UM BENCHMARK PARA SISTEMAS DE GERENCIAMENTO DE WORKFLOW

Fagundes Pereira da Silva

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA

Aprovada por:

Prof. Jorge Lopes de Souza Leão, Dr.Ing.

Prof. Marcos Roberto da Silva Borges, Ph.D.

Prof. Aloysio de Castro Pinto Pedroza, Dr.

Prof. Renata Mendes de Araujo, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2004

SILVA, FAGUNDES PEREIRA DA

Um Benchmark para Sistemas de Gerenciamento de Workflow [Rio de Janeiro] 2004

XII, 79 p., 29,7 cm, (COPPE/UFRJ, M.Sc., Engenharia Elétrica, 2004)

Tese – Universidade Federal do Rio de Janeiro, COPPE

1 – Desempenho

2 – Workflow

3 – Benchmark

4 – SGFT

I. COPPE/UFRJ II. Título (série)

Dedicatória:

Esta Tese é dedicada ao meu filho Gabriel Moreno e aos meus pais Sebastião Agostinho da Silva e Jacira Pereira da Silva, que imersos na sua singeleza sertaneja e privados do acesso aos meios formais de educação foram meus primeiros Mestres, tornando-me possível o acesso ao conhecimento.

Agradecimentos:

Agradecimento especial e cheio de carinho a minha companheira Mônica de Rezende, minha Moma, cúmplice e testemunha.

Ao meu filho Gabriel Moreno e meus enteados Dudu e Felipe. Aos meus queridos irmãos Quinho, Jane e Déa. Ao amigo João Diógenes, irmão escolhido e companheiro de muitos carnavais. A Nivaldo, amigo, conselheiro e compadre.

Aos amigos de vida e de sambas na Lapa José Valentim e sua companheira Alê.

Ao caríssimo Álvaro Ferreira, grande oráculo e biblioteca ambulante, e sua companheira Josi.

Aos amigos do CHORD/NCE: Vivi, Michel, Igor, Naiana, Rosa, Hadeliane, Débora e Mauro.

Aos amigos e Professores do GTA/Coppe, em especial Kleber, Gardel, Saulo, Pedro, Roman e Márcio.

A todos que fizeram parte da vida na Comuna Laranjeiras Village: Mó, Lene e Bido.

Aos amigos, baianos perdidos e apaixonados pela Academia: Clodoaldo Paixão, Gildásio Santana e Francisco Cardoso.

Ao Professor Jorge Leão pela orientação e paciência quase infinita.

Ao Professor Marcos Borges pela participação decisiva e construtiva ao longo deste processo.

Aos Professores Aloysio Pedroza e Renata Araujo por aceitarem o convite para participação na Banca.

Aos funcionários do PEE, Coppe e NCE: Solange, Amanda, Nildo e Seu Didi. O trabalho, dedicação e simpatia de vocês enobrecem o Serviço Público.

Aos Professores Sandra Mariano, Frederico Pontes, Raquel Gomes e Rodolfo Badin pelas oportunidades e pela condescendência durante os momentos críticos.

Ao povo brasileiro, que financiou parte deste trabalho através do CNPQ.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM BENCHMARK PARA SISTEMAS DE GERENCIAMENTO DE WORKFLOW

Fagundes Pereira da Silva

Março/2004

Orientador : Jorge Lopes de Souza Leão

Programa: Engenharia Elétrica

A automação dos processos de negócio é um objetivo das organizações que desejam ser eficientes e competitivas. A utilização de Sistemas de Gerenciamento de Fluxo de Trabalho (SGFT) possibilita a automação dos processos e permite uma efetiva coordenação da execução das atividades que compõem o negócio. Um SGFT é uma ferramenta bastante complexa, o que dificulta o estabelecimento de uma avaliação mais simples. Um *benchmark* estabelece um conjunto de critérios para a avaliação de desempenho, cujas métricas devem ter isenção e relevância. Esta tese propõe a especificação de um *benchmark* para avaliação do desempenho de Sistemas de Gerenciamento de Fluxo de Trabalho. O *benchmark* proposto é utilizado na avaliação de uma plataforma proprietária denominada *IBM MQSeries Workflow*. Os detalhes desta implementação, juntamente com os resultados obtidos, são descritos e a importância deste *benchmark* é discutida.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A BENCHMARK FOR WORKFLOW MANAGEMENT SYSTEMS

Fagundes Pereira da Silva

March/2004

Advisor : Jorge Lopes de Souza Leão

Department: Electrical Engineering

Business process automation is an aim for all organizations trying to be efficient and competitive. Workflow Management Systems (WfMSs) provide for process automation and effective coordination of business task executions. A WfMS is a complex tool, making it hard to establish simple evaluation criteria or procedures. A benchmark describes a set of criteria for performance evaluation, whith metrics characterized by impartiality and relevance. This work proposes a benchmark specification for WfMS performance evaluation. The proposed benchmark is applied to the IBM MQSeries Workflow evaluation. Implementation details and results are described, and the relevance of the proposed benchmark is discussed.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Apresentação

Um Sistema de Gerenciamento de Fluxos de Trabalho (SGFT) é uma ferramenta que tem o propósito de automatizar o controle dos processos de negócios através das especificações computacionalmente tratáveis de modelos de fluxos de trabalho (*work-flows*[1]), promovendo a coordenação e o acompanhamento das execuções das atividades ao longo da encenação dos processos.

Apesar do grande número de implementações de SGFTs, não há clareza na definição de um conjunto de critérios comparativos que auxiliem a discussão sobre desempenho de tais ferramentas. Logo, supondo que uma organização decida pela aquisição (compra) ou mesmo pela implementação de um SGFT, haverá dificuldades na avaliação deste sistema. Os parâmetros que poderão auxiliar a definir a capacidade de processamento de um determinado SGFT não são muito precisos e isentos, o que dificulta a comparação entre ferramentas de diferentes fabricantes.

Os primeiros SGFTs foram construídos há mais de trinta anos. Novas tecnologias relacionadas à interconexão de computadores, o desenvolvimento de linguagens indepen-

[1]Na literatura, o termo *workflow* é utilizado para abreviar dois conceitos. Pode ser utilizado como abreviatura para um modelo de fluxo de trabalho ou como abreviatura para Sistemas de Gerenciamento de Fluxos de Trabalho.

dente de plataformas e a interoperabilidade fornecida por XML foram incorporadas aos SGFTs. A avaliação do desempenho de um sistema com este nível de complexidade necessita ser bastante criteriosa, de forma a evitar testes que não exercitam corretamente os elementos sob avaliação e a interpretação incorreta dos resultados encontrados.

1.2 Problema

Nos últimos anos, as pesquisas sobre fluxos de trabalho têm se concentrado em torno de discussões sobre questões como modelagem de fluxo de trabalho, arquiteturas de SGFTs, tratamento de exceções e padrões de controles de roteamento entre atividades. Entretanto, pesquisas acerca de desempenho e capacidade de processamento de um SGFT não têm sido muito exploradas.

SGFTs são sistemas complexos que podem envolver servidores de http, servidores de bancos de dados, mecanismos de comunicação entre processos, mecanismos de controle de transações, interfaces gráficas com o usuário e outras famílias de ferramentas. Uma mera avaliação de apenas um dos elementos que constitui um SGFT levará certamente a uma interpretação errônea do comportamento da ferramenta.

Alguns fabricantes de SGFT fornecem relatórios acerca do desempenho de suas ferramentas. Devido às peculiaridades de cada abordagem, tem-se uma gama de medidas, configurações e padrões de teste que não estão correlacionados com outros SGFTs. Não havendo um mapeamento explícito entre os testes, torna-se difícil realizar uma comparação. Os elementos da arquitetura que são sobrecarregados ao longo destes testes podem não caracterizar o desempenho de um SGFT, o que compromete a credibilidade das métricas apresentadas e a forma como as medidas como são levantadas.

A automação de processos de negócios ainda é um assunto restrito a uma pequena comunidade no universo acadêmico e empresarial. Uma discussão acerca dos limites de utilização de um SGFT traz esclarecimentos sobre a ferramenta e proporciona à comunidade um conjunto de definições que poderá constituir um instrumento para avaliar os limites de utilização de SGFTs.

1.3 Hipótese

Várias abordagens poderão ser consideradas no estabelecimento dos critérios de avaliação de uma determinada ferramenta de *software*. Uma abordagem que considere termos complexos de serem medidos (p. ex: impacto nos processos da empresa, nível de usabilidade da ferramenta, adaptação da cultura dos empregados ao uso da ferramenta) poderá levar a resultados difíceis de serem comparados. Avaliações desta natureza podem estar associadas à dinâmica das relações sócio-culturais dentro de uma empresa, que não são determinísticas, e a um contexto econômico e tecnológico bastante específico (possivelmente único).

O estabelecimento de uma metodologia para avaliação de desempenho de um SGFT, que leve em consideração a complexidade da ferramenta e possua características de portabilidade, escalabilidade, simplicidade e relevância – um *benchmark* – poderá ser utilizado na comparação de diferentes SGFTs.

1.4 Enfoque da Solução

Historicamente, *benchmarks* têm sido utilizados pela indústria e por pesquisadores como fontes de comparações idôneas. Como exemplo, pode-se citar: *benchmarks* para bancos de dados (TPC, WISCOSIN) e *benchmarks* para arquiteturas de CPUs (SPEC95, SPEC2000).

A avaliação de desempenho tem como objetivo medir a efetividade com que os recursos de *hardware* são utilizados para atender aos objetivos do *software*. Um *benchmark* é um abordagem passível de ser utilizada na avaliação de desempenho e consiste em definir um padrão de medições, considerando o funcionamento em tempo real do sistema, que evidenciem a capacidade de atender um determinado conjunto de requisitos. Normalmente estes requisitos estão relacionados a métricas que descrevem velocidade e vazão de processamento.

Para estabelecer um *benchmark* para SGFTs será implementado um emulador de clientes, que será utilizado na realização dos testes. Serão definidos ainda o contexto em que serão realizadas as medições, as métricas avaliadas e os elementos da arquitetura sobrecarregados ao longo dos testes. Em seguida, será realizada uma discussão sobre os resultados encontrados na avaliação de uma SGFT e a aplicabilidade do *benchmark* proposto.

1.5 Objetivos

Os objetivos deste trabalho são:

1. Especificar um padrão de testes para avaliação de desempenho de SGFTs que possua características de relevância, portabilidade, simplicidade e escalabilidade.

2. Implementar a especificação da avaliação de desempenho em SGFTs.

3. Promover a discussão sobre capacidade de processamento de SGFTs.

1.6 Organização

No capítulo 2 são listados os conceitos básicos relacionados à tecnologia de fluxos de trabalho. São apresentadas taxonomias e definições de Sistemas de Fluxos de Trabalho. Ao final, são exibidas as arquiteturas de SGFTs comerciais.

No capítulo 3 são discutidos os conceitos envolvendo avaliação de desempenho e *benchmark*. É realizada uma descrição dos *benchmarks* do TPC e do SPEC. Também são apresentadas e comparadas as propostas de avaliação de desempenho no contexto de SGFTs.

No capítulo 4 é apresentada a proposta de especificação de um *benchmark*, definem-se detalhes sobre métricas avaliadas, componentes sobrecarregados durante a avaliação e aplicabilidade dos testes.

No capítulo 5 é descrita a implementação do *benchmark* especificado na avaliação de uma plataforma comercial de SGFT e os resultados são apresentados.

No capítulo 6 é realizada uma discussão acerca da contribuição do trabalho e das perspectivas de pesquisas futuras envolvendo o tema.

Capítulo 2

Tecnologia de SGFTs

A idéia de automatizar processos de negócios utilizando SGFTs é discutida desde a década de 70. Naquela época, o grande objetivo dos sistemas criados era a automação de escritórios. A maioria dos produtos envolvia tecnologia bastante complexa para aquele momento e a presença do computador no ambiente de trabalho não era socialmente aceita [13]. Com o passar dos anos, as mudanças sociais promovidas pela larga utilização do computador e o avanço das tecnologias associadas a automação de processos tornaram plausível a construção e a utilização de Sistemas de Gerenciamento de Fluxos de Trabalho.

Neste capítulo serão apresentados conceitos relacionados com SGFTs.

2.1 Modelagem de Fluxos de Trabalho

Um processo pode ser considerado como um conjunto de atividades que, ao serem realizadas, atingem um determinado objetivo de trabalho [14]. Um processo de negócio é um procedimento onde documentos, informação e tarefas transitam entre participantes de acordo com um conjunto de regras definidas, de forma a alcançar o objetivo do negócio [14]. A WfMC[1] define processo de negócios como um conjunto de procedimentos ou atividades que coletivamente efetivam um objetivo do negócio, geralmente inserido no contexto de uma estrutura organizacional, definindo papéis e os relacionamentos entre

[1] Workflow Mangement Coalition – Consórcio de empresas que especifica padrões de interoperabilidade entre SGFTs.

atividades e papéis [3].

Em um processo de negócio, o trabalho flui por um caminho (rota) e sofre interações com os papéis, que são assumidos pelos executores das atividades. A definição da ordem e do contexto destas execuções das atividades constituem um conjunto de regras que estabelecem quando as atividades estarão habilitadas para serem executadas.

Modelos de fluxos de trabalho especificam: as atividades que compõem o processo de negócio, a ordem e as condições que as atividades devem ser executadas, os executores de cada atividade, as ferramentas a serem utilizadas e os documentos manipulados durante sua execução [14].

Apesar de não evidenciar todos os detalhes do funcionamento de uma organização, os modelos de fluxo de trabalho podem possuir informações que sejam suficientes para possibilitar a compreensão, avaliação, reestruturação (reengenharia) e automação dos processos.

Para efetivar a automação dos processos, a modelagem de fluxos de trabalho deve estar numa forma computacionalmente tratável, descrevendo os elementos que compõem o processo de negócio [2, 3, 14]:

- **Atividade**: Descreve uma unidade de trabalho que forma um passo lógico dentro de um processo. Uma atividade pode ser executada manualmente, automaticamente ou de forma mista. Portanto, uma atividade em um modelo de fluxo de trabalho requer interação com um participante do processo (ser humano e/ou com uma máquina) que estará desempenhando um papel.

- **Papel**: Um participante do processo deve assumir um papel para desempenhar as atividades. A definição de um papel abstrai quem (ou o que) irá executar a atividade, e um mesmo participante poderá assumir mais de um papel.

- **Atores (participantes do processo)**: São aqueles que desempenham os papéis. Podendo ser representado por uma pessoa, uma máquina, um programa ou um grupo destes.

- **Documentos**: São abstrações do trabalho que está sendo realizado. Cada atividade possui documentos de entrada e de saída. A partir da avaliação dos documentos é possível estabelecer um significado para o trabalho realizado na execução de uma atividade.

– **Modelo (Definição) de Processo**: É a representação de um processo de negócio em uma forma que suporta manipulação automatizada.

– **Instância de Processo**: Quando um processo está sendo excutado, passa a existir uma instância (um caso particular) do modelo do processo. Várias instâncias de um mesmo processo poderão estar ocorrendo simultaneamente.

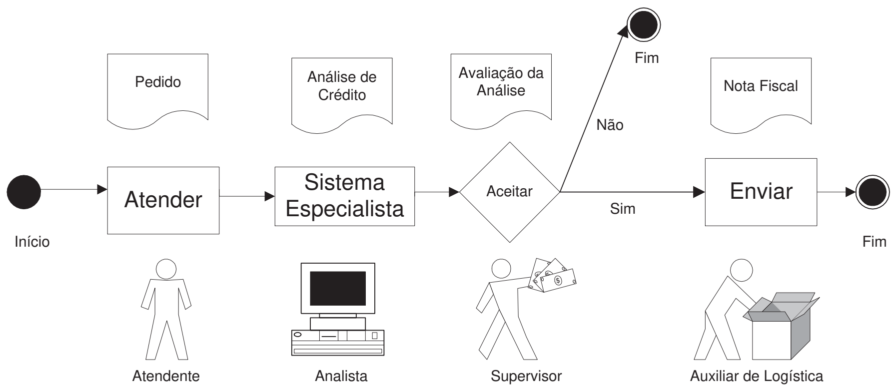

Figura 2.1: Exemplo de modelo de fluxo de trabalho.

A Figura 2.1 representa um modelo de fluxo de trabalho. O processo de negócio descrito é um serviço de compras por tele-atendimento. Neste exemplo, pode-se perceber a seqüência das atividades e papéis associados, o fluxo de execução do processo e os documentos envolvidos.

Através de um modelo Entidade-Relacionamento, exposto na Figura 2.2, Ellis [2] descreve as relações (e as cardinalidades) existentes entre os elementos que descrevem o fluxo de trabalho.

2.2 Sistemas de Gerenciamento de Fluxo de Trabalho (SGFT)

De acordo com a WfMC [4], um SGFT pode ser descrito como um sistema para definição, criação e gerência da execução de fluxos de trabalho através do uso de software, capaz de interpretar a definição de processos, interagir com seus participantes e, quando necessário, invocar ferramentas e aplicações. Um SGFT tem como objetivos a gerência e a automação dos processos [3].

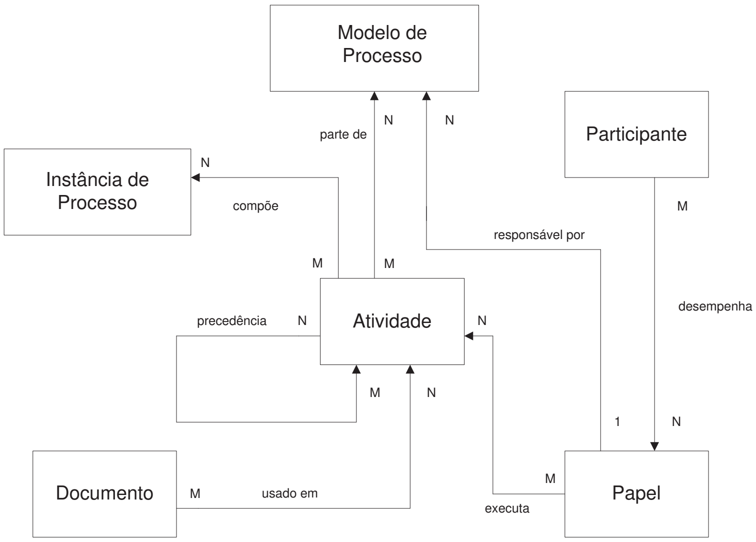

Figura 2.2: Modelo E-R de Fluxo de Trabalho (adaptado de [2])

Ellis [2] descreve que SGFTs contêm conhecimento organizacional, sendo projetados para auxiliar grupos de pessoas a efetivar um trabalho, em contraste com outras ferramentas, tais como serviços de correio eletrônico e vídeo-conferência, que não possuem conhecimento do processo de negócio em que estão envolvidos.

2.2.1 Arquitetura de SGFTs e Padronização

A utilização de um SGFT envolve a interface com dois componentes básicos.

O primeiro é um componente que permite definir os processos de negócio através da descrição dos modelos de fluxo de trabalho, estando normalmente associado às tarefas desenvolvidas por analistas de processos e administradores.

O segundo é o componente relacionado diretamente com a encenação das instâncias de processos, possibilitando aos atores do processo interagir com o SGFT e monitorar a dinâmica das execuções das atividades.

Estes dois componentes estão representados na Figura 2.3, que exibe a estrutura genérica de um SGFT.

Geralmente um SGFT é construído baseando-se numa arquitetura cliente-servidor [2, 8], sendo composto pelos seguintes elementos: servidor do SGFT, cliente do SGFT, repositório de dados, estrutura de comunicação entre servidor e clientes. No servidor encontra-se o **motor do SGFT** (*workflow engine*), responsável pela interpretação e roteamento dos fluxos de execução de atividades. No cliente encontram-se as aplicações para definições de processos, interação, acompanhamento e administração do sistema.

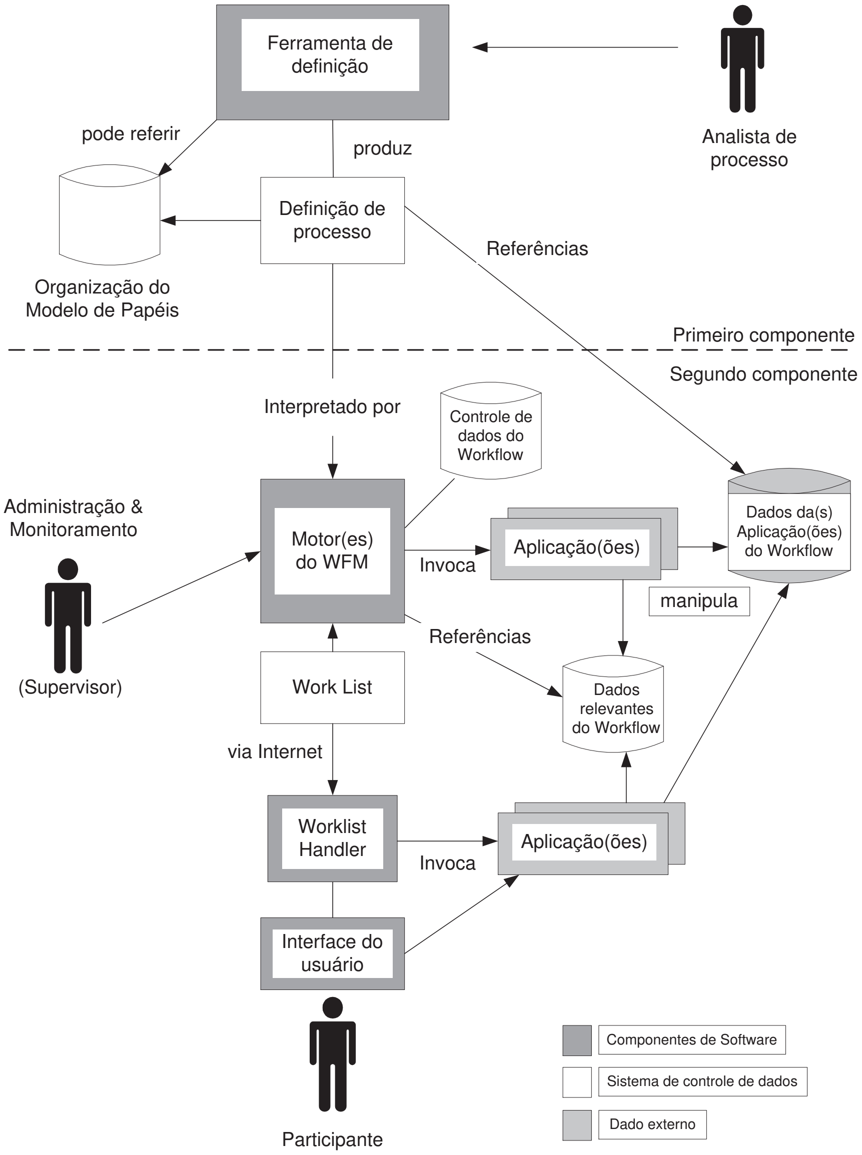

Figura 2.3: Estrutura genérica de um SGFT (adaptado de [3])

À medida que o processo vai sendo encenado, os atores são informados sobre a necessidade de executar atividades através de uma lista de trabalho (*worklist*). Cada item contido nesta lista designa a possibilidade de execução de uma atividade particular de uma determinada instância de processo, sendo denominado de item de trabalho (*work item*).

Áreas funcionais de um SGFT

Com o objetivo de alcançar a padronização, a WfMC estabeleceu uma série de modelos que descrevem os requisitos para um SGFT. A Figura 2.4 mostra as áreas funcionais definidas pela WfMC, que são descritas como [3]:

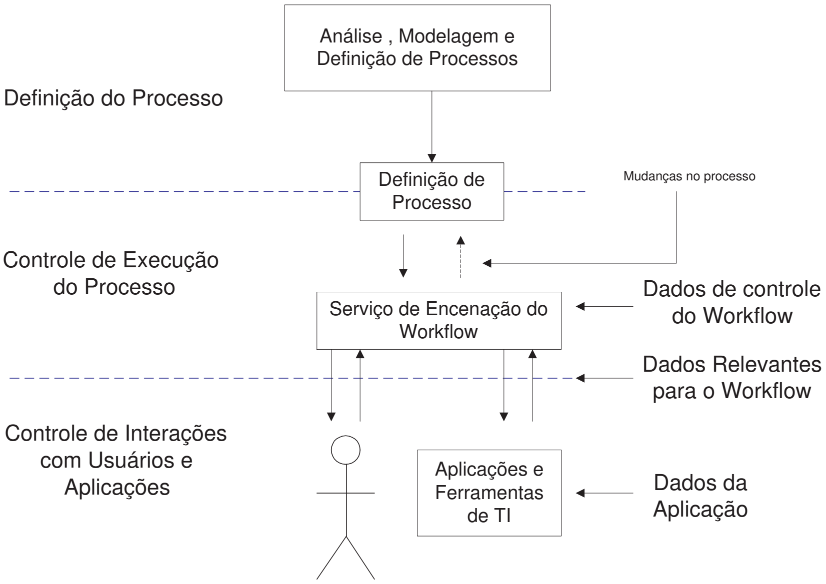

Figura 2.4: Áreas Funcionais de um SGFT (adaptado de [3])

- Definição de processos: Nesta fase, um processo de negócio é caracterizado como um modelo de fluxo de trabalho. A linguagem de descrição de fluxos de trabalho é bastante peculiar para cada ferramenta, como também os padrões suportados para descrever o controle dos fluxos [15].

- Controle de execução de processos: Coordena, habilita e monitora a execução das instâncias dos processos.

- Controle de interações com usuários e aplicações: A execução das atividades não é desempenhada pelo motor do SGFT. No entanto, cabe a ele a coordenação e o acompanhamento do fluxo de execução das atividades, habilitando a execução das atividades para os atores relacionados.

Modelo de Referência da WfMC

O Modelo de Referência proposto pela WfMC descreve os elementos arquiteturais de um SGFT e as interfaces entre tais elementos. A especificação das interfaces tem como objetivo providenciar a capacidade de integração entre SGFTs de diferentes fabricantes [11]. Este Modelo está descrito na Figura 2.5.

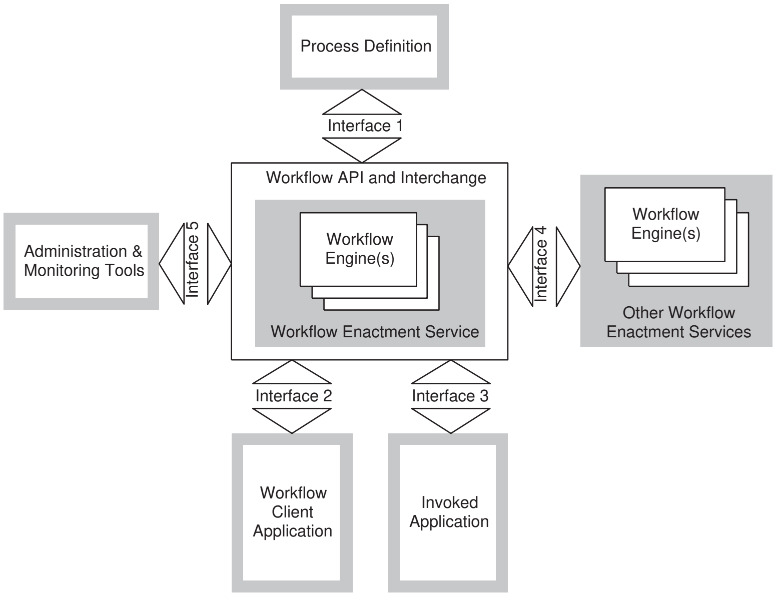

Figura 2.5: Modelo de Referência da WfMC [4]

Descrição das Interfaces do Modelo de Referência da WfMC:

Interface 1 - Ferramentas de Definição de Processos: Especifica uma interface padrão, de forma a permitir a inter-operabilidade entre distintas ferramentas de definição de processos e o motor do SGFT.

Interface 2 - Aplicações Cliente: Definição das APIs[2] para aplicações clientes que requisitam serviços do motor do fluxo de trabalho para controlar a progressão dos processos, atividades e *workitems*.

Interface 3 - Aplicações Invocadas: Definição de um padrão de APIs para permitir que motores de fluxo de trabalho invoquem uma variedade de aplicações através de um agente comum de *software*.

[2] Application Programming Interfaces: Conjunto de convenções, programas e bibliotecas que possibilitam a comunicação com uma ferramenta de *software*.

Interface 4 - Interface com outros Serviços de Encenação: Definição dos modelos de interoperabilidade e os correspondentes padrões para suportar a interconexão com outros motores de SGFT.

Interface 5 - Ferramentas de Administração e Monitoramento: Definição de uma interface comum e padronizada para o acesso a funções de controle, monitoramento e auditoria.

Mapa conceitual de um SGFT

O mapa conceitual proposto por Wang [5], descrito na Figura 2.6, é específico para a plataforma em que seus trabalhos foram desenvolvidos (Meteor[3]). Contudo, existem alguns elementos que são genéricos o suficiente para serem encontrados em vários SGFTs, apesar de não haver uma descrição explícita destes elementos no modelo de referência da WfMC.

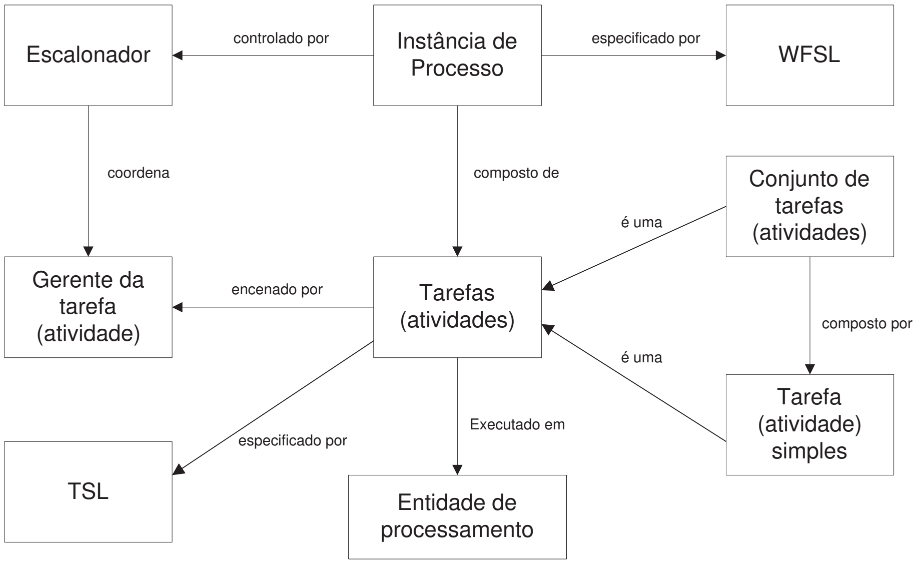

Figura 2.6: Mapa conceitual de um SGFT(adaptado de [5])

O motor (ou um conjunto de motores) do SGFT escalona e monitora as instâncias de processos, gerencia e monitora as execuções das atividades desempenhadas pelos atores nas respectivas entidades de processamento. No mapa conceitual existe ainda a indicação de duas especificações que permitirão definir as atividades (TSL[4]) e os modelos de fluxos

[3]SGFT desenvolvido no Laboratório de Sistemas Distribuídos da Univerisidade da Geórgia.

[4]Task Specification Language: Linguagem de especificação de tarefa (atividade).

de trabalho (WFSL[5]).

2.2.2 Classificação de Sistemas de Fluxo de Trabalho

Na literatura, as classificações de sistemas de fluxo de trabalho são bastante variadas e dependem da abordagem relacionada. Inicialmente, serão discutidas duas taxonomias relacionadas ao tipo de processo a que o SGFT destina-se. Em seguida, serão abordadas uma taxonomia relacionada à forma como o SGFT se comunica com os atores e uma outra relacionada ao grau de independência funcional do SGFT.

Quanto ao grau de estruturação e repetitividade das tarefas realizadas

Esta classificação considera os quesitos de estruturação e grau de repetitividade dos processos aos quais o SGFT está relacionado. A Figura 2.7 exibe a classificação [16, 14], que é descrita a seguir:

- Sistemas *Ad Hoc*: São sistemas em que há pouca estruturação das tarefas realizadas, não há uma padronização para a movimentação da informação entre os atores. O motor do fluxo de trabalho fica dependendo de eventos gerados por decisões humanas, o que dificulta a automação.

- Sistemas Administrativos: São sistemas que envolvem processos previsíveis e repetitivos, com simples regras de coordenação e que manipulam informações com pouca complexidade.

- Sistemas de Produção: Relacionam-se a processos previsíveis e repetitivos, envolvendo processamento complexo de informação e podendo acessar múltiplos sistemas de informação. A coordenação e ordenamento das tarefas possuem características que tornam os processos mais passíveis de serem automatizados.

[5] Workflow Specification Language: Linguagem de Especificação de Modelos de Fluxo de Trabalho

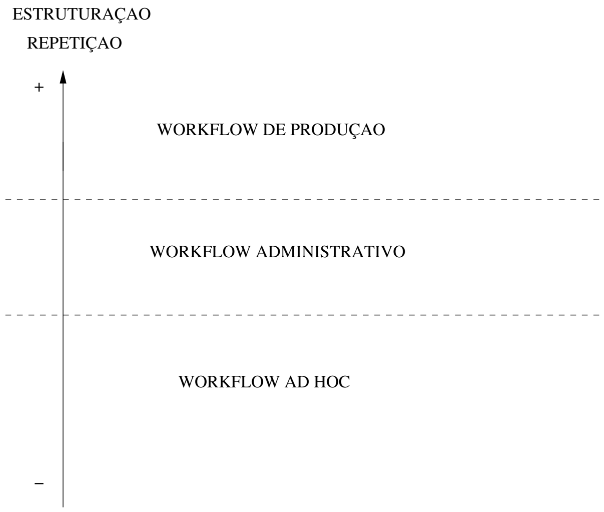

Figura 2.7: Classificação quanto ao grau de repetitividade e estruturação

Quanto ao grau de colaboração

Chaffey [6] define um Sistema de Gerenciamento de Fluxos de Trabalho como um sistema especializado de *software* que fornece o apoio computacional para o trabalho colaborativo. Apesar de manter a mesma nomenclatura da taxonomia anterior, Chaffey sugere ainda uma classificação onde considera o conceito de grau de colaboração.

Um sistema colaborativo é um *software* que deve permitir aos membros de um grupo compartilhar idéias, informações e tarefas, possibilitando ao grupo executar um processo de negócio de uma maneira eficiente.

A Figura 2.8 descreve a relação entre a estruturação dos processos e o grau de colaboração. As redefinições dos termos utilizados estão descritas a seguir:

- Sistemas *Ad Hoc*: estes sistemas são caracterizados pela falta de uma estrutura rígida. Tipicamente há uma forte influência da participação humana ocorrendo colaboração, coordenação e co-decisão. Nestes sistemas a criatividade dos participantes poderá influenciar na forma como as tarefas são desempenhadas e encaminhadas.

- Sistemas Administrativos: são sistemas intermediários em relação a influência da colaboração. Geralmente, são baseados no processamento de formulários.

- Sistemas de Produção: a forte característica de repetitividade e estruturação limita a colaboração e a criatividade dos participantes.

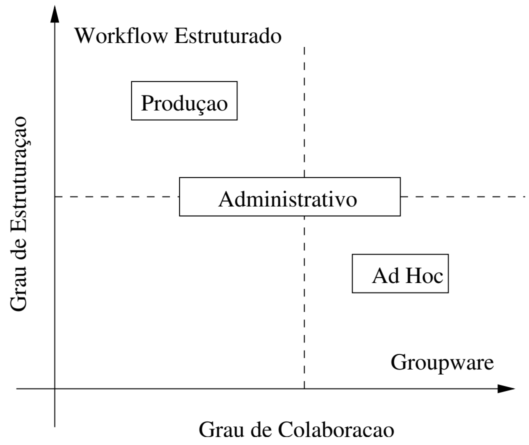

Figura 2.8: Classificação quanto ao grau de estruturação e colaboração [6]

Quanto à natureza das aplicações suportadas

Considerando a forma como o SGFT providencia suporte à iteração com as aplicações, os SGFTs podem utilizar:

– Uma tabela (ou classe) do SGBD[6], que é compartilhada pelos participantes sendo manipulada como um formulário que, à medida que o processo é encenado, tem seus campos modificados.

– Um *middleware*[7] que fornece interoperabilidade entre aplicações em um ambiente heterogêneo. Este serviço pode ser fornecido por um gerenciador de fila de mensagens[8] ou por uma arquitetura de objetos distribuídos[9].

Quanto ao grau de independência funcional

Considerando o fato da tecnologia fluxo de trabalho trazer vantagem competitiva para a empresa e a necessidade de integrar o SGFT aos *softwares* pré-existentes, Muehlen

[6]Sistema de Gerenciamento de Banco de Dados

[7]*Software* que fica abaixo da camada de aplicação e fornece infra-estrutura para possibilitar a comunicação entre diferentes aplicações.

[8]Exemplos de gerenciadores de filas de mensagem: *IBM MQSeries e Oracle Advanced Queuing*.

[9]Exemplos de arquiteturas de objetos distribuídos: *CORBA,COM*

[17] propõe a seguinte classificação, relacionando o grau de independência funcional do SGFT:

- Sistema Autônomo: É definido como um SGFT que é funcional sem qualquer *software* de aplicação, com exceção do SGBD e do *middeware* responsável pelo serviço de fila de mensagens. Características de um Sistema Autônomo: providencia funcionalidade fluxo de trabalho e integra-se com diferentes aplicações. Geralmente é uma arquitetura cliente-servidor.

- Sistema Embutido (*embedded*): É definido como um SGFT que é funcional apenas quando utilizado com um conjunto de *softwares* de aplicação. Características de um Sistema Embutido: é parte de um sistema maior de *software*, a seleção de um SGFT alternativo (depois da introdução de um SGFT embutido) é muitas vezes proibitiva, tem limitação em relação ao fornecedor e aos componentes que podem ser adicionados. Geralmente, apresenta dificuldades de interface com sistemas independentes.

2.3 Exemplos de arquiteturas de SGFTs

O conhecimento da arquitetura de um SGFT é fundamental para a analisar o seu funcionamento. Os requisitos de configuração, a interface de definição de modelos de fluxo de trabalho, os mecanismos de comunicação com os atores e o SGFT, os protocolos de interconexão, as APIs suportadas e todos os demais detalhes de implementação permitem uma interpretação correta do comportamento de um SGFT, possibilitando uma avaliação efetiva.

2.3.1 *Staffware iProcess*

O *Staffware iProcess* é um SGFT embutido; faz parte do pacote *Staffware iProcess Suite*, *software* utilizado em gerência de processos de negócio. Uma descrição lógica da arquitetura dos serviços desta ferramenta é apresentada na Figura 2.9.

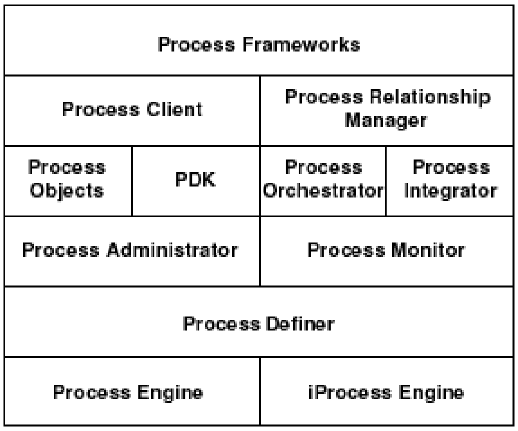

Figura 2.9: Representação Lógica do *Staffware iProcess Suite*

2.3.2 SGFT *W4*

O motor do *W4* faz parte do pacote de ferramentas da empresa W4 denominado *W4 Suite*, sendo composto dos seguintes elementos arquiteturais [7]:

- Um distribuidor central de mensagens, que faz o papel de um ORB[10], descrito na figura como MOM[11];
- Um conjunto de serviços do SGFT conectados permanentemente à base de informações: escalonadores, dicionários, administradores, etc;
- Um conjunto de módulos clientes: serviço de distribuição.

A Figura 2.10 apresenta a arquitetura do W4 com a descrição de vários serviços que poderão ser integrados à ferramenta.

Todos os processos comunicam-se com o ORB W4 por um sistema de filas de mensagens, podendo ser, por exemplo, o *MQSeries* (da IBM) ou o *Oracle Advanced Queuing*. Os dados que descrevem a ocorrência de eventos e são importantes para o funcionamento do motor do SGFT podem ser recuperados a partir das avaliações das mensagens.

[10]Object Request Broker: elemento da arquitetura de objetos distrbuídos que troca mensagens com os objetos.

[11]Manager of Messages: serviços de fila de mensagens.

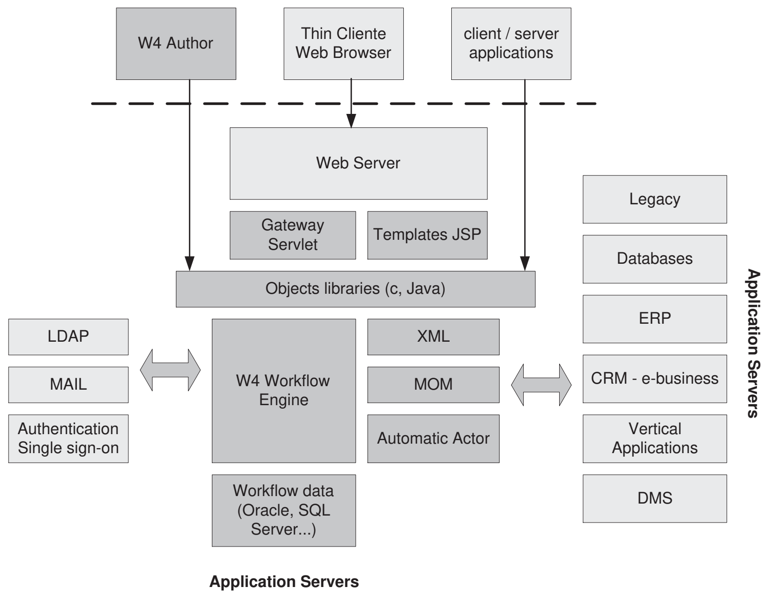

Figura 2.10: Arquitetura do SGFT W4 (adaptado de [7])

2.3.3 SGFT *Ultimus*

A implementação do SGFT Ultimus é baseada nos objetos COM[12], que utilizam a arquitetura Microsoft COM+/DNA[13].

Dentre os módulos que compõem este SGFT, pode-se destacar o Servidor de Banco de dados do Ultimus que é encarregado de armazenar a informação para controlar os fluxos de trabalho, o que inclui as definições de processos e o estado atual de todas as instâncias de processos e atividades. Os objetos COM do Ultimus utilizam as informações para decidir que ações serão tomadas nas conclusões de cada passo do fluxo de trabalho [18].

A comunicação com os clientes pode se dar através de HTTP (no caso de *web clients*) ou objetos COM/DCOM. Na Figura 2.11 estão representadas as arquiteturas do cliente e do servidor do SGFT *Ultimus*.

[12]Component Object Model: Modelo de arquitetura de objetos distribuídos.

[13]Distributes Internet Architecture

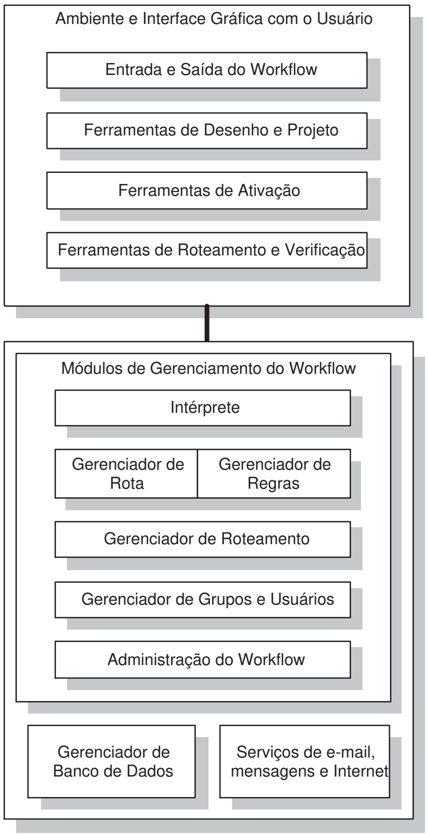

Figura 2.11: Arquitetura do SGFT *Ultimus* (adaptado de [8])

2.3.4 SGFT MQSeries Workflow (MQSW)

MQSeries Workflow é caracterizado como um SGFT autônomo e de produção, possuindo um grande grau de liberdade em relação ao sistema operacional requerido.

É construído sobre um serviço de fila de mensagens, fornecido pelo gerenciador de filas *IBM MQSeries*, utilizando o *IBM DB2* como SGBD para providenciar a camada de persistência de dados (repositório). Na interconexão com os clientes remotos utiliza um aplicativo proprietário que roda em cima dos protocolos TCP/IP ou um *web-client* através da utilização de um *browser*. Na Figura 2.12 há uma descrição dos componentes do *MQSeries Workflow* (figura à esquerda) e detalhes da arquitetura do servidor (figura à direita).

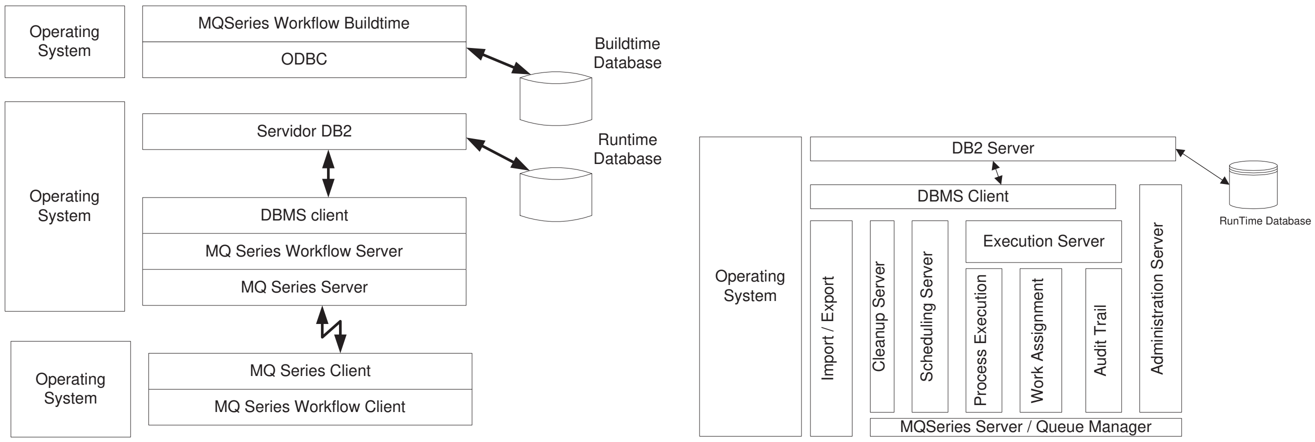

Figura 2.12: Topologia do *MQSW* e Detalhes do Servidor

2.4 Considerações

Neste capítulo foram apresentados os conceitos relacionados à tecnologia de fluxos de trabalho. Foram exibidas algumas definições, taxonomias, esforços para padronização e arquiteturas de SGFTs.

Na maioria dos SGFTs comerciais, os elementos do Modelo de Referência da WfMC são encontrados (em um nível bastante abstrato). No entanto, existe uma série de diferenças nos detalhes de implementação de cada SGFT, o que tem dificultado o estabelecimento de um padrão e, consequentemente, a inter-operabilidade. Esta heterogeneidade dificulta também o estabelecimento de um padrão de testes válido para uma grande variedade de SGFTs.

No próximo capítulo são apresentadas definições de avaliação de desempenho e *benchmark*. São exibidas as especificaçoes de *benchmarks* do TPC e do SPEC e em seguida são apresentadas propostas de avaliação de desempenho em SGFTs.

Capítulo 3

Benchmark e Avaliação de Desempenho

Este capítulo apresenta os conceitos relacionados a avaliação de desempenho e *benchmarks*. São descritas as implementações dos *benchmarks* TPC[1] e SPEC[2]. Ao final, são apresentadas algumas propotas de avaliação de desempenho de SGFTs.

3.1 Conceitos Relacionados a *Benchmarks*

Dada a complexidade de um sistema computacional, sua avaliação pode envolver várias perspectivas, desde aspectos funcionais (p. ex: sistema operacional suportado, capacidade de processamento, suporte a interface gráfica) até aspectos sócio-culturais (p. ex: adaptação do grupo a ferramenta, uso do computador no ambiente de trabalho).

O desempenho é um dos critérios a serem avaliados. O desempenho em um sistema computacional é definido como a efetividade com que os recursos de *hardware* são utilizados para atender aos objetivos do *software* [19]. A avaliação do desempenho consiste em medir esta efetividade, definindo uma semântica para as medidas levantadas e pode ser classificada como:

Avaliação comparativa: Quando o desempenho de um sistema em particular é avaliado mediante o desempenho de outro sistema. Esta avaliação tem os propósitos de classificar sistemas existentes, selecionar um fornecedor ou comparar produtos.

[1]Transaction Process Council: Consórcio que define avaliações para sistemas de processamento de transações.

[2]Standard Performance Evaluation Company: empresa que define *benchmarks* para arquiteturas de CPU

Avaliação analítica: Quando o sistema computacional é avaliado com vários parâmetros de carga e de configuração do próprio sistema. A avaliação analítica tem como objetivos melhorar o desempenho (através do ajuste das configurações), projetar e implementar novos sistemas.

Um *benchmark* define um padrão através do qual algo poderá ser medido ou julgado [20], podendo ser utilizado para avaliar o desempenho de um sistema computacional.

Benchmarks definem uma carga de trabalho em um sistema real, satisfazendo os requisitos para ser considerado como uma avaliação comparativa e analítica. Um bom *benchmark* deve **exercitar** todas as funções do sistema sob avaliação, podendo ser implementado como uma instrução, um programa ou uma seqüência de pedidos de interações com um componente de *software* [19].

De acordo com Gray [21] um *benchmark* deve atender os seguintes critérios:

– **Relevância**: As medidas verificadas deverão descrever funcionalidade e expectativa de desempenho acerca do produto submetido ao *benchmark*. É necessário especificar com clareza o domínio da utilização da ferramenta e as operações a serem submetidas.

– **Portabilidade**: Na elaboração de um *benchmark*, as definições e os termos utilizados devem estar em um nível de abstração que permita transportá-lo para as implementações de diferentes ferramentas. Portanto, não deve haver privilégio na utilização de um contexto que seja particular a uma determinada abordagem. O objetivo desta consideração é tornar o *benchmark* aplicável a uma gama maior de ferramentas a serem avaliadas.

– **Escalabilidade**: O padrão das medições deve atender a pequenos ou grandes sistemas, independente do nível de complexidade dos mesmos. Fatores como monoprocessamento ou multiprocessamento, *clock* e tamanhos de memórias não devem tornar as medidas inconsistentes, nem influenciar na aplicabilidade do *benchmark*.

– **Simplicidade**: Os critérios definidos no *benchmark* devem ser simples e de fácil compreensão. Porém, a relevância das métricas não deve ser negligenciada. Medidas muito simples podem não refletir boas características de julgamento de uma ferramenta. No entanto, se for definido um conjunto de métricas muito complexas e de difícil assimilação pela comunidade interessada nos resultados, poderá ocorrer uma perda de credibilidade do *benchmark*.

3.2 *Benchmarks* TPC

O TPC [22] é um consórcio sem fins lucrativos fundado em agosto de 1988 com a finalidade de fornecer para a indústria a definição e verificação de *benchmarks* para Banco de Dados e Sistemas de Processamento de Transações. Periodicamente, diversos relatórios de *benchmarks* são publicados e um *rank* é apresentado com os diversos produtos que foram submetidos às avaliações.

A diversidade de ferramentas envolvendo processamento de transações originou o aparecimento de vários *benchmarks* relacionados a bancos de dados, a sistemas de tomadas de decisão e a sistemas ambientados na *web*. Os relatórios são denominados de TPC-A, TPC-B, TPC-C, TPC-H, TPC-W. A diferença básica entre os diversos relatórios TPCs está relacionada à constituição do conjunto de transações que compõem os testes. Qualquer fabricante poderá submeter seu sistema computacional para ser avaliado em um dos *benchmarks* definidos pela TPC. Na Figura 3.1 há um exemplo de parte de um relatório do TPC-C.

A definição de*benchmarks* é passível de ser modificada com o passar do tempo, de forma a atender novos tipos de transações. Inclusive, alguns *benchmarks* criados pelo TPC foram considerados obsoletos e/ou inaptos de serem utilizados (como é o caso do TPC-A e TPC-B).

Como exemplo de um *benchmark* do TPC, bastante utilizado, e com o intuito de avaliar os detalhes descritos na implementação é apresentada a seguir uma descrição sucinta do TPC-C.

3.2.1 Especificação do *benchmark* TPC-C

O TPC-C é um *benchmark* utilizado para testar Sistemas de Gerenciamento de Banco de Dados (SGBDs). Suas primeiras versões foram elaboradas no final da década de 80.

As métricas definidas pelo TPC para avaliar vazão sistemas baseado em transações são: transações efetivadas por segundo ou transações efetivadas por minuto. O *benchmark* TPC-C estabelece uma carga de processamento de transações *on-line* que simula as atividades encontradas em uma ambiente de aplicações complexas, caracterizado por [23]:

- Execução simultânea de múltiplos tipos de transações;

- Múltiplas sessões;

Rank	Company	System	tpmC	Price/tpmC	System Availability	Database	Operating System	TP Monitor	Date Submitted
1	IBM	IBM eServer pSeries 690 Model 7040-681	1,025,486	5.43 US $	08/16/04	IBM DB2 UDB 8.1	IBM AIX 5L V5.2	Microsoft COM+	02/17/04
2	hp invent	HP Integrity Superdome	1,008,144	8.33 US $	04/14/04	Oracle Database 10g Enterprise Edition	HP UX 11.iv2 64-Bit Base OS	BEA Tuxedo 8.0	11/04/03
3	hp invent	HP Integrity Superdome	786,646	6.49 US $	10/23/03	Microsoft SQL Server 2000 Enterprise Ed. 64-bit	Microsoft Windows Server 2003 Datacenter Edition 64-bit	Microsoft COM+	08/27/03
4	IBM	IBM eServer pSeries 690 Turbo 7040-681	768,839	8.55 US $	02/29/04	Oracle Database 10g Enterprise Edition	IBM AIX 5L V5.2	TXSeries Developers for AIX V5	09/12/03

Figura 3.1: Exemplo de Relatório do TPC

- Entrada/Saídas de disco significantes;
- Integridade da transação(propriedade ACID[3]);
- Distribuição não-uniforme de acessos aos dados;
- Bancos de dados consistindo de muitas Tabelas, com uma grande variedade de tamanhos, atributos e relacionamentos;
- Concorrência no acesso aos dados.

A métrica de desempenho relatada pelo TPC-C é uma taxa medindo o número de transações finalizadas com sucesso por minuto e é expressa como transações-por-minuto-C (tpmC). Múltiplas transações são utilizadas para simular a atividade do negócio. Cada transação é sujeita a uma restrição de tempo de resposta.

Para adequar-se ao padrão TPC-C, todas as referências aos resultados devem incluir a taxa expressa em tpmC. Os custos associados ao desempenho das plataformas avaliadas, a data em que foram disponibilizados e a configuração do sistema avaliado também são exibidos nos relatórios. Para evitar conflitos e avaliações errôneas, os únicos resultados

[3] Atomicidade, Consistência, Isolamento e Durabilidade

de *benchmarks* comparáveis a um relatório do TPC-C são outros resultados TPC-C de uma mesma versão.

Apesar do TPC-C oferecer um ambiente que emula muitas aplicações, este *benchmark* não reflete todas as aplicações possíveis. Os resultados encontrados são bastante dependentes do contexto. No preâmbulo da especificação (item de especificação 0) é enfaticamente recomendado que os resultados não sejam extrapolados para outros ambientes e que o TPC-C não seja utilizado como um substituto para as aplicações específicas de um dado consumidor. [23]

Itens de Especificação do TPC-C

Especificação TPC-C:

1. Modelo lógico do banco de dados: Uma empresa fornecedora (com uma quantidade de escritórios de venda distribuídos geograficamente) tem o funcionamento modelado como um banco de dados relacional.

2. Perfis dos terminais remotos emulados e das transações: Definição dos requisitos para os terminais de acesso (formatos de exibição dos dados e campos de entrada) e detalhamento das transações utilizadas nos testes (novo pedido, pagamento, *status* do pedido, liberação e nível do estoque).

3. Propriedades do sistema e das transações: Descrição dos requisitos necessários às transações para possuírem as propriedades de Atomicidade, Consistência, Isolação e Durabilidade.

4. Preenchimento do banco de dados e escalonamento: Descrição das regras de preenchimento e definição do escalonamento[4] das tabelas.

5. Tempo de resposta e métricas de desempenho: Define como as transações efetivadas serão medidas, o peso para a participação de cada transação no processo, a função de distribuição dos intervalos de tempo em que as transações serão iniciadas, os intervalos de tempo entre as medições e os requisitos para levantamento do tempo

[4] A intenção do escalonamento é evitar a concentração da emissão das transações em um (ou um grupo) de terminais remotos, mantendo a razão entre a carga de transações presentes no sistema sob teste, a cardinalidade das tabelas acessadas pelas transações, o espaço requerido para armazenamento e o número de terminais gerando transações[23].

de resposta de cada transação. Neste item de especificação são definidos também os intervalos de tempo em que o usuário não interage com o terminal remoto (fração de tempo que o usuário leva para tomar uma decisão).

6. Sistema sob teste, *driver* e definições de comunicação: Descrição da configuração do sistema sob teste, dos pontos em que serão levantadas as medições e das características do *driver* que providencia a emulação do terminal remoto. Neste item de especificação é definida também as características das interfaces de comunicação (banda passante e distância).

7. Levantamento de preço: Descreve a metodologia utilizada no cálculo do preço/transação que levanta os custos de aquisição do sistema e manutenção (considerando os próximos 5 anos).

8. Ampla Divulgação: Define como os documentos deverão ser acessados e como deverão ficar dispostos, facilitando o acesso.

9. Auditoria: Recomendação de uma lista de verificações que devem ser analisadas no caso da necessidade de auditoria nos resultados encontrados no *benchmark* de um determinado produto.

3.3 *Benchmarks* SPEC

O SPEC é uma organização sem fins lucrativos composta de fornecedores de computadores, integradores de sistemas, universidades e organismos de pesquisa, cujo objetivo é estabelecer e manter um conjunto padronizado de *benchmarks* relevantes para sistemas de computadores [24]. Os *benchmarks* definidos pelo SPEC são consituídos de programas de teste e critérios de compilação. Os relatórios com os resultados dos sistemas avaliados são divulgados periodicamente.

3.3.1 Especificação do *benchmark* SPEC CPU2000

O *benchmark* SPEC CPU2000 focaliza no desempenho intensivo do computador, enfatizando CPU, arquitetura da memória e compiladores. Este *benchmark* é composto de dois subcomponentes: CINT200 (12 programas para medições e comparações de processamento intensivo de inteiros) e CFP2000 (14 programas para medições e comparações de computação intensiva com pontos flutuantes). O SPEC CPU2000 não avalia influências de E/S (discos), rede, sistema operacional ou gráficos.

Características do SPEC CPU2000:

- Os programas de teste são desenvolvidos a partir de aplicações de usuário;
- Múltiplos fornecedores utilizam o conjunto de programas em seus produtos;
- Portabilidade;
- Os resultados são divulgados conforme regras pré-definidas para possibilitar comparações e reprodutibilidade.

As métricas avaliadas no CINT2000 e no CFP2000, descritas na Tabela 3.3.1, são calculadas a partir da média geométrica dos resultados encontrados quando o sistema é submetido à carga de processamento definida pelos programas que manipulam inteiros e flutuantes. As diferenças entre essas métricas estão relacionadas com o tipo de otimização utilizada no processo de compilação (razão dos sufixos *base* e *rate* no nome das métricas).

Métricas do CINT2000	Métricas do CFP2000
SPECint2000	SPECfp2000
SPECint_base2000	SPECfp_base2000
SPECint_rate2000	SPECfp_rate2000
SPECint_ratebase2000	SPECfp_ratebase2000

Tabela 3.1: Métricas do SPEC CPU2000

3.4 *Benchmarks* e Avaliação de Desempenho no contexto de SGFTs

Para descrever como avaliações de desempenho são realizadas em SGFTs, procurou-se por fontes que indicassem como este assunto é tratado pela indústria e pelo meio acadêmico. Foram selecionados três artigos acadêmicos, um relatório emitido por uma empresa especializada em auditoria e um relatório emitido pelo próprio fabricante do SGFT(IBM). Para descrever como as propostas de avaliação foram elaboradas foram analisados os processos utilizados, os modelos de carga e as métricas levantadas. Ao final deste capítulo é implementado um quadro comparativo descrevendo as estratégias utilizadas nas diversas avaliações.

3.4.1 Avaliação do *Staffware iProcess* proposta por *Doculabs*

A empresa *Doculabs*[5] avaliou o desempenho do *Staffware iProcess*, um SGFT embutido que faz parte do pacote de serviços de gerenciamento de processos denominado *Staffware Suite*.

Descrição do Processo

O modelo de processo utilizado nos testes é composto por 10 atividades: 8 atividades relacionadas a uma aplicação que se comunica com um banco de dados local, 1 atividade relacionada a uma aplicação que se comunica com um banco de dados remoto e 1 atividade envolvendo a interação com um usuário.

Fluxos de controle encontrados nos roteamentos das atividades: seqüências simples e decisão.

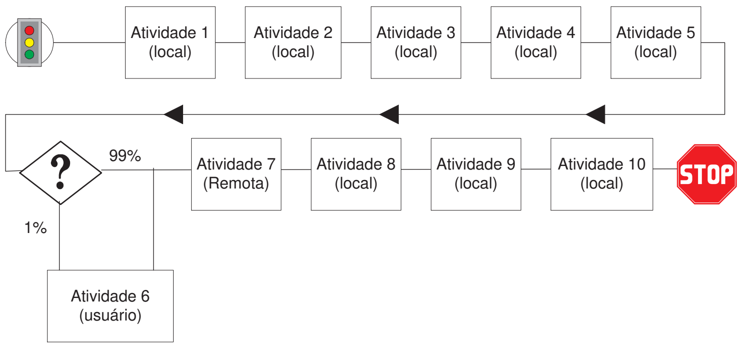

Figura 3.2: Processo utilizado na avaliação do *Staffware iProcess* (adaptado de [9])

Modelo de carga

Foram realizados dois testes distintos, cada um com uma configuração específica de carga e *hardware*.

[5] Empresa especializada em prover serviços de auditoria e avaliação de sistemas. Sítio em http://www.doculab.com.

Teste 1 – Avaliando a taxa de transações:

Em um primeiro momento vários processos foram instanciados (a uma taxa de 16 instâncias por segundo), o que gerou uma grande quantidade de *work items*, garantindo que o sistema teria uma carga de trabalho quando os testes fossem iniciados.

Ao iniciarem os testes, com as tentativas de conexão dos usuários a uma taxa de 10 tentativas por segundo, havia uma série de instâncias de processo totalmente automáticas e uma parcela menor de instâncias ativas esperando pela participação de um usuário. No entanto, em apenas 1% das instâncias de processo ocorreu esta participação. Em seguida, foram levantados os resultados.

Teste 2 – Avaliando o número de usuários conectados:

Inicialmente foram instaciados 200000 processos. Os usuários foram emulados e conectavam-se ao sistema em lotes de 1000. O teste foi conduzido até haver cerca de 30000 usuários *logados*.

Métricas e Resultados Encontrados

As métricas utilizadas para descrever o desempenho foram a taxa de utilização da CPU, a quantidade de usuários que conseguiram conectar-se ao sistema e a taxa de transações. Neste caso, cada transação corresponde a execução de uma atividade (um passo) do processo.

As ferramentas do próprio sistema operacional foram utilizadas para levantar os dados relacionados à utilização da CPU. Para medir os dados relacionados ao funcionamento do *Staffware iProcess*, utilizaram-se os *logs* da própria ferramenta.

Teste	Resultado	Uso da CPU
Transações	1.514 milhões/segundo	Média: 53% / Pico: 81%
Carga de Usuários	29837	Média: 35% / Pico: 83%

Tabela 3.2: Resultados encontrados na avaliação do *iProcess* (adaptado de [9])

3.4.2 Avaliação CSIRO

A avaliação de desempenho descrita a seguir foi realizada pela CSIRO[6] com o produto MQSeries Workflow. O escopo da avaliação reside em levantar algumas métricas associadas ao *middleware* que dá o suporte de comunicação para o MQSeries Workflow.

Descrição do Processo

O processo utilizado nos testes é composto por 6 atividades:

– ***ProcessRouter***: atividade que define o roteamento para uma das três atividades consequentes.

– ***UnknownAct***: atividade que finaliza o processo. Está relacionada a uma uma provável condição de erro.

– ***UPES4A1, UPES4A2, UPES4A3***: atividades executadas automaticamente, retornando apenas informações utilizadas para o controle do fluxo.

– ***UPES4Client***: atividade relacionada ao final (esperado) do processo.

Fluxos de controle encontrados no roteamento das atividades: seqüência simples e decisão.

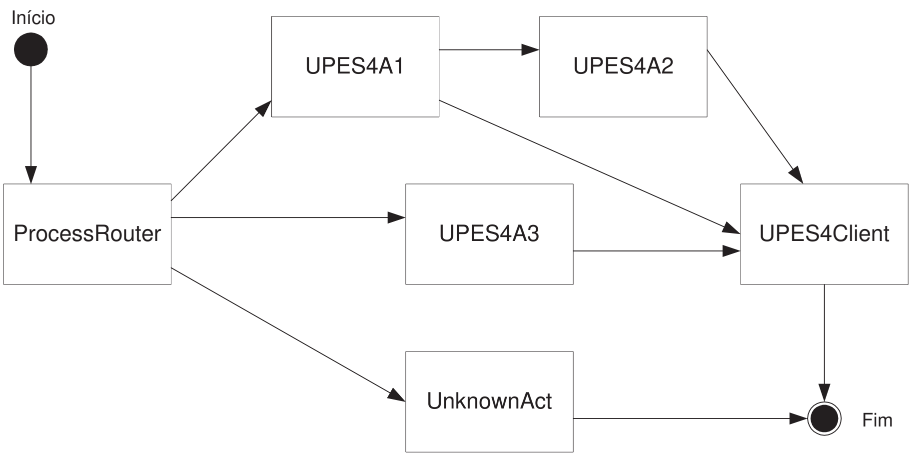

Figura 3.3: Processo utilizado na avaliação realizada por CSIRO (adaptado de [10])

[6]Organização Australiana de fomento a pesquisa.

Modelo de Carga

Durante 30 segundos é emitida uma rajada de pedidos para instânciar e iniciar processos, numa taxa de 5 pedidos/segundo.

Métricas e Resultados encontrados

O objetivo desta avaliação foi verificar o comportamento da infra-estrutura fornecida pelo serviço de fila de mensagens. Os resultados são, portanto, apresentados sob a forma de quantidade de mensagens recebidas e mensagens processadas ao longo do tempo. As curvas levantadas estão descritas na Figura 3.4.

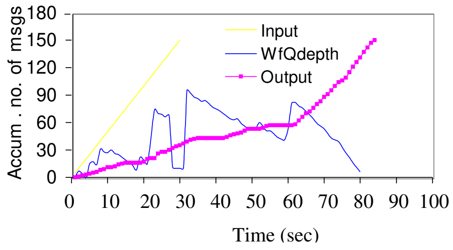

Figura 3.4: Comportamento das filas do MQSW na avaliação Doculabs [10]

3.4.3 Labflow-I

O projeto de pesquisa Labflow-I [12] foi desenvolvido em meados da década de 90 pelo grupo de Banco de Dados da Universidade de Toronto[7]. O objetivo do projeto foi a especificação de um *benchmark* para avaliar o desempenho de SGBDs de alta vazão utilizados em SGFTs. A finalidade do SGFT utilizado era apoiar os processos realizados por laboratórios que faziam parte do Projeto Genoma[8] no *Genoma Center*.

Descrição do Processo

O processo em um laboratório de pesquisa pode ser caracterizado como um fluxo de trabalho de produção, com um relaxamento no quesito de estruturação.

[7] sítio em http://www.db.toronto.edu:8020

[8] Projeto de mapeamento do DNA humano, que envolve centros de pesquisa de vários países.

No laboratório *Genoma Center*, as atividades são organizadas semelhante a linhas de produção de manufatura. Com a chegada de um material para análise, uma seqüência de atividades precisam ser executadas e uma linha de produção é criada. Por exemplo, um cientista encarrega-se dos experimentos, um robô monta pequenas amostras e um computador realiza complexos cálculos estatísticos. As informações geradas ao longo do processo são armazenadas em um banco de dados. Para cada um dos milhões de materiais que chegam, uma estrutura de dados registra o que se tem aprendido sobre ele.

As linhas de produção não trabalham de forma independentes; sendo possível ocorrer interações entre as linhas. As atividades ao longo do processo não são definidas de forma rígida, a seqüência é obtida a partir do histórico das atividades executadas anteriormente. Muitas vezes, se em um dos passos, um resultado não está adequado, várias atividades anteriores necessitam ser executadas novamente.

O processo escolhido para a especificação do *benchmark* é baseado no sequenciamento de DNAs[9] composto de seis atividades que são desempenhadas por aplicações. As atividades estão descritas a seguir e dispostas na Figura 3.5:

criar_clones: programa que simula a criação de clones;

pegar_tclones: inserir no *clone* uma informação (*transposon*);

posição_t: define a posição onde será inserido o *transposon*;

selecionar_tclones: selecionar alguns *t-clones*;

determinar_seqüência;

montar_seqüência: finalização do processo.

Os arcos que ligam as atividades são rotulados com t (rota dos *tclones*) e c (rota dos *clones*). Os arcos que não terminam em uma atividade designam o fim do processo. As ramificações nos arcos indicam cardinalidade.

As transações utilizadas para testar o banco de dados são uma tentativa de generalização daquilo que ocorre nos processos típicos dos laboratórios de pesquisa, sendo compostas de 5 fases. Inicialmente, há uma pesquisa (*query*) no banco de dados em

[9]Uma molécula de DNA pode ser descrita por uma cadeia de caracteres (bases), a determinação da seqüência de bases é denominada de sequenciamento. Uma molécula de DNA possui cerca de 3000 bases longas, denominadas de *clones*. Para faciliatr a leitura de longas cadeias de caracteres, uma técnica de sequenciamento insere uma informação no clone, que passa a ser denominado de *tclone*.

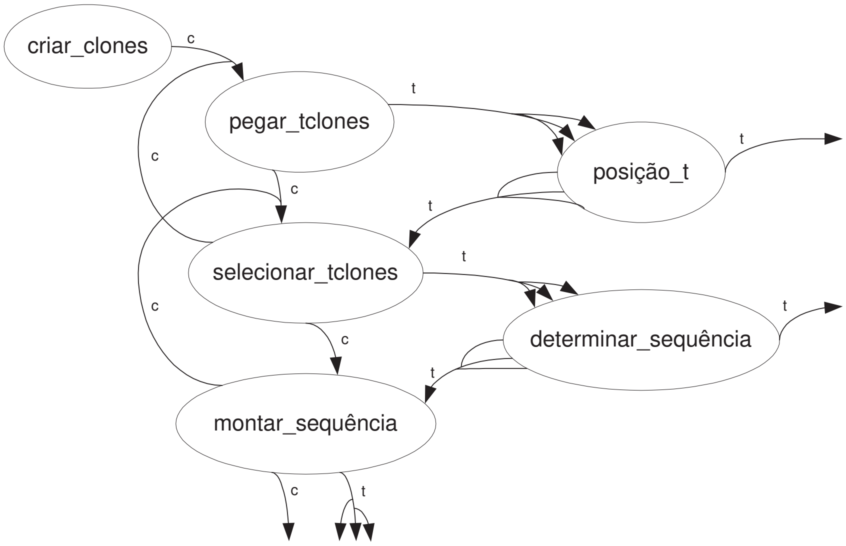

Figura 3.5: Processo utilizado no LAbFlow-I

busca de materiais para serem analisados. Em seguida, serão pesquisados os resultados gerados por outras linhas de produção e, então, ocorrerá a busca por novos materias de análise criados ao longo do processo. Com a finalização de cada atividade (passo), um registro é criado no banco de dados. Baseado nos resultados de cada passo, pode haver a modificação do atributo que define o estado do material.

Modelo de carga

O SGBD é submetido a uma carga de 10000 *queries e updates*.

Métricas e resultados encontrados

Após a carga ser imposta, as métricas são levantadas quando o volume armazenado no banco de dados alcança:

- metade do tamanho da memória principal (0.5X);
- mesmo tamanho da memória principal (1.0X);
- dobro da memória principal (2.0X).

Os testes foram realizados em vários fornecedores de SGBDs (*Texas, Ostore*). Na Tabela 3.3 são apresentados os resultados para um dos servidores de banco de dados utilizados.

Recursos	0.5X	1X	2X
Tempo total (s)	1424	3243	7057
Tempo de CPU - Usuário (s)	1381	2568	4711
Tempo de CPU - Sistema (s)	16	137	812
Tamanho (bytes)	16629760	33824768	62136320

Tabela 3.3: Resultados encontrados no LabFlow-I para o *Ostore* (adaptado de [12])

3.4.4 *Benchmark* proposto por Gillman [1]

Este *benchmark* foi desenvolvido por Gillman [1] na Universidade de Saarland (Alemanha), onde foi desenvolvido um SGFT denominado de *Menthor-Lite*. Os testes tinham o objetivo de compará-lo com um produto comercial.

Descrição do processo utilizado

O processo de negócio utilizado corresponde a um serviço de compras baseado em *e-commerce*. Para descrever o processo, é utilizado um formalismo denominado de *state chart*, apresentado na Figura 3.6. Foram combinadas vários tipos de transações do TPC-C na execução das atividades.

Fluxos de controle encontrados no roteamento das atividades: seqüências simples, *loops*, decisões.

Modelo de carga

Os clientes são emulados e submetem o SGFT a uma carga de processamento modulada pela taxa de emissão de pedidos de instanciação de processos. Os pedidos dos clientes chegam em intervalos distintos de tempo, descrevendo uma distribuição de Poisson. À medida que as taxas vão sendo variadas, as métricas são levantadas.

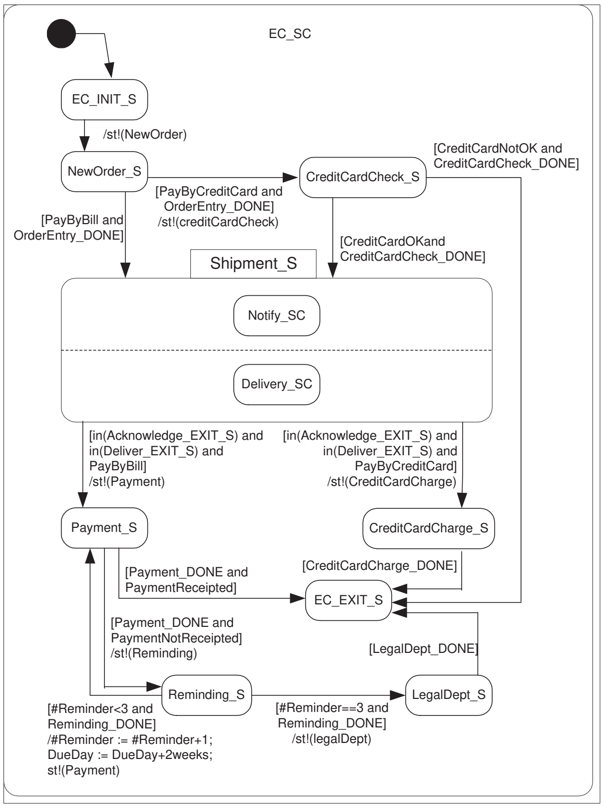

Figura 3.6: Processo utilizado no *benchmark* proposto por Gillman (adaptado de [1]).

Métricas e Resultados Encontrados

Nos testes, são comparados os resultados de dois servidores: o *Menthor-Lite* (desenvolvido pelo grupo de Gillman) e uma versão do *Staffware* em várias configurações.

As curvas apresentadas exibem os comportamentos da taxa de instâncias finalizadas e do tempo de duração de uma instância. Quando as taxas de pedidos aumentam, os resultados apresentados pelas curvas vão configurando uma situação de estresse.

3.4.5 Relatório Técnico de Desempenho do SGFT *IBM MQSeries*

Este relatório foi emitido pela IBM, estabelecendo um conjunto de critérios para avaliar o produto *MQSeries Workflow*, visando estabelecer as capacidades de atendimento a uma demanda de processamento [25].

Caracterizando o Processo

O processo é composto de 100 atividades executadas sequencialmente. Cada atividade é desempenhada de forma automática por um cliente emulado, que executa um programa que retorna apenas informação necessária para o roteamento.

Fluxos de controle encontrados nos roteamentos das atividades: seqüências simples.

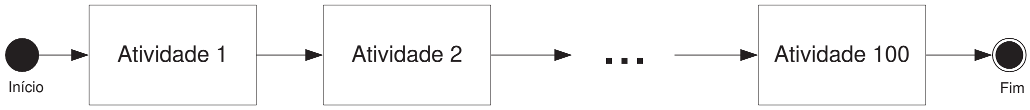

Figura 3.7: Processo utilizado no *benchmark* proposto pela IBM para o *MQSW*

Modelo de carga

Um total de 64 clientes (emulados em máquina distintas) instanciam e iniciam, cada um, um processo. As atividades de cada processo são desempenhadas pelo mesmo cliente. A carga de processamento é decorrente desta rajada de 64 pedidos. Os clientes não fazem novos pedidos de instanciação de processos, ficando conectados apenas para executar as atividades.

Métricas e Resultados Encontrados

Ao final dos testes, quando é executada a centésima atividade do último cliente conectado, os dados de auditoria da própria ferramenta são recuperados. A partir destes dados, é levantada a taxa de atividades por segundo que o servidor conseguiu processar. Esta taxa é denominda de *BWUsec* (unidade básica de fluxo de trabalho (atividade) por segundo).

A quantidade de *BWUsecs* permite fazer um rank dos produtos testados. No relatório [25] estão descritas as métricas relacionadas a diversas máquinas e sistemas operacionais suportados pelo MQSeries Workflow.

3.5 Considerações

Comparando-se as avaliações de desempenho descritas ao longo deste capítulo, nota-se que os *benchmarks* TPC e SPEC possuem definições bastante rigorosas; ao contrário das avaliações realizadas por *Doculabs* e *IBM*, onde as definições de carga e processo de teste são pouco criteriosas. Este procedimento reduz a credibilidade dos testes. A proposta da CSIRO , apesar de ser um pouco mais detalhada, é também pouco criteriosa na definição dos parâmetros dos testes.

Avaliando-se as propostas de *benchmark* relacionadas a SGFTs, Bonner [12] descreve uma avaliação do repositório de dados utilizado e Gillman [1] avalia o motor do SGFT. Apesar de focarem serviços distintos da ferramenta, o que há de comum nas duas avaliações é o fato das métricas levantadas estarem enfaticamente ligadas às transações realizadas nos SGBDs que apóiam as execuções das atividades. Nestes trabalhos, há bastante ênfase no tipo de processo utilizado durante os testes. Em um SGFT, as transações que deverãoser exercitadas devem estar voltadas para as informações de controle, e não para as aplicações dos usuários.

Na Tabela 3.4 é realizada uma comparação entre alguns critérios das avaliações de desempenho realizadas em SGFTs. Estes critérios são discutidos na proposta de um *benchmark* descrita no próximo capítulo. O objetivo desta tabela é compreender as estratégias de cada proposta de avaliação.

Para descrever os fluxos de controle de roteamento é utilizada a seguinte legenda:

1. seqüência simples
2. decisão
3. *loops*

Estratégias de avaliação	CSIRO	Doculab	Relatório IBM	Labflow	Gillman
Controles de Fluxo	1,2	1,2	1	1,2,3	1,2,3
Métricas	$\bullet$	$\clubsuit, \heartsuit, \spadesuit$	$\clubsuit$	$\diamond$	$\oplus, \star$
Proposta de Benchmark	Não	Não	Não	Sim	Sim
Intervalo entre disparos	Uniforme	Uniforme	Uniforme	Uniforme	Exponencial

Legenda:		
	$\clubsuit$ atividades/tempo	$\spadesuit$ número de usuários
	$\diamondsuit$ transações/tempo	$\heartsuit$ uso da CPU
	$\star$ tempo total	$\oplus$ instâncias/tempo
	$\bullet$ número de mensagens	

Tabela 3.4: Quadro comparativo de propostas de avaliações de desempenho em WfMSs.

Capítulo 4

Especificação de um *Benchmark* para SGFTs

O objetivo deste capítulo é especificar um *benchmark* para avaliação de desempenho de SGFTs.

4.1 Introdução

Um dos objetivos do tipo de *benchmark* que se está interessado é comparar métricas que descrevam as condições de funcionamento do motor do SGFT quando submetido a uma determinada carga de processamento.

As informações que definem as ações do motor são dadas pela definição do processo e pela interpretação dos dados enviados durante a encenação. O *benchmark* está interessado nos eventos que ocorrem nesse período e que determinam as durações das atividades e das instâncias de processos. Com estes eventos pode-se caracterizar a taxa de instâncias (vazão) de processos de negócio que é uma das métricas de desempenho do motor do SGFT.

Para especificar o *benchmark* serão explorados os seguintes itens: definição de termos, modelo de comunicação entre cliente e servidor em um SGFT, modelo de carga utilizado, descrição das métricas a serem capturadas, modelo de um processo para os testes, descrição do cenário de testes, programas usados nas avaliações e a forma de descrever os recursos de *software/hardware* utilizados.

4.2 Definição de termos

Inicialmente serão definidos os termos relevantes para a compreensão do contexto das avaliações de um SGFT. Estes termos foram estendidos das definições propostas no glossário da WfMC [3] e por Miller [26]. Os termos expostos na seção 2.1 (atividade, papel, atores, documentos, modelo de processo e instância de processo, *work item* e *work list*) serão utilizados na descrição do modelo do SGFT.

Estado da atividade: informação que define o estado atual da atividade.

Estado da instância do processo: estado atual da instância de processo. Os valores associados ao estado de uma instância de processo podem variar, dependendo do SGFT em questão.

Emulador de clientes: programa que, através da utilização de APIs, efetua comunicação com o servidor, fazendo o papel de um cliente.

Instâncias completas: instâncias de processo que foram finalizadas com sucesso.

Instâncias abortadas: instâncias de processo que foram finalizadas sem sucesso, mediante a ocorrência de um aborto.

Aborto: evento que finaliza uma instância de processo antes do esperado.

Tempo de espera: intervalo de tempo em que o usuário avalia o que ele deverá fazer.

4.3 Modelo de Funcionamento de um SGFT

Não existe um modelo conceitual universalmente aceito para SGFTs, como existe, por exemplo, o modelo relacional utilizado em SGBDs [27]. Apesar dos esforços de padronização da WfMC, criada há mais de dez anos, nem todos os detalhes definidos no Modelo de Referência (principalmente nas Interfaces) são adotados pelos fabricantes de SGFTs. Isto promove a inexistência de uma real interconexão entre as diversas ferramentas. Entretanto, alguns objetivos desta padronização foram alcançados, como, por exemplo, a elaboração de um vocabulário bastante difundido [28].

A descrição arquitetural do sistema que se deseja avaliar o desempenho deve mosrar detalhes que reflitam a relevância das métricas escolhidas para análise [19]. Para descrever um *benchmark* que se proponha a avaliar SGFTs, é necessário descrever um modelo conceitual de SGFT que evidencie as características relacionadas à medição daquilo que impõe carga e processamento ao motor (as transações de controle). Além disso, este modelo deve ser abstrato o bastante para permitir sua utilização em uma quantidade representativa de SGFTs.

Foi descrito no capítulo 2 que a WfMC define as seguintes áreas funcionais para um SGFT [3]:

- Definição de Processos;
- Controle de Execução de Processos;
- Controle de Interações.

As interações com o usuário do SGFT podem ser:

- Interações anteriores ao início da encenação do processo: Definição de Processos.
- Interações após o início da encenação do processo: Controle de Execução de Processos e Controle de Interações.

Supondo que a fase de definição de processos já tenha sido finalizada, o modelo do SGFT deve evidenciar as características relacionadas às interações com os participantes do fluxo de trabalho durante a encenação dos processos. Assume-se ainda a premissa de que os modelos de fluxos de trabalho não sofrerão modificações ao longo da encenação.

O modelo de funcionamento proposto para SGFTs abstrai detalhes de implementação e focaliza-se no levantamento dos detalhes das interações das mensagens trocadas entre o servidor e o cliente do SGFT, baseando-se na máquina de estados da instância de um processo.

Portanto, o modelo limita-se a SGFTs que possuem uma arquitetura cliente-servidor. Apesar desta restrição, alcança-se uma boa representatividade, haja visto que um grande número dos SGFTs utilizam esta arquitetura[8].

4.3.1 Descrição dos estados de uma instância de processo

Os estados que uma instância de processo pode assumir são descritos pela WfMC. No entanto, alguns detalhes como o suporte a transações e manipulação de exceções podem descrever máquinas de estados diferentes.

O *benchmark* está interessado nos estados que descrevam os eventos de início e fim (com sucesso ou não) de uma instância de processo. A lista dos estados está descrita a seguir:

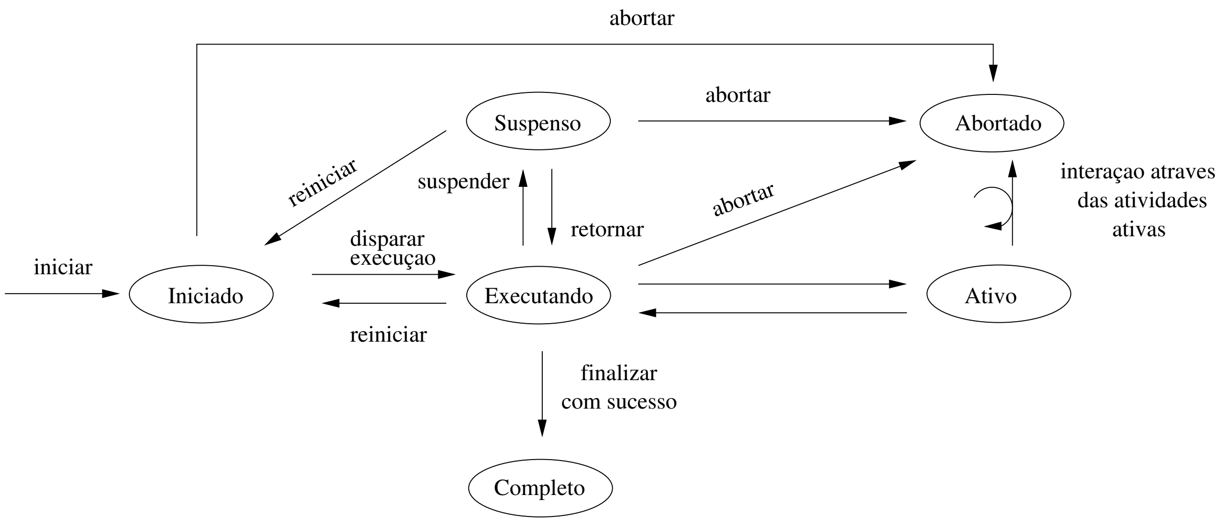

Figura 4.1: Estados de uma Instância de Processo (adaptado de [11])

- **Iniciado**: uma instância de processo foi criada e inicializada, incluindo dados do estado do processo e dados relevantes do SGFT. No entanto, o processo ainda não preencheu todas as condições para iniciar a execução.

- **Executando**: a instância de processo foi iniciada e suas atividades poderão ser executadas. Durante este estado, os itens de trabalho são inseridos nas listas de trabalho dos usuários que tem um papel na execução do processo.

- **Ativo**: uma (ou mais) das atividades relacionadas à instância de processo está sendo executada. O *loop* descrito neste estado representa a possibilidade de execução de uma seqüência de atividades.

- **Suspenso**: a instância do processo está quiescente e nenhuma atividade é disparada até que o processo tenha retornado para o estado *Executando*. Neste estado, o processo poderá também ser reiniciado.

– **Completo**: a instância de processo preencheu as condições para a finalização correta (com sucesso).

– **Abortado**: execução da instância do processo foi abortada e não foi finalizada com sucesso. Em qualquer dos estados anteriores, com exceção do estado Completo, poderá haver um evento que ocasione um aborto e finalize o processo antes do esperado.

4.3.2 Descrição dos estados de uma instância de atividade

De forma semelhante às instâncias de processos, as atividades podem também ser representadas por máquinas de estados que descrevem o ciclo de vida de uma atividade (instância de atividade).

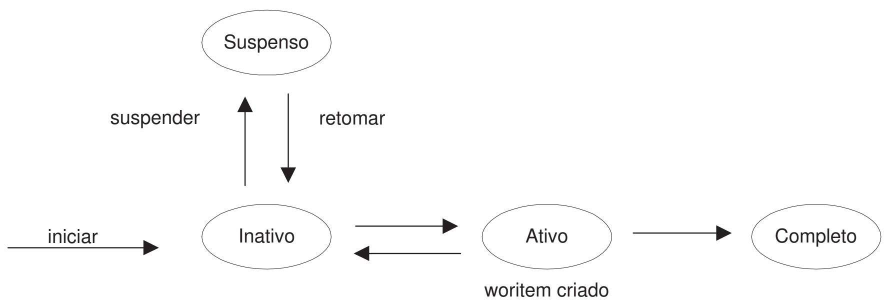

Figura 4.2: Estados de uma Instância de Atividade (adaptado de [11])

– **Suspenso**: a instância da atividade está quiescente e nenhum *work item* será alocado até o retorno para o estado ativo.

– **Inativo**: a atividade foi criada, mas ainda não está no estado Ativo (as condições de entrada da atividade ainda não foram alcançadas). Não há nenhum *work item* para processamento.

– **Ativo**: um *work item* foi criado e atribuído para uma atividade que deverá ser executada.

– **Completo**: execução de uma atividade foi finalizada com sucesso. As condições de transição para as próximas atividades podem ser avaliadas.

4.3.3 Comunicação entre cliente e servidor no SGFT

O modelo descrito na Figura 4.3 descreve a comunicação entre o cliente e o servidor do SGFT. $T1$ e $T2$ são os instantes em que, respectivamente, o cliente inicia a instância e recebe a mensagem de fim de instância de processso.

O *benchmark* busca medir o intervalo entre o Início e o Fim da instância de processo, que é obtido por $T2 - T1$.

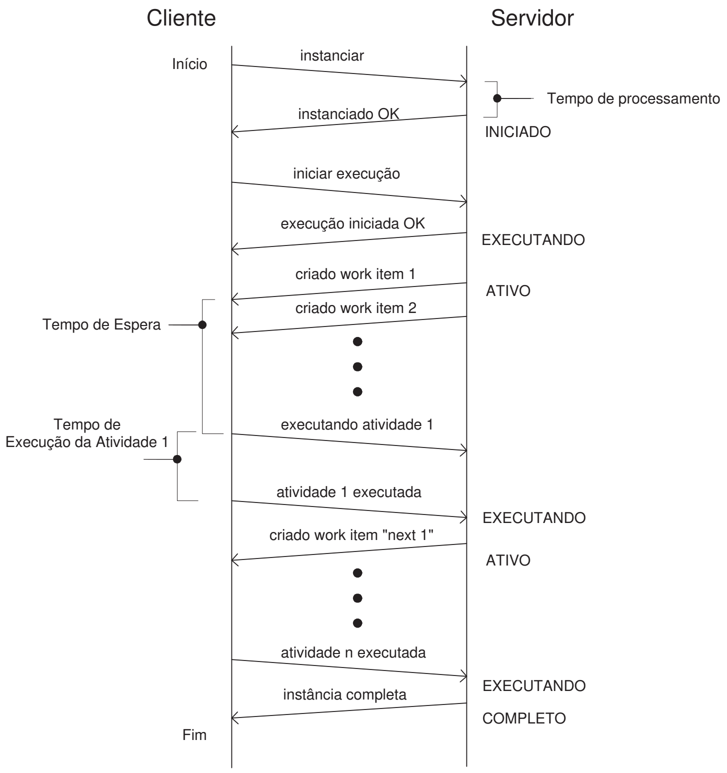

Figura 4.3: Seqüência de interações entre Cliente e Servidor SGFT

4.4 Modelo de carga

A sobrecarga imposta ao motor é independente do tamanho dos dados manipulados nas atividades, haja visto que a execução das atividades se dá, geralmente, nos clientes.

O motor possui a função de habilitar as execuções de instâncias de processos e as atividades mediante consulta aos estados das mesmas. Desta forma, a condição de sobrecarga poderá ser alcançada submetendo o SGFT a uma determinada taxa de criação de instâncias de processos.

Para caracterizar a carga de processamento em um SGFT, busca-se modelar a forma como os pedidos são emitidos pelos usuários do sistema.

Em uma situação ideal, a partir dos registros de utilização (*logs*) de uma quantidade representativa de SGFTs, seriam levantadas e validadas expressões analíticas que descrevessem (ou pelo menos se aproximassem) o tráfego. No entanto, não foi encontrado tal conjunto de informações que pudesse ser utilizado nos testes, optando-se pela utilização de um distribuição de probabilidade relacionada a um processo estocástico que melhor se adequasse ao contexto real de utilização de um SGFT.

Em um SGFT, a carga imposta depende do número de pedidos emitidos pelo cliente e que vão requerer processamento do motor. Modelando-se os instantes de ocorrência destes pedidos, obtém-se uma forma de caracterização da carga imposta.

Em um primeiro momento, os usuários que iniciam processos no SGFT vão requerer a criação de instâncias de processos. Em seguida, serão criados os primeiros *work items*, indicando que os usuários envolvidos no processo instanciado devem interagir com o SGFT para desempenhar os papéis a eles atribuídos e executar as atividades que lhes cabem.

A participação de usuários em atividades interativas está descartada no modelo de avaliação de desempenho, conforme será discutido na próxima seção, e todas as atividades relacionadas ao processo de testes são automáticas. Os usuários são emulados através de programas de aplicações utilizando as APIs fornecidas pelos fabricantes dos SGFTs.

Portanto, um modelo de carga é definido quando se impõe uma taxa de criação de instancias de processo. Existem, possivelmente, outras formas de impor carga de processamento, como simular o envio de mensagens para um SGFT implementado sobre os serviços de um gerenciador de filas de mensagens ou no caso de SGFTs que utilizam os serviços de *Object Brokers*, simular o envio de requisições de serviços oferecidos por objetos. No entanto, estas possibilidades estariam inadequadas aos propósitos do *benchmark*, por serem dependentes de um determinado serviço (e portanto não portáveis) e por simular o cliente ao invés de emular[1].

A opção de ter clientes emulados fazendo uma série de pedidos de criação de instan-

[1]A emulação envolve as respostas em tempo real de uma determinada entidade (*software/hardware*), enquanto a simulação não envolve o conceito de tempo real nas interações

cias de processos permite uma maior abstração em relação aos detalhes de implementação de um SGFT em particular.

4.4.1 Caracterização da distribuição como um Processo de Poisson

Os pedidos de instanciação de processo são variáveis discretas. Considera-se bastante realística a hipótese de que cada pedido é independente dos demais. Na busca de um modelo estocástico que esteja relacionado a este processo, optou-se por utilizar o Processo de Poisson, que possui as seguintes características (adaptadas ao contexto de um SGFT) descritas abaixo[29, 30]:

- O número de pedidos emitidos durante intervalos de tempo (sem sobreposição de janelas de tempo) consituem variáveis randômicas.
- A distribuição do número de pedidos em qualquer intervalo de tempo depende do comprimento do intervalo e não dos limites específicos deste intervalo.
- Em um intervalo suficientemente pequeno de tempo, a probabilidade de obter exatamente um pedido é proporcional ao comprimento do intervalo de tempo.
- A probabilidade de ocorrerem dois pedidos exatamente no mesmo instante é desprezível.
- No instante de tempo $t = 0$, a probabilidade de não ocorrer nenhuma requisição é 1.0, ou seja, não há pedidos no instante $t = 0$.

A equação 4.1 descreve função densidade de probabilidade (fdp) de um Processo Estocástico de Poisson, onde t se refere ao tempo, k a variável randômica discreta e λ o valor esperado.

$$f(k; \lambda t) = \frac{e^{-\lambda t}(\lambda t)^k}{k!} \tag{4.1}$$

Para gerar os números randômicos adequados ao modelo estocástico escolhido, lança-se mão da relação entre a distribuição dos intervalos de chegada dos pedidos e o Processo de Poisson. Se os intervalos entre os pedidos são distribuídos exponencialmente, o número de eventos em um período fixo de tempo descreve uma distribuição de Poisson[31].

Sheldon[30] descreve a seguinte relação entre a média da distribuição dos intervalos de chegada e a média dos pedidos ocorridos: sendo λ o valor esperado de uama variável randômica associada ao Processo de Poisson, os intervalos de chegada terão uma distribuição exponencial com valor esperado de $1/\lambda$.

O algoritmo utilizado para gerar os números randômicos relacionados aos instante do disparo e emissão de requisições estão descritos em Gordon[31] e Shannon[32].

A partir da definição de um modelo estocástico (Poisson) para a emissão de pedidos de criação de instância de processo, o parâmetro a ser manipulado para emular diversas condições de carga é o valor esperado dos intervalos de chegada entre os pedidos $(1/\lambda)$.

4.5 Métricas avaliadas

Um modelo de fluxo de trabalho descreve um processo de negócio que é composto por várias atividades que deverão ser executadas pelos diversos papéis que participam do processo.

Um SGFT controla e coordena a execução de atividades e de processos. As métricas que caracterizem o desempenho de um SGFT estão, consequentemente, ligadas à velocidade de execução destas atividades e processos.

Considerando os intervalos de tempo em que os processos são encenados, chega-se às seguintes taxas: **taxa de execução de atividades e taxa de execução de instâncias de processos**.

Taxa de execução de atividades: é definida como a quantidade de atividades finalizadas com sucesso durante um intervalo de tempo.

Taxa de execução de instâncias de processos: é definida como a quantidade de instâncias de processo que são finalizadas com sucesso durante um intervalo de tempo.

As atividades em um SGFT poderão ser interativas (executadas a partir da interação com um usuário) ou automáticas (executadas sem a participação de usuários) [3]. As atividades interativas dependem da reação do usuário e a caracterização do tempo de resposta deste usuário pode variar bastante, estando fortemente relacionada ao tipo de processo envolvido, podendo levar desde segundos até mesmo semanas. A modelagem do comportamento dos usuários interagindo com uma ferramenta de *software* pode ser bastante complexa, seja considerando um usuário apenas ou uma agregação deles [33].

As atividades automáticas independem da interação humana com o SGFT, estando diretamente relacionadas à capacidade de processamento da máquina onde são executadas.

O *benchmark* proposto utiliza atividades automáticas nas avaliações, cada atividade é desempenhada por um programa de aplicação que é implementado de acordo com a API fornecida pelo fabricante do SGFT, não desempenhando nenhuma transformação útil na informação que transita no processo e retornando dados que são utilizados para controlar as rotas da execução das demais atividades.

Por terem um tempo de resposta menor que a maioria das atividades interativas, as atividades automáticas tendem a fazer com que o SGFT alcance mais rapidamente um estado de sobrecarga, o que justifica a sua escolha para os testes realizados no *benchmark* proposto.

Para descrever o tempo de duração de uma atividade, será utilizado um modelo probabilístico bastante frequente na literatura, a distribuição normal (Gaussiana) [29].

Alguns parâmetros como tempo médio (valor esperado) de duração de uma atividade e as taxas médias de requisição de serviço (instanciação e início de processos) pelo cliente emulado serão variadas ao longo dos testes do SGFT.

Os tamanhos e os tipos de estruturas de dados que estão relacionados às instâncias de modelos de fluxos de trabalho não têm influência direta na capacidade de resposta do motor do SGFT. Este fato implica a não utilização do volume de dados e transações relacionadas à execução das atividades. Em propostas anteriores de avaliação de desempenho, foram utilizadas transações das atividades como métricas de avaliação [9]. No entanto, estas transações sobrecarregam as máquinas onde as atividades são executadas e não o motor do SGFT, sendo, consequentemente descartadas como métricas de avaliação.

O principal objetivo do *benchmark* proposto é medir a capacidade de vazão de negócios de um SGFT, portanto, as métricas levantadas devem estar relacionadas às instâncias de processos. Portanto a métrica que evidencia o desempenho de um SGFT é:

- $\boldsymbol{IPsu/min}$: Instâncias de processo finalizadas com **sucesso** por minuto.

4.6 Condições de realização do *Benchmark*

Durante as medições não são levados em conta os fatores de disponibilidade das ferramentas, abstraindo-se as possíveis falhas que ocorreriam em alguma parte do sistema, considerando-se então os SGFTs disponíveis durante todo o teste [32].

Além da métrica que define a capacidade de processamento do SGFT ($IPsu/min$), outras métricas deverão ser levantadas para auxiliar a medição das restrições de tempo:

- $CIPpe/min$: Pedidos de criação de instância de processo por minuto.
- $CIPac/min$: Criações de instância de processo **aceitas** por minuto.
- $CIPre/min$: Pedidos de criação de instância de processo **recusados** por minuto.
- $IPab/min$: Instâncias de processo **abortadas** por minuto.

As taxas descritas em seguida serão utilizadas para auxiliar a determinar as condições das medições:

- $Kac = \frac{CIPac}{CIPpe}$, onde Kac é a taxa de pedidos de criação de instância de procecsso aceitos.
- $Kre = \frac{CIPre}{CIPpe}$, onde $CIPre = CIPpe - CIPac$ e Kre é a taxa de pedidos de criação de instância de procecsso recusados.
- $Kab = \frac{IPab}{CIPac}$, onde Kab é a taxa de instâncias de processos abortados.

Condição de Regime Permanente

Durante as medições, o SGFT deverá está em condição de trabalho em regime permanente. O tempo necessário para determinar o estado de regime permanente deverá ser levantado empiricamente. Um tempo razoável para início de testes, considerando-se o tempo de finalização de uma instância de processo, é 20 minutos. Durante o estado de regime permanente, a seguinte condição deverá ser satisfeita:

O valor de Kac deve permanecer constante ao longo das medidas.

Condição de validação das medidas

Para as medidas serem consideradas válidas, além do regime permanente é exigido que o SGFT consiga coordenar a finalização com sucesso da grande maioria das instâncias de processos aceitos, como também a maioria dos pedidos de criação de instância de processos devem resultar em instâncias de processos:

$Kab \leq 5\%$

$Kac > 95\%$

Tempo de duração dos testes

O tempo de duração dos testes deve sugerir uma carga equivalente a uma jornada diária de trabalho de 8 horas.

Levantamento das medidas

As medidas são levantadas a partir de avaliação dos *logs* gerados nos clientes emulados e no SGFT. A WfMC [4] define uma tabela de *log*, denominada de *audit trail*, com uma série de campos relacionados aos eventos (e instantes em que ocorrem) que modificam os estados das atividades e processos. Desta forma pode-se rastrear todo o funcionamento do SGFT.

Métricas levantadas no *log* do cliente emulado: $CIPpe/min$

Métricas levantadas no *log* do SGFT: $CIPac/min$, $CIPre/min$, $IPsu/min$ e $IPab/min$.

4.7 Modelo do processo de teste utilizado no *Benchmark*

Para avaliar um SGFT e levantar as métricas necessárias à especificação do *benchmark*, é preciso definir e detalhar um modelo de processo comum a todas às medições.

No modelo de processo de negócio proposto, o trabalho é representado pela informação sendo transportada e modificada ao longo das interações com os atores. O fluxo de controle descreve uma topologia envolvendo decisões, sequências simples, concorrência, *loops*, paralelismo, *splits* e *joins*.

Aalst [15] descreveu uma série de padrões relacionados ao roteamento das atividades de processos de negócio reais. No entanto, não há um estudo que indique qual a freqüência destes padrões (ou mesmo outros padrões possíveis) em processos reais. Uma discussão semelhante foi realizada em linguagens de programação para levantamento das freqüência de ocorrência das estruturas de alto nível (p. ex: atribuições, chamadas de função e estruturas de controle de fluxo de execução) [34].

É importante observar que dada a variedade de sistemas de fluxo de trabalho, é praticamente impossível estabelecer um processo genérico que venha a representar todo e qualquer processo de negócio, aliás não é este o objetivo do processo aqui proposto.

Ao definir um processo padrão, procurou-se estabeler características que acarretassem o exercício das transações de controle efetuadas pelo motor:

– Rotas que contassem com uma quantidade razoável de padrões de controle de fluxo de execução de atividades.

– Atividades com duração aleatória para simular o tempo de processamento.

– Ausências de *deadlocks* e espera ocupada.

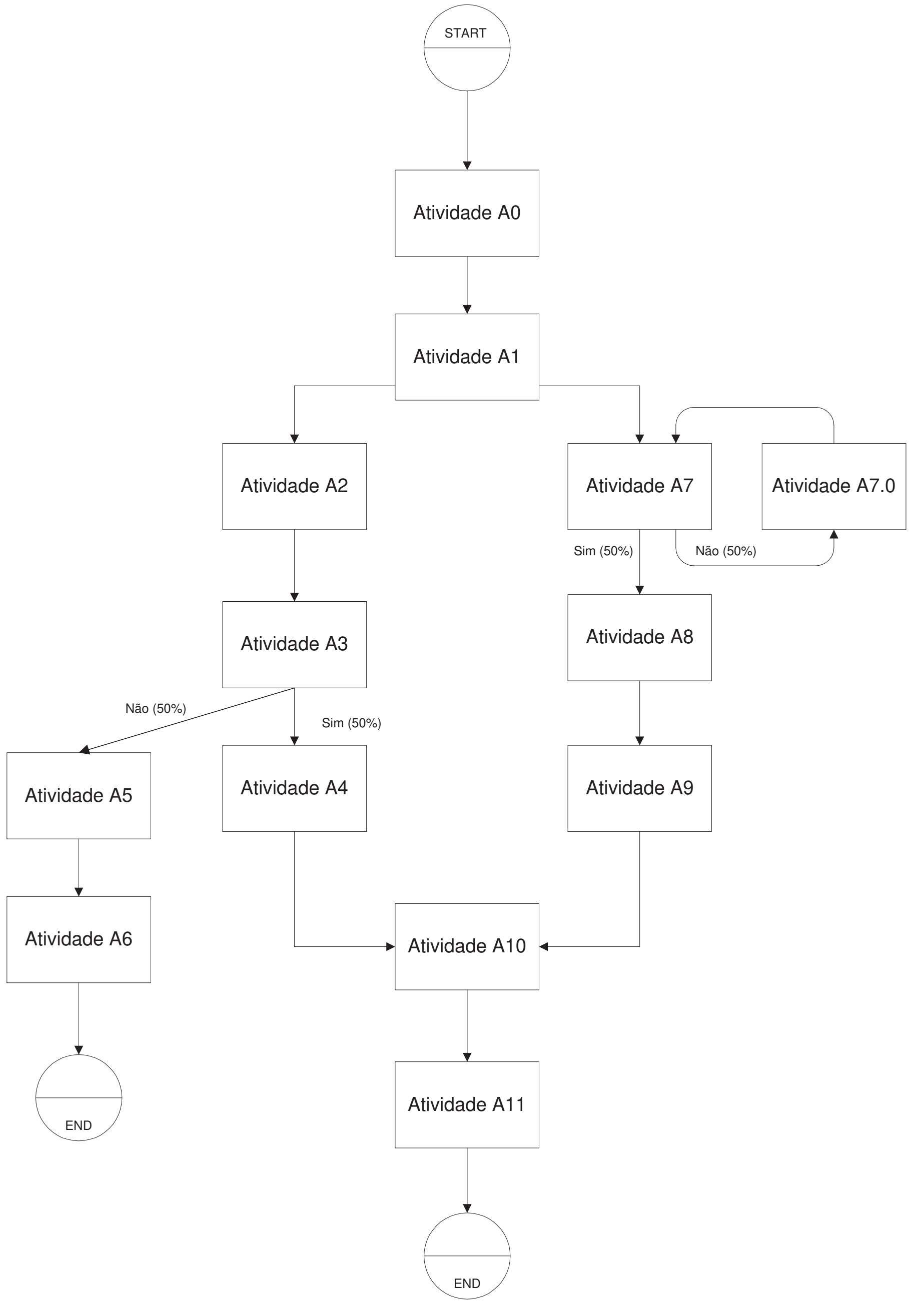

Figura 4.4: Modelo do Processo de Testes

4.8 Descrição do cenário

Os clientes emulados estão localizados em uma ou mais máquinas remotas e conectados, através de uma rede local, à máquina onde está o SGFT. Os protocolos de rede e a conexão remota do servidor não devem impor uma limitação no desempenho a ser medido.

Os pontos onde as métricas estiverem sendo levantadas deverão ser transparentes ao usuário. Para evitar distorções nas medidas, nas máquinas utilizadas durante os testes, apenas os usuários que possuam relação direta com os testes devem estar conectados às mesmas. Todos os serviços (SGBDs, Servidores de Aplicação,etc.) suportados pelo Sistema Operacional que não tiverem uma associação direta com os testes do *benchmark* devem ser suspensos.

4.9 Programas de teste

Os programas de testes são utilizados para comunicar-se com o SGFT e disparar vários pedidos de criação de instância de um mesmo modelo de processo, o que fornecerá carga de processamento.

Para implementar estes programas são utilizadas as APIs fornecidas pelos fabricantes. As linguagens mais comumente encontradas nas APIs são C/C++ e Java. Para o *benchamrk* proposto ser factível é necessário a existência de uma API.

O programa pode ser dividido em três blocos lógicos:

- Gerador de Números aleatórios
- Escalonador
- Iniciador de Processos

O **Gerador de Números Aleatórios** fornece números randômicos mediante a definição de um determinado processo estocástico. Estes números são utilizados para inferir o instante de cada disparo de requisição. Uma descrição mais detalhada é fornecida na seção 4.4.

O **Escalonador** recebe os números aleatórios como dados de entrada e inicializa os métodos que criarão a instância de processo, em seguida comunica-se com o bloco Iniciador de Processos.

O **Iniciador de Processos** instancia processos a partir de modelos de fluxo de trabalho definidos previamente, gerando eventos que coloquem os processos no estado Iniciado. O Iniciador de Processos comunica-se com o SGFT através das APIs fornecidas pelo fabricante.

Na Figura 4.5 está descrito o fluxo de informação entre os blocos do programa de teste, a API do SGFT e o próprio SGFT.

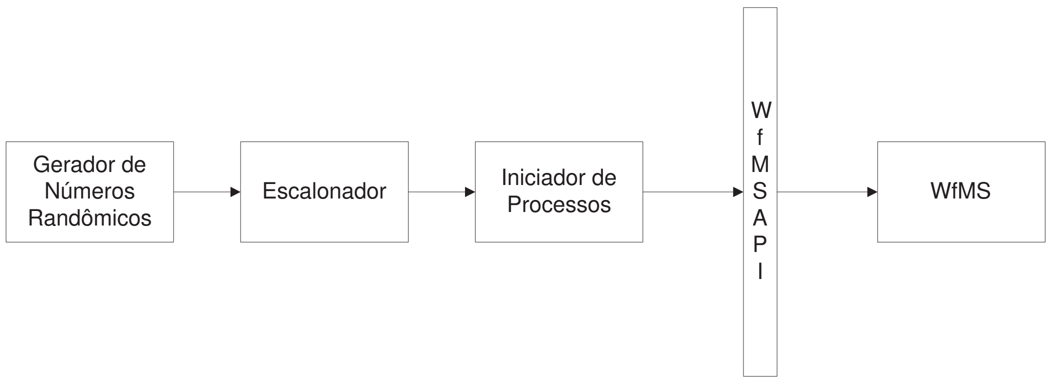

Figura 4.5: Blocos do programa emulador de clientes

4.10 Geração de logs para o *Benchmark*

As variações das métricas avaliadas no decorrer do processo de avaliação necessitam ser registradas para permitir, posteriormente, a descrição da resposta do motor do SGFT ao longo dos testes. São utilizados dois *logs*: o do cliente emulado e do próprio SGFT.

A WfMC propõe, na Interface 5 do Modelo de Referência, os requisitos necessários para as ferramentas que implementam auditoria e monitoramento de um SGFT [11]. A estruturação destes dados, apesar de definidos no Modelo de Referência, pode diferenciar dentre os diversos fabricantes.

Gillman [1] utilizou os *logs* de auditoria para extrair dados que permitissem descrever o desempenho do SGFT. Em casos semelhantes a este, as informações necessárias para levantar as métricas do *benchmark* são recuperadas a partir de consultas à camada de persistência responsável por armazenar os dados do monitoramento.

Uma observação importante na captura e registro dos eventos que caracterizam o *benchmark* é o sincronismo das máquinas onde está o servidor e os clientes emulados, o que garante consistência nos dados relativos aos tempos das medições.

Características que devem estar presentes nos *logs* do SGFT e do emulador: instantes registrados com boa precisão, identificação correta dos eventos e suporte à concorrência.

4.11 Descrição dos recursos de *hardware* e *software*

Durante as avaliações deverão ser descritas as configurações de *hardware* e *software* das máquinas do servidor do SGFT e do cliente. Uma completa descrição destes itens visa caracterizar o ambiente onde foram realizadas as medições.

Configuração da máquina onde está o SGFT

- Fabricante do Processador
- Modelo do Processador
- Clock do Processador
- Tamanho da Memória Principal(RAM)
- Memória Virtual
- Memória Cache
- Disco Rígido
- Protocolo da camada física da rede
- Protocolo das camadas de Enlace/Rede
- Sistema Operacional

Configuração da máquina onde está instalado o Cliente

- Fabricante do Processador
- Modelo do Processador
- Clock do Processador
- Tamanho Memória Principal(RAM)
- Memória Virtual
- Memória Cache
- Disco Rígido

- Protocolo da camada física da rede
- Protocolo das camadas de Enlace/Rede
- Sistema Operacional
- Linguagem de construção dos Programas
- Compiladores/Interpretadores

4.12 Considerações

A finalidade do *benchmark* especificado neste capítulo é avaliar em que circunstâncias uma determinada carga de trabalho fará o motor de um SGFT alcançar um estado de sobrecarga. De posse destas medidas, pode-se comparar o desempenho entre SGFTs distintos (avaliação comparativa) ou entre diferentes configurações de um mesmo SGFT (avaliação analítica).

Características do *benchmark* proposto: portabilidade, simplicidade, relevância, reprodutibilidade e escalabilidade.

Portabilidade

Os requisitos que um SGFT necessitaria possuir para se submeter ao *benchmark* proposto são:

– API que permita a automaçao e a implementação do escalonamento dos pedidos de criação de instância de processo.

– Tabela de auditoria com *log* dos eventos ocorridos durante a encenação dos processos.

– Arquitetura cliente-servidor.

Os itens descritos anteriormente restringem o uso do *benchmark*. Contudo, uma quantidade bastante representativa de SGFTs adequam-se a estes requisitos [8, 2], o que garante a portabilidade.

Simplicidade

O *benchmark* é descrito de forma clara. A métrica de desempenho e as métricas que auxiliam a estabelecer as condições de medição são definidas a partir de consultas ao *log* de funcionamento do SGFT e ao *log* de funcionamento do cliente emulado.

A implementação dos emuladores de clientes também não apresenta complexidade, nem dependência de bibliotecas dependentes de plataforma.

Escalabilidade

A carga de trabalho definida no *benchmark* proposto depende apenas da taxa de pedidos de criação de instâncias de processo. Para gerar grandes cargas de trabalho, poderá ser necessário a utilização de mais de uma máquina emulando clientes. No entanto, não existe qualquer restrição acerca dos limites de capacidade de resposta do SGFT e a aplicabilidade do SGFT.

Reprodutibilidade

A descrição detalhada das condições de medição, a definição de uma carga que exercita as transações de controle do motor e a simplicidade do *benchmark* garantem a capacidade de reprodução dos testes.

Relevância

O modelo de carga proposto tem como objetivo exercitar as transações de controle do motor do SGFT. A métrica $IPsu/min$ mede a máxima vazão de negócios que o SGFT suporta. Logo, esta métrica representa claramente a carga imposta ao motor do SGFT.

Capítulo 5

Resultados

Este capítulo objetivo descreve a implementação e aplicação do *benchmark* proposto ao SGFT *IBM MQSeries Workflow* e analisa os resultados encontrados.

5.1 Avaliação do SGFT MQSeries Workflow

Inicialmente, serão descritos detalhes da arquitetura do *MQSeries Workflow* (MQSW) e em seguida a implementação das especificações descritas em 4.1: modelo de processo, detalhes do cenário, programas de teste, descrição dos recursos de *hardware e software* e a geração de *logs* para o *benchmark*.

5.1.1 Detalhes de Utilização do MQSW

O usuário, durante a utilização do *MQSeries Workflow*, irá deparar-se com as seguites ferramentas:

MQSeries Workflow BuildTime: ferramenta gráfica utilizada para definir um modelo de fluxo de trabalho, seguindo os padrões da linguagem de descrição de processos (FDL)[1], que é particular da própria ferramenta.

[1]Flow Descriptive Language

MQSeries Workflow Runtime: ferramenta utilizada para instanciar processos, monitorar e administrar a execução dos mesmos. Os modelos de processos deverão ter sido criados anteriormente com o *Buildtime*. Nesta ferramenta pode-se observar a encenação do processo, consultar as *worklists*, abortar processos em execução e controlar as instâncias ativas de processos.

Ao final da modelagem no *Buildtime*, os processos deverão ser exportados para um arquivo em formato de texto (com extensão *fdl*). Em seguida o *Runtime* deverá importar a descrição dos processos através da ferramenta *fmcibie*, que é executada em linha de comando. A partir deste instante, é possível a utilização do ambiente do *Runtime*. Na Figura 5.1 há uma descrição do processo de interação entre o *Buildtime* e o *Runtime*.

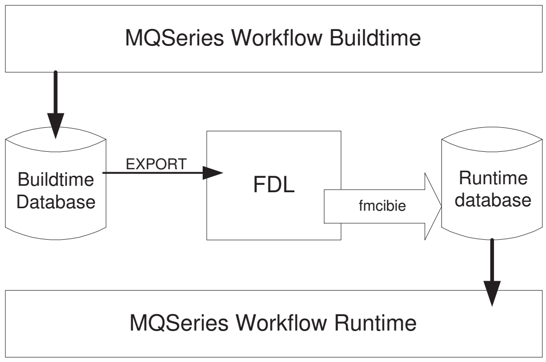

Figura 5.1: Interação entre o *Buildtime* e o *Runtime*

5.1.2 Modelando o processo de teste no *Buildtime*

O processo de testes descrito na seção 4.7 foi modelado no *Buildtime*, segundo as restrições da ferramenta. Na Figura 5.2 é exibida a interface gráfica com parte da descrição do modelo do processo.

Os passos para a descrição de um modelo de fluxo de trabalho , utilizando a ferramenta *Buildtime*, são os seguintes [35]:

1. Modelagem da *staff*: definição dos atores e dos papéis associados.
2. Definição das Estruturas de Dados(Entrada/Saída)[2].
3. Definição dos programas associados à execução das atividades.

[2]Em 4.2, o termo associado a este item é Documento.

4. Descrição do diagrama com fluxo de controle, atividades e papéis associados.

5. Exportação do modelo definido no *Buildtime* para o formato FDL.

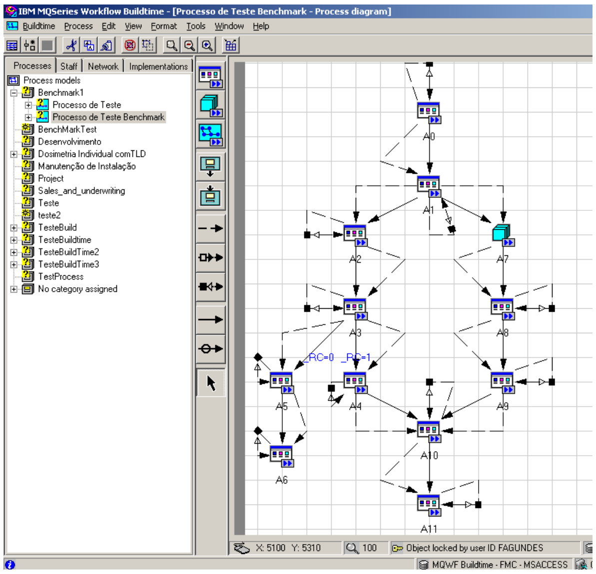

Figura 5.2: Descrição do Modelo implementado no *Buildtime*

Todas as atividades são automáticas e utilizam uma mesma Estruturas de Dados, a opção padrão do *Buildtime* definida como *Default Data Structure*.

Na definição dos programas é estabelecida a forma como ocorrerão as execuções e qual dos servidores de execução será utilizado. Existem duas opções: um servidor denominado de UPES[3] e outro denominado de PEA[4]. Por questão de robustez, o servidor escolhido foi o PEA [36].

No *Buildtime*, os eventos que controlam o roteamento das atividades podem ser: gerados após a execução (como valor de retorno) dos programas nas atividades ou gerados a partir da consulta aos documentos, que têm seus campos modificados ao longo das atividades.

As transições entre as atividades são configuradas nos atributos dos arcos orientados (ver Figura 5.2), que fazem o papel de conectores de controle, e nas condições de entrada/saída de cada atividade.

[3]User Defined Program Execution Server
[4]Program Execution Agent

Como foi discutido por Aalst [15], nem todos os padrões de controle de fluxo passíveis de serem encontrados em um processo de negócio real estão disponíveis nas ferramentas de definição de modelos de fluxos de trabalho dos SGFTs. Quando o processo de testes foi implementado no *Buildtime*, foi observada uma limitação na construção da malha de repetição *while*.

No *Buildtime* há a possibilidade de implementar apenas uma estrutura de repetição do tipo *do - while*. No entanto, esta limitação não deve influenciar nos resultados dos testes. Como o processo de testes não se refere a um processo real, a presença de uma malha com *while* tem uma única finalidade, providenciar uma estrutura de repetição, o que pôde ser alcançado com o *do-while*.

5.1.3 Detalhes do Cenário e Configurações

As máquinas onde foram realizados os testes estão instaladas no Laboratório do Chord situado no NCE/UFRJ[5].

Na seção 5.1.5 há uma discussão acerca da forma como são gerados os *logs*. O cenário em que ocorrem os testes está descrito na Figura 5.3, os pontos onde ocorrem as medidas para a construção dos *logs* estão localizados nas máquinas clientes e no servidor.

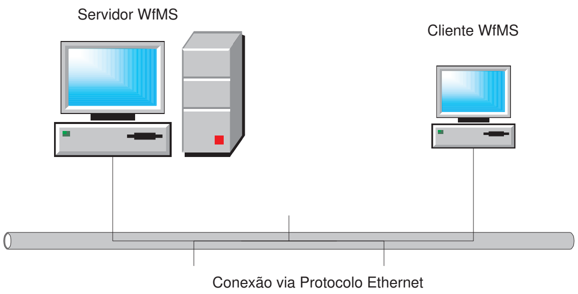

Figura 5.3: Cenário de realização dos Testes

Configurações de *Hardware/Software*

Os requisitos mínimos de *Hardware* para a instalação do servidor do MQSW, considerando a utilização do Windows 2000 como sistema operacional são:

[5]Núcleo de Computação Eletrônica

- Memória Principal (RAM): 256 MBytes

- Espaço no disco rígido: 150 MBytes(Código e amostras), 280 MBytes(Espaço vazio para o DB2) e 2GBytes (recomendado).

Durante as avaliações, os requisitos mínimos de configuração foram satisfeitos. É importante salientar que as máquinas utilizadas nos testes são máquinas consideradas modestas pelos padrões atuais. No entanto, esta consideração não distorce os valores medidos nem o objetivo do *benchmark*, pois a comparação entre diferentes SGFTs terá validade se as máquinas possuírem configurações similares. Nas tabelas 5.1.3 e 5.1.3 estão descritas as configurações das máquinas utilizadas como servidor e cliente, respectivamente.

Descrição do Processador	AMD 1700(+) XP / 1450 MHz
Memória Principal	512MBytes + 256 MBytes(Virtual)
Memória Cache	128 KBytes(L1) /256 KBytes(L2)
Disco Rígido	10 GBytes (livres)
Protocolo de Interconexão	TCP/IP/Ethernet 10 Mbps
Sistema Operacional	MS Windows 2000 Server 5.00.2195 SP4

Tabela 5.1: Configurações da máquina com o servidor SGFT

Descrição do Processador	AMD 1700(+) XP / 1450 MHz
Memória Principal	512MBytes + 256MBytes(Virtual)
Memória Cache	128 KBytes(L1) / 256 KBytes(L2)
Disco Rígido	1.4 GBytes (livres)
Protocolos de Interconexão	TCP/IP/Ethernet 10 Mbps
Sistema Operacional	MS Windows 2000 Professional 5.00.2195 SP 3
Compilador	MS Visual C++ 6.0

Tabela 5.2: Configurações da máquina com o cliente SGFT emulado

5.1.4 Implementando os Programas de Teste

Os programas que implementam as atividades automáticas e disparam pedidos de criação de instâncias de processos foram escritos em linguagem C, utilizando-se o compilador Microsoft Visual C++ 6.0 e a API para C disponível no MQSW. No apêndice A há uma descrição das APIs que o MQSW suporta.

Na descrição dos programas de teste requeridos no *benchmark*, na seção 4.9, é realizada uma descrição bastante abstrata, que não é associada a uma qualquer linguagem específica. Avaliando as APIs do MQSW, optou-se por utilizar API C, porque pareceu uma escolha que levaria a um melhor desempenho, haja visto o fato do MQSW ter sido construído predominantemente em C. Além do mais, a API em C é responsável (ver Figura A.1) pela interface entre as demais APIs e o MQSW (com exceção das mensagens em XML).

Na API C existem vários métodos utilizados para comunicar-se com o servidor, implementar atividades e acessar os documentos que uma atividade manipula [37].

Além de utilizar os métodos das bibliotecas da API em C, foi necessário utilizar algumas funções de bibliotecas específicas de C que são dependentes do sistema operacional.

O trecho do código que implementa os pedidos de criação de instâncias de processos utilizou o mecanismo de *threads* do Windows e os números randômicos foram gerados a partir da função *rand()*, que faz parte de uma biblioteca padronizada pela ANSI[6].

O emulador de clientes é consituído por dois blocos: implementação das atividades e programa que dispara pedidos de criação de instâncias de processo.

Implementação das atividades

Foi construído um único programa que efetua a execução automática em *background* das atividades. Após a execução com êxito dos mesmos, há o envio de um valor inteiro aleatório dentre os dois possíveis (0 ou 1), com probabilidade de 50% para cada cada um. Este valor retornado define a condição de término de um atividade e é a informação utilizada para descrever as condições de transição entre as atividades do modelo do processo.

Cada atividade não é completamente executada assim que é chamada. Logo após o

[6] *American National Standards Institute*: Instituto de Padronização Norte-Americano.

início da execução da mesma e utilizando os recursos do sistema operacional, a atividade é suspensa por uma determinada quantidade de tempo.

A duração do tempo em que a atividade fica suspensa possui uma distribuição normal com valor esperado de 10 segundos e desvio padrão de 4 segundos. A escolha de um intervalo de tempo para a atividade ficar suspensa simula o tempo consumido na execução de uma atividade automática. Considerando-se que os valores serão concentrados em torno da média, optou-se por uma distribuição normal [29].

O programa que implementa as atividades está descrito na Figura 5.4.

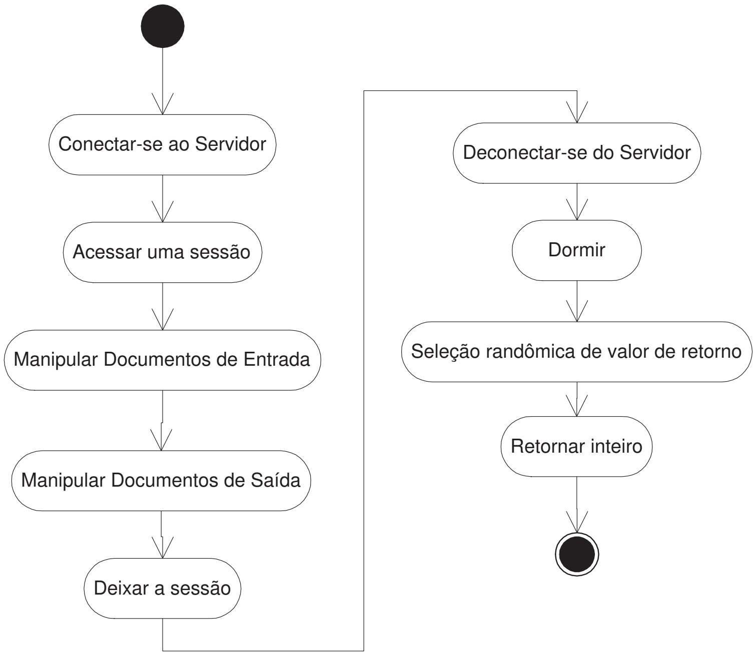

Figura 5.4: Implementação das atividades

Implementação do disparador de processos

No início da execução do programa é aberta uma sessão de comunicação com o SGFT. Em seguida, iniciam-se as chamadas dos métodos que realizam os pedidos de criação de instâncias de processos. Estes métodos são executados em um *thread* diferente do programa principal.

O *thread* do programa principal fica num *loop* infinito, a cada outro *thread* lançado, a execução do *thread* do programa principal é suspensa (utilizando o sistema operacional) por um determinado intervalo de tempo e em seguida um novo *thread* é lançado. A partir da variação deste tempo de suspensão do programa, obtém-se diferentes taxas de requisições de serviço.

Na Figura 5.5 há uma descrição da seqüência de atividades do programa que implementa o cliente do MQSW.

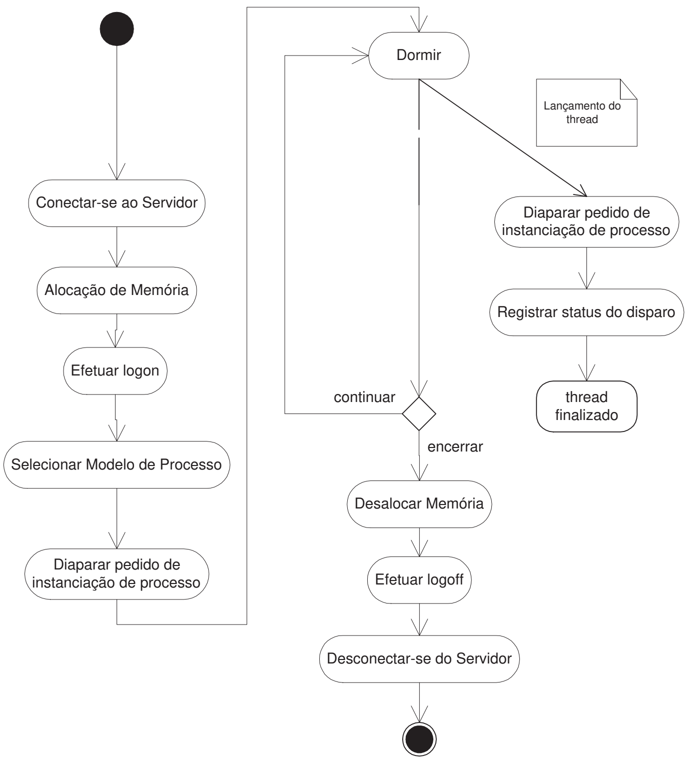

Figura 5.5: Implementação do emulador de cliente.

5.1.5 Geração de Logs

Os arquivos de *log* são gerados no SGFT e no emulador de clientes.

Para contornar possíveis inconsistências (defasagem) na informação de tempo, as máquinas foram sincronizadas no início dos testes, utilizando a ferramenta *net time* do sistema operacional Windows 2000. Cada taxa de requisições foi submetida durante intervalos de tempo e realizado uma série de três repetições para cada teste.

O *log* gerado no emulador possui as seguintes informações: quantidade de pedidos de criação de instância de processos, instante do tempo medido em segundos e mensagens de erro ocorridas nas tentativas de criação de instância de processo.

O *log* gerado no SGFT é gravado na Tabela *audit_trail* e possui vários campos que descrevem os eventos ocorridos durante a encenação dos processos. Identificação das instâncias de processos, papéis, atores e toda uma série de detalhes relacionados à administração do SGFT e à interação com a execução das atividades [36]. Campos da Tabela *audit_trail* que foram avaliados para obter as métricas do *benchmark*:

– CREATED: indica o instante em que foi ocorreu o evento.

– EVENT: código que descreve os eventos associados às instâncias de processos.

– PROCESS_NAME: nome do processo concatenado com uma cadeia de caracteres gerados sequencialmente, o que evita conflitos na identificação das instâncias de processos ativas.

– ACTIVITY_STATE: estado da atividade.

As informações de *audit_trail* foram recuperadas utilizando-se a ferramenta de consulta *SQL* (processador de linha e comando) do SGBD DB2 e exportadas para um arquivo em formato de texto plano. Em seguida, um *script* filtrou este arquivo e gerou um novo conjunto de informações com os dados necessários à caracterização da métrica $IPsu/min$ e análise de desempenho do SGFT.

5.2 Análise de Desempenho

Após os *scripts* filtrarem os *logs*, foram geradas tabelas que possibilitaram a obtenção de curvas que informam o comportamento do motor do SGFT ao longo da avaliação realizada. Para plotar os gráficos foi utilizada a ferramenta *Gnuplot* [38].

O conjunto de avaliações de desempenho implementadas podem ser caracterizadas como avaliações analíticas, onde o SGFT MQSW foi submetido a testes em diferentes contextos de configuração.

Benchmark detectando problemas de configuração

A curva descrita na Figura 5.2 exibe a capacidade de resposta do MQSW em termos de Instâncias de Processos finalizadas com sucesso por hora ($IPsu/hor$) quando a taxa de pedidos de criação de instância de processos ($CIPpe/min$) cresce.

A métrica apresentada na seção 4.5 é diferente daquela utilizada na descrição da curva na Figura 5.2. Esta discrepância deve-se a questões de escala, pois a capacidade de resposta do SGFT foi bastante aquém do esperado.

Este comportamento, aparentemente distorcido, orientou a análise de desempenho no sentido da reavaliação dos ajustes dos parâmetros do MQSW. Foi averiguado que havia um problema na manutenção dos índices das tabelas utilizadas pelo SGBD DB2. Como havia alguns meses que não havia sido realizada esta manutenção, o DB2 gerava grandes atrasos nas consultas realizadas. Este fato influenciou as medições, causando distorções nas medidas de capacidade de resposta do MQSW.

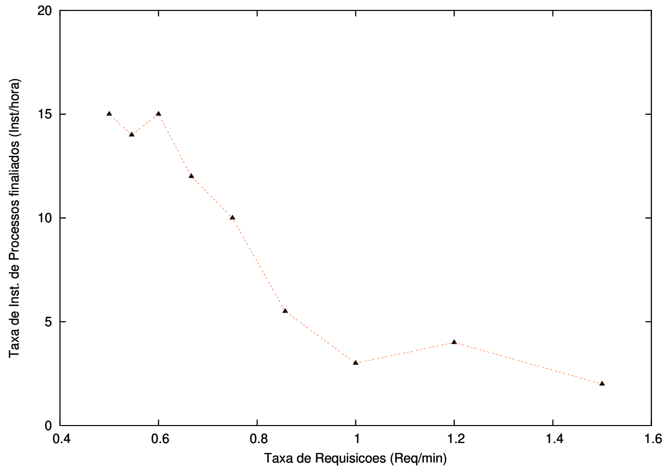

Figura 5.6: Vazão (Taxa de Instâncias) no MQSW - I

Para analisar a possível dispersão dos dados foi plotada a curva descrita na Figura 5.7, que exibe os tempos consumidos na execução das instâncias de processos. Ainda nesta figura, encontra-se uma curva que representa a duração média de uma atividade quando o SGFT está ocioso.

Apesar de poder ser inferido, através de interpolação, um comportamento que sugere a elevação dos tempos consumidos na execução de uma instância de processo, há uma grande dispersão das medidas levantadas.

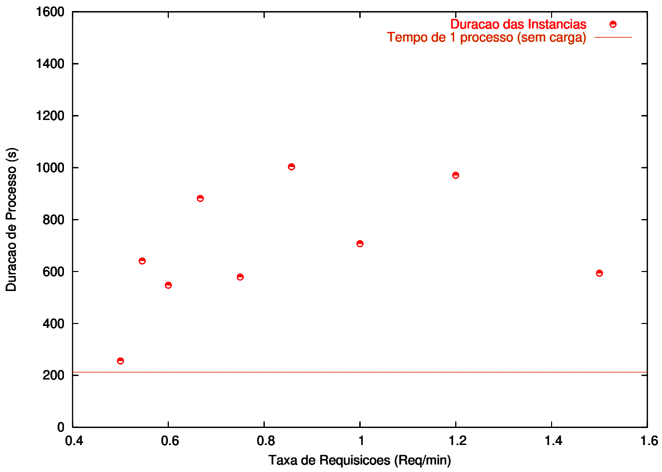

Figura 5.7: Tempo de Duração das Instâncias no MQSW - I

Na Figura 5.8, é exibida a curva que descreve o comportamento da taxa de atividades executadas por hora quando $CIPpe$ é variada. Cada atividade, que tem a sua execução habilitada, promove a criação de um *workitem*.

A taxa de atividades executadas tendeu a um valor constante, o que pareceu razoável. Contudo, era esperado que com o aumento da carga de trabalho, a quantidade de atividades executadas por minuto diminuísse em virtude da falta de recursos do SGFT sobrecarregado. Mais uma vez, é mostrado uma possível distorção dos dados em decorrência dos ajustes necessários ao DB2.

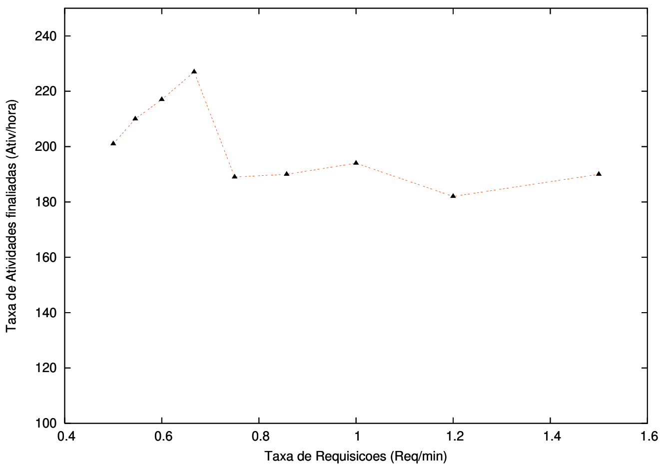

Figura 5.8: Vazão (Taxa de Atividades) no MQSW - I

Benchmark exercitando e avaliando um SGFT

Na Figura 5.9 são descritas duas curvas, uma com taxa uniforme de chegada de pedidos e outra com taxa que segue uma distribuição de Poisson. A taxa com distribuição de Poisson atinge o estado de sobrecarga mais rapidamente que a taxa uniforme. Pode-se avaliar ainda que quando uma determinada carga de pedido de criação de instâncias de processo (em torno de 300 $CIPpe/min$) é submetida ao SGFT, a sobrecarga é bastante onerosa e há grande disputa por recursos.

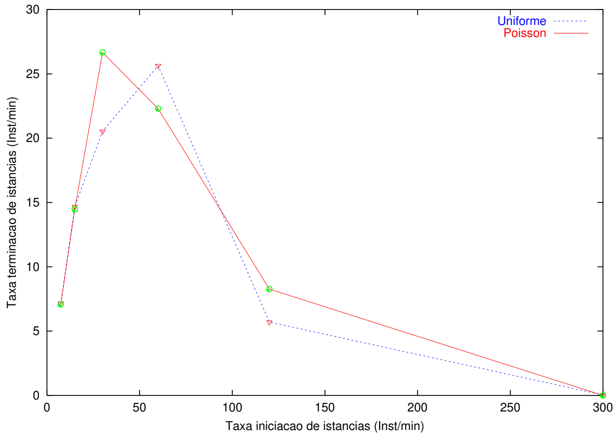

Figura 5.9: Vazão (Taxa de Instâncias) no MQSW - II.

Na Figura 5.10 são descritas curvas que mostram a taxa de atividades finalizadas com sucesso. Mais uma vez a taxa de chegada uniforme tem um comportamento mais lento que a de Poisson.

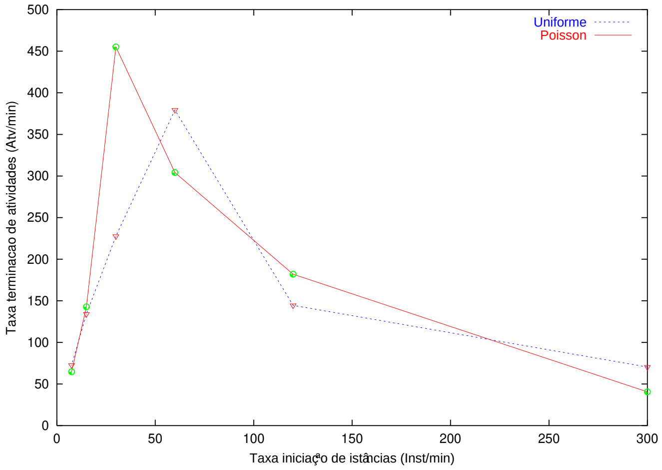

Figura 5.10: Vazão (Taxa de Atividades) no MQSW - II.

Na Figura 5.11 foram plotadas curvas de criação de instâncias de processo aceitas ($CIPac/min$) com diferentes taxas de chegada (uniforme e Poisson). Neste caso fica claro o efeito da hierarquia de memórias da máquina na taxa uniforme.

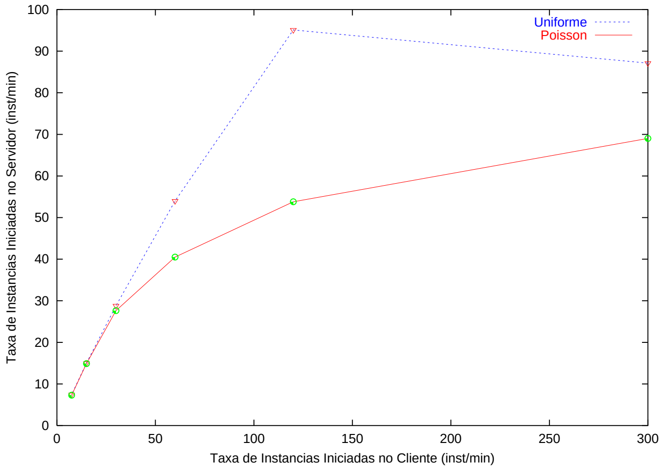

Figura 5.11: Efeito da hierarquia de memória na sobrecarga.

Na Figura 5.12 é exibido o quadro de consumo de recursos do gerenciador de processos do sistema operacional. Pode-se perceber que cerca de 99 % dos recursos da máquina estão disponibilizados para atender ao SGFT, o que prova que o modelo de carga do *benchmark* exercita o motor do MQSW e a métrica escolhida representa o desempenho da máquina.

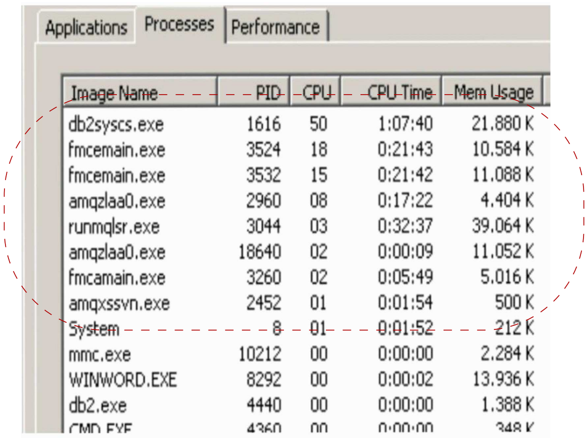
Applications | Processes | Performance

Image Name	PID	CPU	CPU Time	Mem Usage
db2syscs.exe	1616	50	1:07:40	21.880 K
fmcemain.exe	3524	18	0:21:43	10.584 K
fmcemain.exe	3532	15	0:21:42	11.088 K
amqzlaa0.exe	2960	08	0:17:22	4.404 K
runmqlsr.exe	3044	03	0:32:37	39.064 K
amqzlaa0.exe	18640	02	0:00:09	11.052 K
fmcamain.exe	3260	02	0:05:49	5.016 K
amqxssvn.exe	2452	01	0:01:54	500 K
System	8	01	0:01:52	212 K
mmc.exe	10212	00	0:00:00	2.284 K
WINWORD.EXE	8292	00	0:00:02	13.936 K
db2.exe	4440	00	0:00:00	1.388 K
CMD.EXE	4360	00	0:00:00	348 K

Figura 5.12: Consumo de recursos da máquina com o SGFT durante os testes

Capítulo 6

Conclusões

6.1 Sobre o trabalho realizado

O trabalho apresentado propôs a especificação de um *benchmark* para avaliar o desempenho de Sistemas de Gerenciamento de Fluxo de Trabalho.

Um SGFT, assim como um SGBD[1], é um sistema que processa transações. No entanto, as transações em um SGFT são utilizadas com um fim muito específico: armazenar e recuperar informações de controle de roteamento de atividades. Isto torna impraticável a aplicação das métricas e dos testes definidos nos *benchmarks* convencionais de banco de dados (p. ex: TPC-C, Wiscosin) nas avaliações de SGFTs.

As avaliações de desempenho encontradas na literatura ainda são bastante parecidas com o TPC-C. Nestas, é descrito um processo com atividades constituídas de transações complexas e, ao final, é exibida a capacidade de processamento em termos da taxa das transações (ocorridas nas execuções das atividades dos clientes) finalizadas com sucesso. Métricas e testes desta natureza estão mais voltados para avaliar a infra-estrutura que apóia a execução das atividades do que o SGFT propriamente dito. Um *benchmark* que avalie o desempenho do SGFT propriamente dito deve avaliar o desempenho do controle da execução das atividades e não o desempenho da execução das próprias atividades.

[1]Sistema de Gerenciamento de Banco de Dados.

Em um SGFT, o motor (*workflow engine*) é o elemento da arquitetura responsável pela interpretação da definição dos modelos de fluxos de trabalho e pelo roteamento das atividades. Impor carga de processamento para o SGFT, independentemente de um processo específico, implica em exigir uma maior capacidade de resposta do motor em relação a estes aspectos.

Ao longo do trabalho, é mostrado que o estado de sobrecarga do motor pode ser alcançado pela diminuição dos intervalos entre os pedidos de criação de instâncias de processo. Esta situação força o incremento do número de transações necessárias para recuperar e armazenar dados que descrevem os estados de uma atividade e de uma instância de processo. Consequentemente, os elementos do motor que processam estes dados também são sobrecarregados.

Foi mostrado ao longo do trabalho que para exercitar o motor de um SGFT não é sequer necessário que as atividades executadas pelos clientes sejam robustas. Estas atividades podem ser até mesmo implementadas como meros *stubs*[2].

A partir da caracterização da sobrecarga do motor, encaminhou-se a pesquisa no sentido de detalhar um *benchmark* de avaliação de desempenho para SGFTs. Durante esta especificação, procurou-se atender aos requisitos de **portabilidade, relevância, escalabilidade, simplicidade e reprodutibilidade**.

Para detalhar o *benchmark*, foram considerados os seguintes itens: definição de termos, definição do modelo de comunicação entre um cliente e o servidor do SGFT, descrição das métricas que se desejava levantar (taxas de instâncias de workflows completas e atividades finalizadas com sucesso), modelo de processo de teste, descrição de configurações de *hardware/software*, modelo de carga e descrição do cenário de testes.

Na definição de um modelo conceitual que retrata a comunicação entre o cliente do SGFT e o servidor são abstraídos os detalhes de implementação. Logo, tal modelo restringiu-se às mensagens (*requests*/*responses*) trocadas e a relação destas com o trabalho do *motor*. A definição das métricas também está fortemente relacionada com este modelo.

O processo utilizado para os testes possui as seguintes características: um conjunto de padrões de roteamento de fluxos presente em boa parte dos processos de negócio reais, ausência de *deadlock* e a presença de mais de uma rota para alcançar uma atividade final. Apesar do processo sugerido não ter relação explícita com um processo real, as suas

[2]Um programa ou rotina que não faz processamento útil, mas possui os elementos necessários para ser compilado, ligado e executado.

características são relevantes para a avaliação de um SGFT.

O objetivo do modelo de carga utilizado foi aproximar-se de uma situação real, com os pedidos dos clientes chegando a uma taxa que não fosse constante, mas controlável e computacionalmente factível.

Após a fase da especificação, implementou-se o *benchmark* na ferramenta *IBM MQSeries Workflow* (MQSW), disponibilizada para o experimento nos laboratórios do CHORD através do *IBM Scholar Program*.

Inicialmente, foi necessário compreender os detalhes da arquitetura, da implementação e da utilização das ferramentas do MQSW. A documentação da ferramenta fornecida pela empresa foi relevante nesta fase da pesquisa. Contudo, muitos detalhes do funcionamento do MQSW só foram observados à medida que ocorriam os problemas.

Antes da escolha da API C, imaginou-se utilizar a API JAVA para implementar o emulador de clientes e as atividades. Entretanto, analisando as APIS suportadas, optou-se por C. Esta escolha pareceu a mais razoável, haja visto o MQSW ser implementado predominantemente em linguagem C.

6.1.1 Dificuldades encontradas

Inicialmente, a maior dificuldade encontrada foi a configuração correta dos parâmetros do SGFT MQSW.

Durante os testes, o problema mais frequente foi o vazamento de memória no emulador de cliente. Em um primeiro momento os testes foram executados em uma máquina com 192 MBytes (memória principal) e, em seguida, utilizou-se uma máquina de 512 MBytes. O problema parecia ter sido resolvido até que novos vazamentos de memória voltaram a ocorrer, acarretando o estouro (*overflow*) da memória virtual da máquina.

Foi necessário revisar o código do emulador de clientes e avaliar a política de alocação e gerência de memória. Foi constatado que quando ocorriam as negações de serviço (*denial of services*), durante os estados de sobrecarga do SGFT, a memória que tinha sido alocada não era liberada, nem lançava exceção capaz de ser tratada.

Os testes foram realizados em máquinas com poder computacional bastante modestos. Apesar deste fato não invalidar as avaliações, foi necessário balancear a execução das atividades entre clientes emulados em várias máquinas, para alcançar uma determinada quantidade de pedidos de criação de instância de processo ($CIPpe/min$).

Uma outra questão que merece atenção é o espaço consumido pelos *logs* gerados pelo SGFT ao longo dos testes. Nos testes com o MQSW foram gerados aproximadamente 500 MBytes de *log*. Além disso, foi necessário apagar as tabelas *audit trail* ao final de cada etapa dos testes. Isto garantiu que o tamanho das tabelas do SGBD DB2 não interferisse nas medidas.

6.2 Sobre o trabalho que pode ser feito

Com base nos resultados encontrados, uma proposta de trabalho futuro é a realização de medições com uma gama maior de variação de parâmetros (p.ex.: Sistema Operacional, Protocolo de Interconexão, diferentes APIs) e diferentes SGFTs.

A variação de parâmetros sugere ainda a utilização do *benchmark* para comparação entre ferramentas *Open Source* (e *Free Software*) e ferramentas proprietárias, com uma consequente análise do custo destas escolhas.

Um outro trabalho possível seria a avaliação de SGFTs com características adaptativas, avaliando o efeito de mecanismos de adaptação no desempenho dos motores dos servidores.

6.3 Contribuições

A principal contribuição deste trabalho foi a definição de um conjunto de critérios que permitem a avaliação de desempenho de um SGFT, satisfazendo aos requisitos de portabilidade, relevância, simplicidade, escalabilidade e reprodutibilidade. Além disso, uma série de outras contribuições podem ser citadas:

- Uma ferramenta para identificar possíveis gargalos do desempenho de um SGFT.
- Descrição de uma forma de comparar diferentes configurações de um mesmo SGFT.
- Comparação do desemepenho entre diferentes SGFTs.

Referências Bibliográficas

[1] M. Gillmann, R. Mindermann e G. Weikum, "Benchmarking and configuration of workflow management systems", in *Proceedings of the 7th International Conference on Cooperative Information Systems*, vol. 1901, Eilat, Israel, pp. 186–197, September 2000.

[2] C. Ellis, *Workflow Technology in Computer Supported Co-operative Work, Michel Beaudouin Lafon (ed.)*. John Wiley and Sons, 1 ed., 1999.

[3] Workflow Management Coalition, *WFMC-TC-1011 Terminology and Glossary, versão 3.0*, February 1999. Disponível no sítio http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf, acesso em 07 de abril de 2004.

[4] WfMC, "Workflow Management Coalition". Disponível no sítio http://www.wfmc.org, acesso em 10 de janeiro de 2004.

[5] X. Wuang, "Implementation and performance evaluation of corba-based centralized workflow schedulers", Tese de Mestrado, University of Georgia, Athens, Georgia, USA, 1995.

[6] D. Chaffey, *Groupware, Workflow and Intranets: Reengineering the Enterprise with Collaborative Software*. Butterworth-Heinemann, 1 ed., 1998.

[7] World Wide Web Workflow, *Livre Blanc*, 2002. Disponível no sítio http://www.w4global.com, acesso em 07 de abril de 2004.

[8] T. Cruz, *Workflow A Tecnologia que vai Revolucionar Processos*. Editora Atlas, 2 ed., 2000.

[9] Doculabs, *Performance Assessment of Staffware Process Suite*, 2002. Disponível no sítio http://www.staffware.com/downloads/, acesso em 07 de abril de 2004.

[10] P. Tran e J. Gosper, "EAI/BPM Evaluation Series", tech. rep., Commonwealth Scientific and Industrial Research Organisation - CSIRO, November 2002. Disponível no sítio http://www.cmis.csiro.au/mte/reports/BPM_IBMwebsphereMQ332.htm, acesso em 07 de abril de 2004.

[11] D. Hollingsworth, *WFMC-TC00-1003 The Workflow Reference Model, versão 1.1*. Workflow Management Coalition, January 1995. Disponível no sítio http://www.wfmc.org/standards/docs/tc003v11.pdf, acesso em 07 de abril de 2004.

[12] A. J. Bonner, A. Shrufi e S. Rozen, "Labflow-1: A database benchmark for high-throughput workflow management", in *Proceedings of the 5th International Conference on Extending Database Technology*, vol. 1057, Avignon, France, pp. 463–478, March 1996.

[13] C. Ellis e G. Nutt, "Workflow: The process spectrum", in *Proceedings of the NSF Workshop on Workflow and Process Automation in Information Systems*, Athens, Georgia, USA, pp. 140–145, May 1996.

[14] R. M. de Araújo e M. R. da Silva Borges, "Sistemas de workflow", in *Anais da XX Jornada de Atualização Científica - XXI Congresso da Sociedade Brasileira de Computação*, Fortaleza, Ceará, Brasil, pp. 463–478, Agosto 2001.

[15] W. M. P. Van Der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski e A. P. Barros, "Workflow patterns", *Distributed Parallel Databases*, vol. 14, no. 1, pp. 5–51, July 2003.

[16] D. Georgakopoulos, M. F. Hornick e A. P. Sheth, "An overview of workflow management: From process modeling to workflow automation infrastructure", *Distributed and Parallel Databases*, vol. 3, no. 2, pp. 119–153, April 1995.

[17] M. zur Muehlen, "Workflow-architectures: Stand-alone vs. embedded systems", 2000. Universität Münster, Münster, Germany. Disponível em http://www.wi.uni-muenster.de/is/vorlesungen/WF/Winter2000, acesso em 07 de abril de 2004.

[18] Ultimus Inc., *Ultimus Version 5.o Product Guide*, Jun 2001. Disponível no sítio http://www.quixa.com/library/docs/QSLIB-ULT-102.pdf, acesso em 07 de abril de 2004.

[19] L. Svobodova, *Computer Performance Measrement and Evaluation Methods: Analysis and Applications*. Elsevier Scientific Publishing Company, 1976.

[20] "Merriam-webster online dictionary", 2004. Disponível no sítio http://www.webster.com, acesso em 07 de abril de 2004.

[21] J. Gray, *The Benchmark Handbook for Database and Transaction Processing Systems*. Morgan Kaufmann, 2 ed., 1993.

[22] TPC, "Transaction Processing Performance Council", 2004. Disponível no sítio http://www.tpc.org, acesso em 09 de janeiro de 2004.

[23] Transaction Processing Council, *TPC Benchmark C – Standard Specification Revision 5.2.*, December 2003. Disponível no sítio http://www.tpc.org/tpcc/spec/tpcc_current.pdf, acesso em 07 de abril de 2004.

[24] SPEC, "Standard Performance Evaluation Corporation", 2004. Disponível no sítio http://www.spec.org/CPU2000/docs, acesso em 10 de janeiro de 2004.

[25] R. Metzger, "MQSeries Workflow - Performance Estimates for Solution and Capacity Assesments", tech. rep., IBM, October 2002. Disponível no sítio ftp://ftp.software.ibm.com/software/integration/support/supportpacs/individual/wp01.zip, acesso em 07 de abril de 2004.

[26] J. A. Miller, A. P. Sheth, K. Kochut, X. Wang e A. Murugan, "Simulation modeling within workflow technology", in *Proceedings of the 27th Winter Simulation Conference*, Arlington, Virginia, USA, pp. 612–619, December 1995.

[27] W. M. P. van der Aalst, K. van Hee e G. J. Houben, "Modelling and analysing workflow using a petri-net based approach", in *Proceedings of the 2nd Workshop on Computer-Supported Cooperative Work, Petri Nets and Related Formalisms*, Zaragoza, Spain, pp. 31–50, June 1994.

[28] D. Hollingsworth, "The workflow reference model: 10 years on", 2003. Disponível no sítio http://www.wfmc.org/standards/docs/, acesso em 16 de fevereiro de 2004.

[29] P. L. Meyer, *Introductory Probability and Statistical Applications*. Addison-Wesley, 1 ed., 1965.

[30] S. M. Ross, *Introduction to Probability Models*. Academic Press, 1 ed., 1989.

[31] G. Gordon, *System Simulation*. Prentice-Hall, 1 ed., 1969.

[32] R. E. Shannon, *Systems Simulation*. Addison-Wesley, 1 ed., 1975.

[33] M. Crovella e A. Bestavros, “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes”, in *Proceedings of SIGMETRICS’96: The ACM International Conference on Measurement and Modeling of Computer Systems*, Philadelphia, Pennsylvania, USA, pp. 160–169, May 1996.

[34] A. S. Tanenbaum, *Organização Estruturada de Computadores*. Editora LTC, 4 ed., 1998.

[35] D. Wackerow, W. Adam e D. Burton, *MQSeries Workflow for Windows NT for Beginners*. International Business Corporation, 5 ed., Mar 2001.

[36] International Business Corporation, *IBM MQSeries Workflow 3.3 - Administration Guide*, 5 ed., Mar 2001.

[37] International Business Corporation, *IBM MQSeries Workflow 3.3 - Programming Guide*, 7 ed., Mar 2001.

[38] GNU, *GNUplot Documentation*, 2004. Disponível no sítio http://www.gnuplot.info/, acesso em 17 de março de 2004.

Apêndice A

API do MQSeries Workflow

<table>
<tr><td rowspan="3">X
M
L</td><td>ActiveX</td><td>Java</td><td>Lotus Notes</td><td>Visual Basic V2</td></tr>
<tr><td colspan="3">Linguagem C++ V3/V2</td><td>Linguagem C V2</td></tr>
<tr><td colspan="4">Linguagem C V3</td></tr>
</table>

Figura A.1: APIs do MQS Workflow

AS APIs providenciadas pelo MQSeries Workflow suportam as seguintes linguagens: linguagem C, linguagem C++, controloes ActiveX e Objetos OLE, Java, Interface com mensagens XML. As interfaces básicas para requisições de serviço *Runtime* do MQSW são interfaces em API em linguagem C e mensagens XML. Acesso à esta camada de API em C pode ser realizado por vários compiladores de C.
Uma API para a linguagem C++ é providenciada no topo da API de C. Desde que a API do C++ é uma pequena camada de métodos *inline*. As APIs de Java e Active X são implementadas no topo da cama de C++. MQSW utiliza o padrão de mensagens XML 1.0 como linguagem de descrição de documento.

As chamadas para as APIs do MQSW providenciam chamadas API para:

Instanciar modelos de processos, acessar, manipular *worklists* e *workitems*

Monitorar as execuções dos processos

Providenciar funções administrativas

Receber informação enviada por um servidor MQSW

Processar os dados dos documentos associados com a implementação de uma atividade

Existem essencialmente duas tarefas distintas que estão relacionadas à utilização das APIs:

1. Pode-se escrever um cliente apropriado, ao invés de utilizar-se o *Runtime* ou os comandos no *prompt do shell*. Pode-se, por exemplo:controlar as funcionalidades providenciadas para um usuário, apresentar um ambiente de interação customizado, executar determinadas tarefas sem a intervenção de um usuário.

2. Pode-se escrever uma programa que implementa uma atividade ou uma ferramenta de suporte para um determinado modelo de processo.