Título: W4MS Um Sistema Distribuído de Gerenciamento de *Workflow* para a WWW

Responsável: Antônio Carlos Fernandes Nunes

E-mail: antguria@gta.ufrj.br

W4MS
Um Sistema Distribuído de Gerenciamento de *Workflow* para a WWW

Adriana Nemer de Oliveira
adriana@gta.ufrj.br

Andrea de C. M. Bahiense
andreamb@gta.ufrj.br

Antônio Carlos F. Nunes
antguria@gta.ufrj.br

Jorge Lopes de Souza Leão
leao@gta.ufrj.br

Grupo de Teleinformática e Automação – GTA
Universidade Federal do Rio de Janeiro
COPPE/EE – Programa de Engenharia Elétrica
C.P. 68504 – CEP 21945-970 – Rio de Janeiro – RJ – Brasil

Resumo

Atualmente, várias empresas têm investido muito na área de compartilhamento de informação, a fim de tornar o trabalho em grupo mais produtivo e mais econômico. Este fato está levando a um interesse cada vez maior na área de CSCW (*Computer Supported Cooperative Work*), ou Trabalho Cooperativo Apoiado por Computador. Uma classe particular de sistemas CSCW são os sistemas de gerenciamento de *workflow*. Este trabalho apresenta o modelo, a arquitetura, a implementação e os primeiros resultados do desenvolvimento de um Sistema Distribuído de Gerenciamento de *Workflow* para a *World Wide Web*.

Abstract

Nowadays, many companies have been spending a lot of money in the area of information sharing, so that people can work in group more efficiently. Actually, this is increasing the interest in the area of CSCW (Computer Supported Cooperative Work). A particular class of CSCW systems is the Workflow Management Systems. This paper presents the model, the architecture, the implementation and the first results of a Distributed Workflow Management System for the World Wide Web.

1. Introdução

O termo CSCW (*Computer Supported Cooperative Work*), ou Trabalho Cooperativo Apoiado por Computador, designa sistemas que fornecem uma área de trabalho compartilhada, permitindo uma integração direta entre os membros de um grupo. Esses sistemas visam aumentar a qualidade e eficiência do trabalho nas empresas, além de reduzir os custos. Eles envolvem tipicamente atividades tais como: escrita e projeto com coautores [1], áudio e vídeo-conferência, preparação e realização de reuniões virtuais e sistemas de suporte à decisão em grupo (esta englobando duas ou mais das aplicações anteriores).

Uma classe particular de sistemas de CSCW, que apoia processos organizacionais, é conhecida como *workflow* ou fluxo de trabalho. Define-se processo organizacional como aquele que envolve a execução coordenada de múltiplas atividades executadas, sequencial ou concorrentemente, por várias entidades de processamento, humanas e automáticas. Um processo de *workflow* é um processo organizacional cujo encadeamento das atividades é automatizado. Esta automação é normalmente baseada na manipulação de documentos informatizados.

Como a divisão do conjunto de todas as atividades de uma instituição em processos costuma seguir a própria divisão organizacional da instituição, um processo de *workflow* pode ser composto por sub-processos que serão tratados por diferentes seções ou departamentos. É o que chamamos de distribuição dos processos de *workflow*. Isto significa que um processo de *workflow* pode ser implementado em sistemas de gerenciamento de *workflow* fisicamente distribuídos e que se comunicam.

A literatura tanto acadêmica como comercial freqüentemente distingue três tipos de *workflow* [2]:

- ***workflow ad hoc*** tipicamente envolve coordenação, colaboração ou co-decisão humana [3]. A ordenação e a coordenação de tarefas em um *workflow ad hoc* não são automatizadas, mas sim controladas por humanos. Além disso, não existe uma estrutura pré-definida para o processo, sendo a lógica do processo definida pelo usuário, em tempo de execução. Este tipo de *workflow* envolve pequenos grupos de profissionais e tem a intenção de dar apoio a pequenas atividades que requerem uma solução rápida.
- ***workflow* administrativo** envolve processos repetitivos e previsíveis com regras simples de coordenação de tarefas. Tanto a ordenação como a coordenação de tarefas em um *workflow* administrativo podem ser automatizadas. Este tipo de *workflow* não engloba um processamento complexo de informações e não requer acesso a múltiplos sistemas de informação. Sistemas de *workflow* administrativo são baseados tipicamente em correio eletrônico.
- ***workflow* de produção** também envolve processos repetitivos e previsíveis, mas estes são tipicamente processos de informação complexos envolvendo acesso a múltiplos sistemas de informação. Os sistemas de gerenciamento de *workflow* que suportam *workflow* de produção devem prover facilidades para definir dependências entre tarefas e controlar a execução de tarefas com pouca ou sem nenhuma intervenção humana.

O sistema que permite a execução desses diferentes tipos de *workflow* é chamado de sistema de gerenciamento de *workflow* ou simplesmente sistema de WF. Na verdade, um sistema de gerenciamento de WF é o *software* responsável pelo suporte computacional à automação de processos. É ele que fornece a estrutura necessária para a definição, a execução e monitoração de processos de WF [4], sendo capaz de interpretar a definição do processo, interagir com os participantes do WF e, quando necessário, invocar ferramentas e aplicativos de sistemas de informação.

As empresas que desenvolvem sistemas de gerenciamento de *workflow* observaram algumas características comuns em seus produtos, que possibilitariam alguma forma de interoperabilidade. Isto levou ao desenvolvimento de padrões e, finalmente à criação da WfMC (*Workflow Management Coalition*)[4].

A WfMC é uma organização formada por um grupo de empresas e instituições acadêmicas, criada para definir um Modelo de Referência para conectividade, interoperabilidade e terminologia de produtos de WF. O Modelo de Referência identifica 5 interfaces com o servidor de WF, que é o responsável pelo comando, comunicação e controle das atividades que compõem um processo de *workflow*. Estas interfaces são usadas para suportar a integração do servidor com os outros componentes do sistema de WF e a interoperabilidade entre sistemas diferentes. Esta arquitetura pode ser visualizada na Figura1.

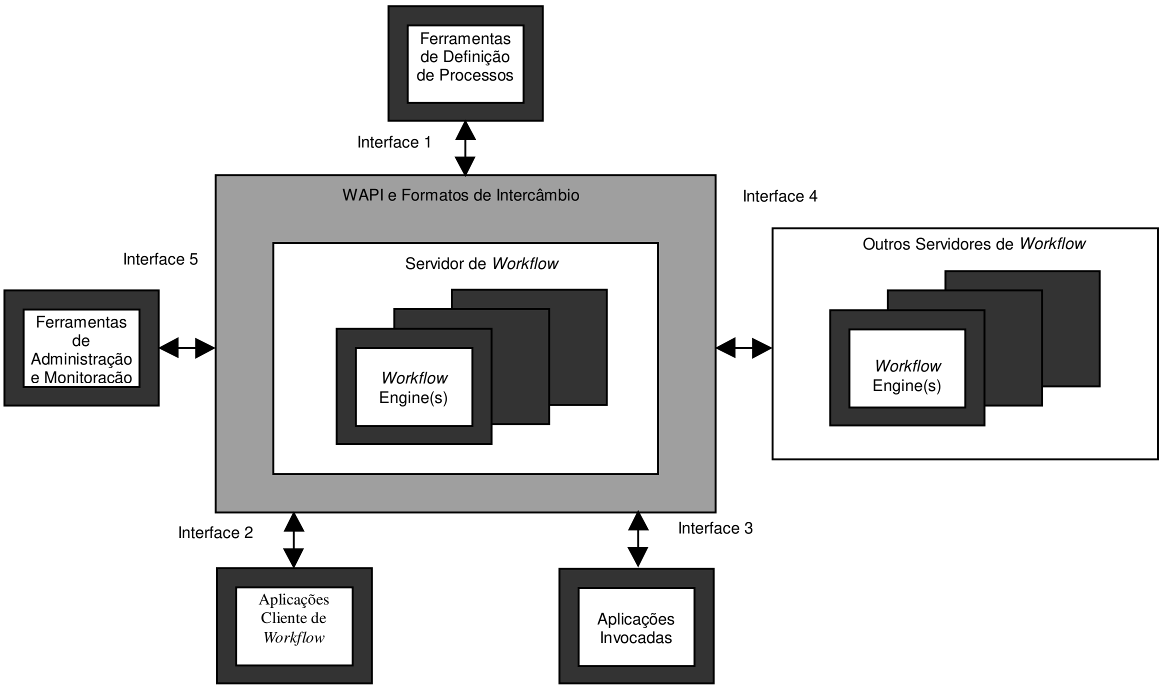

Figura 1 – Arquitetura do sistema de gerenciamento de *workflow* da WfMC.

A interface 1 define a comunicação entre as ferramentas de definição do processo e o servidor de WF. Ela é chamada de interface de importação/exportação das definições de processo de *workflow* e é composta por um formato comum de intercâmbio ou uma API, que permite que a definição de um processo gerada por uma ferramenta de modelagem possa ser usada por diferentes produtos de *workflow*.

A interface entre as aplicações cliente e o servidor de *workflow* é denominada interface 2. Para esta interface, a WfMC define um conjunto de APIs (a WAPI - *Workflow Application Programming Interface*), as quais podem ser utilizadas por uma aplicação de WF para acessar o servidor e as listas de tarefas (*worklists*), independentemente da natureza de implementação do produto. Para isto são providos comandos que controlam as instâncias de processos, assim como manipulam as listas de tarefas.

A interface 3 define a comunicação entre o servidor de *workflow* e uma larga faixa de aplicações de ambientes heterogêneos que devem ser integradas ao sistema de *workflow*. Além disso, aplicações desenvolvidas especificamente para *workflow* também são invocadas através desta interface.

A interoperabilidade entre sistemas de *workflow* heterogêneos é provida através da interface 4, a qual provê a habilidade de se transferir partes de um processo para execução em um outro servidor de *workflow*.

Já a interface 5 é responsável pela interação com as ferramentas de administração. Ela propõe funções de monitoramento e administração permitindo que a aplicação de gerenciamento de determinado fabricante possa ser utilizada em outros servidores de *workflow*.

Para o desenvolvimento de um sistema de gerenciamento de *workflow* para a WWW (*World Wide Web*), foi criada uma arquitetura diferente da WfMC, a fim de adaptá-la ao uso das tecnologias da *Web*. Neste trabalho serão descritas a arquitetura, a implementação e os primeiros resultados do desenvolvimento do W4MS (*World Wide Web Workflow Management System*), um sistema distribuído de gerenciamento de *workflow* para WWW. A seção 2 mostra como foi desenvolvido o projeto, as tecnologias escolhidas para serem utilizadas e a razão delas. A seção 3 define modelagem de processos em geral e como um processo é modelado no W4MS. A seção 4 apresenta a arquitetura de implementação do projeto e o seu funcionamento. Finalmente, nas seções 5 e 6 são apresentados os resultados obtidos e as principais conclusões do projeto.

2. Visão Geral do Projeto

O passo inicial do projeto foi a criação de uma linguagem de definição de processos. Esta linguagem possui uma posição central neste trabalho, pois ela tanto fornece modelos para processos do mundo real, como um modelo para a própria implementação do sistema de gerenciamento de *workflow*.

Na fase de implementação, decidiu-se pela utilização de *FreeSoftware* para os componentes disponíveis e de padrões consagrados na comunidade de *FreeSoftware* para os componentes a serem desenvolvidos. Assim sendo, o sistema utiliza um PC Intel como servidor rodando Unix (FreeBSD ou Linux), um servidor HTTP (Apache), um sistema gerenciador de bases de dados (Postgres95 [5]) e a linguagem C++ (GNU) para o desenvolvimento de programas.

O acesso dos participantes ao sistema de gerenciamento é feito exclusivamente através de *Web browsers*, devido a algumas vantagens das tecnologias da WWW (*World Wide Web*). Estas vantagens são:

- Independência de plataforma [6], já que praticamente todas as plataformas atuais possuem *Web browsers*;
- Interface com o usuário bastante difundida, o que dispensa o custo de treinamento dos usuários;
- Dispensa a necessidade de redistribuição do programa cliente a cada modificação do sistema;
- Utiliza as tecnologias de rede da Internet, permitindo que o sistema seja utilizado em Intranets, Extranets ou na própria Internet.

3. Modelagem de Processos

3.1 O processo de modelagem

A modelagem de processos é o ponto de partida para qualquer organização que esteja implantando um sistema de gerenciamento de *workflow*. Ela é utilizada tanto na criação de processos completamente novos, como na otimização ou automação de processos já existentes. É na fase de modelagem, que processos do mundo real são transformados em diagramas, gráficos e textos para, mais tarde, serem executados pelo servidor de *workflow*.

O primeiro passo para a modelagem de um processo é a divisão das tarefas em dois tipos: aquelas que precisam da intervenção humana para serem realizadas e aquelas que são realizadas automaticamente pelo sistema. O passo seguinte é, então, definir o encadeamento entre estas tarefas, especificando-se a sequência de execução das mesmas, as aplicações que serão invocadas, os participantes envolvidos, entre outras informações relevantes ao processo.

3.2 A linguagem LDP

Uma variedade de ferramentas pode ser usada para analisar, modelar, descrever e documentar um processo de *workflow*. Considerando a simplicidade de uma linguagem textual, criou-se uma Linguagem de Definição de Processos – LDP, para a qual foi definida uma interpretação que associa a cada descrição de processo uma Rede de Petri Estendida - RPE constituída de:

- Lugares, chamados aqui de bandejas ou *bins*;
- Transições, chamadas aqui de atividades ou *activities* e
- Arcos direcionados simples, que podem ser de dois tipos: os transportadores de fichas e os inibidores.

A esta RPE associa-se um conjunto de objetos chamados *memos*, que devem residir em uma memória não-volátil. Estes objetos possuem uma referência a um documento e a uma aplicação que deverá ser usada para abrí-lo. Eles possuem um formato fixo com os seguintes componentes:

- *ID*, identificador numérico do próprio *memo*;
- *Instance*, identificador da instância do processo que manipula o *memo*.
- *Status*, que pode assumir os valores: *Processing*, *Done*, *InWL*, *TimedOut*, *Failed*;
- *Bin*, identificador de um *bin*, onde podem ser armazenadas referências aos *memos*;
- *WL*, lista de participantes, isto é das *worklists* correspondentes nas quais existem referências a este *memo*, como será visto mais à frente;
- *Doc*, contém a localização completa de um documento;
- *Aplic*, identificador de uma aplicação registrada que deve ser usada para abrir e processar o documento;
- *FirstDate*, *LastUpDate*, *ExpirationDate*, datas relativas à criação, atualização e expiração em uma *worklist* de um *memo*, respectivamente;

A cada *memo* cujo *Status* é diferente de *Processing* corresponde um único *bin*. Como cada *bin* pode estar associado a vários *memo*s, podemos dizer que há uma relação *1:n* de *bin*s x *memo*s. Esta relação, na verdade, é a marcação da Rede de Petri, onde os *memo*s podem ser vistos como fichas coloridas, já que eles possuem uma identidade que os diferencia uns dos outros.

A cada *memo* que pertencer à relação *bins* x *memos* pode corresponder um ou mais participantes. Cada conjunto de elementos da relação *memo*s x participantes que possua o mesmo participante é chamado de *worklist* deste participante. Isto pode ser melhor compreendido se pensarmos que uma *worklist* é uma lista de tarefas designadas a um determinado participante, sendo que para cada tarefa existe um documento e uma aplicação apropriada a ele. Além disso, cada tarefa, representada por um *memo* na *worklist*, pode ser executada por mais de um participante. Diz-se, então, que existe uma relação *n:m* de *memo*s x participantes.

Além dos *memo*s e das *worklists*, associa-se também à RPE um conjunto de variáveis que podem ser de três tipos: inteiro (*int*), cadeia de caracteres (*string*) e *memo*. Os dois primeiros tipos correspondem a tipos usuais nas linguagens de programação, enquanto que uma variável do tipo *memo* é uma referência a um objeto *memo*. Como será visto adiante, o disparo de uma transição pode ser visto como uma transação. A variável *memo* será implementada como um *cache*, em memória temporária, do *memo*, que reside na memória estável.

Esta RPE evolui como uma rede de Petri condição-ação. Uma atividade está habilitada se uma expressão, chamada VARCOND, envolvendo variáveis *int* e *string* for TRUE e se cada *bin* ligado a ela por um arco (transportador de fichas) de entrada possuir pelo menos um *memo* que atenda às condições da cláusula *where*. Ao ser disparada, uma atividade retira os *memos* dos *bins* de entrada e executa uma série de comandos definidos na sessão ACTION do programa. Estes comandos podem criar e destruir *memo*s, enviar *memos* para *bin*s de saída, mudar campos de um *memo* e invocar aplicações. Estas aplicações incluem por exemplo, o acesso ao sistema de *e-mail* da Internet (*sendmail*). Nesta sessão também é possível usar os comandos IF e ELSE, como em Pascal.

3.3 Um pequeno exemplo

O trecho de descrição mostrado a seguir ilustra alguns desses pontos.

```
process Exemplo1 is

var
 int i, j, k;
 string Linha, Campo, Temp;
 memo m1, m2, m3;

applications
   Appl1, Appl2, Appl3;

activities
activity A1 is

 varcond
   i>3;
   j=k;

 memocond
   Bin1 =? m1 where ((WL='contador') and
   (Status='Done'));

 action
   create m1;
   m1.Status=inWL;
   m1.ExpirationDate=DateTimeNow+TimeInHours (24);
   m1.WL='contador';
   Bin2 =! m1;

end activity A1;
end process Exemplo1;
```

Como pode ser visto no exemplo descrito acima e esquematizado na Figura 2, um programa inicia e termina com o nome do processo cujas atividades serão descritas. Inicialmente, são declaradas todas as variáveis utilizadas no processo, além das aplicações que serão invocadas. Em seguida são definidas as atividades.

A descrição de cada atividade é composta das seções VARCOND, MEMOCOND e ACTION, todas opcionais. Neste exemplo, a atividade A1 poderá ser disparada se `((i>3) and (j=k))` e se no *Bin1* houver um *memo* cujos campos *WL* e *Status* sejam respectivamente ‘contador’ e ‘Done’.

O símbolo “=?” indica que o *memo* será retirado do *Bin1*. Já o símbolo “=!” indica que o *memo* referenciado pela variável *m1* será colocado no *Bin2*.

Ao ser disparada a atividade, os campos *Status*, *ExpirationDate* e *WL* do *memo* serão modificados e o *memo* é colocado no *Bin2*.

Figura 2 – Exemplo de um trecho de um processo de *workflow*.

A única forma de interação de um participante com a execução de uma descrição de processo de *workflow* é feita pela seleção de um *memo* na sua *worklist* e pela abertura do documento correspondente. A execução da aplicação correspondente tipicamente muda o estado do *memo* de *InWL* para *Done*.

4. Implementação do Sistema Distribuído de Gerenciamento W4

Para implementar o Sistema Distribuído de Gerenciamento W4, desenvolveu-se uma arquitetura para um Módulo Servidor W4. Este módulo é acessado pelos participantes através de *Web browsers*, que obviamente podem estar sendo executados em estações distantes. Por outro lado, um sistema pode ser composto por vários Módulos Servidores sendo executados em estações distribuídas e se comunicando através da interface *Peer*.

A arquitetura de um Módulo Servidor é constituída por componentes sobre um sistema operacional Unix. Alguns destes componentes são *FreeSoftware*, como o servidor HTTP (Apache) e o sistema gerenciador de bases de dados Postgres95, enquanto outros, como as Aplicações de Interface, as Aplicações Internas e o Motor W4, foram desenvolvidos e codificados em C++.

A arquitetura de um Módulo Servidor é mostrada na Figura 3.

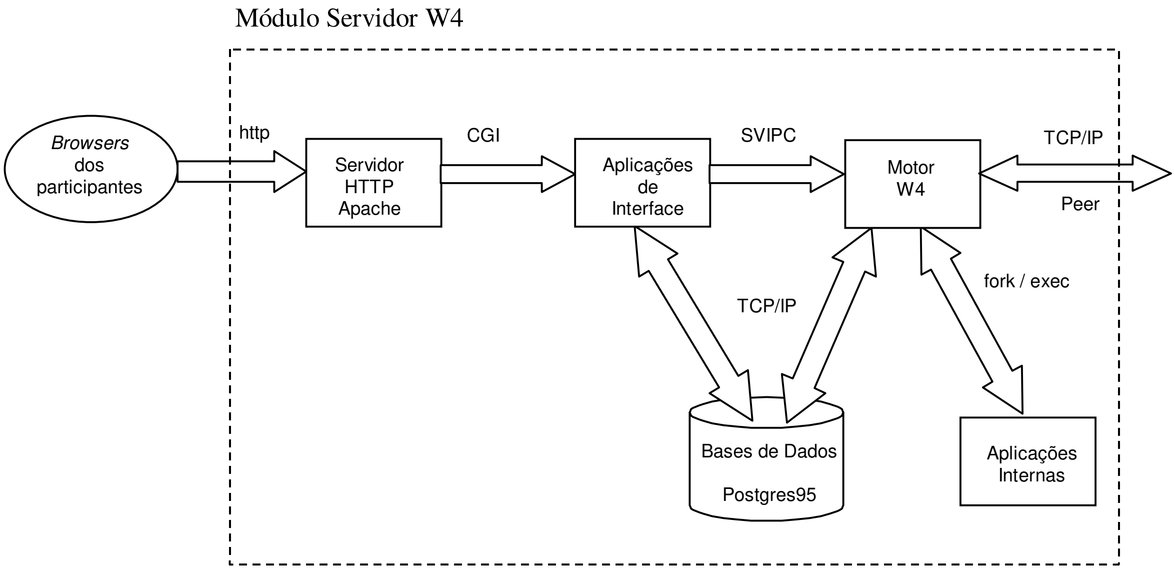

Figura 3 – Arquitetura do Sistema Distribuído de Gerenciamento de *Worklfow* W4MS.

Como pode ser visto nesta figura, os participantes acessam através de *browsers* o servidor HTTP, que serve um conjunto de páginas HTML. Estas páginas possuem normalmente *links* CGI para invocar as Aplicações de Interface e podem ainda usar outros recursos como Java*Script*, *applets* Java, etc.

As Aplicações de Interface, por sua vez, comunicam-se com o gerenciador de bases de dados Postgres95 através de soquetes TCP/IP. Estas aplicações são as seguintes:

- Ferramenta de Administração
  - Registrar ou eliminar um participante;
  - Registrar ou eliminar uma aplicação usada em *memos*.
- Ferramenta do Participante
  - Apresentar a *Worklist* de um participante;
- Ferramenta de Autoria
  - Editar, validar, registrar ou eliminar um modelo de processo.
- Ferramenta de Controle
  - Suspender ou retomar a interpretação de uma instância;
  - Abortar a interpretação de uma instância.

As bases de dados principais de um Sistema Distribuído de Gerenciamento W4 são:

- *Participant* – armazena o nome, a senha e outras informações relevantes sobre um participante do sistema;
- *Memo* – armazena os *memos* do sistema;
- *Assoc* – informações sobre as Aplicações de Interface e suas localizações (*paths*) correspondentes;
- *Worklist* – possui uma lista de referências a *memos* para cada participante;
- *Partrole* – papéis associados a cada participante;
- *Process Models* – guarda os modelos de processos;
- *Instances* – guarda as instâncias dos processos.

Um componente central de um Módulo Servidor W4 é o Motor. Ele recebe comandos das Aplicações de Interface, possivelmente de várias delas, já que podem haver vários participantes acessando o servidor HTTP ao mesmo tempo. Estas aplicações enviam os comandos através uma fila de mensagens do mecanismo SVIPC (*System V Interprocess Communication*) [8]. Desta forma, as mensagens são serializadas e processadas seqüencialmente, simplificando o problema de manutenção da integridade dos dados que vão ser manipulados pelo Motor W4.

Dos vários comandos existentes, 3 deles podem ser considerados básicos. O primeiro deles é o que determina a criação de uma nova instância de um processo já modelado e a sua interpretação. Os dois outros são: o que permite recuperar a *worklist* de um participante e o que permite a execução de um item de uma *worklist*.

A interpretação de uma instância é feita por um componente do Motor W4, o interpretador LDP, que carrega um arquivo pré-compilado de uma definição de processo feita em LDP. Esta interpretação é feita segundo a semântica de programas LDP apresentada anteriormente (RPE).

Os dois outros comandos básicos não são processados pelo interpretador LDP mas, como a execução destes implica no acesso a estruturas de dados que também são acessadas pelo interpretador, o processamento sequencial dos comandos pelo Motor W4 permite que se controle a integridade destes dados.

O interpretador LDP garante que as atividades de um mesmo processo sejam interpretadas de forma atômica. Como não existe concorrência na interpretação destas atividades, é simples construir um interpretador que garanta as propriedades de consistência e isolação. Esta interpretação eventualmente realiza acessos a dados em memória não-volátil (arquivos em disco) e estável, permitindo garantir a propriedade de durabilidade. Desta forma, a interpretação de uma atividade pode ser vista como uma transação (ACID) [9][10].

Por outro lado, a interpretação de uma atividade pode invocar a execução de uma Aplicação Interna. Neste caso, o interpretador LDP cria um processo filho para executar a aplicação e suspende a interpretação desta instância de processo de *workflow*. O interpretador pode então prosseguir com a interpretação das outras instâncias de processos de *workflow* com um impacto menor sobre o tempo de resposta para estas outras instâncias. Em relação às Aplicações Internas, o interpretador LDP pode ser visto como um servidor concorrente e, como é usual, o número máximo de processos filhos é limitado.

Além dos componentes descritos anteriormente, existe uma interface baseada em soquetes TCP/IP que permite a comunicação entre Motores W4 de Módulos Servidores distintos. Assim sendo, o projeto de um sistema complexo e distribuído de gerenciamento de *workflow* deve começar pela concepção e modelagem dos processos que serão alocados aos vários Módulos Servidores. Neste caso, um processo será composto por um processo inicial e vários sub-processos. O processo inicial é alocado a um Módulo Servidor que, no momento apropriado, ativa a criação de um sub-processo em outro Módulo Servidor através da interface *Peer*. Este procedimento pode ser repetido, criando assim vários sub-processos distribuídos em várias estações.

A Figura 4 apresenta uma configuração com várias conexões entre Módulos Servidores.

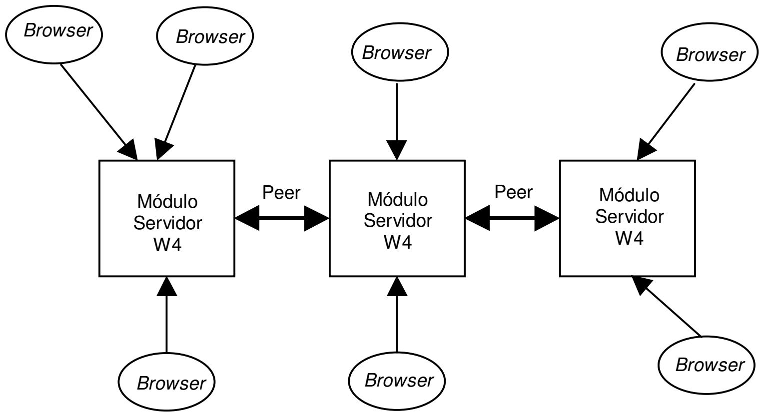

Figura 4 – Configuração do W4MS com vários Módulos Servidores.

5. Resultados

No estágio atual de desenvolvimento do Sistema de Gerenciamento W4, vários componentes já foram ou estão sendo desenvolvidos. Em particular podemos citar que um analisador léxico e sintático para a linguagem LDP está em testes. Uma ferramenta de análise que verifica a criação, destruição e manipulação dos *memos* nas atividades está sendo projetada. Toda a estrutura básica de acesso através da *Web*, aplicações de interface acessadas por CGI, a comunicação com o Motor W4 através SVIPC e os tipos básicos de documentos (formulários HTML) já foram implementados e testados. Uma primeira versão do Motor W4 está sendo desenvolvida.

Para testar tanto a concepção como a implementação do sistema, alguns processos administrativos da própria universidade foram escolhidos para serem usados como "veículos de teste".

O primeiro deles é o processo que leva à defesa de uma tese de mestrado na COPPE/UFRJ. O processo inclui desde a escolha dos membros de uma banca, sua aprovação nos colegiados apropriados, a marcação de uma data e de um horário, de uma sala, a compra e reserva de passagens para membros externos, o depósito de um exemplar para exame público com a antecedência requerida e a divulgação apropriada.

O segundo processo que está sendo modelado é um processo de solicitação e acompanhamento de serviços gerais, tais como limpeza, carpintaria, eletricista e outros, à Seção de Manutenção da COPPE/UFRJ.

6. Conclusões

A escolha do acesso ao sistema por meio de *Web browsers* mostrou-se muito apropriada, uma vez que estes são largamente difundidos e dispensam custos com treinamento, além de serem independentes da plataforma utilizada.

O uso de *FreeSoftware* contribuiu de forma positiva no desenvolvimento do sistema devido à facilidade de acesso à documentação e por serem bastante conhecidos na comunidade acadêmica. Tem-se como objetivo disponibilizar posteriormente o Sistema W4MS como *FreeSoftware*.

A característica distribuída da arquitetura possibilita a divisão de processos em sub-processos que podem ser executados por diferentes Módulos Servidores fisicamente distantes dentro de uma organização. Esta característica foi bastante apreciada na modelagem do processo que leva à defesa de uma tese, pois existe um subprocesso de aprovação de banca que é administrativa e fisicamente independente do processo principal.

Como desenvolvimentos futuros, já estão sendo pensados mecanismos de tolerância a falhas e processos para a recuperação de erros nos casos de transações abortadas e o uso de um subsistema de *news* (*usenet*) para aumentar os recursos de trabalho em grupo.

.

7. Agradecimentos

Os autores Antônio Carlos Fernandes Nunes e Adriana Nemer de Oliveira agradecem a Capes e a autora Andrea de Castro Martins Bahiense agradece ao Cnpq pelo apoio concedido com bolsas de pesquisa.

8. Bibliografia

[1] PALMER, J.D., FIELDS, N.A., BROUSE, P.L., "Multigroup Decision-Support Systems in CSCW", IEEE Computer, pp.67-72, maio de 1994.

[2] GEORGAKOPOULOS, D., HORNICK, M. E SHETH, A., "An Overview of *Workflow* Management: From Process Modeling to *Workflow* Automation Infrastructure", Distributed and Parallel Databases, Vol.3, n.2, pp.119-153, abril de 1995.

[3] SCHAEL, T., ZELLER, B., *Design Principles for Cooperative Office Support Systems in Distributed Process Management*, in Support Functionality in the Office Environment, A. Verrijn-Stuart (ed), North Holland, 1991.

[4] HOLLINGSWORTH, D., "The *Workflow* Reference Model", Technical Report TC00-1003, Issue1.1, The *Workflow* Management Coalition, Brussels, Belgium, novembro de 1994.

[5] YU, A., CHEN, J., *The POSTGRES95 User Manual*, Version 1.0, Computer Science Div., Dept. of EECS, University of California at Berkeley, setembro de 1995.

[6] MILLER, J.A., PALANISWAMI, D., SHETH, A.P., KOCHUT, K.J., "The Future of *Web*-Based *Workflows*", Proc. of the International Workshop on Research Directions in Process Technology, Nancy, França, julho de 1997.

[7] THE *WORKFLOW* MANAGEMENT COALITION, "Interface 1: Process Definition Interchange", Technical Report TC-1016, Brussels, Belgium, maio de 1996.

[8] VAHALIA, U., *UNIX Internals: The New Frontiers*, 1st ed, Upper Saddle River, New Jersey, USA, Prentice-Hall, 1996.

[9] GRAY, J., REUTER, A., *Transaction Processing: Concepts and Techniques*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[10] KRISHNAKUMAR, N., SHETH, A., "Managing Heterogeneous Multi-System Tasks to Support Enterprise-Wide Operations", Distributed and Parallel Databases, 3(2):155-186, abril de 1995.