodando`, uma vez que, quando uma instância de tarefa está `suspensa`, obviamente, não está rodando. Em níveis de granulosidade mais baixa, foram definidos alguns outros estados. O estado `pronta` é dividido ainda em dois sub-estados: o estado `paraAntecipar` e `paraExecutar`. O estado `nãoPronta` faz distinção entre os sub-estados `antecipável` e `nãoAntecipável`.

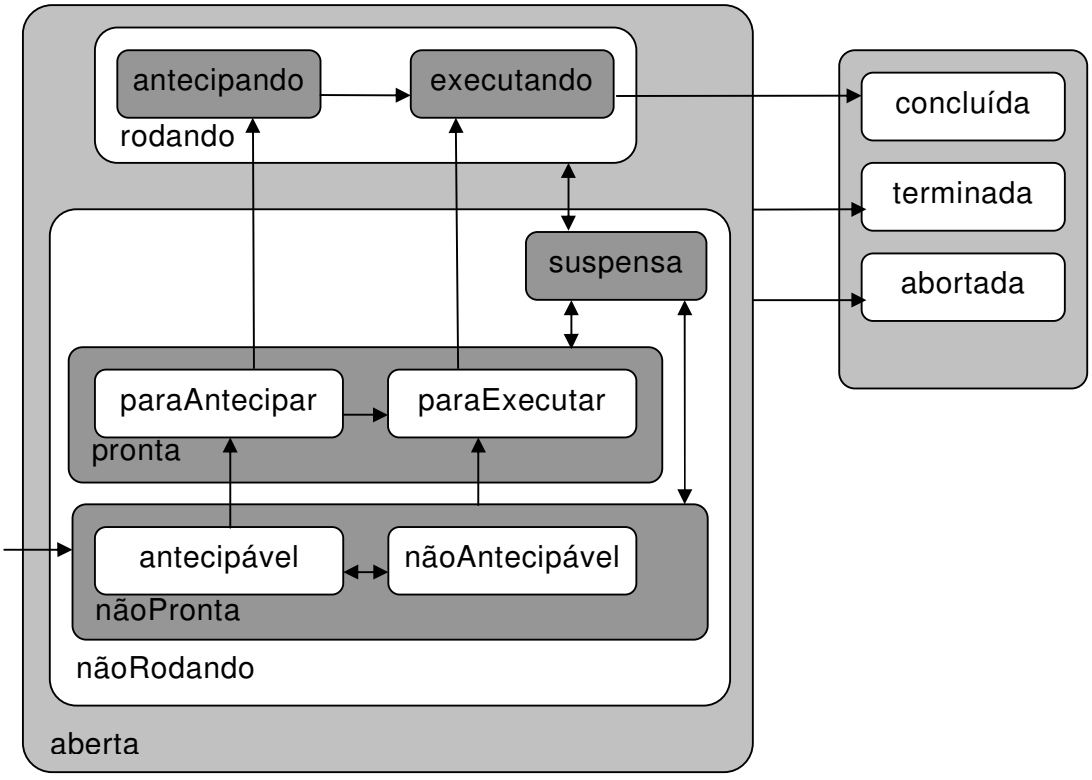

Figura 2. Diagrama de Transição de Estados para as tarefas

A seguir são apresentados os estados propostos e a descrição de cada um:

- `suspensa`: continua com o mesmo significado, indicando que uma instância de tarefa foi parada, porém pode ser retomada a qualquer instante;
- `nãoPronta`: é o estado inicial das instâncias. Indica que a tarefa foi instanciada, porém ainda não tem suas condições satisfeitas para ser disponibilizada para execução;
    - `antecipável`: indica que a instância da tarefa poderá ser antecipada se tiver suas condições de antecipação satisfeitas;
    - `nãoAntecipável`: indica que a instância da tarefa não poderá ser executada antecipadamente;
- `pronta`: a instância teve suas pré-condições (de execução ou antecipação) satisfeitas e está pronta para ser iniciada;
- `paraAntecipar`: indica que a instância da tarefa teve as pré-condições de antecipação satisfeitas e aguarda ser iniciada;
    - `paraExecutar`: a instância pode ter sua execução iniciada, pois suas pré-condições de execução foram satisfeitas por completo;
    - `antecipando`: tarefa está sendo rodada de maneira antecipada;
    - `executando`: tarefa está executando e pode ser concluída normalmente.

Quando se instancia um processo, todas suas tarefas são instanciadas. Quando instanciadas as tarefas são levadas para seu estado inicial. Neste caso o estado inicial das instâncias de tarefa é o estado `nãoPronta`. A instância é então mapeada para um dos dois sub-estados: `antecipável` ou `nãoAntecipável`. Tal classificação é realizada durante a fase de modelagem do processo (como descrito na seção anterior). A transição bidirecional existente entre estes dois estados se deve à possibilidade de alteração desta classificação mesmo durante a execução do processo. Enquanto a instância da tarefa não passou ao estado pronta é possível que o gerente modifique sua condição passando-a de `antecipável` a `nãoAntecipável` e vice-versa. Possibilitando a mudança de estado durante a execução aumenta-se a flexibilidade do sistema, deixando inclusive que agentes solicitem a antecipação de tarefas modeladas como não antecipáveis. Além disso é possível adequar cada instância de um processo para suas necessidades e características próprias apenas alterando o estado de uma tarefa, sem que seja preciso modificar o modelo do processo, ou evoluir o esquema do *workflow*. A complexidade inserida por tal transição é mínima se comparada à complexidade necessária para evoluir o esquema do processo para cada instância em tempo de execução.

A seguir os comportamentos das tarefas antecipáveis e não antecipáveis serão explicados, através do detalhamento das transições entre os estados.

4.1. Tarefas não-antecipáveis

Primeiramente serão atacadas as tarefas que não podem utilizar a antecipação em seu andamento, isto é, as tarefas `naoAntecipaveis`. Uma tarefa `naoAntecipavel` terá seu funcionamento similar ao funcionamento das tarefas em sistemas de *workflow* tradicionais.

Sendo assim, uma tarefa naoAntecipavel se torna pronta.paraExecutar assim que todas suas condições de disparo forem satisfeitas (suas antecessoras forem concluídas e os dados de entrada forem resultados *finais*). A exceção são as tarefas iniciais do processo, que passam ao estado pronta.paraExecutar quando o processo é instanciado. Neste estado a tarefa fica disponível na lista de trabalho (*worklist*) dos agentes capazes de executá-las.

Para que a tarefa passe então do estado pronta.paraExecutar para o estado executando é necessário que esta seja escolhida por algum dos agentes para a qual esta foi oferecida e este comece a executá-la. Ao passar para tal estado a tarefa pode fornecer resultados (*intermediários* e/ou *finais*) e pode ser concluída, quando suas pós condições forem satisfeitas.

Além desta transição, uma tarefa executando pode passar ao estado suspensa a qualquer instante. Este estado se refere às tarefas que foram paralisadas por escolha do agente responsável (ou do gerente), podendo esta tarefa ser retomada a qualquer instante. Quando retomada, a tarefa volta ao estado executando, podendo ser concluída ou suspensa novamente.

4.2. Tarefas antecipáveis

Já uma tarefa classificada como antecipável pode ter seu comportamento modificado justamente pela flexibilidade oferecida pela antecipação. O fato de uma tarefa ser dita antecipável não garante que esta será antecipada, apenas permite que esta possa ser antecipada. É apenas uma alternativa para possibilitar a cooperação entre os agentes (através da troca de resultados *intermediários*) e ajudar na redução do tempo de execução dos processos.

Uma tarefa antecipável pode seguir a trajetória de uma tarefa naoAntecipavel. Isto é, quando suas pré-condições forem satisfeitas passa ao estado pronta.paraExecutar, ficando disponível na lista de tarefas dos agentes capazes de executá-la. Ao ser escolhida por um agente, passa a estar executando, podendo então ser suspensa ou concluída.

O comportamento de uma tarefa antecipável, porém, pode ser totalmente distinto daquele de uma tarefa nãoAntecipável. Quando uma tarefa antecipável tem suas pré condições de antecipação (definidas quando da modelagem) satisfeitas, esta passa ao estado pronta.paraAntecipar. Este estado funciona de maneira similar ao estado pronta.paraExecutar. Quando pronta.paraAntecipar, a tarefa aparece na lista de tarefa dos agentes capazes de executá-la. A partir deste estado é possível que a tarefa se torne executável ou passe a estar antecipando.

A transição entre o estado pronta.paraAntecipar e executável é ativada quando as pré condições de disparo da tarefa são satisfeitas antes desta iniciar sua antecipação. Neste caso, a tarefa volta a seguir o fluxo normal, aparecendo como pronta.paraExecutar na lista de tarefas dos usuários.

Já a transição entre pronta.paraAntecipar e antecipando é ativada quando algum dos agentes capazes de executar a tarefa selecionam-na para antecipação. A partir deste momento a tarefa está antecipando. Neste estado o comportamento é muito parecido com uma tarefa executando, estando a diferença no fato das tarefas não poderem ser concluídas enquanto estão antecipando. Para poderem ser concluída uma tarefa necessita primeiramente passar pelo estado executando. Esta restrição se deve à necessidade de garantir que a tarefa está obedecendo às suas pré-condições de execução (e não de antecipação).

5. Fluxo de Dados

A fim de prover flexibilidade para o fluxo de dados, foram criados também dois estados novos para os elementos de dados: intermediário estável e intermediário instável. Tais estados visam possibilitar a liberação de resultados intermediários, que esteja estáveis ou ainda instáveis. O diagrama de transição de estados para os elementos de dados é apresentado na Figura 3, sendo explicado em seguida. Tais estados visam possibilitar a modificação da condição de disparo das tarefas, com relação ao fluxo de dados.

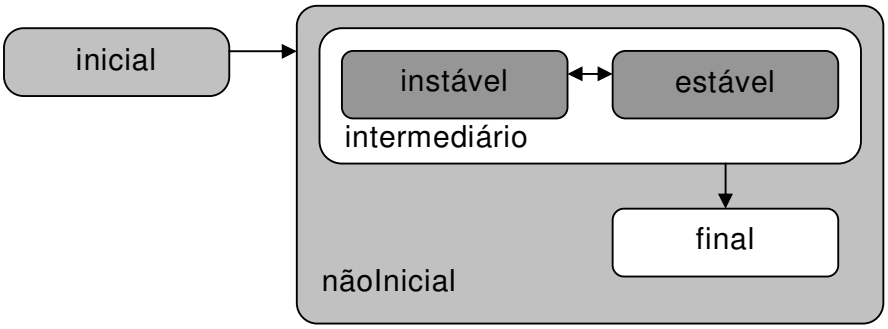

Figura 3. Diagrama de Transição de Estados para os elementos de dados

Na abordagem tradicional, um elemento de dado se torna *inicial* assim que este passa a ser utilizado em uma tarefa. Para que um resultado seja disponibilizado para outras tarefas é necessário que a tarefa seja concluída e que os elementos de dados de saída apresentados sejam resultados finais. Portanto, para que as dependências de dados de entrada de uma tarefa sejam satisfeitas, é necessário que seus dados de entrada sejam resultados finais de outras tarefas.

Na presente proposta, é possível que um elemento de dado entre em um estado `intermediário` antes de ser considerado `final`. Este novo estado visa permitir a habilitação de tarefas sem a necessidade dos dados de entrada serem `finais`. Isto é, quando uma tarefa for antecipável, e existir qualquer resultado não inicial disponível, este pode ser utilizado para satisfazer (parcialmente) as condições de disparo da tarefa. Um dado no estado `intermediário` pode ser classificado ainda como `instável` ou `estável`. Tal classificação se fez necessária para que fosse possível deixar os agentes cientes dos possíveis problemas que podem existir quando do uso destes resultados intermediários. A classificação como `estável` ou `instável` é de responsabilidade do agente que fornece os resultados, sendo de grande importância para o bom andamento do processo.

Para exemplificar a diferença entre um resultado `intermediário instável` e um resultado `intermediário estável` será utilizado novamente o processo de produção de um artigo científico. Um agente pode disponibilizar os rascunhos de toda uma seção para revisão. Desta maneira o documento pode ser editado tanto pelo editor quanto pelo revisor, passando a existir uma espécie de edição cooperativa (resultado `intermediário instável`). Porém, o agente pode liberar uma subseção já numa versão final para revisão, garantindo que esta não mais será alterada, podendo o agente responsável pela revisão realizar sua tarefa sem preocupação com possíveis inconsistências (resultado `intermediário estável`).

Como dito anteriormente, um resultado dito *instável* significa que a parte do documento ou o documento disponibilizado ainda será alterado na tarefa de origem. Este tipo de resultado pode ser usado como entrada das tarefas subseqüentes, porém, existe grande risco de perda de trabalho, ou de necessidade de negociação. Na proposta aqui apresentada está prevista a existência de elementos de dados neste estado, porém não nos preocuparemos em solucionar os problemas relacionadas a este tipo de resultado. No contexto deste trabalho, dados neste estado serão liberados somente para leitura para as outras tarefas.

Um resultado `intermediário estável` é um resultado pronto para ser utilizado como entrada para tarefas. São resultados "prontos" para serem modificados pelos outros agentes em outras tarefas. Quando um agente A disponibiliza um de seus resultados e o classifica como `estável` este "perde" o direito de alterá-lo. Ao utilizar este resultado, um agente B tem plenos direitos sobre ele, podendo lê-lo e alterá-lo como desejar. Qualquer alteração realizada por A neste elemento de dado será desprezada quando outro resultado (`intermediário` ou `final`) for liberado, e houver a união das "versões" dos agentes A e B. No caso de haverem inconsistências entre as duas versões, a versão utilizada (ou alterada) por B será sempre considerada a versão válida.

6. Trabalhos Relacionados

Muita pesquisa tem sido desenvolvida a fim de adicionar maior flexibilidade aos modelos de workflow, tornando possível a sua utilização em um número maior de domínios de aplicação. [Agostini and Michelis, 1996] tratam os modelos de processo como se fossem recursos para o mecanismo de percepção (*awareness*) guiar os participantes de um processo cooperativo. Esta proposta traz grande flexibilidade ao fluxo de controle, porém não leva em conta o fluxo de dados. A presente proposta apresenta um meio de flexibilizar tanto o fluxo de controle quanto o fluxo de dados, através da troca de resultados intermediários.

Uma tentativa de flexibilizar a troca de dados é apresentada em [Hagen and Alonso, 1999]. Para tal é proposto um meio de acabar com o encapsulamento dos dados de um processo, permitindo que uns processos possam ter acesso aos resultados de tarefas de outros processos. Apesar de acabar com o isolamento dos processos, as tarefas continuam executando de maneira isolada e produzindo apenas resultados finais. A quebra de tal isolamento e a possibilidade de troca de resultados intermediários é o foco de nossa proposta.

Uma outra importante abordagem é a proposta do operador *Coo* [Godart *et al*., 1999][Godart *et al.,* 2000]. Este novo operador visa permitir que sejam definidas as tarefas que necessitam trocar resultados intermediários, já na fase de modelagem. A grande desvantagem desta proposta é justamente a criação do novo operador, alterando e aumentando a complexidade do modelo tradicional de definição de processos.

[Joeris, 1999] defende que a definição correta do comportamento das tarefas é a base para modelagem de *workflows* menos restritivos e para suporte a alterações dinâmicas. Neste sentido é apresentada uma maneira de especificar diferentes tipos de dependência no fluxo de controle para cada tarefa, visando a definição de *workflows* menos restritivos. Tais tipos de dependência são definidos basicamente por regras do tipo E-C-A (Evento-Condição-Ação). O mecanismo de disparo trata as tarefas como se estas fossem componentes reativos, que encapsulam seu comportamento interno e interagem com outras tarefas através de passagem de mensagens/eventos. Mais uma vez existe o aumento na complexidade durante a modelagem, sendo necessário criar as novas regras de dependência para cada tarefa, definindo o comportamento desta durante a execução do processo.

No projeto ADEPT [Reichert et al., 2003] [Reichert and Dadam, 2003] existe a proposta de modificar o modelo do *workflow* na instanciação dos processos. Para tal são definidos métodos que

permitem, a omissão, a adição e a troca na ordem de execução das tarefas. Tudo isto aliado a uma série de restrições que garantem que o processo continua consistente. Tal proposta viabiliza a "quebra" de tarefas possibilitando, desta maneira, a antecipação das mesmas. Porém, tal antecipação continua sendo feita de maneira rígida, isolada. O trabalho aqui proposto, tem como objetivo fornecer maneiras de haver a antecipação apenas identificando tal fato quando da instanciação das tarefas, porém, permitindo que tarefas seqüenciais possam trocar resultados que não sejam finais.

A abordagem mais próxima desta proposta, e que tem servido de base para a pesquisa, é o *Coo-flow* [Grigori *et al.*, 2004], uma tecnologia baseada em workflow para coordenar processos com interações criativas, adicionando-se flexibilidade. Em tal abordagem são propostas alterações na máquina de workflow, permitindo o disparo de tarefas antes do cumprimento de suas pré-condições. Apesar de apresentar como prover antecipação, não são criados meios de identificação das tarefas, a fim de classificar aquelas que não podem ser antecipadas.

7. Conclusões e Trabalhos Futuros

Este artigo apresentou uma maneira de trazer flexibilidade a Sistemas Gerenciadores de *Workflow* de forma a permitir que tarefas possam ser antecipadas. A antecipação é provida através da possibilidade de disparar algumas tarefas antes mesmo de suas pré-condições serem satisfeitas. A maneira com que isto é atacado no artigo é guiada por três alterações no modelo tradicional de *workflow*:

(i) a criação de tipos de dependência entre as tarefas, possibilitando escolher o comportamento do fluxo de controle no disparo de cada uma das tarefas;
(ii) a utilização de um diagrama de estados modificado para apoiar tal antecipação;
(iii) uma modificação nos possíveis estados dos elementos de dados envolvidos no modelo, permitindo que dados intermediários sejam utilizados.

Nós acreditamos que tal abordagem pode ser muito útil, pois tende a levar a tecnologia de *workflow* a muitos outros domínios de aplicação onde se faz necessária a interação entre agentes de diferentes tarefas. Além disso, é possível aumentar a eficiência dos processos que envolvem recursos humanos e escrita e revisão de documentos de maneira simples uma vez que o paralelismo existente entre tarefas aumenta visivelmente.

Alguns experimentos estão sendo realizados levando-se em conta a antecipação de tarefas, tentando prever o comportamento humano durante tal processo. Com certeza o uso desta abordagem trará perdas com relação ao esforço de comunicação relativo à troca dos resultados intermediários. O que se pretende ter como resultado de tais experimentos, é a relação existente entre antecipação e perda com comunicação, podendo-se prever os casos em que a antecipação traria ganhos com relação ao tempo de execução dos processos.

Um mecanismo de percepção (*awareness*) está sendo definido para apoiar a antecipação das tarefas, diminuindo as possíveis perdas com a comunicação. A validação desta proposta está prevista através da implementação da mesma junto à máquina de *workflow* [França, 2004] implementada no projeto CEMT [de Lima *et al.,* 2001]. As alterações propostas durante a modelagem do processo, deve ser implementada no ambiente *Amaya Workflow* [Telecken, 2004], através da inclusão de atributos nos elementos tarefa do modelo.

Agradecimentos

O autores deste trabalho são gratos pelo financiamento por parte do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq - Brasil).

Agradecimentos aos integrantes do grupo SIGHA, do qual faz parte o projeto CEMT, pelo apoio, dicas e pelo ambiente de trabalho oferecido. Muito obrigado, em especial, a Tiago Lopes Telecken e Níccholas Rodriguez Vidal.

Referências Bibliográficas

Aalst, W., Hee, K. (2002) *Workflow Management: Models, Methods, and Systems*, The MIT Press, Cambridge.

Agostini, A. and Michelis, G. D. (1996). Modeling the document flow within a cooperative process as a resource for action. Ctl-dsi report, University of Milano.

Canals, G., Godart, C., Molli, P., and Munier, M. (1998). A criterion to enforce correctness of indirectly cooperating applications. In *Information Sciences*, volume 110, pages 279–302.

de Lima, J. V., Quint, V., Laya¨ıda, N., Edelweiss, N., Kirsch-Pinheiro, M., and Telecken, T. (2001). The conception of cooperative environment for editing multimedia documents with workflow technology (cemt). In *Protem-CC- Projects Evaluation Workshop - 2001 International Cooperation NSI/INRIA*, pages 542–560.

França, M. B. (2004). Proposta e implementação de uma máquina de workflow para o projeto CEMT. Master's thesis.

Godart, C., Charoy, F., Perrin, O., and Skaf-Molli, H. (2000). Cooperative workflows to coordinate asynchronous cooperative applications in a simple way. In *7th International Conference on Parallel and Distributed Systems*.

Godart, C., Perrin, O., and Skaf, H. (1999). Coo: A workflow operator to improve cooperation modeling in virtual processes. In *RIDE*, pages 126–131.

Grigori, D. (2001). *Eléments de flexibilitè des systèmes de workflow pour la définition et l'exécution de procédés coopératifs*. PhD thesis.

Grigori, D., Charoy, F., and Godart, C. (2004). Coo-flow: a process technology to support cooperative processes. *International Journal of Software Engineering and Knowledge Engineering*, 14(1):1–19.

Hagen, C. and Alonso, G. (1999). Beyond the black box: Event-based inter-process communication in process support systems. In *International Conference on Distributed Computing Systems*, pages 450–457.

Joeris, G: (1999) Defining Flexible Workflow Execution Behaviors, in *Enterprise-wide and Cross-enterprise Workflow Management - Concepts, Systems, Applications*, GI Workshop Proceedings - Informatik'99, n. 99-07, University of Ulm.

Reichert, M. and Dadam, P. (2003). Adeptflex - supporting dynamic changes of workflows without losing control. In *Journal of Inteligent Information Systems*.

Reichert, M., Rinderle, S., and Dadam, P. (2003). Adept workflow management system: Flexible support for enterprise-wide business processes. In *Proc. International Conference on Business Process Management (BPM '03)*, pages 371–379.

Telecken, T. L. (2004) “Um estudo sobre modelos conceituais para ferramentas de definição de processos de workflow”, Master Thesis in Computer Sciences – Instituto de Informática, UFRGS, Porto Alegre, 2004. p. 123.

WfMC, “Workflow Management Coalition Home Page” [Online]. Disponível em: http://www.wfmc.org. Acesso em Setembro, 2004.