UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Carlos André de Sousa Rocha

ANÁLISE DE DESEMPENHO EM AMBIENTES CLIENTE/SERVIDOR 2-CAMADAS E 3-CAMADAS

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Paulo José de Freitas Filho, Dr. Eng.

Florianópolis, agosto de 2002

ANÁLISE DE DESEMPENHO EM AMBIENTES CLIENTE/SERVIDOR 2-CAMADAS E 3-CAMADAS

Carlos André de Sousa Rocha

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, Área de Concentração Sistemas de Computação, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____

Prof. Fernando Álvaro Ostuni Gauthier, Dr. Eng.
Coordenador do Curso

Banca Examinadora

_____

Prof. Paulo José de Freitas Filho, Dr. Eng.
Orientador

_____

Prof. João Bosco da Mota Alves, Dr.

_____

Prof. Luiz Fernando Jacintho Maia, Dr.

Dedicatória

Dedico esta pesquisa à minha amada mãe Edisa e ao meu amado irmão caçula Alexandre pelo apoio incondicional não só na realização deste trabalho de dissertação, mas por todo o apoio ao longo da minha vida, e por não duvidar em nenhum momento da minha capacidade em concluí-lo.

Agradecimentos

À Direção do Centro de Ensino Superior do Pará (CESUPA), hoje orgulhosamente Centro Universitário do Pará, pela a coragem em promover o mestrado, juntamente com a Universidade Federal de Santa Catarina, e por trabalhar pelo desenvolvimento da região.

Ao amigo e orientador Paulo José de Freitas Filho, por comprar a idéia deste trabalho de pesquisa, pelo apoio dado nas horas de desespero, pela confiança e paciência mostradas e também pelos puxões de orelha nos momentos de necessidade.

Ao meu melhor amigo Paulo Francioli Honorato Jr. pelo incentivo inicial de encarar o mestrado e pelo apoio nos momentos em que pensei desistir.

Às amigas e conterrâneas Chris e Cynthia , amigas é uma definição pequena para todo carinho que tenho por elas, que me receberam, e me acolheram, no seu apartamento durante meses, o que permitiu grande parte da elaboração desta pesquisa.

Aos amigos Lídio (Toledo) Mauro, Gustavo (Guga) Campos e Luis Lacerda pelas idéias sugeridas, pelo apoio aos trabalhos não concluídos, pela confiança depositada e pela leitura e críticas dos primeiros capítulos depois daquela bendita noite no CESUPA, não é galera? Naquela noite teve inicio este trabalho.

Ao amigo Mauro Wilkens, pelas noites na frente do computador, algumas vezes até resfriado, ensinado o caminho das pedras no ARENA e por todos os livros e CD´s emprestados.

À grande amiga professora Silvia Modesto Nassar, diga-se de passagem também originária da terrinha (Belém do Pára), por todo o apoio, pela sala e computador cedidos, pela explicação sobre os métodos estatísticos usados neste trabalho, pelas conversas que matavam as saudades de Belém e pelo sempre altíssimo astral.

Ao conterrâneo professor João Bosco da Mota Alves, por participar da banca examinadora, pelas sugestões dadas e pelo apoio após a defesa.

À amiga Rita de Cássia Cerqueira Gomes pelo material cedido e pelas conversas de apoio.

Ao professor Luiz Fernando Jacintho Maia por participar da banca examinadora.

Sumário

Lista de Tabelas

Lista de Figuras

Resumo

Por volta de 1980, juntamente com o aparecimento dos PC (computadores pessoais), tem-se o surgimento do modelo cliente/servidor baseado em 2-camadas: o cliente e o servidor (representando na maioria das vezes por um servidor de banco de dados). Apesar dos benefícios conseqüentes deste novo modelo, como o surgimento de diversos ambientes de desenvolvimento pertencentes a empresas diferentes, permitindo uma melhor escolha da ferramenta adequada para uma determinada aplicação, a maneira de desenvolver softwares exigiu uma significativa mudança, pois passou a considerar elementos antes ignorados no modelo legado como: largura de banda das redes, capacidade de processamento das estações clientes, etc. Estas novas variáveis provocaram no sistema cliente servidor 2-camadas uma evolução para o modelo baseado em 3-camadas, onde a carga de processamento seria dividida por um terceiro elemento entre o cliente e o servidor chamado de servidor de aplicação. Esta nova camada atenderia as solicitações dos clientes e as gerenciaria com o lado servidor.

Este trabalho visa através das metodologias de Planejamento de Capacidade e Avaliação de Desempenho de Sistemas Computacionais, utilizando técnicas de simulação, através do software de simulação ARENA, comparar o desempenho dos dois modelos citados, 2-camadas e 3-camadas. Para tanto foram desenvolvidos um Modelo de Carga e um Modelo de Desempenho, tendo como base para construção destes dois modelos um levantamento realizado com o Núcleo de Informática do Departamento de Trânsito do Estado do Pará. O modelo de carga, e seus componentes básicos, é representado por instruções SQL do tipo *Insert* e o modelo de desempenho representa uma parte do ambiente computacional corporativo, apenas a LAN foi considerada, do Departamento do Estado do Pará, utilizando um determinado servidor de banco de dados. Partindo-se destes modelos, serão desenvolvidos para as arquiteturas citadas dois modelos no simulador, e as variáveis de desempenho envolvidas, como tempo de resposta, medidas e comparadas para se ter uma medida de desempenho.

Abstract

Around 1980, along with the birth of PC (Personal Computers), comes the client/server model, which was built based on two tiers: the client and the server (usually represented by a database server). Despite this new model's benefits, among them the creation of several development environments by different companies, allowing a better choice of an adequate tool to certain applications, it demanded a significant change in software development, since elements that were ignored in legacy applications should then be considered, such as: network bandwidth, processing capacity in client workstations, among others. These new factors originated an evolution from the two-tier model to the three-tier model, where the processing load would be distributed to a third element, which stood between the client and the server: the application server. This new tier would respond to client requests and manage those requests along with the server side.

Following a methodology called Planning Capacity and Performance Analysis in Computer Systems and using the simulation software ARENA, this thesis proposes a comparison between the performance of two-tier and three-tier models. With that goal in mind, two models were developed: a Load Model and a Performance Model, which were based on information obtained with the Computing Department of State of Pará's Transit Authority. The Load Model contains basic components, which represent *Insert* SQL statements. The Performance Model represents part of State of Pará's Transit Authority corporate computing environment (only the LAN was considered), which uses a database server. From these two models, two simulation models will be developed. Performance factors such as reponse time will be measured and compared so that a performance baseline can be established.

1. Introdução

O que representou o paradigma cliente/servidor dentro do panorama computacional mundial? Uma revolução? Uma evolução? Ambas indicam mudança. A revolução indica uma mudança radical, violenta e inovadora de conceitos e pensamentos,ou seja, uma transformação mais imediata de um conceito em outro. A evolução considera um aspecto gradual dentro de um determinado tempo, considerando ainda que a próxima geração contenha traços característicos da anterior. Da análise de ambas chega-se à conclusão de que o paradigma cliente/servidor 2-camadas tratou-se de uma revolução dentro do modelo computacional tradicional, modelo legado, ”evoluindo” mais tarde para o modelo baseado em 3-camadas e N-camadas. Tais mudanças serão exploradas ao longo deste trabalho de dissertação.

1.1. Motivação

O que significa planejar e desenvolver Sistemas de Informação (SI) nos dias atuais? Considerando todo o aparato tecnológico representado pelos mais variados dispositivos de hardware existentes e as mais variadas técnicas de análise para desenvolvimento de projetos computacionais conceituais, é fácil desenvolver e planejar SI dentro de um ambiente corporativo?

O paradigma centralizado ou tradicional, baseado no modelo computacional dos mainframes e minicomputadores, ignorava elementos hoje fundamentais para o projeto de um SI: o tipo de informação inerente a um determinado ambiente, o desempenho deste SI dentro do ambiente, etc, este paradigma considerava como fator mais relevante o custo do ambiente computacional.

Por volta de 1967 o especialista em computação Herb Grosch afirmou: o poder computacional de um processador é proporcional ao quadrado de seu preço. Tal afirmação que ficou conhecida como a Lei de Grosch encaixava-se muito bem para o ambiente dos mainframes, onde quanto mais caro a máquina mais rápida ela era. Hoje se pode adquirir um microcomputador que atinja um desempenho mais significativo do que muitos mainframes da década de 1980 por um preço bem mais viável.

Na tentativa de estabelecer uma comparação entre as mudanças no tipo de informação do ambiente tradicional para um ambiente cliente/servidor, dentro do contexto deste trabalho de pesquisa, defini-se como informação dados do tipo: nome, valores ou instruções SQL (*Structured Query Language*) que trafegam através de uma rede de computadores qualquer, abstraindo dentro do escopo desta pesquisa o tipo de informação trocada pelas primitivas de serviço nas camadas de uma determinada pilha de protocolos. Então, pode-se dizer que a informação "evoluiu" de um caráter estático para um distribuído, considerando a Internet e suas variações intranets, extranets e mais a necessidade comum em distribuir e compartilhar informações entre LANs, WANs e as mais variadas redes que se interconectam.

Levando-se em consideração este panorama de informações distribuídas, a forma de projetar SIs dentro da arquitetura cliente/servidor envolve uma considerável complexidade, abrindo um leque para os mais variados estudos iniciais sobre:

- A informação que trafegará em uma LAN, WAN, uma interconexão entre LANs ou entre WANs;

- Qual o número de requisições num dado espaço de tempo que um servidor precisará suportar;

- Que tipo de cliente é o mais adequado para um domínio de aplicação qualquer, etc.

- Que tipo de arquitetura utilizar?

Isto é, a heterogeneidade (software e hardware) típica do ambiente cliente/servidor deve ser previamente analisada e caracterizada para se estabelecer um domínio de aplicação. Mas e quando a abordagem 2-camadas, não é mais satisfatória? O que fazer?

Enquanto o "Mundo Computacional" assimilava as mudanças nos conceitos provocados pelo paradigma cliente/servidor 2-camadas, novas mudanças provocadas pelo surgimento da *World Wide Web* na Internet fizeram com que outras arquiteturas ganhassem destaque, a 3-camadas e N-camadas.

Esta nova abordagem adequou-se melhor considerando o aspecto distribuído inerente do modelo cliente/servidor, pois introduz uma camada intermediária entre o cliente e o servidor, aumentando o balanceamento de informações trocado por ambos. Esta pesquisa

aborda os paradigmas 2-camadas e 3-camadas, com ênfase ao modelo baseado em 3-camadas como solução para ambientes com informações distribuídas.

Uma motivação para o desenvolvimento deste trabalho foi o resultado insatisfatório obtido na tentativa de substituir o sistema de informação do Departamento de Trânsito do Estado do Pará baseado em mainframe, que não supre de forma satisfatória a instituição com informações gerenciais e representar um elevado custo para a mesma, para um sistema baseado no paradigma cliente/servidor 2-camadas o que acabou conduzindo à busca de uma solução baseada no modelo 3-camadas. Abaixo são descritos alguns dos resultados insatisfatórios mencionados anteriormente:

- A alta taxa no tempo de resposta;
- Complexo processo de atualização do software cliente;
- E a baixa escalabilidade dentro do ambiente computacional.

O resultado satisfatório no aumento de gerenciamento da manutenção do software do Módulo Agência (sistema utilizado para consultas de saldos e transferências de contas correntes dos correntistas do Banco do Estado do Pará), que é parte integrante da solução integrada Multiserv (um acrônimo para multi-serviços) responsável pelo pagamento dos funcionários dos órgãos conveniados ao Banco do Estado, de uma proposta baseada em 2-camadas para uma baseada em 3-camadas também serviu como incentivo a este trabalho de pesquisa.

1.2. Problema a Ser Abordado

A dificuldade em desenvolver um sistema de informação no paradigma cliente/servidor 2-camadas para ambientes computacionais com informações distribuídas por pontos remotos, devido às limitações do modelo. E o modelo cliente/servidor N-camadas como solução para estas dificuldades e limitações, tendo como foco central o modelo 3-camadas, sua integração com servidores de bancos de dados e a infraestrutura de rede necessária para ambos os modelos.

1.3. Objetivos

1.3.1. Objetivo Geral

Dado um ambiente de rede avaliar o desempenho do paradigma cliente/servidor 2-camadas e 3-camadas definindo seus domínios de aplicação.

1.3.2. Objetivos Específicos

- Revisão da literatura como alicerce para o entendimento do ambiente;
- Abordar os pontos falhos do modelo cliente/servidor 2-camadas sugerindo uma solução;
- Mostrar o que são servidores de aplicação, suas vantagens e desvantagens;
- Construir um modelo de simulação para cada arquitetura;
- Criar cenários com os modelos de simulação, para avaliação das medidas de desempenho;
- Comparar as duas alternativas de sistemas utilizando intervalo de confiança e os desvios padrão obtidos em cada cenário de simulação;
- A simulação como fator comprobatório na avaliação de desempenho de arquiteturas distintas de sistemas, como por exemplo 2-camadas e 3-camadas.

1.4. Estrutura do Trabalho

O capítulo 1 contém os objetivos e a apresentação inicial deste trabalho. O capítulo 2 introduz os paradigmas cliente/servidor 2-camadas e 3-camadas, mostrando suas características, vantagens, desvantagens e benefícios, abordando também o porquê do surgimento deste novo paradigma representado pelos servidores de aplicação. O capítulo 3 é dedicado à metodologia abordada para atingir os objetivos deste trabalho de pesquisa. O capítulo 4 apresenta a aplicação da metodologia considerando os dois modelos de ambientes que serão analisados, modelados e simulados no software ARENA, e mais as simulações e a análise dos resultados obtidos. O capítulo 5 contém as considerações finais e conclusões deste trabalho e sugestões para trabalhos futuros.

2. Paradigma Cliente/Servidor uma Abordagem 2-Camadas e 3-Camadas

2.1. Histórico da Evolução no Ambiente Computacional

O projeto de SI vem ao longo das décadas sofrendo modificações, acompanhando as mudanças que ocorrem nos paradigmas da informática.

Da década de 1960 até meados da década de 80, o modelo computacional centralizado (tradicional) ou legado, baseado em mainframes e minicomputadores dominou o cenário da computação mundial. As empresas possuíam um pequeno número de computadores e dada a inexistência de um meio eficiente e confiável que os interliga-se, trabalhavam de forma independente uns dos outros e desenvolver SI era algo bastante caro e demorado.

Por volta de 1980 com o nascimento dos computadores pessoais (PC) surgiu um novo paradigma computacional, o cliente/servidor. Ele introduziu o conceito da divisão do processamento computacional entre os dois extremos, o cliente e o servidor, de uma conexão. Parecia que este novo modelo substituiria por completo o anterior, que era de elevado custo, complexo e pouco flexível.

Porém novas mudanças acarretam novos problemas. Apesar da indústria de software mostrar um crescente amadurecimento no lado servidor com os *Relational Data Bases Management Systems* (RDBMS) e os Sistemas Operacionais (SO) de Rede, e no lado cliente com suporte a várias ferramentas de desenvolvimento como o *Microsoft Visual Basic*, *Borland Delphi*, *Borland C++ Builder*, o modelo tradicional cliente/servidor baseado em 2-camadas mostrava alguns pontos falhos como: O inevitável aumento do número de clientes na rede que implica em um constante *upgrade* de hardware e software cliente, e apesar destas constantes mudanças o tempo de resposta dos SI não diminuía [LINDGREN, 2001].

Nos anos 90 surge um novo conceito, o da orientação a objetos, que consegue sanar alguns problemas de desenvolvimento de sistemas dentro do modelo cliente/servidor. A explosão da Internet para fora do nível acadêmico-profissional através da *Web*, por volta de 1994, provoca uma revolução na forma de disseminar e prover informações, com o advento da *World Wide Web* (WWW) e sua interface colorida e amigável através dos *browsers*. Sem sombra de dúvida, esse crescimento continuará nos próximos anos, e é provável que ele influencie o desenvolvimento tecnológico e seja determinante para o uso da Internet no próximo milênio [TENENBAUM, 1996].

Dentro desta nova abordagem de desenvolvimento de sistemas surgem os servidores de aplicação (AppServer). Por serem baseados no modelo orientado a objetos, os AppServer beneficiam-se dos pilares básicos deste paradigma: abstração, encapsulamento, polimorfismo e herança, o que possibilita reutilização de código e facilidade de manutenção dentro do ambiente computacional. Eles representam a evolução da arquitetura cliente/servidor baseado em 2-camadas.

2.2. Uma Visão Geral dos Sistemas Computacionais

Os sistemas baseados em mainframes e minicomputadores eram compostos pelos chamados “terminais burros”, pois não realizavam qualquer tipo de processamento, sendo baseados em caracteres alfanuméricos. Sua interface, representada pela tela verde e preto dos terminais, com o usuário era muito pouco amigável e desenvolver sistemas neste ambiente era pouco produtivo e muito caro [LINDGREN, 2001], pois os programas eram codificados através de fitas ou cartões perfurados, os terminais não acessavam simultaneamente um mainframe e minicomputador, para isso seriam necessários dois terminais, e sua mão de obra especializada muito cara.

Já o modelo cliente/servidor, representado pelas redes de computadores e computadores pessoais, valendo-se do poder de computação representado pelos PC permite o desenvolvimento de novos tipos de aplicações baseadas nos elementos cliente e servidor.

Porém a existência neste novo ambiente dos mais variados clientes e servidores e a sobrecarrega em demasia do lado cliente, tornam as aplicações pouco manuteníveis necessitando de computadores com capacidade computacional cada vez maior e um maior entendimento sobre as várias camadas de softwares como os protocolos de comunicação e as *Applications Programming Interfaces* (API) responsáveis pela comunicação entre os dois lados.

Então uma camada intermediária é introduzida entre o cliente e o servidor na tentativa de amenizar os problemas encontrados com o tradicional modelo cliente/servidor em 2-camadas, ganhando popularidade com o surgimento e crescimento da *Web*.

2.3. O Paradigma Centralizado (Tradicional) ou Legado

Caros, com seus programas e dados manipulados apenas por uma seleta equipe de profissionais e indicados para ambientes de missão crítica os “Grandes Dinossauros” da era computacional moderna encontraram seu declínio por volta da década de 1980. Essas grandes máquinas tiveram e têm até hoje seu maior representante a figura da *International Business Machines Corporation* (IBM). Na metade dos anos 80 a maioria das grandes e médias empresas americanas, como *General Motors e AT&T*, possuíam mainframes IBM ou IBM compatíveis.

Porém os altos custos destas máquinas levaram a indústria de hardware, a buscar novas alternativas, nascem então os minicomputadores como uma alternativa mais barata e um aceitável poder de processamento. O sistema VAX da *Digital Equipment Corporation* (DEC) foi seu maior representante [LINDGREN, 2001]. Com o enfraquecimento do mercado de minicomputadores a IBM introduz sua família de produtos AS/400, conseguindo comercializar por volta de 450.000 unidades. Hoje uma modificada linha de produtos AS/400 ainda são comercializados como servidores *Web* pela IBM.

As características inerentes à abordagem do processo computacional centralizado conduzem ao seu declínio: ambientes e SO proprietários, controle centralizado, recursos não reutilizáveis, arquitetura hierárquica e estática e a interface pouco amigável com o usuário. A figura 2.1 mostra uma visualização de um ambiente centralizado.

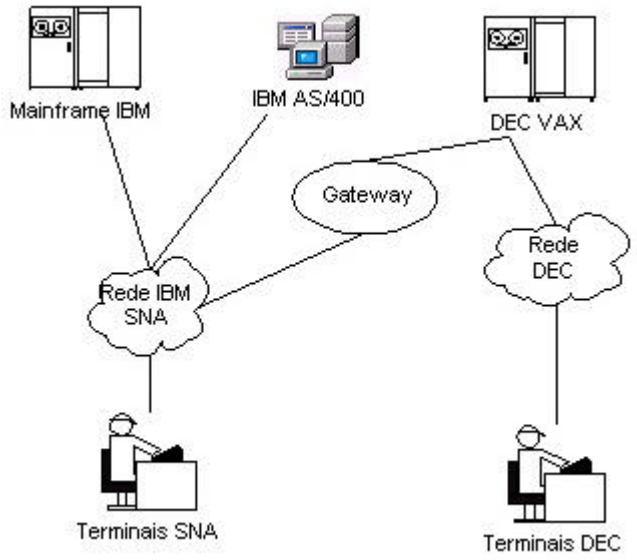


Fig. 2.1 Sistema centralizado hierárquico

Instituições governamentais, educacionais e comerciais têm constantemente realizado mudanças no sentido de substituir seus sistemas tradicionais pelos baseados na arquitetura

cliente/servidor. Nos dias atuais os “Grandes Dinossauros” mais “adaptados” do que seus “irmãos mais velhos” das décadas de 60 e 70, trazem agora pilhas de protocolos encapsulados baseados em padrões abertos como o TCP/IP. Entretanto apesar dos mainframes atuais apresentarem uma maior facilidade para a integração com redes e SI atuais, o custo para sua manutenção e operabilidade ainda é bastante considerável.

2.4. O Paradigma Cliente/Servidor

O avanço das tecnologias de rede juntamente com a proliferação e evolução dos PC provocam uma evolução nos conceitos do processamento computacional, o surgimento do paradigma cliente/servidor. Depois que entendemos, podemos perceber o quanto essa nova abordagem de computação é importante [BOCHENSKI, 1994].

A denominação “cliente/servidor” define muita bem a essência desta nova abordagem de processamento computacional. Ela baseia-se em duas camadas uma cliente, computadores pessoais e *workstations*, que solicitam dados e serviços a uma outra camada, a servidora. Geralmente cliente e servidor encontram-se em plataformas diferentes, porém estas duas camadas podem fazer parte de uma mesma máquina com elas comunicando-se através de interprocessos. O lado cliente contém a interface lógica com o usuário, navegação e apresentação do SI, o lado servidor é responsável pela base de dados e sua lógica de manipulação. Eles comunicam-se através de protocolos de rede (TCP/IP, IPX/SPX, etc) e através de uma API. A figura 2.2 mostra um ambiente cliente/servidor.

Fig. 2.2 Sistema cliente/servidor

O paradigma cliente/servidor é envolto em um paradoxo, pois apesar de ser conceitualmente simples de entender, o desenvolvimento de SI utilizando seus conceitos é

algo complexo, pois seus elementos são de uma maior complexidade que as do modelo baseado na computação centralizada: a ambientação do usuário, definição de quais tarefas serão realizadas por eles e quais serão realizadas pelo SI, definir quais as linguagens de programação utilizar, a existência de API de diferentes fornecedores, SO e gerenciadores de bancos de dados mais adequados para esse novo ambiente, a quantidade e o tipo de dados que irão trafegar na rede, a segurança destes dados e clientes interagindo com os mais diversos sistemas nas mais diversas plataformas. A disponibilidade dos serviços é outro fator a considerar, antes localizados e distribuídos em várias camadas dentro de um único sistema computacional (sistema legado), onde suas várias camadas de software relativamente independente entre si comunicavam-se via SNA. Agora no modelo ciente/servidor essas camadas comunicam-se, entre si, através de máquinas diferentes utilizando protocolos de comunicação.

O lado cliente é um elemento complexo e delicado, pois geralmente é a partir dele que será moldado todo o ambiente de processamento computacional, e apesar da aparente complexidade em desenvolver SI, questões econômicas envolvendo o ambiente legado e a pouca diversidade de ferramentas para desenvolver sistemas, faz com que a arquitetura cliente/servidor ganhe força e torne-se uma realidade. Certamente não veremos todos os sistemas antigos serem substituídos por novos sistemas cliente/servidor ou sistemas distribuídos. Várias empresas terão de combinar o novo e o velho durante muitos anos, mas haverá uma mudança imediata na forma de como os sistemas serão desenvolvidos e empregados [BOCHENSKI, 1994].

2.5. Paradigma Cliente/Servidor 3-Camadas e N-Camadas

Inicialmente, visando a solução de uma determinada classe de aplicações dentro de um ambiente corporativo, esta nova abordagem ganha projeção acompanhando a evolução de outros modelos da computação: Internet, intranets, orientação a objetos e objetos distribuídos. E agora com o Mundo inteiro conectado à *rede das redes*, pelos mais variados dispositivos, SI e topologias de redes, o modelo cliente/servidor baseado em 2-camadas não é mais adequado, cedendo lugar para um novo baseado em 3-camadas e N-camadas, permitindo que as mais variadas organizações integrem seus sistemas baseados no modelo legado com o modelo cliente/servidor e baseados na *Web* de forma mais segura e consistente. O AppServer introduz

a idéia de uma camada intermediária entre os elementos cliente e servidor. A primeira camada, camada cliente, que antes agregava a Interface Gráfica de Usuário (GUI) e toda a lógica do negócio, agora contem somente a GUI introduzindo o modelo *thin client* (cliente magro). A segunda camada, AppServer, é responsável pelo controle da lógica do negócio. A terceira, representada pelo servidor de banco de dados, comporta a base de dados e sua lógica de manipulação, mais as aplicações do modelo legado, se for o caso. Como conseqüência do "cliente magro", o lado servidor passa a ter uma característica *fat server* (servidor gordo) uma vez que o ambiente 3-camadas move a lógica dos negócios para uma camada servidora de aplicação. Porém as variáveis do ambiente cliente/servidor 2-camadas ainda devem ser levadas em conta porque ele oferece uma maior facilidade para desenvolvimento entre várias plataformas e uma maior integração para acompanhar o crescimento de um ambiente computacional. As figuras 2.3 e 2.4 mostram dois ambientes baseados em um AppServer.

Fig. 2.3 Sistema cliente/servidor 3-camadas

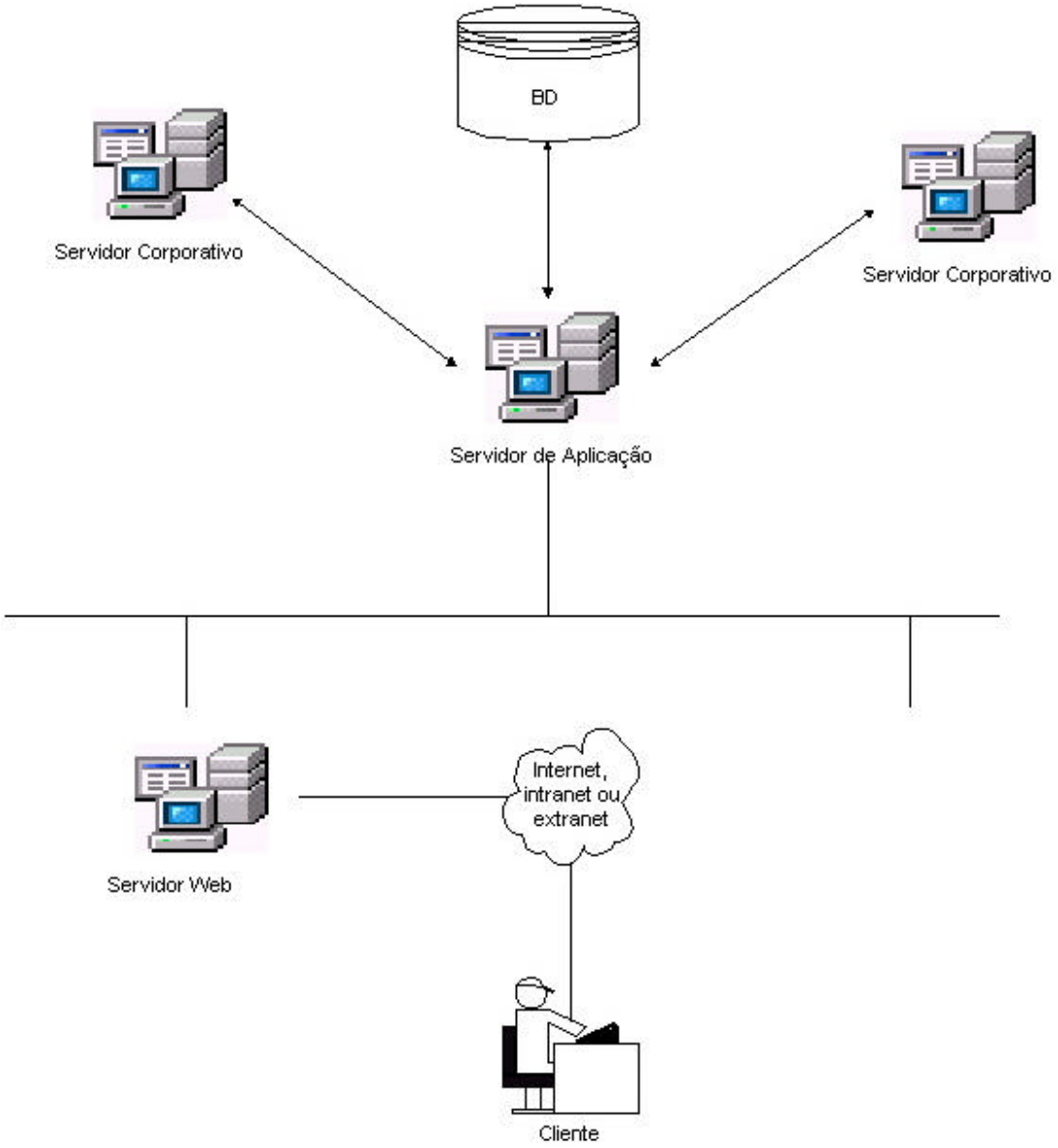


Fig. 2.4 Sistema cliente/servidor 3-camadas dentro do modelo Web

2.6. Introdução ao Paradigma Cliente/Servidor 2-Camadas

Da resultante entre microcomputadores mais poderosos e os avanços nas comunicações das redes de computadores por volta de 1980, emerge o paradigma cliente/servidor. Esse novo modelo traz a computação para mais próxima do usuário na medida que integra os recursos dos computadores pessoais às GUI e recursos das aplicações multimídia, garantindo uma maior integração ao ambiente corporativo.

O termo cliente/servidor significa que existe uma divisão, ou um compartilhamento, de processos entre estas duas camadas. Ou seja, um processo cliente faz solicitações a um processo servidor o qual atende a essas solicitações. Exemplos destes processos podem ser: acesso a registros de um banco de dados, solicitação para validação do acesso de um usuário aos recursos de uma rede, solicitação de execução de uma *store procedure*, as solicitações de um cliente *browser*, os softwares clientes de correio eletrônico, etc.

De [BOCHENSKI, 1994] define-se cinco características principais para o modelo cliente/servidor:

- A existência de um processo cliente e outro servidor, que podem ser distinguidos, mas que cooperam entre si;
- Os elementos clientes e servidor normalmente operam em plataformas diferentes, mas isso não é uma regra;
- Os lados cliente e servidor podem ser atualizados independentes um do outro;
- Os servidores podem atender a múltiplos clientes e os clientes acessar múltiplos servidores;
- As aplicações cliente/servidor possuem alguma capacidade de operar em rede.

Complementa-se o item três ressaltando que uma mudança de banco de dados, lado servidor, implicaria em uma mudança de API, requerendo assim atualização dos clientes.

2.6.1. Elementos do Paradigma Cliente/Servidor

Existem basicamente três elementos na composição do modelo cliente/servidor: o cliente, o servidor e uma infraestrutura de rede que oferece suporte à comunicação entre ambos.

2.6.1.1. O Cliente

O cliente ou *front end* representa a camada número um no modelo cliente/servidor. Ele pode variar desde aplicativos simples para consultas em um banco de dados à ambientes sofisticados para desenvolvimento de sistemas de aplicação clientes baseados em uma GUI, podendo ser também um processo automatizado. No entanto é importante salientar que um cliente não precisa ter necessariamente uma interface baseada em uma GUI ele pode ter uma interface baseada no SO DOS por exemplo, um dos maiores representantes deste lado cliente, desde o inicio dos computadores, foram as planilhas eletrônicas. O *Query Analyser* do *Microsoft SQL Server* e o *Oracle WorkSheet* da *Oracle* podem ser consideradas ferramentas simples de consultas a banco de dados, o *Microsoft Visual Basic* e o *Borland Delphi* são ambientes mais sofisticados para desenvolvimento de aplicações clientes, e o *ISQL* do *Microsoft SQL Server* e o cliente *ftp* do Windows 98, aplicações clientes baseadas em DOS.

As GUI representaram papel fundamental para o avanço do conceito cliente/servidor, uma vez que suas interfaces mais amigáveis aumentaram o entendimento do usuário em relação aos SI e na medida que facilitaram o desenvolvimento de novas aplicações clientes aumentando a produtividade do ambiente computacional corporativo.

De maneira geral alguns fatores devem ser considerados quando se escolhem softwares para desenvolvimento de SI clientes:

- Suporte aos SO padrões de mercado;
- Suporte para o padrão SQL (*Strutured Query Language*);
- Suporte aos mais variados sistemas gerenciadores de bancos de dados e SO de rede;
- Possuir padrão de desenvolvimento baseado no paradigma de objetos;
- Permitir uma interface com o modelo *Web*;
- Suporte a GUI;
- Possuir softwares que permitam a depuração da aplicação cliente.

De um modo geral o cliente deve:

- Rodar em um computador pessoal e possui uma GUI para captura e visualização de dados ao usuário;

- Realizar solicitações a um ou mais processos servidores localizados em máquinas remotas;

- Executar um determinado protocolo de rede e API.

2.6.1.2. O Servidor

O servidor ou *back end* representa a camada número dois no modelo cliente/servidor. Ele é o conjunto hardware/software que oferece um conjunto de serviços necessários aos mais variados tipos de solicitações realizados pelo cliente.

Existem vários tipos de servidores: servidores de arquivos, servidores de impressão, servidores de banco de dados, servidores de *groupware*, servidores de objetos, servidores *Web* e os AppServer, que serão o objeto de estudo deste trabalho de dissertação, serão explorados mais adiante. Independente do tipo, um servidor precisa possuir as seguintes características:

- Estações de trabalho mais poderosas, com mais memória, disco rígido com tolerância à falhas, barramento interno mais veloz. Possuem SO de rede como o *Red Hat Linux* (versão servidor) ou *Windows 2000 Server*. Estas estações possuem software e hardware especializado para prover as variadas solicitações das entidades clientes;

- Não inicia a troca de mensagens com um cliente, tratando-se de um elemento passivo que atende as várias solicitações dos clientes.

2.6.1.3. A Rede

De curiosidade acadêmica no início da década de 1980 a um elemento vital aos meios corporativos, governamentais e acadêmicos, as redes de computadores de alta velocidade foram sem dúvida uma das maiores conquistas para o mundo computacional moderno, possibilitando o surgimento de novas arquiteturas como a cliente/servidor.

A ARPANET (*Advanced Research Projects Agency*), hoje DARPA, do Departamento de Defesa dos EUA foi a motivação inicial que deu origem ao que hoje se chama Redes de Computadores. O objetivo inicial de desenvolver a ARPANET foi da necessidade de compartilhamento de recursos dos sistemas de tempo compartilhado da década de 1960. Entretanto nos primeiros anos da década de 1980 a ARPANET foi dividida em duas redes, a MILNET com objetivos militares e uma versão da antiga ARPANET, dando origem ao que nos dias atuais se conhece como Internet. Hoje a chamada rede das redes, é composta pelos mais variados softwares e hardwares de rede rodando sobre uma pilha de protocolos TCP/IP.

2.6.2. Benefícios, Vantagens e Desvantagens do Paradigma Cliente/Servidor 2-Camadas

O paradigma cliente/servidor provocou uma mudança radical nos conceitos relativos na forma de manipular e distribuir informações em um ambiente computacional, na medida que provocou modificações na forma de desenvolvimento de sistemas de aplicação e permitiu que o ambiente corporativo se valesse de decisões estratégicas mais ágeis e rápidas, além de fomentar a busca de novas pesquisas tecnológicas de hardware e software que melhor tirassem benefícios deste novo modelo pelos centros de pesquisas acadêmicas e pela industria de informática.

Outros benefícios também podem ser observados:

- O surgimento de soluções baseadas em sistemas abertos como o modelo de referência OSI (*Open System Interconnection*), o padrão ODBC (*Open Data Base Connection*), que permite o acesso às mais variadas fontes de dados;

- O surgimento e o crescimento de novas companhias como a *Microsoft, Novell, Oracle, Sun, Sybase, Informix*, etc. Aumentando assim a concorrência e a diversidade de soluções baseadas em software/hardware;

- O surgimento de novos paradigmas como a orientação a objetos e os sistemas distribuídos.

Algumas vantagens de adotar o modelo cliente/servidor são mostradas abaixo:

- Ambiente mais produtivo para o desenvolvedor e para o usuário final baseado em uma GUI;

- Custo mais baixo de software/hardware do que no ambiente legado;

- Curva de aprendizado mais suave para o usuário final que levava até meses de treino para ser considerado proficiente em um SI legado;

- Maior disponibilidade dos dados corporativos;

- Possibilidade de criação de grupos cooperativos de trabalhos;

- Modularidade do ambiente computacional.

Porém, apesar das vantagens listadas acima, este novo paradigma possui algumas desvantagens:

- A lógica do negócio toda inerente ao cliente e sua repetição por todos os clientes da rede, gerando uma complexa atualização destes clientes, em função da necessidade de um grande gerenciamento do software espalhado pela rede;

- Pouca flexibilidade do ambiente de hardware devido ao processo de atualização;

- Segurança dos dados ao nível do cliente. O que permite que usuários mal intencionados manipulem as informações do banco de dados;

- Baixo encapsulamento de dados, uma vez que o cliente manipula dados do banco de dados;

- Baixo reuso da aplicação devido à lógica de controle do negócio estar centralizada no cliente;

- O desempenho do sistema pode ser comprometido. Instruções *SQL* enviadas através da rede com dados sendo baixados no cliente para análise, aumentando o tempo de resposta;

- Nenhuma integração com sistemas baseados no modelo legado;

- Pouca integração com o modelo *Web*;

Se por um lado as GUI foram uma grande vantagem trazida pelo modelo cliente/servidor, por outro, a possibilidade de diversificá-la e manipular a lógica de suas aplicações sobrecarregaram o lado cliente a lógica de negócio, levando à evolução para o modelo mais flexível baseado em 3-camadas e N-camadas.

2.6.3. Porque o Paradigma Cliente/Servidor 3-Camadas

Considerando as desvantagens citadas sobre o ambiente cliente/servidor tradicional, o paradigma 3-camadas advém como uma alternativa no desenvolvimento de SI mais adequados ao ambiente distribuído. Nos dois tópicos a seguir será mostrada a arquitetura cliente/servidor 3-camadas, bem como seus benefícios, vantagens e desvantagens.

2.6.3.1. Sua Arquitetura

O modelo 3-camadas é representado por aplicações, baseadas em componentes de software, denominados *Servidores de Aplicações*. Esses componentes de software contêm camadas intermediárias responsáveis pela comunicação cliente/servidor, tendo uma interface de usuário solicitando serviços de um AppServer, e esses serviços armazenando e devolvendo dados de um banco de dados ou de outro AppServer. A seguir mostraremos sua estrutura e algumas de suas características. A figura 2.5 mostra a representação das camadas de um AppServer e a figura 2.6 mostra um maior detalhamento desta representação.

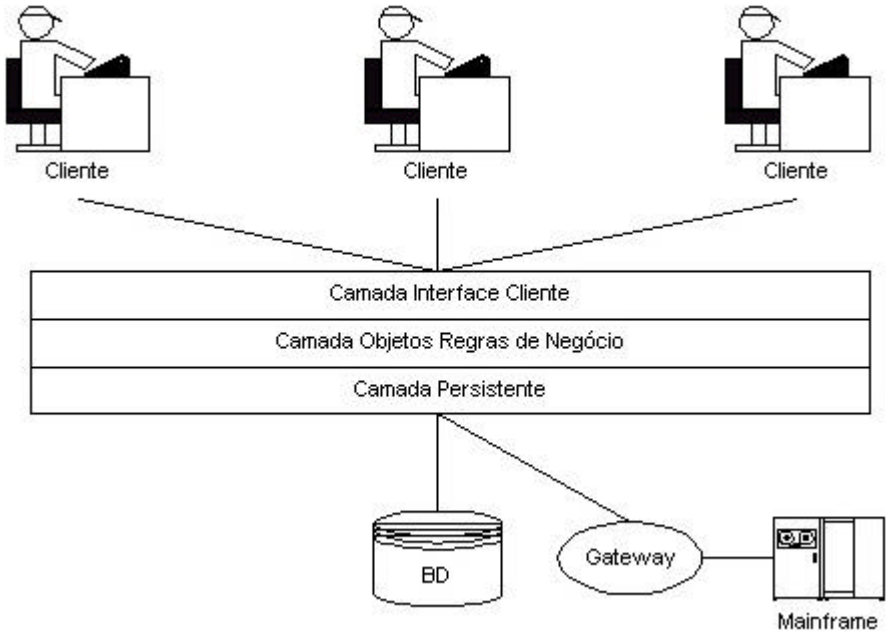


Fig. 2.5 Camadas básicas de um AppServer

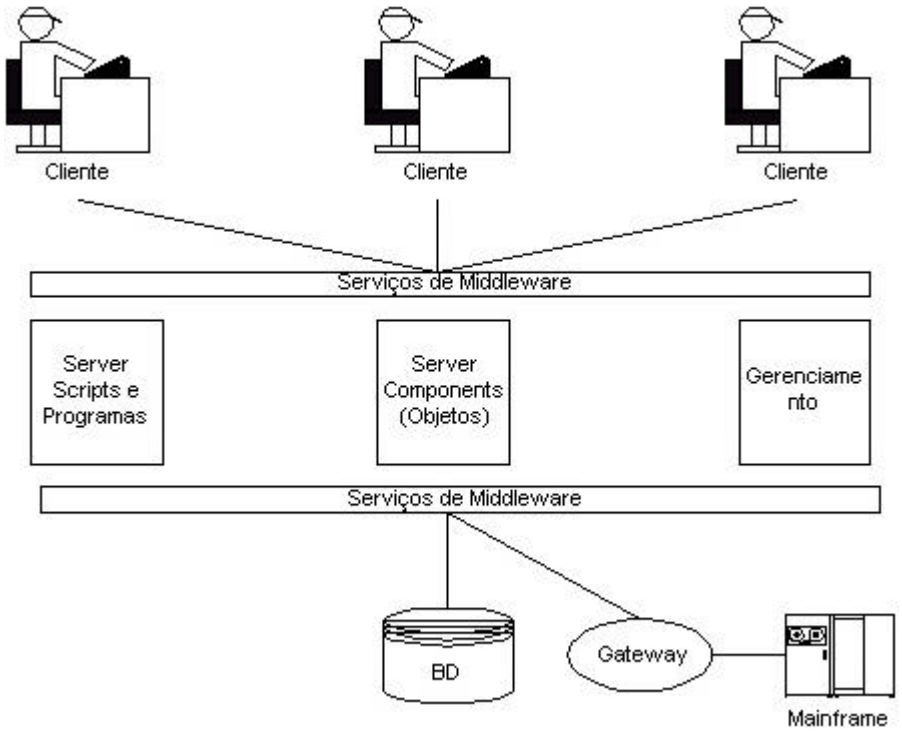


Fig. 2.6 Camadas básicas mais detalhadas de um AppServer

Considerando as figuras acima, considera-se que um AppServer é basicamente dividido em três partes lógicas:

- Camada de Serviços da Interface Cliente: Ela é composta pelos programas de interface com os usuários que contém os *serviços de middleware* do lado cliente, isto é, dependente da aplicação cliente. Estes serviços podem ser: chamada de procedimentos remotos (RPC), ou serviços como a solicitação de dados de um condutor como nome, endereço, etc;

- Camada Objetos Regras de Negócio: Esta camada encapsula os processos, as regras do negócio e as interações entre o cliente e o AppServer. Por esse motivo seus modelos de objetos precisam ser independentes de linguagem de programação e generalizados, possibilitando assim sua interação com as mais variadas aplicações. Os elementos representativos de software desta camada podem ser: *Broadcast*, controle de filas e gerenciamento das instâncias de conexão dos clientes, os objetos do negócio baseados em sua regra, etc;

- Camada Persistente: Esta camada serve de interface entre um banco de dados e aplicações das outras camadas do AppServer, e também com sistemas

baseados em ambientes mainframe. Ela mapeia as solicitações da camada da interface cliente, instanciando os objetos de negócio pertencente à segunda camada da seguinte forma:

- Camada da interface cliente necessita de um objeto de negócio. Ela então envia uma solicitação à camada persistente;

- A camada persistente então localiza e instancia este objeto caso ele esteja na memória. Caso contrário ela localiza-o, instância-o, carrega os dados no objeto instanciado, retornando a referência para a camada da interface cliente ;

- O objeto de negócio não sendo mais necessário, a camada persistente desfaz a referência ao objeto.

A figura 2.7 ilustra as etapas citadas acima.

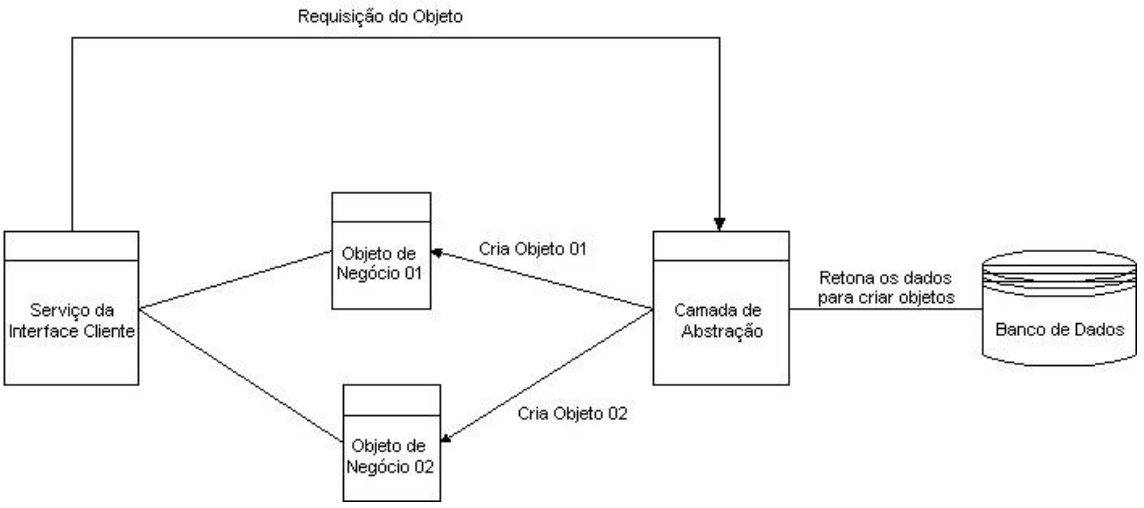


Fig. 2.7 A camada persistente

2.6.3.2. Vantagens e Desvantagens

AppServer são complexas peças de software não adequadas a todo tipo de aplicação. SI que manipulam um grande volume de informações distribuídas em uma LAN, uma WAN

ou interconexão de ambas são candidatos a utilizar o modelo dos AppServer. Este tópico aborda algumas vantagens e desvantagens da utilização desta tecnologia.

Algumas vantagens relacionadas ao paradigma cliente/servidor multicamadas:

- Escalabilidade: O crescimento de um ambiente computacional cliente/servidor normalmente exige uma constante atualização de seus elementos clientes e servidores, com a mudança para servidores com um maior poder computacional sendo sempre necessária. Os AppServer acrescentam a este ambiente uma maior escalabilidade na medida que sua arquitetura permite a distribuição do balanceamento de carga entre vários servidores;

- Processamento Distribuído: Processos cooperativos entre AppServer distribuídos em uma rede de computadores podem ser utilizados para conclusão de tarefas específicas. Se entradas de compra são concluídas na cidade A, e o estoque e produção concluídos na cidade B, então usuários localizados na cidade C podem através da interconexão dos elementos servidores localizados nestas cidades realizar consultas nas compras, estoques e produção sem a necessidade dos dados estarem totalmente armazenados no banco de dados da terceira cidade;

- Objetos Reutilizáveis: Baseados na orientação a objetos e objetos distribuídos, e sendo um *container* de serviços representados pelas regras de negócios, este paradigma permite a reutilização dos objetos de negócio na medida que garante sua utilização em várias aplicações.

- Objetos Processando as Regras do Negócio: O ambiente cliente/servidor tradicional sobrecarrega o lado cliente com as regras do negócio, fazendo com que a camada de serviços da interface cliente manipulem diretamente dados do banco de dados. As *store procedures* possibilitam que uma determinada carga de processamento passe para o lado servidor, porém elas são limitadas quando comparadas aos ambientes de desenvolvimento sofisticados, baseadas em objetos, oferecidos pelas ferramentas utilizadas para a construção destes componentes de softwares. Nos ambientes multicamadas as mesmas regras são modeladas e encapsuladas aos objetos permitindo uma melhor integração para um ambiente

computacional. A figura 2.8 mostra os passos lógicos desde a invocação de um serviço chamado “CONSULTA ESTOQUE” até a sua execução através do objeto “CONSULTA ESTOQUE”;

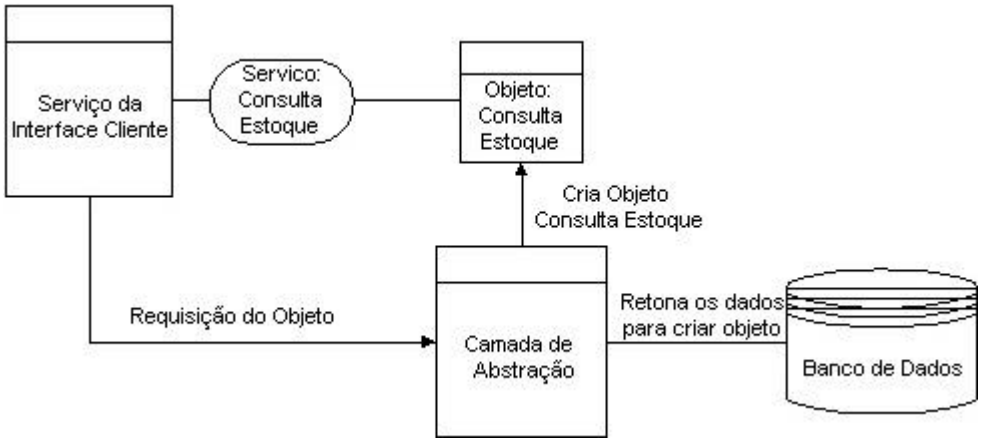


Fig. 2.8 Objeto processando regra

- Integração Entre Plataformas: Uma sofisticada IDL (*Interface Definition Language*) pode prover integração com várias linguagens de programação. Uma vez que os objetos serão definidos e modelados utilizando regras, esta IDL pode gerar códigos que podem ser compilados em diferentes linguagens.

Algumas desvantagens relacionadas ao paradigma cliente/servidor multicamadas:

- Investimento em Longo Prazo: Devido sua complexa arquitetura e o elevado nível de abstração necessário para a concepção de seus objetos, ele necessita de uma considerável curva de investimentos e aprendizado até a sua completa adequação dentro do ambiente computacional;

- Escolha da Plataforma: Existem basicamente três padrões que servem como modelo para o desenvolvimento destes componentes de software: A tecnologia de objeto COM (*Component Object Model*) da *Microsoft Corporation* centrada na plataforma windows, a EJB (*Enterprise Java Beans*) que é uma extensão do modelo *JavaBeans*, modelo padrão de componentes para Java no desenvolvimento do lado servidor. E o padrão de objetos baseados em CORBA (*Common Object Request Broker Architecture*). Alguns destes padrões para objetos distribuídos apesar de já possuírem uma considerável maturidade, como o modelo de objetos

baseados em Java, possui uma certa escassez de mão de obra especializada. Por outro lado outros são relegados a uma plataforma específica como é o caso do modelo de objetos baseados no padrão COM da Microsoft, [LINDGREN, 2001] cita o modelo CORBA como o representante mais maduro e completo dos modelos de objetos citados;

- Objetos Reutilizáveis: A preconização da reutilização de objetos, sob um certo aspecto, atua como um fator de demasiada relevância para tempo de desenvolvimento de um servidor de aplicação. Porque se por um lado devem ser devidamente especializados para realizar os serviços aos quais são destinados, necessitam ser suficientemente abstrato para permitir seu reuso e acompanhar o crescimento dos mais variados processos distribuídos em um ambiente computacional.

O paradigma cliente/servidor 3-camadas e N-camadas acrescentam um novo enfoque ao modelo cliente/servidor tradicional, na medida em que sua arquitetura tenta representar através da maior abstração possível um ambiente de negócio na forma de elementos de software representados pelos AppServer, resultando em uma maior disciplina, coordenação e padronização no desenvolvimento de softwares.

2.6.4. Comparação Entre os Paradigmas Cliente/Servidor 2-Camadas e 3-Camadas

As duas abordagens no desenvolvimento de SI usando os conceitos de 2-camadas e 3-camadas são diferentes, de um modo geral a tabela 2.1 abaixo mostra resumidamente uma comparação entre os dois modelos.

	Modelo C/S 2-Camadas	Modelo C/S 3-Camadas
Administração do Sistema	Mais complexo devido a lógica do negócio centrada no cliente.	Mais simples devido a lógica do negócio centrada no servidor.
Segurança	Pequena devido a camada cliente manipular dados	Melhor pelos dados serem manipulados pelo AppServer
Encapsulamento de dados e Reutilização de Objetos	Baixo nível de encapsulamanto de dados e métodos objetos, devido a distribuição em muitos clientes, dificultando a reutilização de objetos	Encapsulamento de dados e métodos muito maior devido a centralização das regras de negócio baseadas em objetos, permitindo sua reutilização
Escalabilidade (hardware) e Flexibilidade (software)	Limitada escalabilidade e flexibilidade, pois com o surgimento de novos clientes cresce o número de links de comunicação e as regas de negócio são somadas à esse número	Muito melhor devido ao controle de conexões clientes concentradas no AppServer ou distribuídas através de outros servidores, da mesma forma que as regras de negócio baseadas em objetos
Integração com sistemas centralizados (Mainframe)	Não permite	Possível através da implementação de *gateways* utilizando RPC
Integração em ambientes com bancos de dados heterogêneos	Não permite devido aos RDP (*Remote Database Protocols*), como ODBC (*Open Database Connectivity*) ou DAO (*Data Access Objects*) estarem concentrados no cliente	Um mesmo objeto responsável por uma determinada transação de negócio pode acessar múltiplos bancos de dados

Tab. 2.1 Comparação dos paradigmas 2-camadas e 3-camadas

2.6.5. Trabalhos Envolvendo os Paradigmas Cliente/Servidor 2-Camadas e 3-Camadas

Como já citado anteriormente no item 2.5 o conceito de três camadas surgiu antes do crescimento da Internet, com o aparecimento da *World Wide Web*, ganhando projeção com a sua evolução. O comentário sobre “Camadas” em [HANSON e CRAIN, 2000] já mostrava a preocupação com surgimento desta nova abordagem computacional: “Antes da interweb (Internet e *World Wide Web*), revistas comercias de softwares pareciam obcecadas com a

noção do cliente/servidor 3-camadas". Já o trabalho de [WOLFRAM, 2000], ressalta a importância deste modelo para o *E-Business* apesar de não estar maduro o suficiente. Nele também é abordada a iniciativa de grandes empresas mundiais, como a IBM, de desenvolver aplicações servidoras como o *WebSphere Application Server* para integração de dados e transações corporativas com a *Web*. Simon Edward em [SIMON, 2000], gerente de marketing para sistemas corporativos da IBM, aponta a tendência das aplicações *middleware* bem como a estratégia da IBM de portar sua linha de servidores para este tipo de aplicação.

Por volta de 1989 a *AT&T* através da USL (*Unix System Laboratory*) lança uma das primeiras aplicações usando AppServer, o Tuxedo, que passou em 1996 a ser parte integrante da *BEA Systems Inc.*, empresa de software líder mundial em infraestrutura de aplicações. Em [MERTON, 2000] é realizado um estudo de caso da evolução de desempenho em um ambiente cliente/servidor distribuído utilizando uma base de dados *Oracle* rodando em um IBM RS/6000, clientes windows 95 e uma aplicação *middleware* Tuxedo como controlador de transações. Em [SOMIN e GROSS, 2000] observa-se um benefício do modelo 3-camadas em relação ao 2-camadas: "Enquanto a configuração em 2-camadas consiste de clientes conectados diretamente ao banco de dados, a configuração em 3-camadas contém a adicional camada de um AppServer. A introdução da camada AppServer incrementou a flexibilidade, reduziu o tráfego na rede e melhorou o desempenho, mas necessitou mudanças no planejamento de capacidade devido à nova configuração distribuída".

Uma importante característica do paradigma 3-camadas é a possibilidade de integração entre o "novo e o velho", isto é, integrar ou migrar ambientes legados para ambientes baseados no modelo cliente/servidor como observado em [CRABB, 2000] e [KALM, 2000].

Com o crescimento de aplicações baseadas no modelo 3-camadas muitas empresas passaram a portar seus SI utilizando os modelos 2-camadas 3-camadas como mostrado em [SCHUMACK, 1999]. A estrutura complexa das aplicações ERP (*Enterprise Resource Planning*) pode utilizar a arquitetura do modelo 3-camadas como modelo integrador destas aplicações às corporações como mostrado em [DHILLON, 2000]. As abordagens baseadas no modelo cliente/servidor 3-camadas ou N-camadas como integradores de negócios corporativos torna-se cada vez mais comum nos mais diversos ambientes computacionais, indo desde significativos *midllewares* como o *dynamic TAO*, um componente baseado em CORBA que suporta configurações dinâmicas de mecanismos para suporte de monitoração, mostrado em [KON et al., 2000], a modelos complexos como SPE (*Software Performance Engineering*) baseados no paradigma multicamadas abordado em [SMITH, 1999].

2.6.6. Fatores e Grandezas de Desempenho Consideradas em Sistemas Cliente/Servidor

Para construção de modelos de desempenho no ambiente cliente/servidor, seja ele 2-camadas, 3-camadas ou baseados no modelo Internet, as variáveis de desempenho e os fatores que influenciam seu comportamento precisam ser definidos. Este item tem como objetivo abordar estas variáveis e seus fatores de influência.

O trabalho de [MENASCÉ e GOMMA, 2000] aborda a integração de um modelo de projeto e um modelo de desempenho, de modo que o modelo de desempenho seja definido antes da implementação, o objetivo é analisar o projeto sob a perspectiva do desempenho, sendo que o modelo de desempenho é baseado no enfileiramento em rede (*queuing network*). Essa abordagem garante a flexibilidade de mapeamento da arquitetura de software para sistemas clientes servidor 2-camadas ou 3-camadas. A metodologia abordada utiliza dois parâmetros: intensidade da carga de trabalho e demanda por serviços aos vários recursos como CPU, subsistemas de I/O e segmentos de rede. O desempenho de instruções *selects* no banco de dados é descrito em função dos números de I/O gerados por estas instruções, e os números de I/O função do tipo de plano de acesso (*joins* aninhados, unidos ou híbridos) que é função do sistema gerenciador de banco de dados utilizado. Um das arquiteturas mais populares para o desenvolvimento de servidores de aplicação, nos dias atuais, é a Java 2 e o trabalho de [JURIC et al., 2000] aborda a comparação e otimização destas, utilizando os dois protocolos para comunicação entre os objetos localizados no cliente e no servidor de aplicação o RMI (*Remote Method Invocation*) e o RMI-IIOP (*Remote Method Invocation over Inter-ORB Protocol*), o artigo utiliza um método independente do modelo de objeto distribuído que minimiza a influência do *overhead* do desempenho para o controle de componentes. As variáveis de desempenho consideradas foram o tempo de resposta, o *throughput* e a degradação do desempenho. Outro trabalho referente a Java RMI é mostrado em [AHUJA e QUINTAO, 2000] que avalia o desempenho do Java RMI empiricamente comparando-o com o Java Sockets API, o foco é direcionado para avaliação do tempo de resposta, que é utilizado como base de comparação para estas duas tecnologias.

Existem ferramentas especificas para a analise de desempenho de aplicações Java como a *Jinsight EX*, que em [SEVITSKY et al., 2001] é aplicada a um ambiente cliente/servidor 3-camadas composto por um cliente, um Java Server, representando o servidor de aplicação, e um banco de dados relacional. Para a análise de otimização do banco de dados são consideradas três variáveis: o *overhead* devido o estabelecimento das conexões

com o banco de dados e a compilação das *queries,* a execução da *query* e o processamento dos resultados. A ferramenta de análise permite através da observação das *theads* nos servidores de aplicação e banco de dados, definir qual das três variáveis pode ser otimizada para melhorar o desempenho do banco de dados, através da comparação do número total de execuções com o número de chamadas realizadas para cada uma delas. Testes de *benchmark* são também utilizados para mensurar a carga de trabalho submetida a servidores de aplicação baseados em Java como visto em [LUO e JOHN, 2001] que utiliza dois testes: o VolanoMark que é um avaliador puro Java baseado em 2-camadas, e o SPECjbb2000 que é o primeiro SPEC's *benchmark* para avaliar desempenho de aplicações servidoras Java baseado no modelo 3-camadas. As aplicações Java são comparadas com o *SPECint2000,* e o impacto da *multithreading* em função do aumento dos clientes é investigado. Inicialmente o *benchmark* SPECint é utilizado para determinar as características da aplicação servidora, em seguida o número de *threads* simultâneas são incrementadas para determinar como a *multithreading* impacta a microarquitetura do processador. Os resultados finais mostram que o aumento do número de *threads* ocasiona uma melhora no comportamento das instruções devido o aumento do acesso local, que para o VolanoMark é mais acentuado, mas quando existem muitas conexões clientes o número de instruções que o processador executa aumenta, afetando a escalabilidade do sistema negativamente.

Outra abordagem tecnológica para os servidores de aplicação refere-se ao modelo CORBA de objetos distribuídos. [FATAH e MAJUMDAR, 2002] descrevem o desempenho de três diferentes arquiteturas cliente/servidor baseadas em CORBA. O modelo H-ORB (*Handle-Driven ORB*) no qual o cliente inicialmente solicita o endereço a um agente (*middleware*) para comunicar-se diretamente com o servidor, que pode ser um servidor de banco de dados, sem a comunicação entre o agente e o servidor; o FORB (*Forwarding ORB*) em que a solicitação cliente é automaticamente transferida pelo agente para um servidor apropriado, com a existência de comunicação entre o agente e o servidor, que retorna o processamento do resultado ao cliente e o P-ORB (*Process Planner ORB*) no qual o agente combina a transferência de solicitações com a invocação de múltiplos servidores, como complexas solicitações que requerem serviços de múltiplos servidores. Quatro fatores foram considerados para o controle da carga de trabalho: Numero de clientes, tempo de comunicação entre o cliente, o agente e o servidor, tempo de solicitação de serviço e o tamanho da mensagem. As medidas de desempenho consideradas são: o tempo médio de resposta, o *throughput* médio de todo sistemas em função das solicitações dos clientes por segundo e utilização da CPU. A tabela 2.2 mostra o comportamento do tempo de resposta e

throughput do sistema para as três arquiteturas quando o tamanho de mensagem (L) e o número de clientes (N) são variados.

L(bytes)	H-ORB		F-ORB		P-ORB	
	Tempo de resposta (sec) N = 1	Throughput (Solicitações clientes/sec) N = 24	Tempo de resposta (sec) N = 1	Throughput (Solicitações clientes/sec) N = 24	Tempo de resposta (sec) N = 1	Throughput (Solicitações clientes/sec) N = 24
4800	1.7212	7.7356	1.3142	2.8625	0.9015	1.8524
9600	1.7425	7.4533	1.3448	2.5935	0.9030	1.7938
19200	1.7933	6.8688	1.4141	2.4378	0.9125	1.7592

Tab. 2.2 Comportamento do tempo de resposta e throughput

Para uma carga de trabalho pequena, N = 1, o P-ORB apresenta um melhor tempo de resposta, seguido pelo F-ORB e o H-ORB para todos os valores de L. Para cargas medias e altas, o H-ORB apresenta um melhor *throughput* em comparação com o P-ORB e o F-ORB, exceto é claro para pequenos valores de N. [KEMPER et al., 1999] aborda alguns aspectos de desempenho do SAP R/3, sistema aberto utilizado como padrão para ERP (*Enterprise Resource Planning*). A arquitetura cliente/servidor 3-camadas do SAP R/3 é divida na camada de apresentação contendo as aplicações clientes, a camada de negócio representada por uma ou mais servidores de aplicação e a camada do sistema gerenciador de banco de dados. Os clientes podem fazer parte da LAN, que contém os servidores, ou podem fazer parte de uma WAN. Determinar o hardware para o servidor de aplicação e para o servidor de banco de dados é um fator crucial para o ambiente SAP R/3, uma vez que isso determinará como o acesso concorrente de vários usuários será suportado. Um ambiente com cem clientes implicaria em um servidor monoprocessado e um servidor de banco de dados sob uma *Ethernet*, já um com mil clientes implicaria em um servidor multiprocessado e o servidor de banco de dados rodando sob uma *Fast Ethernet*. O número de LUW (*Logical Unit of Work*), termo que define uma transação no ambiente SAP, no servidor de aplicação também precisa ser cuidadosamente definido uma vez que um grande número de pequenas transações subutilizarão a CPU, e por outro lado, muitos ciclos de CPU podem ser gastos para processar grandes transações.

[NIXON, 2000] descreve um *framework*, o PeRF (*Performance Requirements Framework*), para o gerenciamento das variáveis de desempenho em sistemas de informação em um dado domínio de aplicação. O PeRF integra e cataloga uma base de conhecimento referente a desempenho e sistemas de informação como: conceitos de desempenho e

princípios SPE (*Software Performance Engineering*), método de construção de sistemas com o objetivo da adequação de desempenho [SMARKUSKY et al., 2001]. Este *framework* foi aplicado no estudo dos requisitos de desempenho de sistemas universitários que armazenam dados referentes a estudantes, tendo o SRS (*Student Records System*) da Universidade de Toronto como estudo de caso. O SRS contém por volta de 54.000 registros, 400 tabelas e utiliza o DB2 como gerenciador de banco de dados. A figura 2.9 mostra os conceitos básicos de desempenho catalogados pelo PeREF. O desempenho é considerado sob dois aspectos: de tempo e de espaço. O tempo pode ser decomposto em: *throughput*, tempo de resposta e gerenciamento de tempo. O espaço pode ser particionado em memória principal e armazenamento secundário.

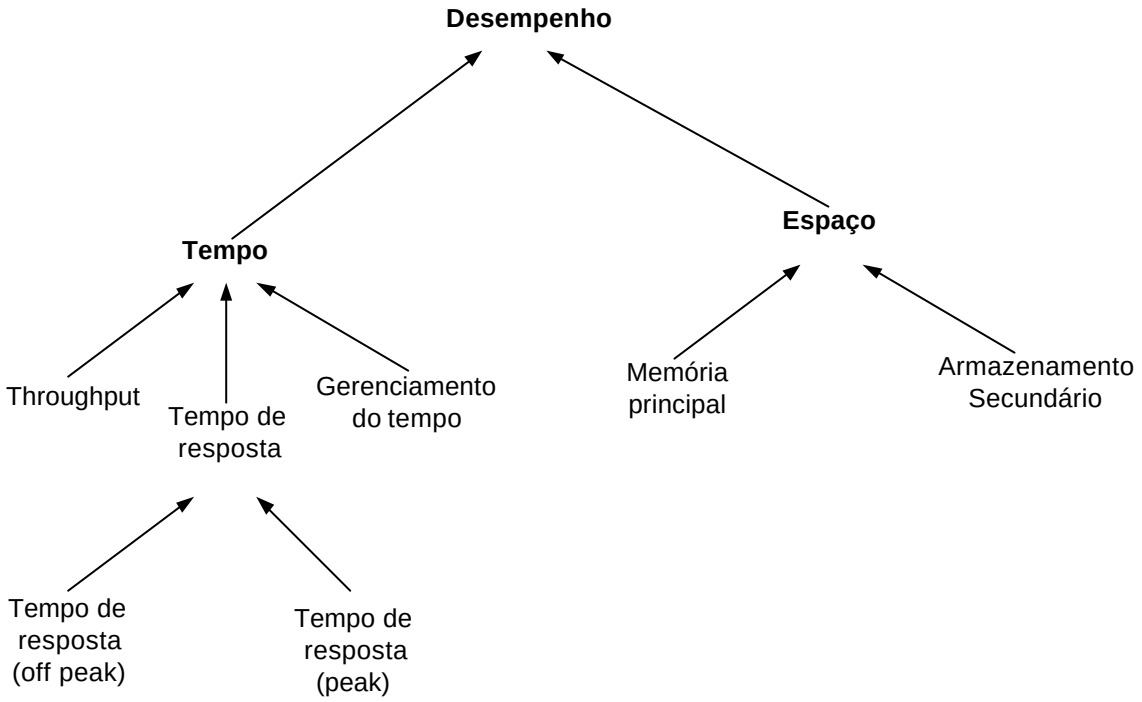


Fig. 2.9 Conceitos básicos de desempenho

[KIM e HAN] descrevem três abordagens analíticas baseadas no modelo de filas e métodos de análise matemática, o objetivo é descrever os resultados da análise como uma maneira de auxiliar o entendimento das variadas configurações nas arquiteturas de *workflow* para sistemas cliente/servidor. As estimativas de desempenho utilizadas foram: demanda por recursos, tempo de processador, e *throughputs* considerado na contenção dos processos de software e hardware. [HELLERSTEIN et al., 2000] revela que na análise de sistemas com informações distribuídas existe uma correlação entre classes de serviços, em parte em função da interação cliente/servidor. Partindo desta premissa objetiva-se clarificar as implicações de

desempenho destas correlações para o comportamento do sistema como um todo, onde o principal dado a considerar para analise de desempenho são as filas de tarefas no servidor em função da carga de trabalho exercida por um grande número de clientes e as interações dos clientes podem ser entendidas como abertura de banco de dados, abertura de *views*, leitura de registros em um banco de dados, etc.

A integração entre uma ferramenta de especificação de softwares como a UML e um modelo de desempenho é mostrada por [SMARKUSKY et al., 2001] que define um *framework* de modelagem de desempenho, que permite a transição de sistemas orientados a objetos especificados em UML (*Unified Modeling Language*) para uma arquitetura de HPM (*Hierarchical Performance Model*), que fornece um mecanismo para modelos de desempenho de sistemas distribuídos orientados a objetos. HPM representa uma visão vertical da modelagem de desempenho, onde cada um dos quatro níveis, sistema, tarefa, modulo e operação, representa uma diferente visão e camadas de abstração para o sistema especificado. A figura 2.10 mostra a modelagem de desempenho do ambiente, que representa a transição de um modelo orientado a objetos desenvolvido em UML para cada nível da arquitetura HPM. Uma breve descrição de cada nível é mostrada a seguir:

- Nível de sistema: representa o maior nível de abstração, enfileiramento em rede (*queuing network*) é utilizado para modelar os parâmetros de entrada e a interação dos processos de software;

- Nível de tarefa: representa uma visão física do sistema sob estudo e tem como foco o mapeamento processo-processador;

- Nível de modulo: especifica e analisa componentes de software, procedimentos e funções;

- Nível de operação: é o mais baixo nível de abstração para a análise do modelo de desempenho. Focaliza as interações entre instruções primitivas com a arquitetura do processador.

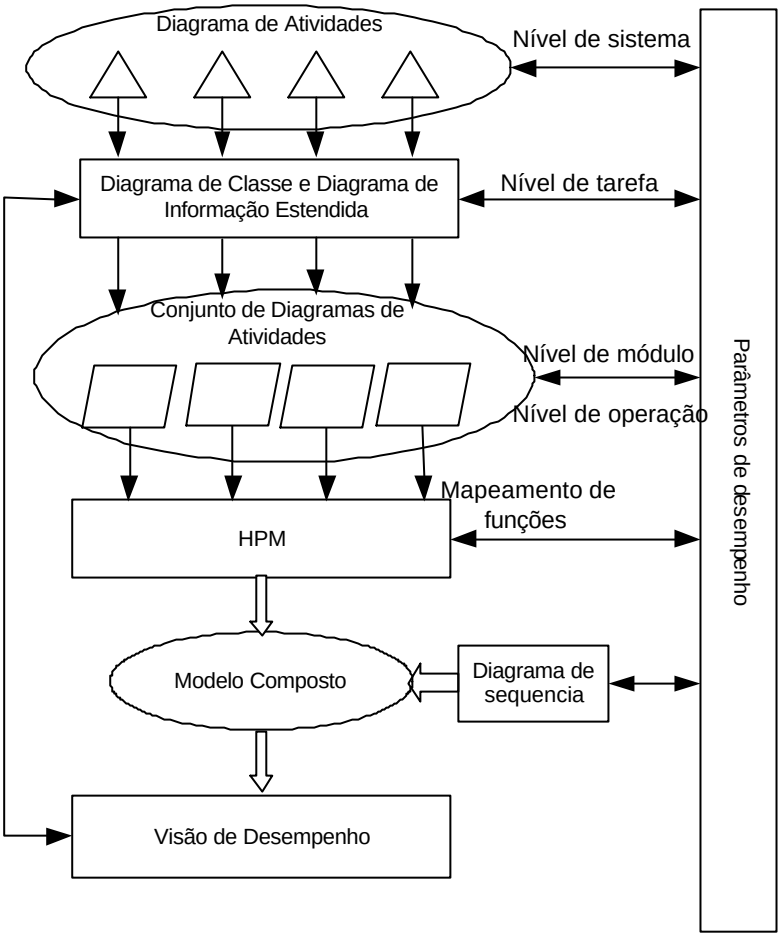


Fig. 2.10 Modelagem de desempenho do ambiente

[NICOLA e JARKE, 2000] analisam modelos de desempenho para sistemas de banco de dados distribuídos e replicados. O objetivo central é analisar e clarificar as diferentes alternativas de modelagem, suas combinações e interdependências nos modelos de desempenho de banco de dados distribuídos através de modelos analíticos. Alguns estudos de simulação são também abordados, devido aos muitos conceitos básicos de modelagem, sistemas de filas (*queuing systems*), por exemplo, dependerem do modelo de avaliação da metodologia. As métricas de desempenho utilizadas são o tempo de resposta e *throughput*, grandezas mais comumente consideradas nas transações envolvendo bases de dados distribuídas. Os modelos de desempenho sejam analíticos ou de simulação, possuem um conjunto de parâmetros de entrada que podem ser estimados precisamente como tempo de serviço em discos e velocidade de CPU, outros parâmetros são dependentes da aplicação e por esta razão difíceis de determinar, como: número de itens de dados no banco de dados, número de solicitações por transação, etc.

Este capítulo fez uma revisão de literatura sobre o nascimento de algumas arquiteturas computacionais, como o modelo cliente/servidor, e a evolução de outros, a abordagem cliente/servidor baseada em 2-camadas evolui para a baseada em 3-camadas e N-camadas. Mostrou-se também uma introdução ao paradigma cliente/servidor 2-camadas abordando seus elementos, características, benefícios e desvantagens. Considerou-se o porquê do modelo 3-camadas, sua arquitetura, vantagens e desvantagens e uma comparação de ambos, além das variáveis de desempenho relacionadas a sistemas de informação baseados no modelo cliente/servidor e fatores relevantes que podem alterar o comportamento destas variáveis.

O capítulo seguinte é dedicado à metodologia utilizada para atingir os objetivos deste trabalho.

3. Metodologia Abordada

3.1. Descrição Geral das Metodologias para Planejamento de Capacidade e Avaliação de Desempenho de Sistemas Computacionais

Projetistas e analistas de sistemas de informação, geralmente, mergulhados na introspecção da modelagem de sistemas acabam por desconsiderar um fator de grande relevância para o desenvolvimento de sistemas de informação, o seu comportamento e interações referentes ao desempenho. [GOMES, 2002] cita esta consideração em relação a um estudo de caso de um projeto de vídeo conferência, e [CRAIN e HANSON, 1999] complementam considerando o aspecto evolutivo das redes de computadores como um fator desempenho.

Planejar capacidade de sistemas clientes/servidor requer uma série de etapas executadas de maneira sistemática, que segundo [MENASCÉ e ALMEIDA, 1998] dividem-se em: compreensão do ambiente, caracterização da carga de trabalho, calibração e validação do modelo de carga, desenvolvimento do modelo de desempenho, calibração e validação do modelo de desempenho, previsão da carga de trabalho, previsão de desempenho, desenvolvimento de um modelo de custo, previsão de custo, e análise custo/desempenho. Três modelos são utilizados pela metodologia: modelo da carga de trabalho, modelo de desempenho e o modelo de custos. Neste trabalho de dissertação não serão abordadas algumas etapas da metodologia de planejamento de capacidade como calibração e validação do modelo de carga, e o tocante a análise dos modelos de custo e desempenho. A metodologia de avaliação de desempenho será a abordada por [JAIN, 1991] que define uma abordagem sistemática para sua aplicação, e a considera como necessária em cada estágio do ciclo de vida de um sistema, o que inclui seu projeto, produção, uso, atualizações, etc.

Uma vez que a metodologia vista em [JAIN, 1991] não detalha alguns aspectos como a compreensão do ambiente, e a caracterização da carga de trabalho, será utilizada a abordagem proposta por [MENASCÉ e ALMEIDA, 1998].

O objetivo deste capítulo é apresentar as metodologias de planejamento de capacidade e avaliação de desempenho que serão utilizadas para o estudo do ambiente do Departamento de Trânsito do Estado do Pará (DETRAN-PA).

De acordo com [MENASCÉ e ALMEIDA, 1998] planejar capacidade de sistemas “É o processo que visa prever futuros níveis de saturação da carga de trabalho de um sistema, considerando aspectos de custo beneficio e o tempo que levará para o sistema saturar. A previsão deve considerar a carga de trabalho devido às aplicações existentes, o surgimento de novas, além da necessidade de novos níveis de serviços”.

[SCHMIDT, 2002] define o objetivo do planejamento de capacidade como “Gerar um nível aceitável de serviço computacional à organização, ao responder as demandas de cargas geradas pelo sistema, permitindo melhorias como a adequação de hardware e software, com vistas a evitar a deficiência de capacidade”.

A importância do fator humano para a avaliação e planejamento de capacidade de sistemas é lembrada em [HANSON CRAIN, 1999], que aborda a união do SPE (*Software Performance Engineering*) com os aspectos do HFE (*Human Factors Engineering*). A heterogeneidade do software, do hardware, das aplicações e do comportamento dos usuários aumenta a complexidade de metodologias de avaliação e planejamento de capacidade de sistemas computacionais, motivando pesquisas como a mostrada em [HULL, 1999]. Apesar do aspecto humano não estar sendo considerado neste trabalho é importante ressaltar que eles atuam como um fator importante dentro da metodologia de planejamento de capacidade, uma vez que pessoas são responsáveis por definir alguns aspectos de desempenho como o tempo de resposta de um sistema.

3.2. O Planejamento de Capacidade

Segundo [MENASCÉ e ALMEIDA, 1998] a figura 3.1 mostra o fluxograma das etapas da metodologia de planejamento de capacidade.

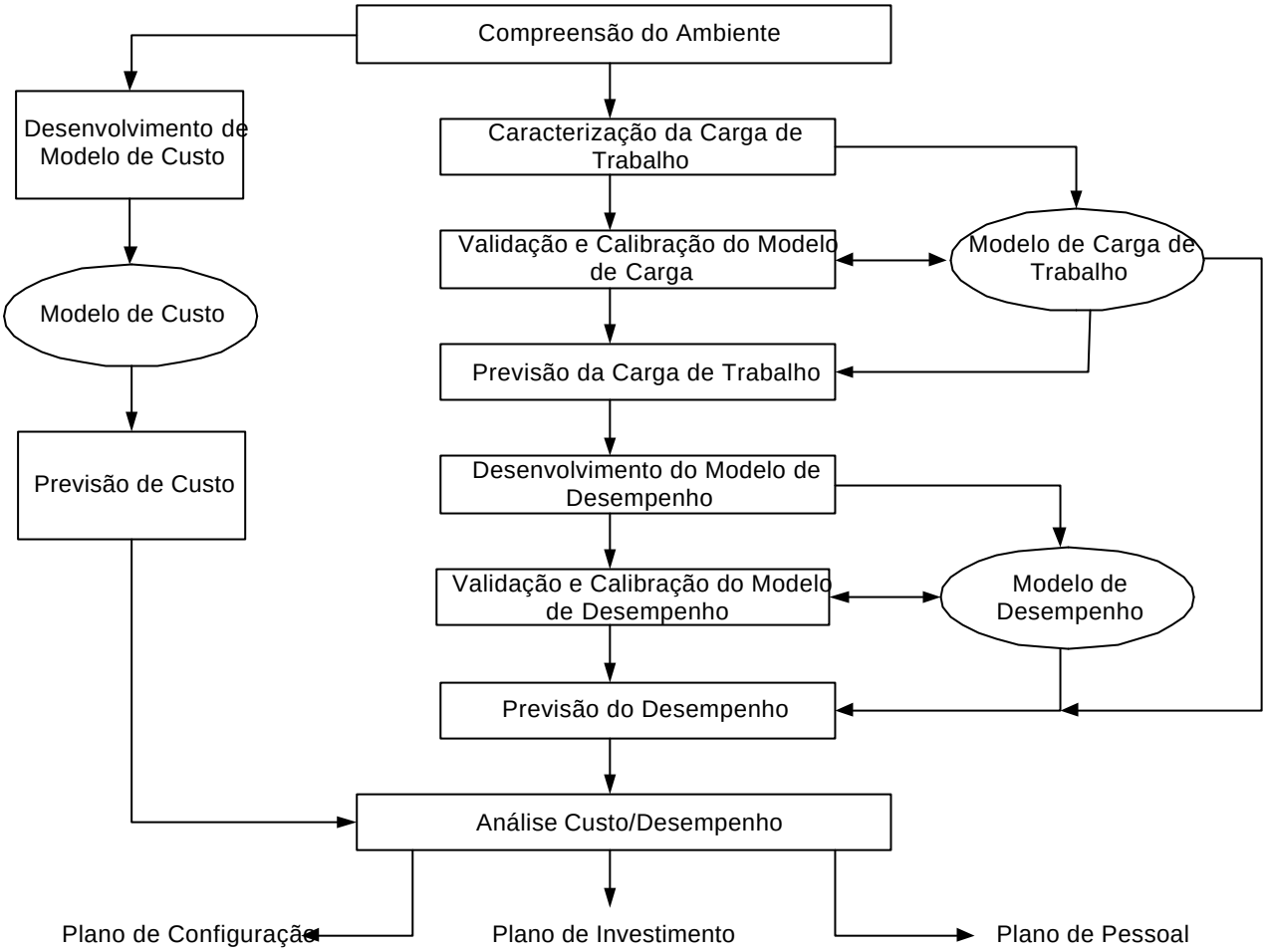


Fig. 3.1 Metodologia de planejamento de capacidade

A seguir são descritas as etapas da metodologia de planejamento de capacidade.

3.2.1. Compreensão do Ambiente

Esta etapa consiste no conhecimento do ambiente sobre o qual se pretende aplicar a metodologia de planejamento e adequação de capacidade, isto é, identificar que tipo de *hardware*: micros clientes, configuração dos servidores de banco de dados e aplicação; *software*: sistemas operacionais clientes e de rede, editores de texto, gerenciadores de banco de dados, linguagens de programação, *middlewares*, etc; *elementos de conectividade*: modems, *hubs*, *switches*, roteadores, etc e protocolos estão presentes neste ambiente cliente/servidor. Essas informações são normalmente obtidas através de reuniões, entrevistas e possíveis documentos do projeto. Esta etapa é muito importante, pois ajuda o analista a limitar o alcance do estudo, e reduz tempo e custos envolvidos no projeto.

3.2.2. Caracterização da Carga de Trabalho

A caracterização da carga de trabalho é a descrição, com precisão, da carga total a que um sistema computacional é submetido, considerando seus principais componentes. Cada componente primário é dividido em componentes básicos, sendo cada um caracterizado por sua intensidade e pelos parâmetros de demanda de serviço sobre cada recurso utilizado. A figura 3.2 mostra esta divisão.

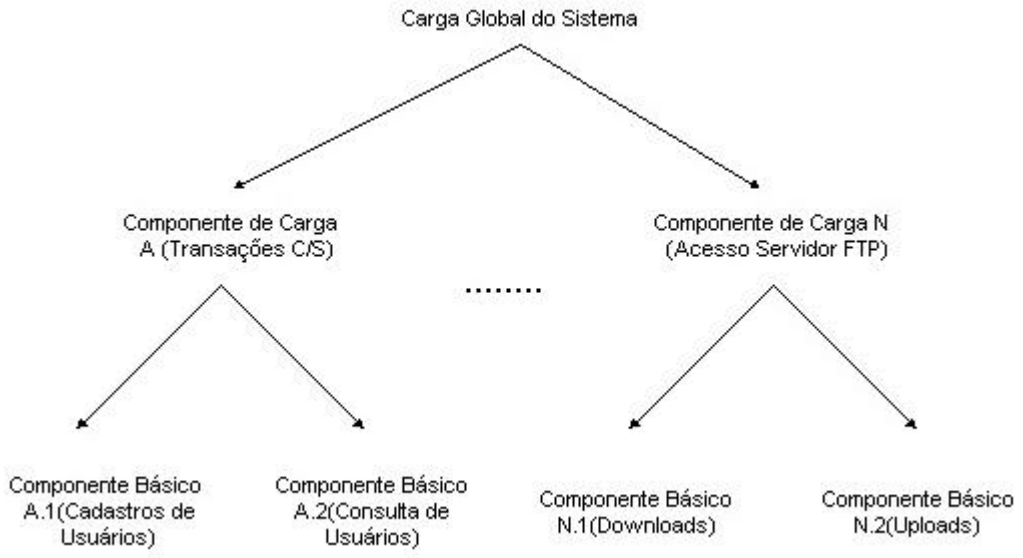


Fig. 3.2 Componentes elementares da caracterização da carga de trabalho

A carga total de um sistema computacional é representada por todas as entradas, transações e requisições que ele pode receber durante um determinado intervalo de tempo. Esta etapa objetiva estabelecer um modelo de carga de trabalho para o sistema sob estudo. Ele é uma representação da carga real do sistema, e como tal precisa ser compacto e conter as características mais importantes e relevantes da carga real para que se possa abstrair-se das características dos milhares de componentes básicos que um sistema computacional pode possuir. Então para obter-se um modelo de carga compacto os recursos escolhidos são em geral aqueles que mais afetam o desempenho do sistema.

A caracterização da carga de trabalho é composta pelas seguintes etapas:

3.2.2.1. Representação dos Componentes da Carga de Trabalho

Os sistemas de informação atuais normalmente manipulam quantidades consideráveis de informação, refletindo assim em uma grande quantidade de dados coletados. Desta forma torna-se necessário abstrair-se do todo e concentrar a análise em aspectos do sistema que demandem, por exemplo, um maior tempo de resposta aos usuários. Por exemplo, para realizar a aquisição de um determinado produto utilizando a Internet realiza-se: **(1)** Requisição da compra, **(2)** Aceitação da compra, **(3)** Processamento da compra, **(4)** Lançamento da compra. As etapas 2 ou 3 podem ser demoradas fazendo com que o tempo de resposta seja alto, indicando que a análise devesse ser centrada em uma destas etapas. De posse destes componentes básicos, deve-se definir sua parametrização considerando:

- Intensidade da Carga de Trabalho: Indica o número de entradas, transações e requisições a que estão sendo submetidos os recursos do sistema.

- Demanda de Serviço: Solicitação por demanda aos recursos do sistema por parte de cada componente básico.

3.2.2.2. Monitoração e Coleta de Dados

Esta é a etapa de parametrização dos valores dos componentes básicos do modelo de carga de trabalho, através dos seguintes passos:

- Identificar a janela de tempo que define a sessão de medição: Corresponde ao intervalo de tempo necessário para observação da carga de trabalho e a sua demanda por recursos do sistema. Por exemplo, o instante de tempo em que um servidor de banco de dados recebe o maior numero de requisições de serviços;

- Monitorar e medir as atividades do sistema computacional durante a janela de tempo: Utilizar monitores de desempenho incorporados aos sistemas operacionais de rede ou sistemas gerenciadores de banco de dados, analisar seus arquivos de *log*, etc;

- Mediante a coleta de dados definir a padronização de valores para os componentes básicos da carga de trabalho.

3.3. Avaliação de Desempenho

Avaliação de desempenho é segundo [SENE, 2000] o processo utilizado para responder perguntas sobre o comportamento de um sistema em estudo. [JAIN, 1991] afirma que escolher a técnica de avaliação e uma métrica são dois pontos importantes em qualquer projeto de avaliação de desempenho.

As técnicas de avaliação de desempenho permitem que sejam escolhidos os métodos que melhor exprimem as características do sistema real que se deseja avaliar, as principais técnicas utilizadas são: as medições, a modelagem analítica e a simulação.

As medições, realizadas diretamente no sistema real são métodos simples, mas que requerem a construção propriamente dita do modelo real ou de um protótipo a ser avaliado. Para a simulação e modelagem analítica, faz-se necessário a criação de modelos de representação da realidade para a realização dos experimentos e coleta de resultados e posterior avaliação. As diferentes técnicas, empregadas na simulação e modelos analíticos, estão relacionadas com o tipo de modelo criado para representar um sistema real. Os modelos analíticos permitem a construção de uma representação da realidade através do uso do formalismo matemático, com ou sem auxílio de estruturas gráficas. As três principais abordagens de modelagem analítica são as Cadeias de Marcov, as Redes de Petri e a Teoria das Filas.

[PRADO, 2000] define simulação como uma técnica de solução de um problema pela análise de um modelo que descreve o comportamento do sistema usando um computador digital. Para [JAIN, 1991] simulação é a técnica mais comum para análise de desempenho de sistemas computacionais. Ela é muito útil quando o sistema que precisa ser caracterizado não esta disponível, um modelo de simulação provê uma maneira fácil de predizer desempenho ou comparar alternativas de sistemas. [SILVA, 2002] afirma que hoje, com a crescente competitividade, a simulação tornou-se uma ferramenta muito poderosa para planejamento, projeto e controle de sistemas.

3.3.1. Etapas da Avaliação de Desempenho

Como já citado no item 3.1 [JAIN, 1991] prevê uma série de etapas para a avaliação de desempenho, com devida análise dos resultados. Estas etapas são mostradas nos itens a seguir.

3.3.1.1. Determinar os Objetivos e Definir o Sistema

Em um projeto de avaliação de desempenho o primeiro passo é determinar os objetivos do estudo e definir o que constitui o sistema, delimitando as suas fronteiras. A escolha das fronteiras do sistema afeta suas métricas de desempenho como a carga de trabalho utilizada para comparar duas propostas de modelos de sistemas. Em geral a definição da fronteira esta associada aos objetivos de estudo.

3.3.1.2. Listas de Serviços e Resultados

Todo sistema possui uma lista de serviços com um conjunto de resultados e cada resultado, dentro da perspectiva do usuário, pode ser desejado ou não, por exemplo, quando um usuário clica o botão gravar dados cadastrais de um cliente, em sua interface gráfica, ele espera que os dados sejam gravados com um tempo de resposta aceitável. Esta etapa objetiva definir as listas de serviços e seus possíveis resultados, selecionando as métricas de desempenho e cargas de trabalho mais adequadas.

3.3.1.3. Seleção das Métricas

Definir as métricas do sistema significa estabelecer critérios para a comparação do desempenho. Em geral estas métricas são: velocidade, precisão, e disponibilidade dos serviços. De forma geral existem três categorias para classificação dos resultados: o serviço é realizado corretamente, incorretamente, ou recusa em realizar o serviço.

Quando o serviço é realizado corretamente, seu desempenho é medido pelo tempo para realizar o serviço (tempo de resposta), a taxa na qual o serviço é realizado (*throughput*) e os recursos consumidos (utilização) durante a realização do serviço.

Se o serviço é realizado incorretamente, uma mensagem de erro é gerada. Isto auxilia na classificação dos erros e na determinação da classe de cada erro.

O serviço pode ainda não ser realizado, neste caso diz-se que o sistema *caiu*, *falhou* ou esta *indisponível*. Isto auxilia na classificação das falhas e na determinação da classe de cada falha. A figura 3.3 mostra os três estados de uma solicitação de serviço.

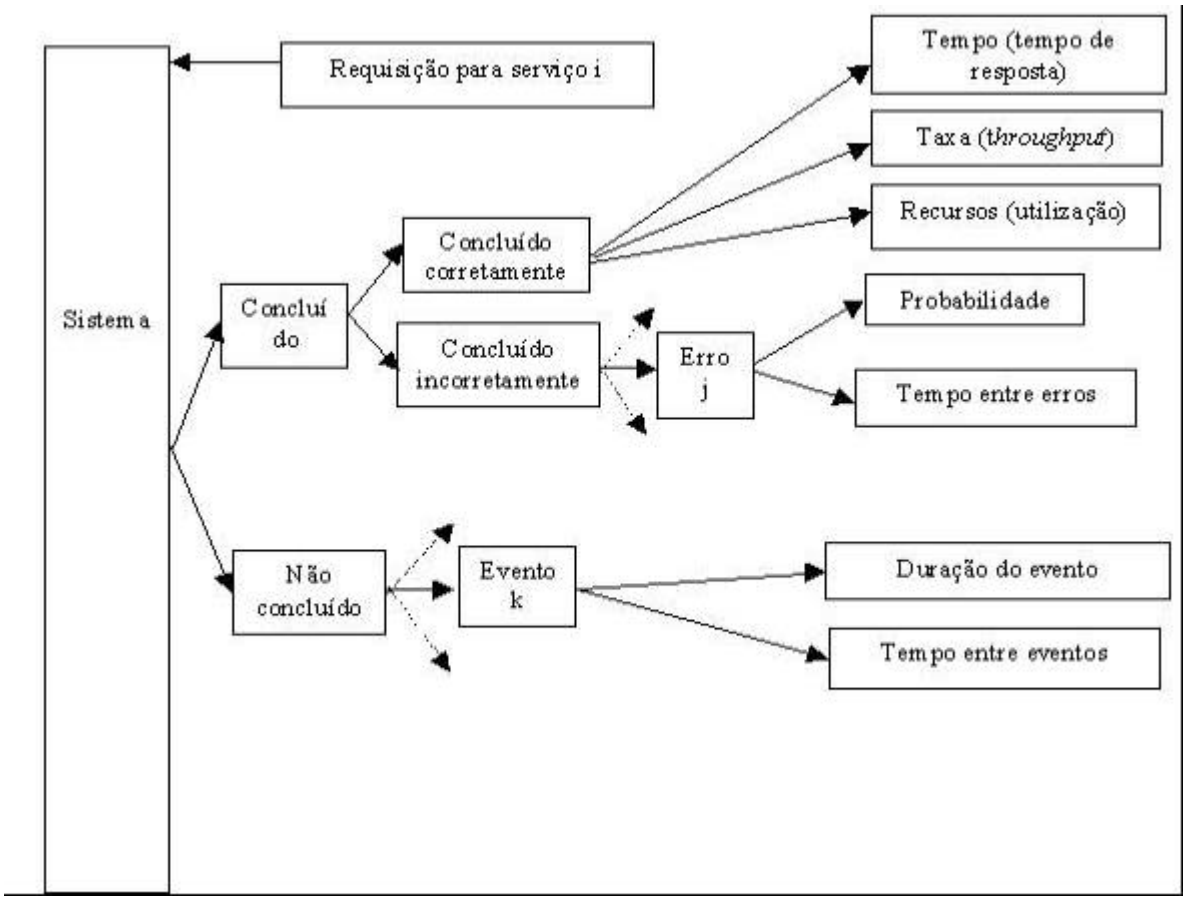


Fig. 3.3 Três estados de uma solicitação de serviço

As métricas de desempenho mais usadas são:

- Tempo de resposta: É o intervalo de tempo entre a solicitação do cliente e as respostas do sistema;

- Throughput: É a taxa (solicitações por unidade de tempo) na qual as solicitações podem ser atendidas pelo sistema;

- Utilização de recurso: É a medida da fração de tempo na qual o recurso esta ocupada atendendo uma solicitação;

- Confiabilidade do sistema: É normalmente medido pela probabilidade de erros ou pelo tempo médio entre erros;

- Disponibilidade do sistema: É a fração de tempo no qual o sistema esta disponível para atender as solicitações.

As métricas em função de sua utilidade, podem ser divididas em:

- Quanto mais alto melhor (HB – *High is Better*): Usuários e gerentes do sistema preferem grandes valores para suas métricas. Ex: *Throughput*;

- Quanto mais baixo melhor (LB – *Lower is Better*): Usuários e gerentes do sistema preferem pequenos valores para suas métricas. Ex: Tempo de resposta;

- Aproximado é melhor (NB – *Nominal is Better*): Grandes e pequenos valores para as métricas são indesejáveis, neste caso um valor na média é melhor. Ex: Utilização de recurso.

3.3.1.4. Lista de Parâmetros

Nesta etapa são definidos todos os parâmetros que podem influenciar o desempenho do sistema. Eles dividem-se em parâmetros de sistema e de carga de trabalho. Os de sistema englobam parâmetros de hardware e software, que geralmente devem ser padrões em todas as instalações do sistema. Os referentes à carga de trabalho são os parâmetros característicos das solicitações clientes, que variam de uma instalação para outra. A lista pode ou não estar completa, isto é, se um novo parâmetro foi descoberto ele pode ser adicionado à lista, permitindo-se determinar quais dados necessitam ser coletados antes e durante a na análise.

3.3.1.5. Seleção de Fatores para Estudo

A lista de parâmetros pode ser divida naqueles que variam durante a avaliação e aqueles que não variam. Os que variam são chamados de *fatores* e seu valores de *níveis*. Pode-se iniciar com uma pequena lista de fatores e um pequeno número de níveis, e expandi-los nas próximas fases do projeto se os recursos assim o permitir. É necessário dar preferência aos fatores que mais influenciam o desempenho, logo um fator de significante relevância não pode ser ignorado simplesmente em função da dificuldade que o envolve. Ao selecionar os fatores, deve-se considerar a economia, política e restrições tecnológicas envolvidas, além das limitações impostas pelos responsáveis pelas decisões.

3.3.1.6. Seleção das Técnicas de Avaliação

As três técnicas para avaliação de desempenho são: a modelagem analítica, a simulação e medições do sistema real.

3.3.1.7. Seleção da Carga de Trabalho

Consiste da lista de serviços requerida para o sistema. Dependendo da técnica de avaliação escolhida a carga de trabalho será expressa de diferentes formas. Nos modelos analíticos geralmente é definida como a probabilidade de várias solicitações ao sistema real. Nas simulações poderia ser representada pela execução passo a passo das solicitações no sistema real. E nas medições ela consiste em *scripts* que serão executados no sistema. Sendo que em todos os casos a carga de trabalho deve representar o uso do sistema na vida real.

3.3.1.8. Projeto de Experimentos

A partir dos fatores e seus níveis, são decididos a seqüência de experimentos que obtenham o máximo de informação com o mínimo de esforço. O projeto de experimentos

pode ser definido em duas fases, a primeira onde o número de fatores é grande mas o número de níveis é pequeno, com o objetivo de determinar o efeito relativo dos vários fatores. E em uma segunda fase, o número de fatores é diminuído é o número de níveis daqueles que tiveram uma maior relevância é incrementado.

3.3.1.9. Análise e Interpretação dos Resultados

É importante identificar que medições e simulações, por tratarem-se de experimentos randômicos, conduzem a quantidade de dados diferentes a cada medição ou simulação toda vez que são repetidas. Na comparação de duas alternativas de sistemas, é necessário levar em conta a variabilidade dos resultados, pois simplesmente comparar os meios pode conduzir a errôneas conclusões. A interpretação dos resultados é a chave para entender que uma análise conduz a resultados e não a conclusões, resultados estes que servirão como base para que os responsáveis pelas decisões possam desenhar conclusões.

3.3.2. Modelo de Desempenho

O modelo de desempenho deve ser representativo daquele existente no mundo real com as solicitações e os recursos do sistema sendo representadas graficamente por entidades interconectadas no modelo de simulação. Depois de desenvolvido, o modelo de simulação ele deve ser verificado e validado. A validação tem foco voltado para a representação das suposições consideradas sobre o comportamento do sistema real, no desenvolvimento do modelo, e a verificação considera as correções na implementação.

3.3.2.1. Verificação do Modelo de Desempenho

Validação e verificação são diferentes conceitos, podendo o modelo de desempenho se encaixar em uma das quatro categorias: inválido e não verificado, inválido e verificado, válido e não verificado, ou válido e verificado. Um modelo inválido e verificado é aquele que

implementa corretamente as suposições, porém elas estão distantes da realidade. Se um modelo de simulação tem sua modelagem e programação realizada por pessoas (ou equipes) diferentes, a modelagem representaria a validação e a programação à verificação.

3.3.2.2. Validação do Modelo de Desempenho

A verificação deve garantir, no mínimo de forma razoável, que as suposições, se implementadas corretamente para o modelo, produziram resultados compatíveis com os observados no sistema real. A validação depende das suposições, então, logo depende também do sistema modelado. Dessa forma, diferentemente das técnicas de verificação, que podem ser generalizada para outros modelos, a validação não é aplicável desta forma.

A validação do modelo consiste em três aspectos chaves:

- Suposições;
- Valores dos parâmetros de entrada e distribuições;
- Valores de saída e distribuições.

Estas três chaves podem estar sujeitas a um teste de validade pela comparação através das seguintes fontes:

- Intuição do especialista;
- Medidas do sistema real;
- Resultados teóricos.

Isto leva a nove possíveis testes de validação, mas é claro que nem sempre será possível utilizar todas as combinações. Por exemplo, nenhuma medida do sistema real ou resultados teóricos podem estar disponíveis. De fato, em muitas situações reais, as nove

possibilidades não serão praticáveis, uma vez que a simulação é aplicada se não houver mais nenhum meio confiável de obter os dados.

3.4. Comparação de Duas Alternativas de Projetos de Sistemas

[BANKS et al., 1999] afirma que um dos mais importantes usos da simulação é a comparação de projetos alternativos de sistemas, e apresenta métodos estatísticos que podem ser usados para comparar duas ou mais projetos de sistemas com base em alguma medida de desempenho. Duas abordagens estatísticas serão apresentadas, uma baseada no cálculo de um *intervalo de confiança* entre as diferenças dos valores esperados da medida de desempenho, tempo de resposta, por exemplo, para cada alternativa de projeto de sistema. Esta técnica de abordagem quantifica estas diferenças, se estas existirem. E outra baseada na realização de um *teste de hipóteses*, para verificar se as diferenças nos valores esperados das duas variáveis, expressivamente, não conduzem a um valor nulo, de modo que os resultados entre as diferenças conduzem a uma aceitação ou rejeição da hipótese testada. A técnica estatística utilizada será baseada no cálculo do intervalo de confiança. Os desvios padrão obtidos serão analisados para verificar de que forma os resultados obtidos distribuem-se, isto é, se a distribuição é mais, ou menos, dispersa.

A medida média de desempenho para o sistema i será denotada por θ_i (i = 1,2). O objetivo dos experimentos de simulação é obter pontos e intervalos estimados das diferenças no desempenho médio, denotado por $\theta_1 - \theta_2$. Quando se comparam duas alternativas de sistemas, faz-se necessário decidir o tamanho das replicações $T^{(i)}_E$ para cada modelo (i = 1,2), e o número de replicações R_i realizada para cada modelo, então para r replicações do sistema i, obtém-se uma estimativa Y_{ri} da medida média de desempenho, θ_i. Das simulações dos dois modelos de sistema obtém-se os valores médios denotado através de θ_i, para i variando no intervalo (1,2), sendo assim, do resultado de r simulações para cada modelo de sistema obter-se-á um valor médio. Ao término das simulações realizadas dos dois sistemas modelados obtém-se os valores médios, θ_1 e θ_2, que indicam estimativas (pontuais) da verdadeira média de cada um. Como o objetivo é a constatação da existência, ou não, de diferenças entre estes dois sistemas ou alternativas, opera-se $\theta_1 - \theta_2$, que se trata de uma estimativa, também pontual, para a diferença entre ambos. A conclusão final será baseada na relevância das

diferenças entre $\theta_1 - \theta_2$, então se pressupõem a determinação de um intervalo de confiança para a estimativa pontual da diferença d ($d = \theta_1 - \theta_2$) entre os valores médios.

Três métodos de computar o intervalo de confiança $\theta_1 - \theta_2$ serão discutidos, considerando os valores de saída encontrados, e três possíveis conclusões poderão ser tiradas após o cálculo do intervalo:

- O intervalo de confiança de $\theta_1 - \theta_2$ está à esquerda de zero. Neste caso, afirma-se que existe uma forte evidência a favor da hipótese de que $\theta_1 - \theta_2 < 0$ ou, de forma equivalente, que $\theta_1 < \theta_2$;

- O intervalo de confiança de $\theta_1 - \theta_2$ está à direita de zero. Neste caso, afirma-se que existe uma forte evidência a favor da hipótese de que $\theta_1 - \theta_2 > 0$ ou, de forma equivalente, que $\theta_1 > \theta_2$;

- O intervalo de confiança de $\theta_1 - \theta_2$ contém o zero. Neste caso, com base nos dados disponíveis, não se pode afirmar que exista uma forte evidência estatística a favor da hipótese de que um dos sistemas seja melhor do que o outro.

Para uma melhor compreensão dos três métodos de computar o intervalo de confiança, a distinção entre significâncias estatística e prática, para desempenho de sistemas, deve ser estabelecida. [BANKS, 1999] define *significância estatística* como a função dos experimentos de simulação e dos seus dados de saída. Se os dados coletados são suficientes para garantir que as diferenças observadas são reais e é possível concluir considerando 1 ou 2, então as evidências estatísticas existem, e pode-se afirmar que existe uma significância estatística entre as diferenças. Mas se a conclusão recai no item 3, as evidências citadas não existem, logo não existe significância estatística entre as diferenças mesmo considerando que os dois modelos de sistemas são diferentes.

[BANKS, 1999] define *significância prática* como a função da real diferença entre os sistemas, independente dos experimentos simulados. Então será a diferença entre $\theta_1 - \theta_2$ suficientemente grande para justificar a decisão a ser tomada? Considerando a hipótese da existência de dois projetos alternativos de sistemas, sistemas 1 e 2, onde o tempo de resposta dos dois é tão insignificante que passa desapercebido pelo usuário, pode-se concluir que do ponto de vista prático tanto faz a implementação de 1 ou 2.

Pode-se concluir, considerando as significâncias estatística e prática, que a técnica de intervalos de confiança não condiciona a uma tomada de decisão considerando a pergunta relacionada à questão da significância prática. Entretanto se aspectos científicos precisam ser considerados, a significância estatística aumenta em relevância, podendo-se afirmar com probabilidade 1- α que a real diferença entre θ_1 e θ_2 encontra-se entre limites definidos. Então estas considerações dependerão, significativamente, dos aspectos considerados importantes envolvidos, e o problema sob análise que precisa ser resolvido.

3.4.1. Determinação do Intervalo de Confiança para Avaliar Diferenças Entre Dois Projetos Alternativos de Sistemas

Para se estabelecer o intervalo de confiança da diferença entre as duas médias amostrais, é necessário que o número de observações presentes nas duas sejam iguais logo $r_1 = r_2$. A partir desta igualdade são então estabelecidos pares de valores, uma vez que cada amostra do modelo de sistema 1 possui um correspondente no modelo de sistema 2. Deste conjunto de pares calcula-se as diferenças entre cada elemento Y_{1j} da amostra 1 e seu par correspondente Y_{2j} da amostra 2, isto é, $D_j = Y_{1j} - Y_{2j}$, para $j = 1, 2, ...r$. Desta forma, os D_j's, são variáveis aleatórias i.i.d (independentemente e identicamente distribuídas) e $E(D_j) = \delta$ será uma estimativa pontual da média destas diferenças. Do valor δ deseja-se calcular o intervalo de confiança, chegando-se à estimativa da média amostral.

Para o cálculo do intervalo de confiança, primeiramente define-se o valor de $\overline{D}_{(r)}$, isto é, a média das diferenças para todas as r amostras.

$$\overline{D}_{(r)} = \frac{\sum_{j=1}^{r} D_j}{r}$$

Fig. 3.4 Média das diferenças

A variância deste valor será dada por:

$$S^2_{[\overline{D}_{(r)}]} = \frac{\sum_{j=1}^{r}[D_j - \overline{D}_{(r)}]^2}{r(r-1)}$$

Fig. 3.5 Variância

O desvio-padrão será:

$$s_{[\overline{D}_{(r)}]} = \sqrt{S^2_{[\overline{D}_{(r)}]}}$$

Fig. 3.6 Desvio padrão

[BARBETTA, 1998] caracteriza a variância e o desvio padrão como medidas que fornecem informações complementares à informação contida na média aritmética, usada no cálculo da variância. Elas avaliam a dispersão do conjunto de valores em análise.

O valor do intervalo de confiança, para 100(1-α) é dado por:

$$\overline{D}_{(r)} \pm t_{r-1,1-\alpha/2} s_{[\overline{D}_{(r)}]}$$

Fig. 3.7 Intervalo de confiança

O intervalo de confiança será igual à probabilidade 1-α se os D_j's forem normalmente distribuídos. Do contrário a probabilidade será próxima de 1-α, na medida que r aumentar, obedecendo o teorema do limite central. Para $r < 30$, normalmente, emprega-se os valores da distribuição t.

A abordagem fornecida pela figura 3.7 para o cálculo do intervalo de confiança, para as diferenças entre amostras pareadas, reduz um problema baseado em duas amostras, para um envolvendo uma única, as diferenças D_j's.

Este capítulo descreveu de maneira geral o uso da metodologia para avaliação de desempenho de sistemas computacionais, abordando suas etapas na compreensão do ambiente estudado, e a definição e caracterização da sua carga de trabalho para o desenvolvimento de modelos de cargas e desempenho visando estabelecer padrões para caracterização do modelo real de um ambiente computacional. Além de técnicas estatísticas para a comparação de duas alternativas de projetos de sistemas.

O capítulo seguinte apresenta a aplicação da metodologia considerando os dois modelos de ambientes que serão analisados, modelados e simulados no software ARENA, suas simulações, bem como a análise dos resultados obtidos.

4. Aplicação da Metodologia no Contexto DETRAN-PA

Este capítulo aborda a aplicação da *Metodologia de Planejamento de Capacidade e Análise Desempenho de Sistemas* mostrada no capítulo 3. Neste capítulo serão descritos: O entendimento do ambiente, considerando seus elementos relevantes para os objetivos a que se propõe este trabalho, para a avaliação de desempenho dos dois modelos cliente/servidor, 2-camadas e 3-camadas, as variáveis de desempenho associadas ao negócio, a carga e o modelo de carga, ao que o ambiente será submetido. Nele também será desenvolvido o modelo de desempenho, definido o projeto experimental e analisado os resultados obtidos a partir dos experimentos simulados.

4.1. Entendimento do Ambiente DETRAN-PA

O Departamento de Trânsito do Estado do Pará (DETRAN-PA) é composto por uma LAN localizada em sua sede, na capital Belém, mais o chamado posto avançado que se localiza a alguns quilômetros da sede e uma WAN que interliga os demais pontos distribuídos pelo Estado. A LAN é composta por 415 pontos na sede dispostos em várias diretorias e setores em uma área de aproximadamente 10.600 m^2 quadrados e 21 pontos no posto avançado, localizado também na capital a uns 7 quilômetros da sede. Este posto comunica-se com a sede através de um roteador utilizando um link de 120 Kbytes. Estes pontos são todos conectados a *Hubs* de 10 e 100 Mbits e interligam PC também com placa de rede de 10 e de 100 Mbits. As chamadas CIRETRANS (Circunscrições Regionais de Trânsito), que compõem a WAN, são em número de 15 e possuem 131 pontos espalhados por 15 municípios, interligados por *hubs* de 10 e 100 Mbits e placas de rede de 10 e 100 Mbits interligando-se com a sede através de roteadores utilizando um link dedicado de 64 KBytes. Dentre os vários setores que compõem o ambiente computacional do DETRAN-PA o foco de estudo será o setor de atendimento referente à habilitação de condutores que é composto por 19 PC e utilizam *hubs* de 10 Mbits. A figura 4.1 ilustra de maneira simplificada o ambiente sob estudo.

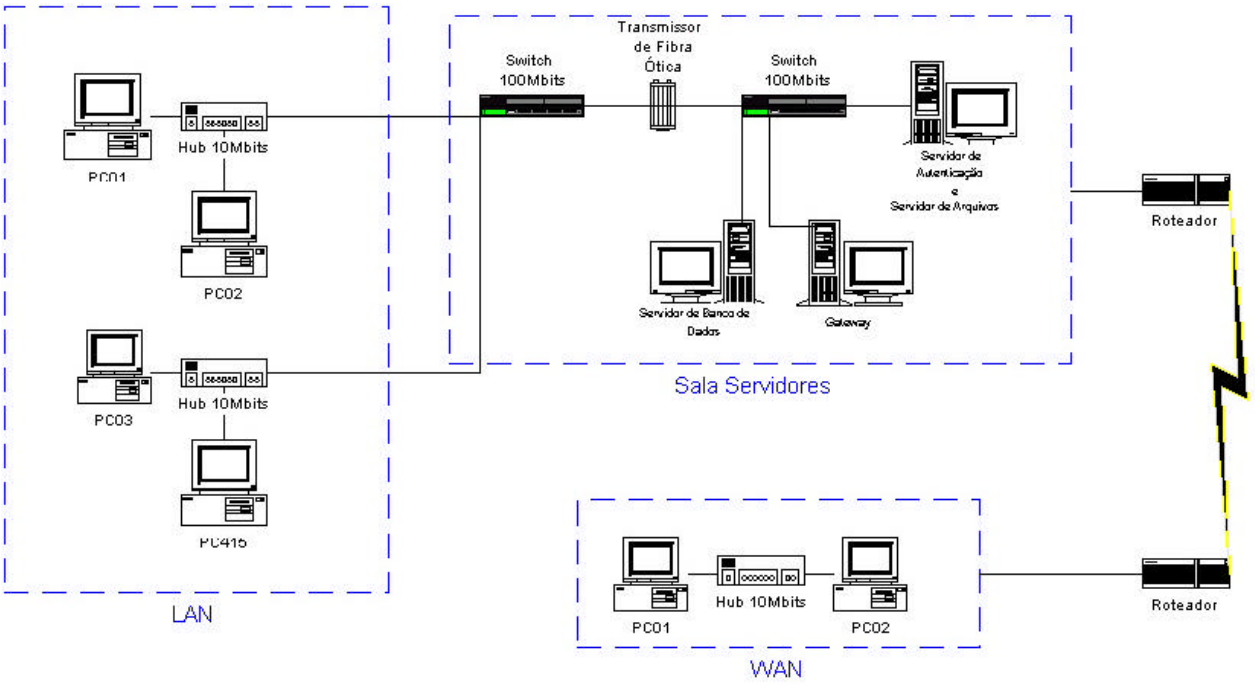


Fig. 4.1 O ambiente DETRAN-PA

- Hardware: Os PC clientes, incluindo LAN e WAN, são processadores da família Intel do tipo Celeron, Pentium II e algumas máquinas da família AMD. O servidor de autenticação/arquivos possui um processador Pentium III, o servidor de banco de dados é uma máquina HP RISC L200 multiprocessada, o servidor de aplicação localiza-se em outra RISC L1000 com um processador, e o *gateway* possui um processador Pentium III. Existem ainda mais dois servidores não representados na figura 4.1, tratam-se dos servidores *Web* e *DNS*. O servidor *Web* é da plataforma Intel e possui dois processadores Pentium III, e assume também o papel de servidor *DNS*. A rede local é uma Ethernet conectada por placas de rede e *hubs* de 10 e 100 Mbits. Existem ainda três *switches ethernet* nível 3 de 100 Mbits que alimentam todos os *hubs* e mais um quarto, também nível 3 e de 100MBits, ao qual os outros estão conectados através de meio ótico, este quarto *switch* também alimenta os três servidores.

- Softwares: A figura 4.1 mostra alguns dos servidores do Departamento de Trânsito do Estado do Pará. A máquina RISC roda o sistema operacional de rede HPUX e contém duas instâncias banco de dados *Oracle8i,* uma de desenvolvimento contendo os dados migrados do banco ADABAS e outra contendo as informações referentes ao sistema de Legislação, que controla um dos

requisitos necessários para emissão da Carteira Nacional de Habilitação, que está em produção. Esta máquina também possui o sistema de informação do DETRAN-PA, desenvolvido em linguagem de programação NATURAL que acessa um banco de dados ADABAS. O servidor para autenticação dos usuários da rede roda o sistema operacional Windows2000, tratando-se assim de uma rede *Microsoft*. O *gateway* rodando o *Microsoft SNA Server* sob Windows NT 4.0, para integração com o sistema legado, possui uma DLL (*Dynamic Linked Library*), desenvolvida no ambiente *Microsoft Visual C++*, responsável pela comunicação do DETRAN com o SERPRO (Serviço Federal de Processamento de Dados), uma vez que algumas informações referentes aos condutores e veículos necessitam estar replicadas na base SERPRO para controle de fraudes em âmbito nacional. O servidor *Web* é também Windows2000 com *Internet Information Server*, os servidores de *ftp* e o servidor de *e-mail Exchange Server*, além de servidor *DNS*. Todos os PCs clientes rodam o SO Windows 98 e possuem clientes de *e-mail* e *browsers*, além do pacote *office (Staroffice da Sun Microsystems ou MSOffice da Microsoft)* contendo processador de texto, planilha eletrônica e software para criação de apresentações. Os PC do setor de informática possuem clientes *ftp* para recebimento dos arquivos texto das multas recolhidas pela Prefeitura de Belém, além de emuladores de terminal utilizados para acessar o sistema de informação do DETRAN. As máquinas clientes deste setor possuem ainda o ambiente para desenvolvimento de sistemas de informação Delphi 5 e software cliente para acesso ao banco de dados *Oracle*, algumas delas possuem ainda o banco de dados Interbase que se encontra sob estudo para desenvolvimento de futuros sistemas, softwares para criação e publicação de *home pages*, como o *Microsoft Front Page*, também são utilizados. As máquinas clientes do setor de atendimento aos usuários dos serviços do DETRAN utilizam apenas os emuladores de terminal e o pacote *office*. O setor de legislação, responsável pela impressão das provas sobre legislação de trânsito, possui um SI que controla a impressão e correção das provas além do cliente para acessar o banco de dados *Oracle*. Outros softwares, editoração de imagens, como *Corel Draw* e *Adobe Photoshop* são utilizados pelas acessórias como a Acessória de Comunicação.

- Elementos de Conectividade e Protocolos: A LAN roda sob o conjunto de protocolos Internet TCP/IP, porém o *gateway* roda sob o protocolo LU6.4 que é

padrão para comunicação entre mainframes. A WAN também roda sobre a pilha de protocolos TCP/IP.

4.2. As Variáveis de Desempenho Associadas ao Negócio

A Lei nº 9.503, de 23 de setembro de 1997, institui o novo Código de Trânsito Brasileiro (CTB). Deste modo todos os DETRAN's são obrigados a adequar-se às novas regras estabelecidas pelo CTB como: organizar e manter o Registro Nacional de Carteiras de Habilitação (RENACH) e o Registro Nacional de Veículos Automotores (RENAVAM) [DENATRAN, 2002]. Estes registros contêm os dados cadastrais sobre todos os condutores de veículos e todos os veículos automotores de um Estado. Além disto alguns dados identificatórios referentes aos condutores e veículos precisam ser replicados ao SERPRO para o exercício de um controle em âmbito nacional de possíveis fraudes envolvendo estes condutores e veículos.

Com o novo CTB e suas modificações o DETRAN-PA adequou seu SI às novas modificações, o sistema compreende dois módulos, RENACH e RENAVAM, responsáveis por todos os serviços envolvendo condutores e veículos.Os dois módulos citados foram originalmente portados para o ambiente centralizado, utilizando um IBM 9672 R24 pertencente ao PRODEPA (Processamento de Dados do Estado do Pará). O DETRAN-PA utilizava apenas terminais e o tempo de resposta do sistema não era comprometido apesar de possuir muitos pontos espalhados pelo Estado, o banco de dados era portado para mainframe e mesmo em momentos de pico o *throughput*, envolvendo o número de requisições por segundo, no banco de dados ADABAS também não era problema.

A administração de 1997 iniciou, substituição dos terminais por PC na sede e em todos os pontos espalhados pelo Estado, a criação das LAN e WAN do DETRAN-PA baseadas em redes *Microsoft* utilizando servidores Windows NT. Os PC utilizavam emuladores de terminal para acessar o banco de dados na máquina IBM e todos os usuários necessitavam ser autenticados para utilizar os recursos da rede. O tempo de resposta e o *throughput* do sistema permaneceram inalterados com as modificações pelo qual passou o Departamento.

O elevado custo do ambiente legado e a dificuldade na manutenção de seus programas, levaram, em 1999, ao início do projeto de um novo sistema baseado no modelo cliente/servidor 2-camadas que substituiria o sistema legado. Ele possuía como *front end* uma

GUI desenvolvida no ambiente *Borland Delphi 5.0* e como *back end* o banco de dados *Oracle8i,* os clientes eram processadores da família *Celeron* com SO Windows 98 e um servidor de com processador *Pentium II* para servidor de banco de dados. O serviço de primeira habilitação foi o primeiro a ser testado, devido representar o maior número de requisições ao banco de dados em relação aos outros serviços envolvendo condutores. O teste inicial foi no Departamento de Informática utilizando cinco clientes, o teste seguinte ocorreu no posto avançado do DETRAN-PA. No primeiro teste o tempo de resposta permaneceu aceitável, porém no segundo foi muito elevado, o *throughput* foi desconsiderado nos testes, mas ficou claro que seria necessário um servidor mais robusto para o banco de dados. Havia ainda um complicador, pois o servidor do SERPRO onde as informações locais de cada DETRAN estão armazenadas é um mainframe que se comunica através do protocolo LU6.4 enquanto que a rede DETRAN baseava-se nos protocolos TCP/IP. Existia a possibilidade do uso de ODBC, mas era uma alternativa de custo elevado. Essa problemática levou ao desenvolvimento de uma DLL, localizada no *gateway*, responsável pela comunicação com o servidor do SERPRO. A figura 4.2 mostra o modelo de requisições ao banco de dados considerando apenas a LAN.

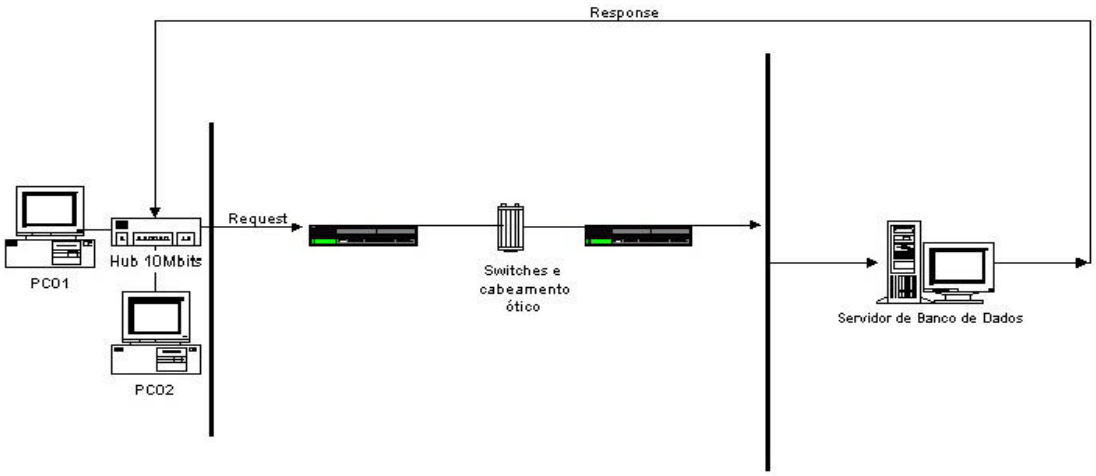


Fig. 4.2 Requisições ao Servidor de Banco de Dados

Recentemente foi realizado o *downsizing* da máquina IBM para a máquina RISC, eliminando assim a utilização do mainframe. Uma alternativa de projeto utilizando o conceito de AppServer está sendo estudada para viabilizar a utilização do SI do DETRAN-PA dentro da arquitetura cliente/servidor 3-camadas.

4.3. Descrição da Carga e do Modelo de Carga para o Ambiente em Estudo

Este item tem como objetivo descrever a carga de trabalho a que esta submetida o ambiente computacional do DETRAN-PA segundo a metodologia abordada no capítulo 3. Esta descrição limita-se à utilização do servidor de banco de dados, uma vez que os dois modelos de simulação serão submetidos à mesma carga.

Os dois módulos, RENACH e RENAVAM, do SI do DETRAN-PA são responsáveis pela carga total a que o servidor de banco de dados é submetido. Cada serviço, referente a condutor ou veículo, do Departamento de Trânsito pode ser decomposto em um determinado número de transações SQL: *Select*, *Insert*, *Delete* e *Update.* As informações sobre estas transações foram obtidas durante a modelagem do novo sistema cliente/servidor mais informações adicionais fornecidas pelo Departamento de Informática, o que permitiu definir a quantidade de *Inserts* e *Updates* para os servidores envolvendo o módulo RENACH. Porém por falta de informações mais detalhadas os *Selects* e *Deletes*, serão ignorados, e os *Updates* por compreenderem uma fase onde o intervalo de tempo para sua execução não é determinado, dependendo de interações externas, serão também ignorados.

4.3.1. Caracterização da Carga de Trabalho

A figura 4.3 mostra a caracterização dos componentes básicos de requisições ao servidor de banco de dados. Enquanto a tabela 4.1 mostra os componentes básicos e os parâmetros de intensidade (IC) e demanda por recursos (DS).

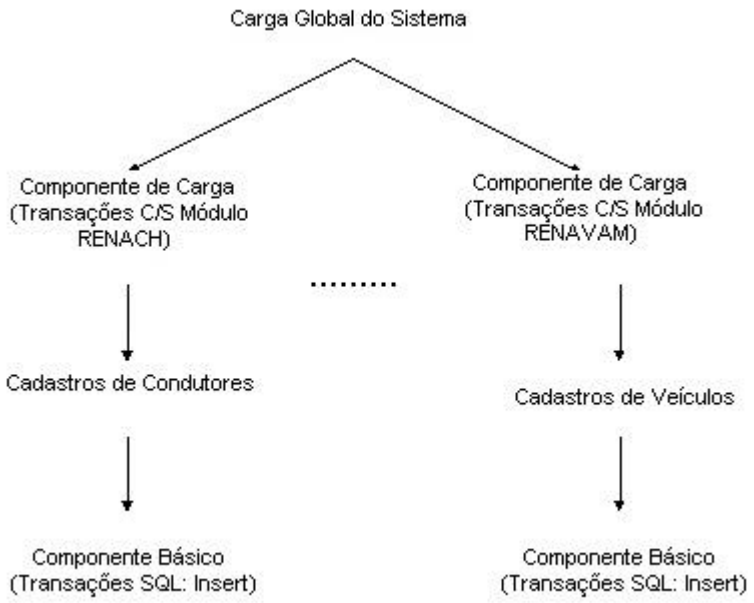


Fig. 4.3 Caracterização dos componentes básicos

Componente básico	Parâmetro
Nº Clientes	IC
Nº Transações/Cliente	IC
Nº de I/Os ao servidor de banco de dados	DS
Utilização da CPU servidor de banco de dados	DS
Tamanho das mensagens enviadas/recebidas pelo servidor de banco de dados	DS

Tab. 4.1 Componentes básicos e parâmetros

Cada serviço envolve quatro principais tabelas, que para simplificação serão denotadas: Tabela01 até Tabela04. Cada tabela recebe um tipo de *Insert*, em consideração às suas diferentes estruturas, que pode ser considerado padrão, quando analisado um conjunto de dois ou mais serviços. O serviço de primeira habilitação, por exemplo, gera 4 *Inserts*.A tabela 4.2 mostra o número de I/O realizado no servidor para um tamanho médio de mensagens envolvendo a operação de *Insert*.

Tabela	TextData (bytes)	I/Os
01	545	50,3
02	219	29,0
03	153	34,1
04	151	26,5

Tab. 4.2 I/O para requisições Insert

Assim o modelo de carga a do ambiente DETRAN-PA pode ser compreendido como um conjunto de requisições SQL, *Inserts*, às quais o servidor de banco de dados é submetido. Faz-se necessário esclarecer que apesar do banco de dados do Departamento de Trânsito ser o *Oracle 8i*, os testes acima foram realizados no banco de dados *Microsoft SQL 2000 Server*.

4.3.2. Os Serviços e seu Fluxo de Solicitação

Serão considerados apenas os serviços envolvendo condutores de veículos, uma vez que as informações existentes são referentes a esta classe de serviço. A Primeira Habilitação pode ser utilizada como exemplo para compreensão dos demais serviços, pois contém o maior número de requisições a tabelas do banco de dados, os demais podem ser entendidos como um subconjunto destas requisições. Os serviços somam um número de sete: Primeira Habilitação, Reabilitação, Renovação, Mudança de Categoria, Segunda Via da CNH, Emissão da CNH Definitiva e Adição de Categoria. Cada cliente pode realizar qualquer serviço, que corresponde um número de requisições *Inserts* ao banco de dados, e o serviço é dado como concluído quando uma seqüência destas requisições é processada por completo. Como a reabilitação é uma nova primeira habilitação, e as sua porcentagem em relação aos outros serviços é pequena, eles serão considerados apenas um serviço.

As requisições possuem todas a mesma prioridade com os servidores atendendo-as segundo o padrão FIFO (*fisrt in first out*), em ambientes multiprocessados existe a possibilidade de várias requisições serem processadas ao mesmo tempo (*multithread*). O AppServer, no entanto, segue uma regra de prioridade para as requisições: mensagens de serviço completadas possuem prioridade sobre as de serviço solicitado, assim quando uma mensagem de serviço completada chega ao AppServer ele a processa e se não houver outras

na fila passa a atender às solicitações de serviços. A tabela 4.3 mostra a relação de serviços e as suas respectivas quantidades de requisições.

Serviços	Nº Inserts
Renovação	3
Primeira Habilitação	4
Reabilitação	4
Emissão da CNH Definitiva	3
Mudança de Categoria	3
Segunda Via da CNH	3
Adição de Categoria	3

Tab. 4.3Serviços com suas requisições

A figura 4.4 mostra o fluxo das requisições, para a arquitetura 2-camadas, ao banco de dados considerando a LAN do DETRAN-PA. Embora o modelo 3-camadas não esteja concluído a figura 4.5 mostra a descrição lógica deste modelo.

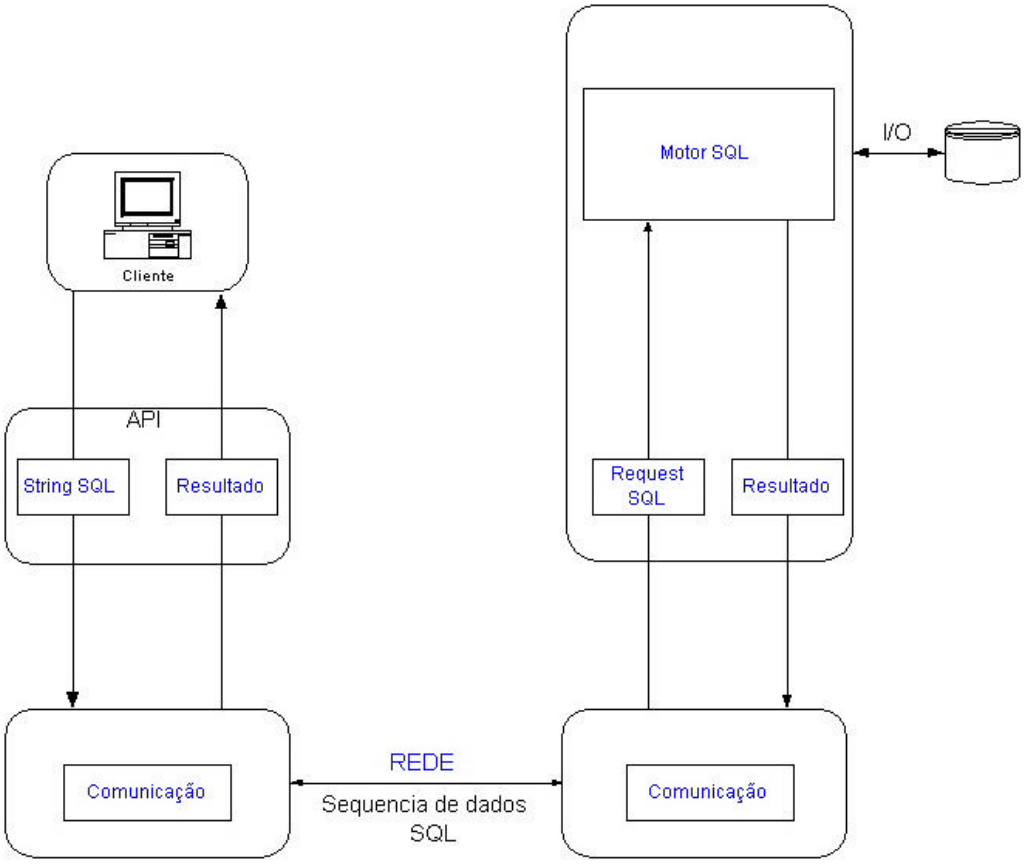


Fig. 4.4 Fluxo Informação modelo 2-camadas

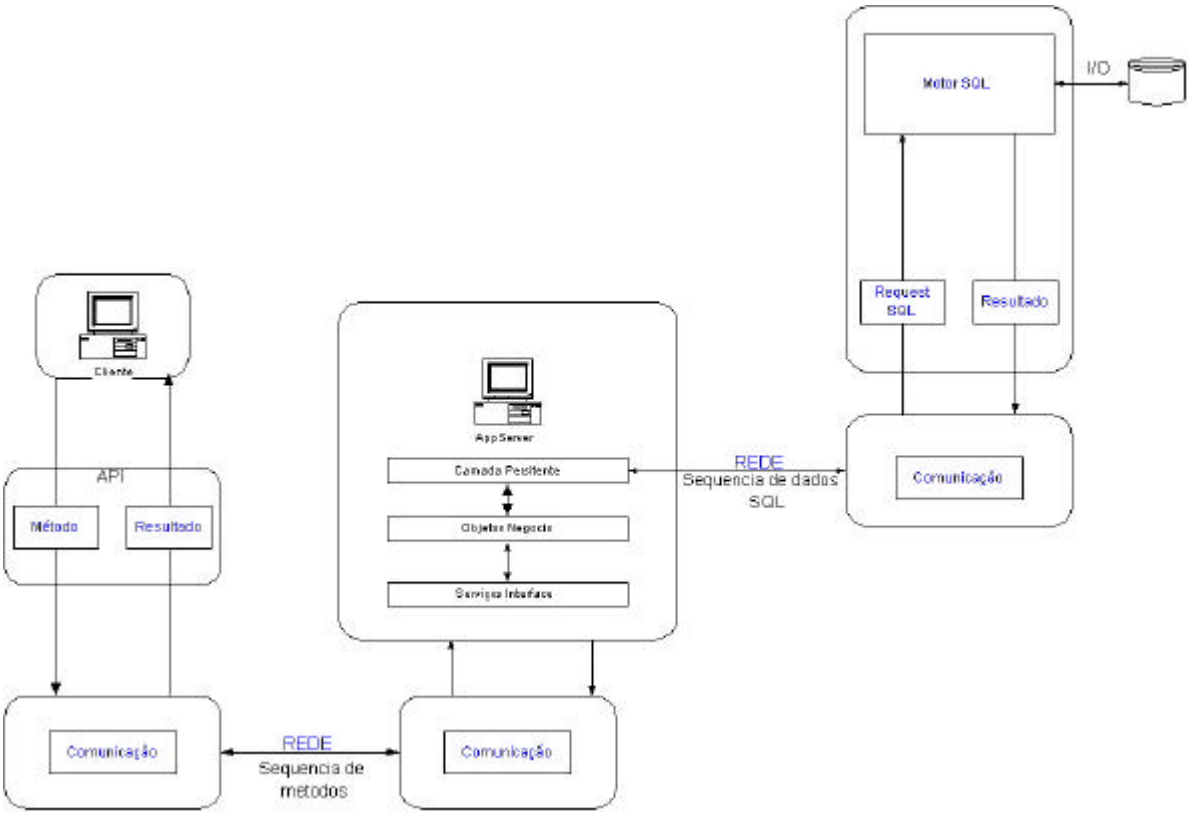


Fig. 4.5 Fluxo Informação modelo 3-camadas

4.4. Desenvolvimento do Modelo de Desempenho para o Ambiente Sob Estudo

Como descrito no item 4.2 o *downsizing* do mainframe IBM para a plataforma cliente/servidor 2-camadas provocou um alto tempo de resposta no sistema quando inicializado remotamente, daí o surgimento da alternativa em 3-camadas. A maior parte do SI foi desenvolvido em 2-camadas, sendo que apenas alguns pequenos módulos funcionam em 3-camadas e sua maioria existe só na forma de proposta de projeto. Os modelos de simulação compreenderão apenas a LAN e seu clientes, isto devido ao fato da dificuldade na coleta de informações perante o setor de Infraestrutura do Departamento de Trânsito referente aos roteadores, *links* e um mapeamento mais detalhado da WAN. Este item descreverá a criação de dois modelos de simulação no ARENA (*Systems Modeling Corporation*) para análise do comportamento das duas alternativas de sistemas.

Como mencionado anteriormente, item 4.3.2, os serviços do DETRAN que serão simulados referem-se aos condutores, módulo RENACH, totalizando um número de sete. Os

dois modelos de desempenho foram construídos baseados no item 4.3.2. O anexo A mostra a descrição gráfica dos modelos.

As distribuições que representam o tráfego gerado pelas solicitações clientes foram extraídas de uma planilha, localizada no anexo B deste trabalho, fornecida pelo Departamento de Informática do DETRAN. Ela contém o número de serviços RENACH realizados durante o período de 2001. O número de serviços de cada mês foi dividido pelo valor 43.200 (quantidade de minutos em 1 mês) para obtenção do número de requisições por minuto ao servidor de dados. As doze taxas, correspondentes aos doze meses do ano, de requisições obtidas foram submetidas ao *Input Analyser* (ferramenta para análise de dados fornecida pelo ARENA) que forneceu as distribuições mostradas na tabela 4.4

Serviços	Distribuição
Renovação	4.35 + LOGN (1.29, 0.964)
Primeira Habilitação e Reabilitação	TRIA (1.5, 2.5, 3.5)
Emissão da CNH Definitiva	0.5 + WEIB (1.39, 3.72)
Mudança de Categoria	NORM (0.756, 0.108)
Segunda Via da CNH	TRIA (0.4, 0.583, 0.79)
Adição de Categoria	LOGN (0.0582, 0.0334)

Tab. 4.4 Serviços com suas distribuições

Os tempos entre respostas do servidor de banco e do servidor de aplicação foram obtidos utilizando como base o *Full Disclosure Report* para o Oracle9iAS (*Oracle 9i Application Server*) [THESERVERSIDE, 2002], a parte relevante deste *report* está localizado no anexo C deste trabalho. *Full Disclosure Report* é a referência ao relatório imprimível em formato PDF ou HTML parte de um *Full Disclosure*. *Full Disclosure* é informação gerada quando um resultado de *benchmark*, o *ECperf*, é finalizado. O *ECperf* é definido pela [SUN MICROSYSTEMS, 2002] como: a maneira de medir a escalabilidade e desempenho de servidores J2EE e *containers*. Os tempos dos servidores foram determinados, considerando o *ECperf* [SUN MICROSYSTEMS, 2002] e o *Full Disclosure Report* [THESERVERSIDE, 2002], segundo as premissas mostradas abaixo:

- 90% (noventa por cento) das requisições devem estar abaixo de 2 segundos;

- A média para o tempo de resposta deve ser igual a 0,805 segundo;

- A capacidade de processamento do servidor de banco de dados deve ser duas vezes a do servidor de aplicação.

Como os três primeiros serviços da tabela 4.4 representam aproximadamente 90% do total, os demais foram descartados e as distribuições dos restantes alteradas para 4.35 + LOGN (1.29, 0.964). Os testes foram então realizados no modelo de simulação 3-camadas até que o valor médio para o tempo de resposta atingisse, aproximadamente, o valor 0,805 segundo. Os valores de tempo determinados, 0,174 segundo para o servidor de aplicação e 0,087 segundo (0,174 / 2) para o servidor de banco de dados, foram usados como base para os demais cenários de simulação. Os resultados das simulações constam do anexo D e o cenário de simulação alcançado em função destes tempos será denominado de cenário de equilíbrio.

Para cada cenário, envolvendo as duas propostas de sistema, foram realizados no Arena, 6 replicações de 500 segundos, totalizando 3000 segundos (50 minutos). As figuras 4.6 e 4.7 mostram os fluxogramas para os modelos 2-camadas e 3-camadas.

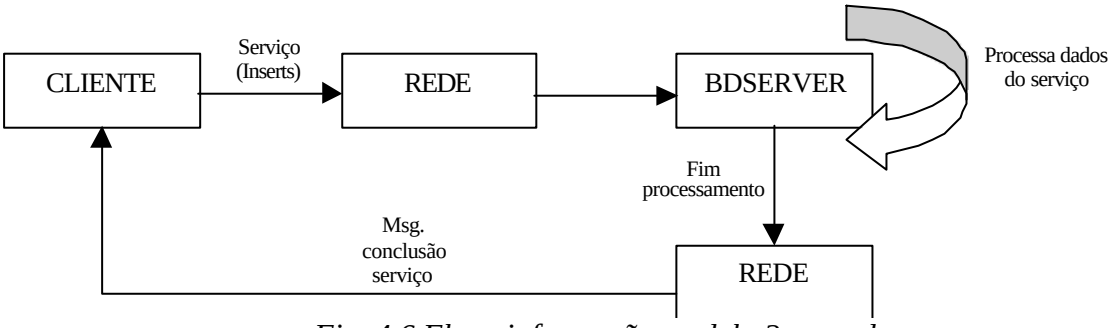


Fig. 4.6 Fluxo informação modelo 2-camadas

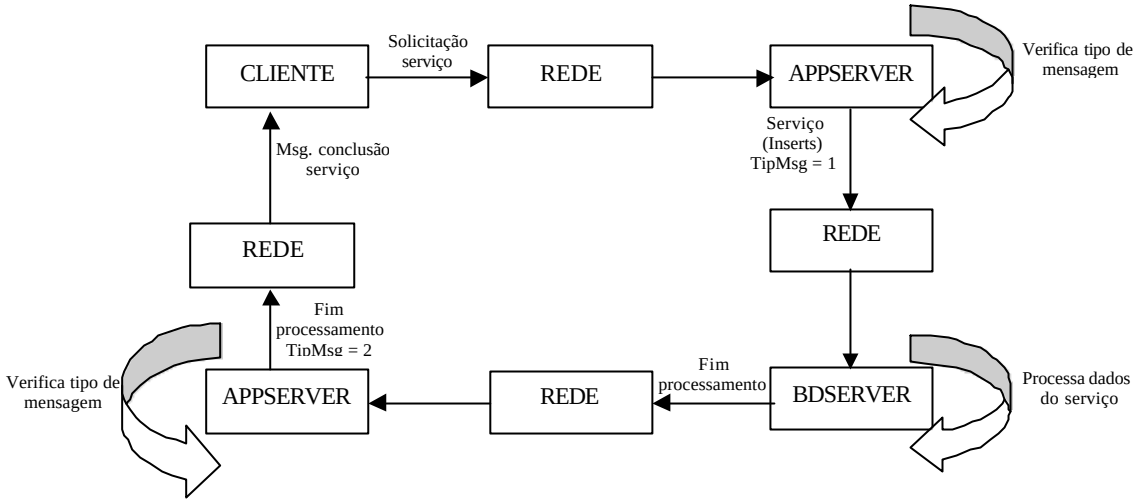


Fig. 4.7 Fluxo informação modelo 3-camadas

4.4.1. Seleção dos Fatores para as Medidas de Desempenho

Parte-se inicialmente de uma lista completa de todos os fatores que podem influenciar as grandezas de desempenho. Abaixo segue uma descrição da lista:

- A carga de serviços submetida ao sistema (transações SQL);
- A quantidade de clientes;
- Tempo de processamento dos clientes;
- Tempo de processamento do servidor de banco de dados;
- Tempo de processamento do servidor de aplicação.

Os fatores da lista acima considerados foram:

- A carga de serviços submetida ao sistema: Como mostrado no item 4.4 a carga de trabalho foi modificada para atender as premissas discutidas neste item;

- Tempo de processamento do servidor de banco de dados: Como o servidor de banco de dados tem duas vezes a velocidade de processamento do servidor de aplicação, ele terá, em hipótese, maior influência para o desempenho dos sistemas;

- Tempo de processamento do servidor de aplicação: Sua análise dentro do modelo 3-camadas é importante para análise dos dois sistemas uma vez que é o elemento diferencial para as duas arquiteturas.

4.4.2. Seleção das Métricas de Desempenho

Considerando que na visão do usuário o tempo de resposta de um sistema é a grandeza de desempenho mais significativa, ele será o elemento de análise para a avaliação comparativa entre as duas propostas de projeto de sistemas.

Para a estimativa do tempo de resposta não é considerado o tempo que o usuário leva para acionar a solicitação de um serviço. Ele passa a ser computado a partir do momento em que a solicitação é realizada, até o recebimento da mensagem de conclusão do serviço. Para o serviço de primeira habilitação, por exemplo, a contagem para o tempo de resposta seria iniciada no momento em que o usuário clicar no botão de “Gravar Serviço” de sua inteface gráfica, e seria finalizado quando recebesse a mensagem “Dados Gravados”.

4.4.3. Projeto Experimental

A etapa do projeto experimental compreende a identificação dos fatores relevantes que podem influenciar as grandezas de desempenho analisadas, no caso deste trabalho o tempo de resposta para a conclusão de um determinado serviço.

Como mostrado no 4.4.1 os fatores considerados foram:

- A carga de serviços submetida ao sistema;

- Tempo de processamento do servidor de banco de dados;

- Tempo de processamento do servidor de aplicação.

Os fatores tempo são variados par a par, a carga de trabalho foi mantida constante depois de alcançado o cenário de equilíbrio e cada variação dos fatores representa um cenário de simulação para cada par de modelos das arquiteturas propostas. As tabelas abaixo mostram os fatores e seus níveis utilizados para a realização de todos os cenários de simulação. Os tempos são dados em segundos.

- Cenário 1

Fatores	Níveis
Tempo de processamento do servidor de aplicação	0,174
Tempo de processamento do servidor de banco de dados	0,087 (0,174 / 2)

Tab. 4.5 Níveis dos fatores para cenário 1

- Cenário 2

Fatores	Níveis
Tempo de processamento do servidor de aplicação	0,174
Tempo de processamento do servidor de banco de dados	0,0435 (0,174 / 4)

Tab. 4.6 Níveis dos fatores para cenário 2

- Cenário 3

Fatores	Níveis
Tempo de processamento do servidor de aplicação	0,174
Tempo de processamento do servidor de banco de dados	0,174

Tab. 4.7 Níveis dos fatores para cenário 3

- Cenário 4

Fatores	Níveis
Tempo de processamento do servidor de aplicação	0,087 (0,174 / 2)
Tempo de processamento do servidor de banco de dados	0,174

Tab. 4.8 Níveis dos fatores para cenário 4

- Cenário 5

Fatores	Níveis
Tempo de processamento do servidor de aplicação	0,0435 (0,174 / 4)
Tempo de processamento do servidor de banco de dados	0,174

Tab. 4.9. Níveis dos fatores para cenário 5

- Cenário 6

Fatores	Níveis
Tempo de processamento do servidor de aplicação	0,0174 (0,174 / 10)
Tempo de processamento do servidor de banco de dados	0,174

Tab. 4.10 Níveis dos fatores para cenário 6

Após obtenção dos resultados das simulações, os resultados dos tempos médios de resposta foram submetidos ao software Statistica, que realiza os cálculos referentes às teorias apresentadas no capítulo 3 da metodologia proposta, para análise dos intervalos de confiança e os respectivos desvios padrão, como proposto no item 3.4. Abaixo serão apresentados os resultados obtidos nas simulações utilizando o ARENA:

- Cenário 1

Fatores				
Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,087 (0,174 / 2)	1	Serviço 01	Serviço 02	Serviço 03
		0,41406	0,50248	0,41611
	2	Serviço 01	Serviço 02	Serviço 03
		0,44565	0,53280	0,41596
	3	Serviço 01	Serviço 02	Serviço 03
		0,40491	0,52279	0,42611
	4	Serviço 01	Serviço 02	Serviço 03
		0,42405	0,51212	0,41375
	5	Serviço 01	Serviço 02	Serviço 03
		0,43105	0,52326	0,42083
	6	Serviço 01	Serviço 02	Serviço 03
		0,43353	0,51249	0,43445

Tab. 4.11 Resultado para cenário 1 no modelo 2-camadas

Fatores					
Tempo de processamento APPServer	Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	0,087 (0,174 / 2)	1	Serviço 01	Serviço 02	Serviço 03
			0,77292	0,85183	0,76535
		2	Serviço 01	Serviço 02	Serviço 03
			0,79430	0,88397	0,76981
		3	Serviço 01	Serviço 02	Serviço 03
			0,75023	0,87610	0,77705
		4	Serviço 01	Serviço 02	Serviço 03
			0,77174	0,86252	0,75974
		5	Serviço 01	Serviço 02	Serviço 03
			0,77951	0,86454	0,76610
		6	Serviço 01	Serviço 02	Serviço 03
			0,78465	0,86385	0,78826

Tab. 4.12 Resultado para cenário 1 no modelo 3-camadas

- Cenário 2

Fatores				
Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,0435 (0,174 / 4)	1	Serviço 01	Serviço 02	Serviço 03
		0,21114	0,25467	0,21141
	2	Serviço 01	Serviço 02	Serviço 03
		0,22210	0,26500	0,21022
	3	Serviço 01	Serviço 02	Serviço 03
		0,20877	0,26545	0,21794
	4	Serviço 01	Serviço 02	Serviço 03
		0,21628	0,25952	0,21234
	5	Serviço 01	Serviço 02	Serviço 03
		0,21857	0,26514	0,21860
	6	Serviço 01	Serviço 02	Serviço 03
		0,21790	0,25826	0,21732

Tab. 4.13 Resultado para cenário 2 no modelo 2-camadas

Fatores					
Tempo de processamento APPServer	Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	0,0435 (0,174 / 4)	1	Serviço 01	Serviço 02	Serviço 03
			0,56036	0,60682	0,56251
		2	Serviço 01	Serviço 02	Serviço 03
			0,56913	0,61225	0,55891
		3	Serviço 01	Serviço 02	Serviço 03
			0,56437	0,61131	0,56663
		4	Serviço 01	Serviço 02	Serviço 03
			0,55851	0,60838	0,56654
		5	Serviço 01	Serviço 02	Serviço 03
			0,56962	0,60794	0,57276
		6	Serviço 01	Serviço 02	Serviço 03
			0,57179	0,59596	0,56633

Tab. 4.14 Resultado para cenário 2 no modelo 3-camadas

- Cenário 3

Fatores				
Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	1	Serviço 01	Serviço 02	Serviço 03
		0,89616	1,0908	0,88483
	2	Serviço 01	Serviço 02	Serviço 03
		0,95563	1,1597	0,90692
	3	Serviço 01	Serviço 02	Serviço 03
		0,87453	1,1451	0,92022
	4	Serviço 01	Serviço 02	Serviço 03
		0,89895	1,1054	0,87139
	5	Serviço 01	Serviço 02	Serviço 03
		0,93752	1,1126	0,87935
	6	Serviço 01	Serviço 02	Serviço 03
		0,94622	1,1150	0,93027

Tab. 4.15 Resultado para cenário 3 no modelo 2-camadas

Fatores					
Tempo de processamento APPServer	Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	0,174	1	Serviço 01	Serviço 02	Serviço 03
			1,2486	1,4492	1,2360
		2	Serviço 01	Serviço 02	Serviço 03
			1,2982	1,5048	1,2580
		3	Serviço 01	Serviço 02	Serviço 03
			1,2274	1,4894	1,2718
		4	Serviço 01	Serviço 02	Serviço 03
			1,2531	1,4562	1,2223
		5	Serviço 01	Serviço 02	Serviço 03
			1,2897	1,4661	1,2222
		6	Serviço 01	Serviço 02	Serviço 03
			1,2996	1,4633	1,2822

Tab. 4.16 Resultado para cenário 3 no modelo 3-camadas

- Cenário 4

Fatores				
Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	1	Serviço 01	Serviço 02	Serviço 03
		0,89616	1,0908	0,88483
	2	Serviço 01	Serviço 02	Serviço 03
		0,95563	1,1597	0,90692
	3	Serviço 01	Serviço 02	Serviço 03
		0,87453	1,1451	0,92022
	4	Serviço 01	Serviço 02	Serviço 03
		0,89895	1,1054	0,87139
	5	Serviço 01	Serviço 02	Serviço 03
		0,93752	1,1126	0,87935
	6	Serviço 01	Serviço 02	Serviço 03
		0,94622	1,1150	0,93027

Tab. 4.17 Resultado para cenário 4 no modelo 2-camadas

Fatores					
Tempo de processamento APPServer	Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,087 (0,174 / 2)	0,174	1	Serviço 01	Serviço 02	Serviço 03
			1,0717	1,2623	1,0577
		2	Serviço 01	Serviço 02	Serviço 03
			1,1281	1,3298	1,0807
		3	Serviço 01	Serviço 02	Serviço 03
			1,0497	1,3153	1,0917
		4	Serviço 01	Serviço 02	Serviço 03
			1,0738	1,2742	1,0410
		5	Serviço 01	Serviço 02	Serviço 03
			1,1140	1,2809	1,0521
		6	Serviço 01	Serviço 02	Serviço 03
			1,1219	1,2808	1,1013

Tab. 4.18 Resultado para cenário 4 no modelo 3-camadas

- Cenário 5

Fatores				
Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	1	Serviço 01	Serviço 02	Serviço 03
		0,89616	1,0908	0,88483
	2	Serviço 01	Serviço 02	Serviço 03
		0,95563	1,1597	0,90692
	3	Serviço 01	Serviço 02	Serviço 03
		0,87453	1,1451	0,92022
	4	Serviço 01	Serviço 02	Serviço 03
		0,89895	1,1054	0,87139
	5	Serviço 01	Serviço 02	Serviço 03
		0,93752	1,1126	0,87935
	6	Serviço 01	Serviço 02	Serviço 03
		0,94622	1,1150	0,93027

Tab. 4.19 Resultado para cenário 5 no modelo 2-camadas

Fatores					
Tempo de processamento APPServer	Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,0435 (0,174 / 4)	0,174	1	Serviço 01	Serviço 02	Serviço 03
			0,97915	1,1743	0,96807
		2	Serviço 01	Serviço 02	Serviço 03
			1,0401	1,2387	0,99226
		3	Serviço 01	Serviço 02	Serviço 03
			0,95745	1,2260	1,0034
		4	Serviço 01	Serviço 02	Serviço 03
			0,98183	1,1865	0,95409
		5	Serviço 01	Serviço 02	Serviço 03
			1,0203	1,1938	0,96231
		6	Serviço 01	Serviço 02	Serviço 03
			1,0295	1,1961	1,0139

Tab. 4.20 Resultado para cenário 5 no modelo 3-camadas

- Cenário 6

Fatores				
Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,174	1	Serviço 01	Serviço 02	Serviço 03
		0,89616	1,0908	0,88483
	2	Serviço 01	Serviço 02	Serviço 03
		0,95563	1,1597	0,90692
	3	Serviço 01	Serviço 02	Serviço 03
		0,87453	1,1451	0,92022
	4	Serviço 01	Serviço 02	Serviço 03
		0,89895	1,1054	0,87139
	5	Serviço 01	Serviço 02	Serviço 03
		0,93752	1,1126	0,87935
	6	Serviço 01	Serviço 02	Serviço 03
		0,94622	1,1150	0,93027

Tab. 4.21 Resultado para cenário 6 no modelo 2-camadas

Fatores					
Tempo de processamento APPServer	Tempo de processamento BDServer	Replicação	Tempo médio de resposta (segundos)		
0,0174 (0,174 / 10)	0,174	1	Serviço 01	Serviço 02	Serviço 03
			0,92636	1,1190	0,91504
		2	Serviço 01	Serviço 02	Serviço 03
			0,98590	1,1860	0,93713
		3	Serviço 01	Serviço 02	Serviço 03
			0,90489	1,1733	0,95039
		4	Serviço 01	Serviço 02	Serviço 03
			0,92933	1,1335	0,90162
		5	Serviço 01	Serviço 02	Serviço 03
			0,96810	1,1409	0,90967
		6	Serviço 01	Serviço 02	Serviço 03
			0,97650	1,1434	0,96061

Tab. 4.22 Resultado para cenário 6 no modelo 3-camadas

Observa-se, comparando-se os cenários 1 e 2, que com a diminuição do tempo de processamento, diminuição de 0,087 segundo para 0,0435 segundo, do servidor de banco de

dados ouve uma significativa diminuição no tempo médio de resposta nas duas arquiteturas de sistemas.

4.4.4. Análise dos Resultados

Para cada cenário de simulação os resultados de tempos obtidos foram submetidos ao Statististica para comparação dos intervalos de confiança e obtenção dos tempos médios e desvios padrão como forma de comprovar, utilizando também resultados gráficos, que o tempo de processamento do servidor de banco de dados é o fator que tem efeito mais relevante na métrica de desempenho adotada para análise. A seguir, para cada cenário, são mostrados os resultados estatísticos obtidos submetendo as três variáveis, a seguir, ao software: A) Arquitetura com os valores 1 para arquitetura 2-camadas e 2 para a arquitetura 3-camadas; B) Serviços com os valores 1 para renovação, 2 para primeira habilitação e 3 para emissão da CNH definitiva; C) Tempo de resposta médio coletados nas simulações do ARENA. Para o auxílio da análise dos resultados foram gerados dois tipos de gráficos, tempo em função da arquitetura e serviço em função da arquitetura. Abaixo são mostradas as distribuições dos tempos médios de resposta para os resultados dos cenários de simulação.

- Cenário 1

Arquitetura	Tempo (segundos)		
	Média	Desvio Padrão	N
2-Camadas	0,455	0,047	18
3-Camadas	0,805	0,047	18
Total	0,630	0,183	36

Tab. 4.23 Descrição do tempo de resposta para cenário 1

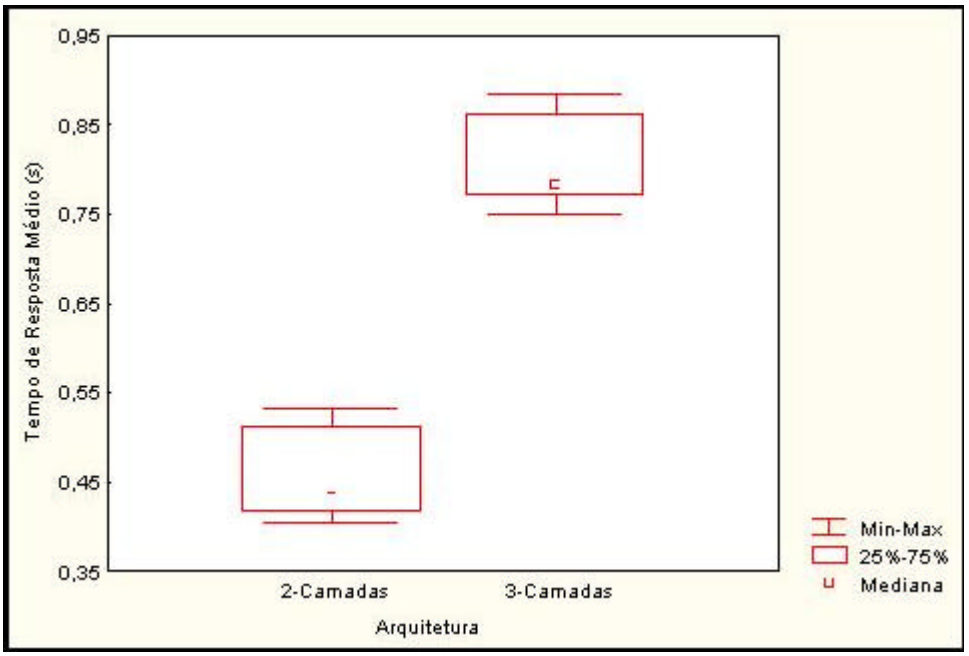


Fig. 4.8 Distribuição do tempo de resposta para cenário 1

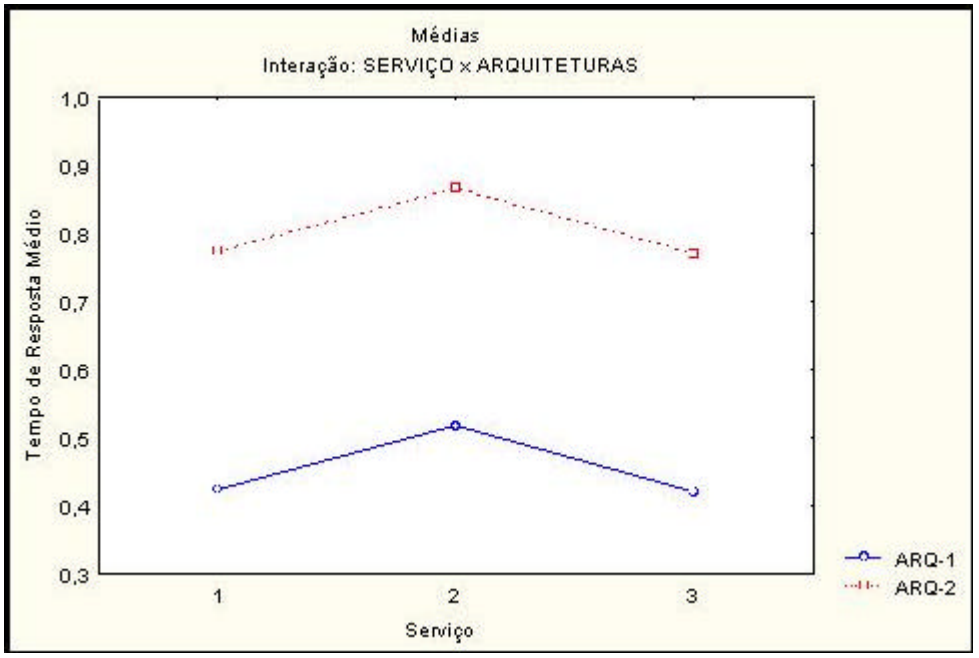


Fig. 4.9 Serviço x arquitetura para cenário 1

- Cenário 2

Arquitetura	Tempo (segundos)		
	Média	Desvio Padrão	N
2-Camadas	0,220	0,023	18
3-Camadas	0,600	0,021	18
Total	0,405	0,178	36

Tab. 4.24 Descrição do tempo de resposta para cenário 2

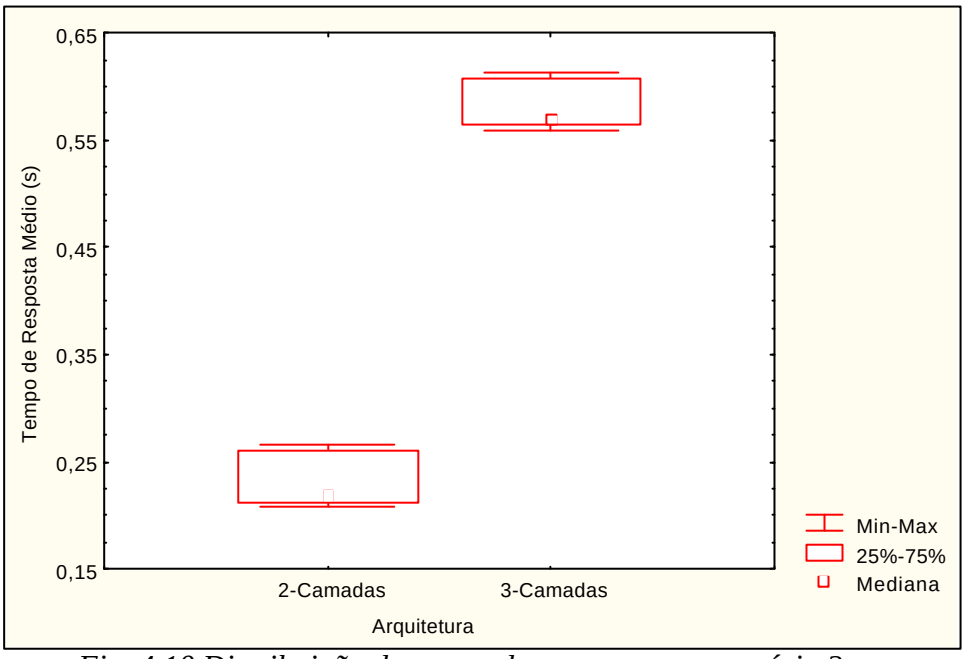


Fig. 4.10 Distribuição do tempo de resposta para cenário 2

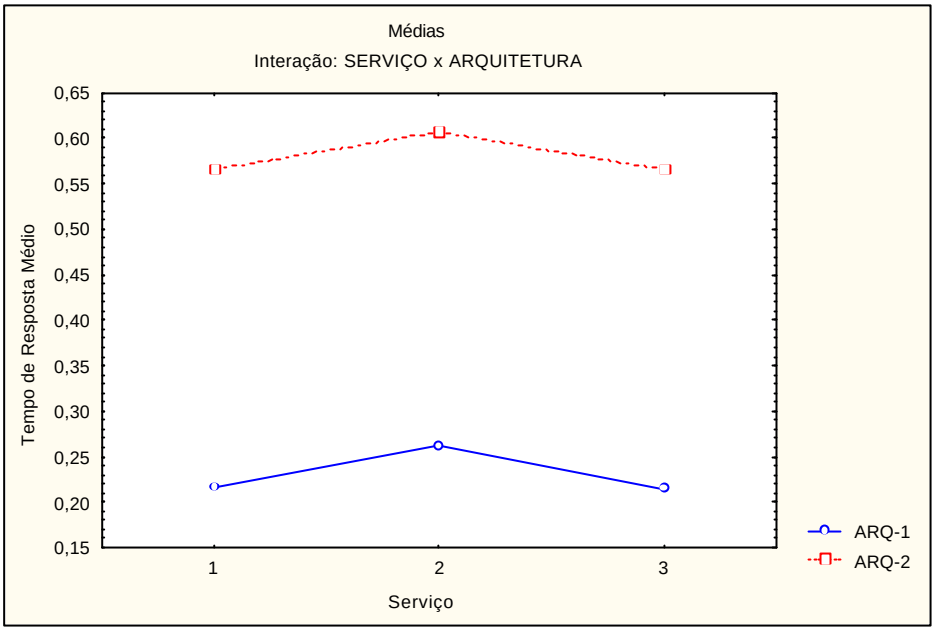


Fig. 4.11 Serviço x arquitetura para cenário 2

- Cenário 3

Arquitetura	Tempo (segundos)		
	Média	Desvio Padrão	n
2-Camadas	0,979	0,107	18
3-Camadas	1,329	0,106	18
Total	1,155	0,206	36

Tab. 4.25 Descrição do tempo de resposta para cenário 3

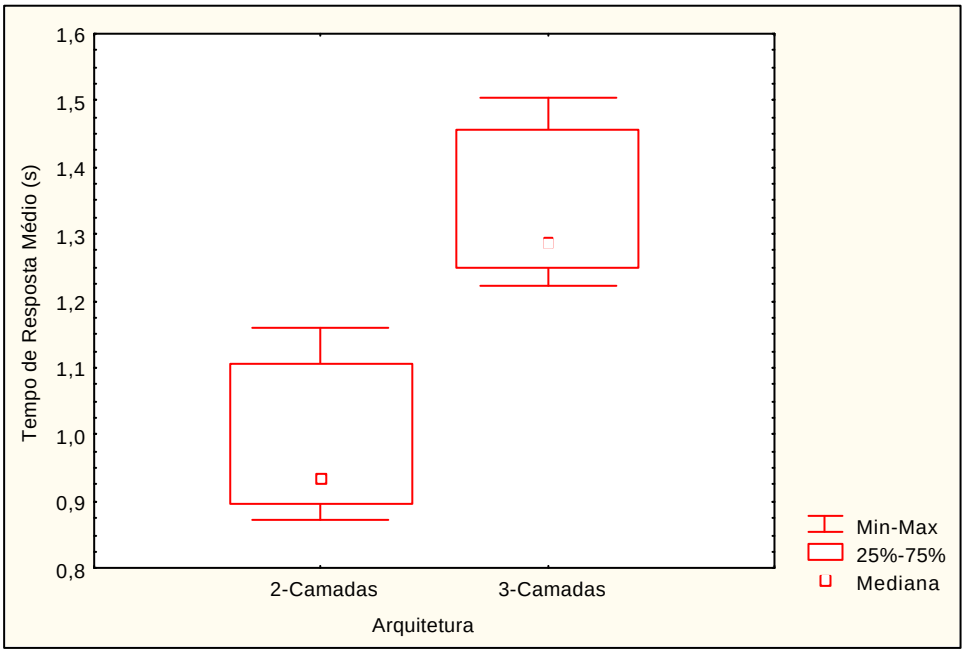


Fig. 4.12 Distribuição do tempo de resposta para cenário 3

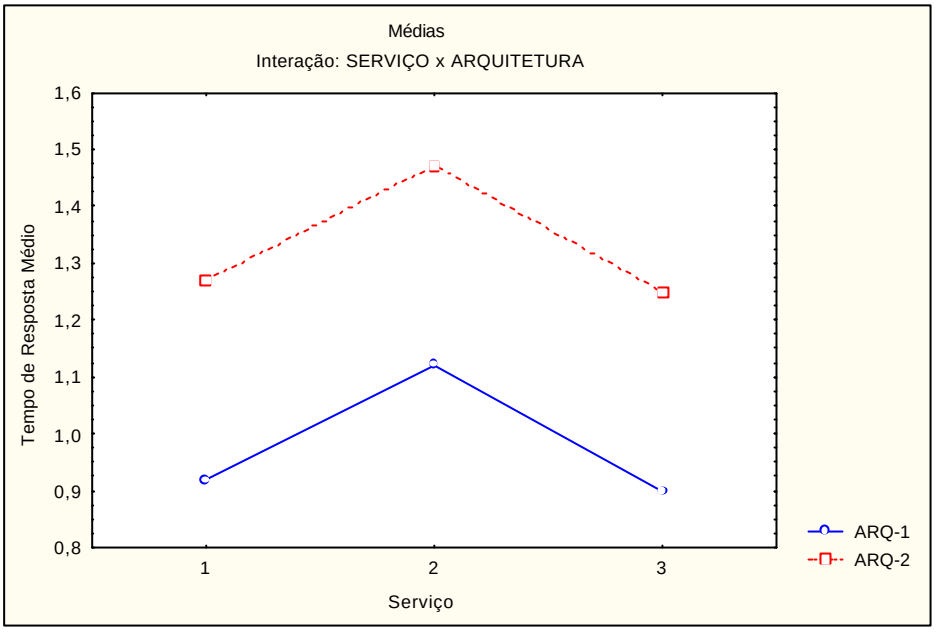


Fig. 4.13 Serviço x arquitetura para cenário 3

- Cenário 4

Arquitetura	Tempo (segundos)		
	Média	Desvio Padrão	N
2-Camadas	0,979	0,107	18
3-Camadas	1,151	0,105	18
Total	1,066	0,136	36

Tab. 4.26 Descrição do tempo de resposta para cenário 4

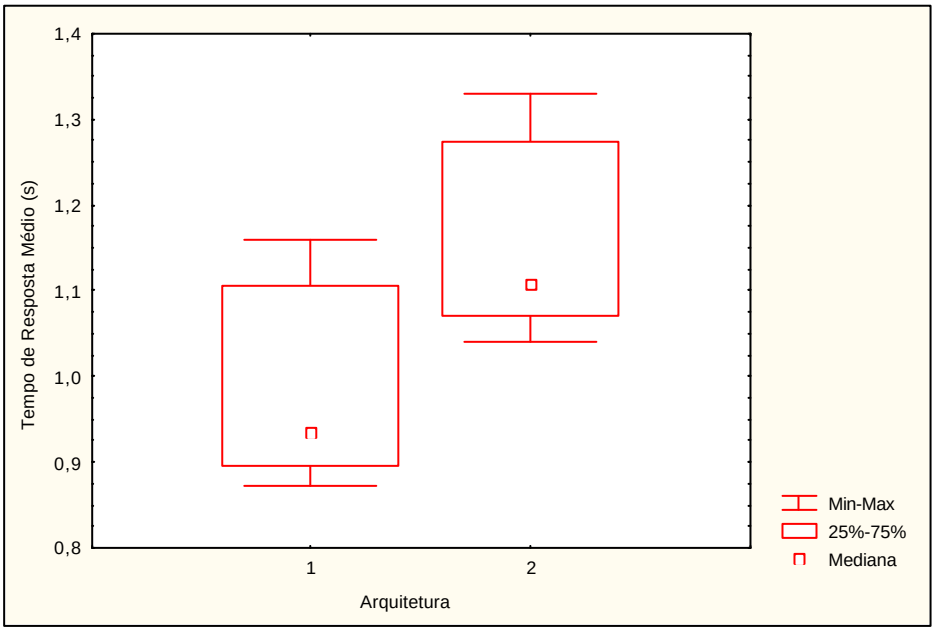


Fig. 4.14 Distribuição do tempo de resposta para cenário 4

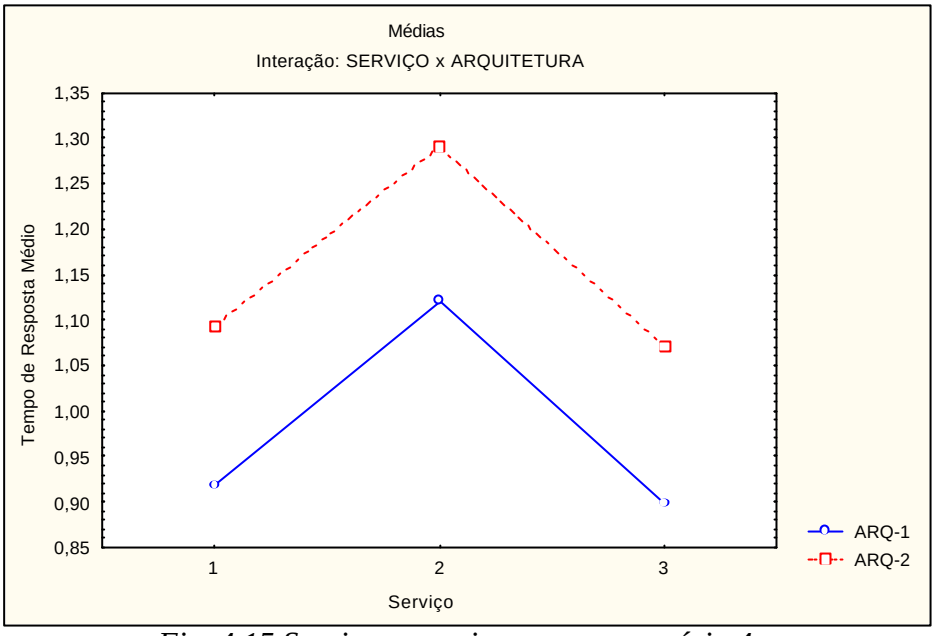


Fig. 4.15 Serviço x arquitetura para cenário 4

- Cenário 5

Arquitetura	Tempo (segundos)		
	Média	Desvio Padrão	N
2-Camadas	0,979	0,107	18
3-Camadas	1,062	0,106	18
Total	1,020	0,113	36

Tab. 4.27 Descrição do tempo de resposta para cenário 5

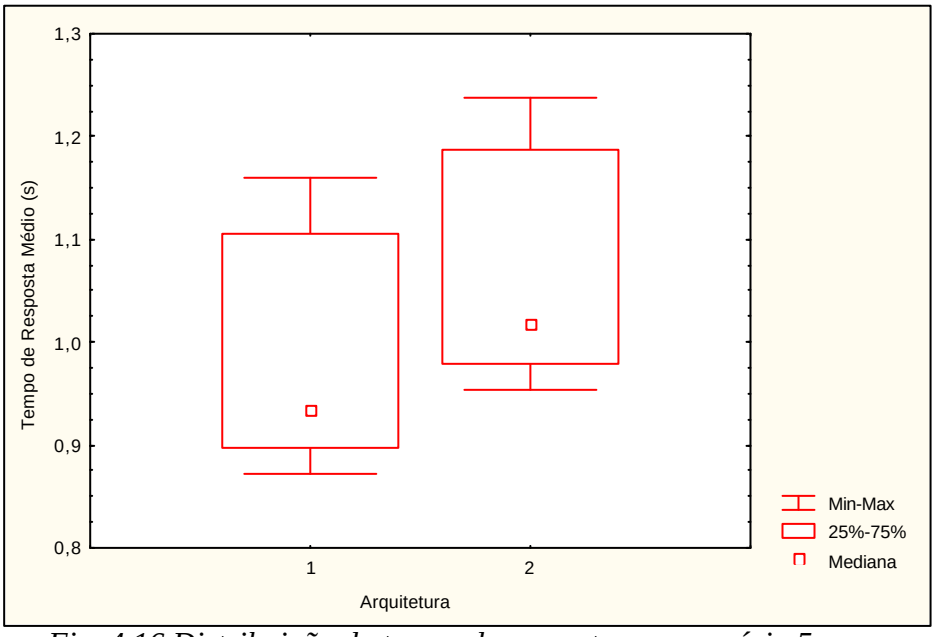


Fig. 4.16 Distribuição do tempo de resposta para cenário 5

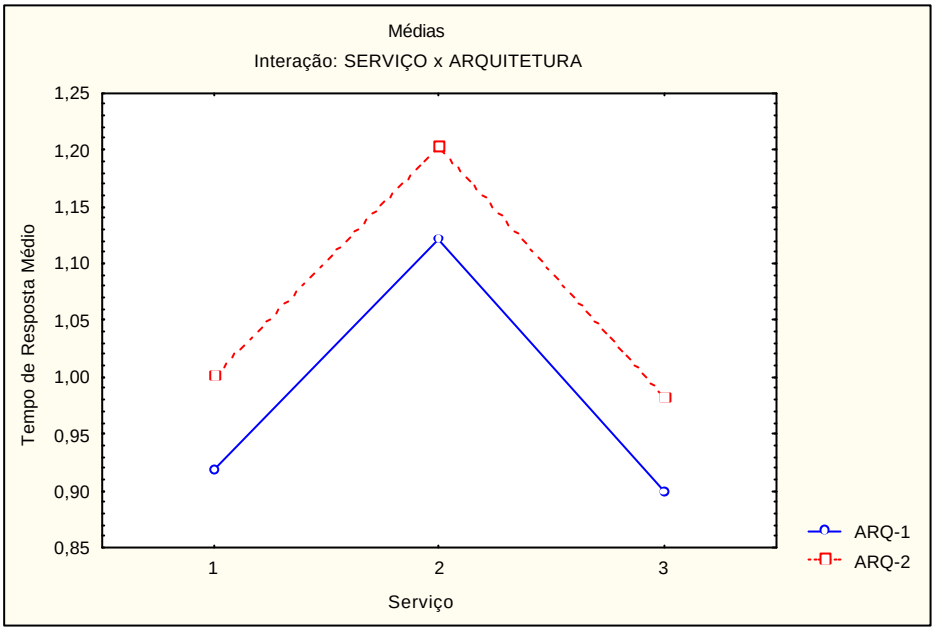


Fig. 4.17 Serviço x arquitetura para cenário 5

- Cenário 6

Arquitetura	Tempo (segundos)		
	Média	Desvio Padrão	N
2-Camadas	0,979	0,107	18
3-Camadas	1,01	0,106	18
Total	0,994	0,106	36

Tab. 4.28 Descrição do tempo de resposta para cenário 6

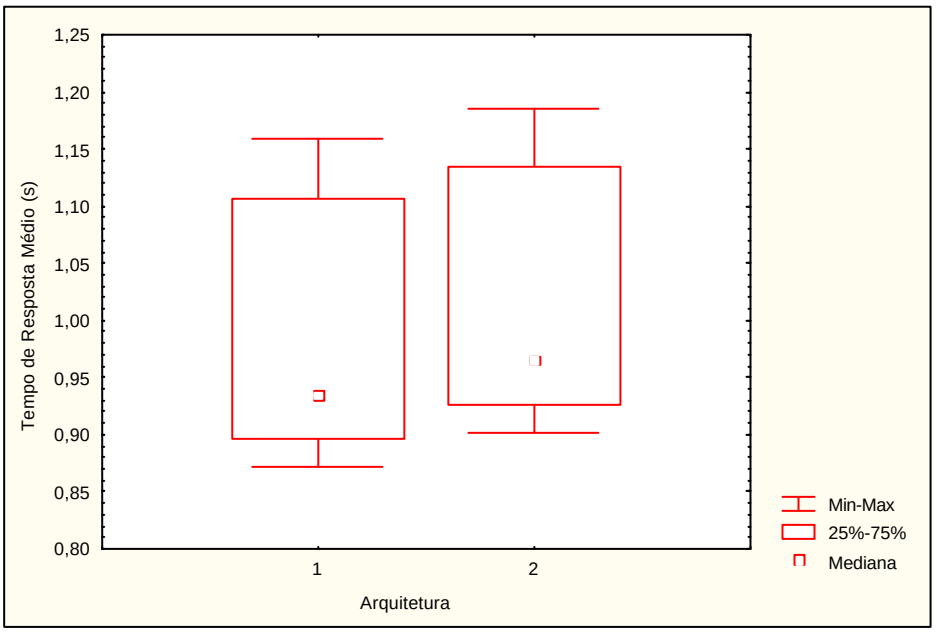


Fig. 4.18 Distribuição do tempo de resposta para cenário 6

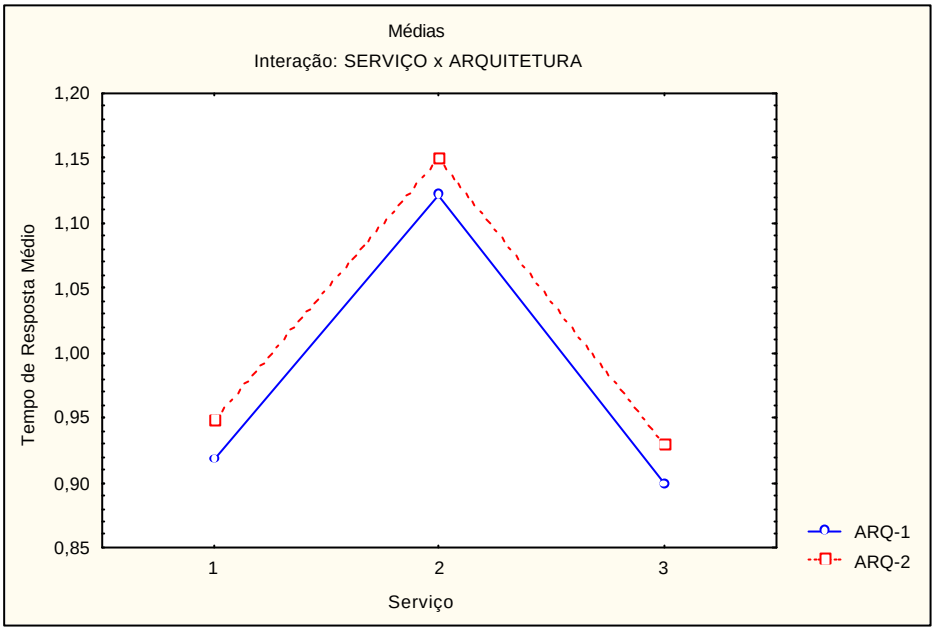


Fig. 4.19 Serviço x arquitetura para cenário 6

As figuras que mostram o tempo em função das arquiteturas permitem concluir que quanto maior o tempo de processamento do servidor de banco de dados, maior é a diferença entre as duas arquiteturas considerando a medida de desempenho adotada. Se o tempo de processamento do servidor do banco de dados for mantido constante e o tempo de processamento do servidor de aplicação for aumentado o suficiente, existirá um tempo médio de resposta aproximado para as duas arquiteturas, concluindo-se então que a escolha de uma ou da outra arquitetura seria indiferente.

A observação das distribuições dos serviços em função das arquiteturas mostra que os serviços possuem um padrão nas duas abordagens de sistemas, apenas são deslocados no tempo em função dos tempos de resposta diferentes. Assim ratificam a hipótese levantada de que com o aumento do tempo de processamento do servidor de aplicação, mantendo-se constante o tempo de processamento do servidor de banco de dados, a variável de desempenho analisada teria o mesmo comportamento para as duas arquiteturas.

A análise dos desvios padrão mostra que estatisticamente as duas arquiteturas são equivalentes, mostrando que as dispersões dos dados de ambas arquiteturas são similares. Porém se a premissa para adoção de uma das arquiteturas fosse apenas o tempo médio de resposta, o modelo 2-camadas seria adotado, pois apresentaram os melhores resultados para o tempo de resposta e do ponto de vista de análise da variável de desempenho existe a significância prática.

Este capítulo tratou da aplicação da metodologia apresentada no capítulo 3 e da análise dos resultados obtidos. O capítulo seguinte trata das conclusões finais e da sugestão de trabalhos futuros.

5. Conclusões Finais

A proposta desta pesquisa é avaliar o desempenho de dois projetos de sistemas, ambos, baseadas no modelo cliente/servidor. O primeiro modelo, chamado 2-camadas, é representado por uma quantidade de clientes (camada 1), uma interface de rede e um servidor de banco de dados (camada 2) que processa as requisições dos clientes. O segundo modelo, denominado 3-camadas, é composto pelos clientes (camada 1), uma interface de rede, um servidor de aplicações (camada 2) responsável por processar as solicitações da camada 1 e o servidor de banco de dados (camada 3) que processa as requisições do servidor de aplicação.

O ambiente do Departamento de Trânsito do Estado do Pará foi utilizado para este trabalho de pesquisa. O DETRAN-PA possui um sistema baseado no sistema legado e decidiu migrá-lo para a plataforma cliente/servidor devido, basicamente, ao alto custo do sistema e a dificuldade de manutenção de suas peças de software.

O primeiro sistema concebido foi o baseado no modelo 2-camadas que nos testes iniciais no Departamento de Informática do DETRAN-PA foi satisfatório, mas quando estendido para outras áreas do DETRAN acusou um alto tempo de resposta. Assim a alternativa 3-camadas passou a ser modelada, como uma forma de viabilizar o novo sistema de informação do Departamento de Trânsito do Estado do Pará.

Com alguns módulos do sistema 3-camadas desenvolvidos os testes iniciais foram realizados, porém o modelo 3-camadas apresentou um tempo de resposta alto. Considerando estas duas propostas de sistemas, foram desenvolvidos dois modelos de simulação com o objetivo de avaliar o desempenho de ambas através do tempo de resposta. Os tempos de processamento do servidor de banco de dados e do servidor de aplicação foram obtidos através de considerações extraídas dos testes de *benchmark* ECperf. De posse destes tempos vários cenários foram simulados e os dados coletados. Para a simulação no ARENA foram considerados dois fatores: o tempo de processamento do servidor de banco de dados e o tempo de processamento do servidor de aplicação.

Os resultados das simulações foram submetidos à análise estatística para determinação dos intervalos de confiança e dos desvios padrão. Com a análise dos dados concluí-se que estatisticamente os resultados das dispersões são equivalentes, uma vez que os desvios padrão mostraram uma pequena dispersão nos resultados obtidos nas simulações. Os dois tipos de gráficos de resultados apresentados foram construídos considerando três variáveis submetidas ao Statistica: Arquitetura, serviço e tempo médio de resposta. Avaliando a variável de

desempenho, tempo de resposta, as duas arquiteturas são diferentes, onde da análise dos gráficos do tempo de resposta em função da arquitetura observa-se que o fator tempo de processamento do servidor de banco de dados tem mais relevância para a determinação do modelo 2-camadas como a melhor alternativa, quando a grandeza de desempenho tempo de resposta é considerada. Os gráficos dos serviços em função da arquitetura mostram um comportamento semelhante para todos os serviços nas duas arquiteturas de sistemas, com a alternativa 3-camadas apresentando um maior deslocamento ao longo do tempo.

Os resultados indicam também que o modelo 2-camadas é mais adequado que o modelo 3-camadas em termos de tempo de resposta, considerando as simplificações dos modelos e os dois fatores apresentados. Apesar do escopo desta pesquisa ser a avaliação de desempenho de duas arquiteturas cliente/servidor, 2-camadas e 3-camadas, faz-se necessário ressaltar que no desenvolvimento de um sistema de informação, o tempo de resposta pode não ser o único fator determinante para o seu desenvolvimento, algumas características e funcionalidades são também necessárias para um sistema: manutenibilidade e fácil distribuição, aos vários clientes de uma rede corporativa; integração com outras plataformas, sistema legado ou sistemas baseados no modelo Internet, etc, desta forma, considerando as características citadas anteriormente, a abordagem 3-camadas adequasse de maneira muito mais favorável pois é fundamentada nos conceitos de orientação a objetos, objetos reutilizáveis e objetos processando as regras do negócio.

Porém alguns objetivos, como a definição dos domínios das aplicações 2-camas e 3-camadas, propostos neste trabalho, não foram atingidos em função das simplificações que viabilizaram a definição dos modelos de desempenho e do projeto de experimentos, simplificações estas realizadas em função da indisponibilidade de informações. Para tanto seria necessário adicionar outros fatores e grandezas de desempenho aos modelos, fatores como: número de clientes, a carga de trabalho, velocidade das CPU dos servidores, etc. E grandezas como: *throughput*, tempo e tamanho das filas nos servidores, etc, além da consideração da WAN, para avaliação das reais características distribuídas do sistema. Entretanto este trabalho serve de base para o desenvolvimento destas outras características elementares aos dois modelos cliente/servidor.

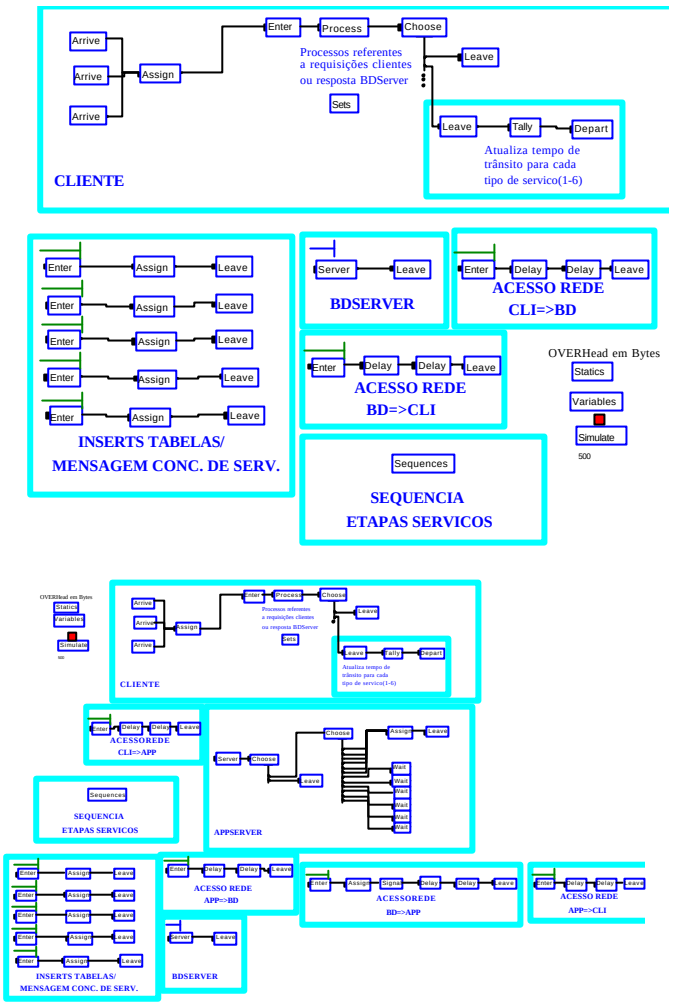
Vale ressaltar a importância da simulação como excelente técnica para a coleta de resultados e concepção de sistemas, uma vez que permitiu a criação de uma abordagem 3-camadas, para o sistema do Departamento de Trânsito do Estado do Pará, apesar da existência apenas de alguns módulos no mundo real, além de possibilitar a avaliação de desempenho de duas arquiteturas diferentes de sistemas.

5.1. Sugestões para Trabalhos Futuros

Uma das sugestões de continuidade desta pesquisa é a simulação considerando novos fatores e novas métricas de desempenho, adicionando ao modelo características de uma WAN para que juntamente com a análise de novos cenários utilizando outros servidores de aplicação, determine-se o domínio de aplicação das duas arquiteturas. Outra sugestão seria definir novos tempos de processamento para o servidor de banco de dados e servidor de aplicação utilizando um estudo de caso de um sistema 3-camadas ou analisar um estudo de caso onde exista a possibilidade de dois sistemas e comparar seus resultados com os obtidos neste trabalho. Um maior detalhamento do servidor de aplicação considerando novos processos como equilíbrio de carga, com o conseqüente aumento da carga de trabalho, seria também uma idéia interessante para continuidade deste trabalho de pesquisa.

Anexos

Anexo A – Modelos de Simulação no ARENA



Anexo B – Distribuição de Serviços do DETRAN-PA

	JAN	FEV	MAR	ABR	MAI	JUN
REN	5.534	3.491	4.338	3.354	4.038	3.683
PH	1.967	1.688	1.336	1.509	2.171	1.724
CNH.DEF.	1.001	829	1.245	1.071	1.165	1.144
MUD. CAT.	542	427	428	540	622	499
SEG VIA	542	324	438	382	452	431
AD. CAT.	106	39	44	71	27	41
REAB	3	1	0	0	0	1
TOTAL DO MÊS	9.695	6.799	7.829	6.927	8.475	7.523

JUL	AGO	SET	OUT	NOV	DEZ	TOTAL	MÉDIA
4.916	4.156	3.742	3.850	4.059	3.466	48.627	4.052
2.323	2.110	1.833	1.724	2.432	1.704	22.521	1.877
1.580	1.588	1.186	1.354	1.159	1.143	14.465	1.205
608	523	561	500	716	562	6.528	544
539	458	316	407	424	394	5.107	426
25	45	29	20	44	13	504	42
2	4	2	2	1	2	18	2
9.993	8.884	7.669	7.857	8.835	7.284	97.770	8.148

Anexo C – Full Disclosure Report: ECperf Benchmark for Oracle9iAS Release 2

FULL DISCLOSURE REPORT

ECperf Benchmark for Oracle9*i*AS Release 2
running on Sun Fire V480 and Oracle9*i*
Database Release 2 running on Sun Fire V880

June 24, 2002

Benchmark results are highly dependent upon a number of factors, including workload, specific application requirements, system hardware, system software, and system design and implementation. Results under other conditions may vary significantly.

THIS FULL DISCLOSURE REPORT INCLUDING ALL INFORMATION CONTAINED HEREIN (THIS “DOCUMENT”) IS PROVIDED ON AN "AS IS" BASIS. ORACLE EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. ORACLE MAKES NO WARRANTY THAT THIS DOCUMENT IS ERROR-FREE, ACCURATE OR RELIABLE. ORACLE MAKES NO WARRANTY ON SYSTEM PERFORMANCE, PRICE AND PRICE / PERFORMANCE. ORACLE RESERVES THE RIGHT TO MAKE CHANGES OR UPDATES AT ANY TIME WITHOUT NOTICE.

IN NO EVENT SHALL ORACLE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES, OR DAMAGES FOR LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY YOU OR ANY THIRD PARTY, WHETHER IN AN ACTION IN CONTRACT OR TORT, ARISING FROM YOUR ACCESS TO, OR USE OF, THIS DOCUMENT.

Some jurisdictions do not allow the limitation or exclusion of liability. Accordingly, some of the above limitations may not apply to you.

Sun Microsystems, Inc., believes that the information in this document is accurate as of its publication date. The information in this document is subject to change without notice. Sun Microsystems, Inc., assumes no responsibility for any errors that may appear in this document. The pricing information in this document is believed to accurately reflect prices in effect on June 4, 2002. However, Sun Microsystems provides no warranty on the pricing information in this document. The performance information in this document

is for guidance only. System performance is highly dependent on many factors including system hardware, system and user software, and user application characteristics. Customer applications must be carefully evaluated before estimating performance. Sun Microsystems, Inc., does not warrant or represent that a user can or will achieve a similar performance. No warranty on system performance or price / performance is expressed or implied in this document.

7.2 General Full Disclosure Requirements

7.2.3 Summary Statement

Please see section 7.3 for the Summary Statement

7.2.4 Sponsors

A statement identifying the benchmark sponsor(s) and other participating companies must be provided. The sponsor(s) must provide a list of third-party companies who permissions have been obtained for publication of this result.
Oracle Corporation and Sun Microsystems, Inc. sponsored and executed the benchmark.

7.2.5 Diagrams of both measured and priced configurations

Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:
- *Number and types of processors*
- *Size of allocated memory, and any specific mapping/partitioning of memory unique to the test*
- *Number and types of disk units (and controllers, if applicable)*
- *Number of LAN (e.g. Ethernet) connections, including routers, etc., that were physically used in the test*
- *Type and run-time execution location of software components (e.g. EJB Server/Containers, DBMS, client processes software load balancers, etc.)*

This section provides detailed information about the priced configuration.

The tested configuration is the same as the priced configuration except for the following:

- 6 x 36GB internal disks were used in the database server instead of the 6 x 73GB disks the system was priced with.
- 9 x 18GB disks and 256MB cache in the StorEdge T3 disk array were used with the database server instead of the 9 x 36GB disks and 1GB cache the system was

priced with.

• 2 x 18GB internal disks were used in one of the three J2EE servers instead of 2 x 36GB disks the system was priced with.

Priced Configuration

Application Servers Configuration:

Hardware

Sun Fire V480 (three systems)

4 X 900 MHz UltraSPARC-III Cu CPU

16GB RAM

Software

Solaris 8, 2/02

Oracle9*i*AS Release 2 Standard Edition v9.0.2.1.0

Oracle JDBC Driver v9.2.0.2.0 (thin)

Java2 Runtime Environment Standard Edition v1.4.0

Database Server Configuration:

Hardware

Sun Fire V880

8 X 900MHz UltraSPARC-III Cu CPU

16 GB RAM

Database Software

Solaris 8, 2/02

Oracle9*i* Database Release 2 Enterprise Edition with Partitioning Option v9.2.0.1.0,

Number and type of disk units:

Application Server

Internal: 2 x 36GB internal disks attached to internal FC-AL controller

Database Server

Internal: 6 x 73GB disks on a single FC-AL controller

External: StorEdge T3 disk array attached to a PCI FC-AL controller 1 logical unit (LUN)

LUN1: RAID 1 (stripe) of 8 x 36.2 GB disks

Network Connections

An 8 port 10/100Mbps switch (Ark Technologies Model #CT2208D3) was used to connect the client machine to the J2EE servers.

Number of LAN connections used: 7 point-to-point networks

100 Mbit Ethernet:

Driver/Emulator (E420R) QFE PCI 10/100 baseT card

10/100Mbs switch CT2208D3

100 Mbit Ethernet (x3)

10/100Mbs switch CT2208D3

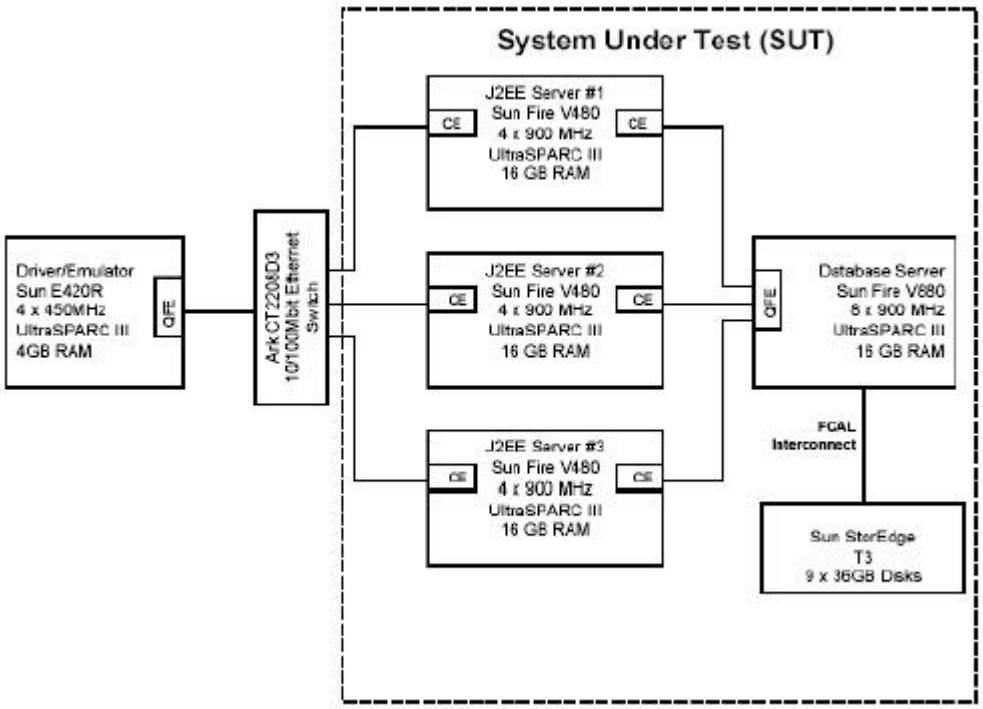
J2EE Server (SF V480) motherboard CE PCI 10/100/1000 baseT network interface

100 Mbit Ethernet (x3):

J2EE Server (SF V480) motherboard CE PCI 10/100/1000 baseT network interface

Database Server (SF V880) QFE PCI 10/100 baseT card

System Configuration Diagram:



7.3 Summary Statement

7.3.1 Summary Statement

The summary statement is a high-level view of the ECperf benchmark configuration and run results. An example of the Summary Statement is presented in Appendix B. The Summary Statement must include all of the information contained in this example in the same format for the benchmark being reported.

Application Server

Sun: Sun Fire V480

Oracle: Oracle9*i*AS Release 2 Standard Edition v9.0.2.1.0

Oracle: Oracle JDBC Driver v9.2.0.2.0 thin

Database Server

Sun: Sun Fire V880

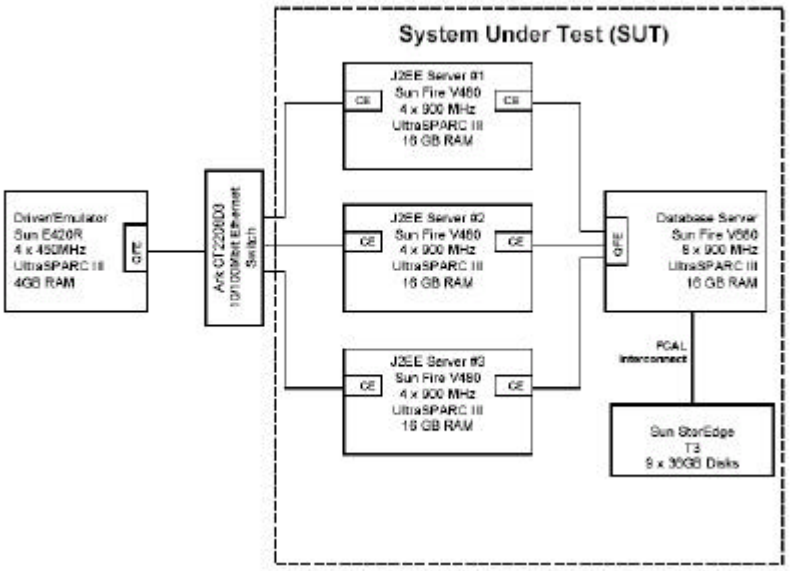
Oracle: Oracle9*i* Database Release 2 Enterprise Edition with Partitioning Option v9.2.0.1.0

Metrics: 26039.37 BBops/min@Std $16/BBops/min@Std

Availability Date: August 10, 2002

Bean Deployment Mode: CMP

Diagram of System Under Test (SUT)



System	Software	CPUs	Memory	Disk
Application Server Nodes 1, 2, 3 Sun Fire V480	Oracle9*i*AS Release 2 Standard Edition v9.0.2.1.0 Oracle JDBC Driver v9.2.0.2.0 (thin) Solaris 8, 2/02 Java 2 Runtime Environment Standard Edition v1.4.0	4 X 900 MHz UltraSPARC-III Cu	16GB	Internal: 2 x 36GB
Database Server Sun Fire V880	Oracle9*i* Database Release 2 Enterprise Edition with Partitioning Option v9.2.0.1.0 Solaris 2.8, 2/02	8 X 900 MHz UltraSPARC-III Cu	16GB	Internal: 6 x 73GB disks External: Sun StorEdge T3 TableTop 8 X 36GB disks

7.3.2 Driver Summary Report

The driver summary reports must appear as part of the Summary Statement. These include the Ecperf.summary, orders.summary and Mfg.summary files.

Contents of the ECperf.summary file

```
                    ECPerf Summary Report
                    Version : ECperf 1.0 Update 2


Run Parameters :
runOrderEntry = 1
runMfg = 1
txRate = 255
rampUp (in seconds) = 600
rampDown (in seconds) = 300
stdyState (in seconds) = 1800
triggerTime (in seconds) = 225
numOrdersAgents = 1, numMfgAgents = 1
dumpStats = 0
Benchmark Started At : Fri Jun 07 13:10:58 EDT 2002


Orders Summary report is in : Orders.summary
```

```
Orders Detailed report is in : Orders.detail
Orders Transaction Rate : 15014.23 Transactions/min

Manufacturing Summary report is in : Mfg.summary
Manufacturing Detail report is in : Mfg.detail
Manufaturing Rate : 11025.13 WorkOrders/min

ECperf Metric : 26039.37 BBops/min
```

Contents of the Orders.summary file

Orders Summary Report
Version : ECperf 1.0 Update 2

Orders Transaction Rate : 15014.23 Transactions/min

TRANSACTION MIX

Total number of transactions = 450427

TYPE	TX. COUNT	MIX	REQD. MIX.(5% Deviation Allowed)	
NewOrder:	224741	49.90%	50%	PASSED
ChangeOrder:	90548	20.10%	20%	PASSED
OrderStatus:	90153	20.02%	20%	PASSED
CustStatus:	44985	9.99%	10%	PASSED

ECPerf Requirement PASSED

RESPONSE TIMES	AVG.	MAX.	90TH%	REQD. 90TH%
NewOrder	0.805	19.414	2.000	2
ChgOrder	0.503	7.251	1.300	2
OrderStatus	0.240	6.072	0.700	2
CustStatus	0.294	5.781	0.800	2

ECPerf Requirement for 90% Response Time PASSED
ECPerf Requirement for Avg. Response Time PASSED

CYCLE TIMES	TARGETED AVG.	ACTUAL AVG.	MIN.	MAX.
NewOrder	4.964	5.136	0.000	25.000 PASSED
ChgOrder	4.962	5.038	0.000	25.000 PASSED
OrderStatus	4.976	4.995	0.000	25.000 PASSED
CustStatus	4.950	4.974	0.000	25.000 PASSED

MISC. STATISTICS

Average items per order	28.579	
Widget Ordering Rate	214099.233/min	PASSED
Percent orders that are Large Orders	10.05	PASSED
Average items per Large order	150.075	PASSED
Largeorder Widget Ordering Rate	112976.567/min	PASSED
Average items per Regular order	15.007	PASSED
Regular Widget Ordering Rate	101122.667/min	PASSED
Percent orders submitted from Cart	50.22	PASSED
Percent ChgOrders that were delete	10.19	PASSED

LITTLE'S LAW VERIFICATION

Number of users = 1275
Sum of Avg. RT * TPS for all Tx. Types = 1269.172019

Anexo D – Resultados de Simulação Colhidos no ARENA Dispostas por Cenário

```
ARENA Simulation Results
                        Silvia - License #9400000

                     Summary for Replication 1 of 6

Project:                                    Run execution date :  11/19/2002
Analyst:                                    Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          .41406     (Insuf)    .37603     .98503         90
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        810
tallyTempServ2          .50248     (Insuf)    .46329     1.1590         90
tallyTempServ3          .41611     (Insuf)    .37603     .93961         90
tallyTempServ4              --         --         --         --          0
BDServer_R_Q Queue Tim  .00902     .00525     .00000     .17400       1170
tallyTempServ5              --         --         --         --          0
tallyTempServ6              --         --         --         --          0

                           DISCRETE-CHANGE VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Final Value
-------------------------------------------------------------------------------

ProCliente_R Busy       .01296     3.9940E-04 .00000     3.0000     .00000
BDServer_R Available    1.0000     (Insuf)    1.0000     1.0000     1.0000
# in ProCliente_R_Q     .00000     (Insuf)    .00000     .00000     .00000
BDServer_R Busy         .20358     .00597     .00000     1.0000     .00000
# in BDServer_R_Q       .02111     (Insuf)    .00000     2.0000     .00000
ProCliente_R Available  19.000     (Insuf)    19.000     19.000     19.000

                                COUNTERS

                    Identifier                Count   Limit
                    ------------------------------------------

                    CountServSaida01             90  Infinite
                    CountServSaida02             90  Infinite
                    CountServSaida03             90  Infinite
                    CountServCheg01              90  Infinite
                    CountServCheg02              90  Infinite
                    CountServCheg03              90  Infinite


                        ARENA Simulation Results
                        Silvia - License #9400000

                     Summary for Replication 2 of 6

Project:                                    Run execution date :  11/19/2002
Analyst:                                    Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          .44565     (Insuf)    .37603     .98503         87
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        795
tallyTempServ2          .53280     (Insuf)    .46329     1.1590         89
tallyTempServ3          .41596     (Insuf)    .37603     .89803         89
tallyTempServ4              --         --         --         --          0
BDServer_R_Q Queue Tim  .01375     .00552     .00000     .17400       1149
tallyTempServ5              --         --         --         --          0
tallyTempServ6              --         --         --         --          0

                           DISCRETE-CHANGE VARIABLES
```

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01272	4.1802E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.19993	.00691	.00000	1.0000	.00000
# in BDServer_R_Q	.03159	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.40491	(Insuf)	.37603	.98503	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	809
tallyTempServ2	.52279	(Insuf)	.46329	1.1590	88
tallyTempServ3	.42611	(Insuf)	.37603	.89803	92
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.01065	(Corr)	.00000	.17400	1165
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01294	4.4368E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.20262	.00724	.00000	1.0000	1.0000
# in BDServer_R_Q	.02482	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	92	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.42405	(Insuf)	.37603	.98503	92
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	819
tallyTempServ2	.51212	(Insuf)	.46329	1.1590	91
tallyTempServ3	.41375	(Insuf)	.37603	.89803	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.01036	(Corr)	.00000	.17400	1183
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01310	4.6885E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.20584	.00731	.00000	1.0000	.00000
# in BDServer_R_Q	.02451	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	90	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.43105	(Insuf)	.37603	.98503	86
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	.52326	(Insuf)	.46329	1.1590	89
tallyTempServ3	.42083	(Insuf)	.37603	.89803	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.01227	(Corr)	.00000	.17400	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01272	3.8487E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.19993	.00633	.00000	1.0000	.00000
# in BDServer_R_Q	.02820	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite

CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.43353	(Insuf)	.37603	.98503	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ2	.51249	(Insuf)	.46329	1.1590	90
tallyTempServ3	.43445	(Insuf)	.37603	.89803	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.01271	.00447	.00000	.17400	1174
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01301	3.2862E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.20428	.00555	.00000	1.0000	.00000
# in BDServer_R_Q	.02984	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 4.97 minutes.
Simulation run complete.

```
attProcAPP=0.174
ProcBDServer = attProcAPP/2

                          ARENA Simulation Results
                          Silvia - License #9400000

                       Summary for Replication 1 of 6

Project:                                       Run execution date :  11/19/2002
Analyst:                                       Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          .77292     (Insuf)    .71753     1.5875         90
tallyTempServ2          .85183     (Insuf)    .80456     1.2395         90
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        810
tallyTempServ3          .76535     (Insuf)    .71753     1.4135         90
tallyTempServ4              --         --         --         --          0
tallyTempServ5              --         --         --         --          0
BDServer_R_Q Queue Tim  .00728     .00369     .00000     .17400       1170
tallyTempServ6              --         --         --         --          0
APPServer_R_Q Queue Ti  .00931     .00489     .00000     .34800        540

                           DISCRETE-CHANGE VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Final Value
-------------------------------------------------------------------------------

ProCliente_R Busy       .00810     2.3350E-04 .00000     3.0000     .00000
# in ProCliente_R_Q     .00000     (Insuf)    .00000     .00000     .00000
BDServer_R Available    1.0000     (Insuf)    1.0000     1.0000     1.0000
BDServer_R Busy         .20358     (Corr)     .00000     1.0000     .00000
APPServer_R Available   1.0000     (Insuf)    1.0000     1.0000     1.0000
# in BDServer_R_Q       .01704     (Insuf)    .00000     2.0000     .00000
# in APPServer_R_Q      .01005     (Insuf)    .00000     2.0000     .00000
APPServer_R Busy        .18792     .00502     .00000     1.0000     .00000
ProCliente_R Available  19.000     (Insuf)    19.000     19.000     19.000

                                  COUNTERS

                    Identifier                Count   Limit
                    ------------------------------------------

                    CountServSaida01             90  Infinite
                    CountServSaida02             90  Infinite
                    CountServSaida03             90  Infinite
                    CountServCheg01              90  Infinite
                    CountServCheg02              90  Infinite
                    CountServCheg03              90  Infinite


                          ARENA Simulation Results
                          Silvia - License #9400000

                       Summary for Replication 2 of 6

Project:                                       Run execution date :  11/19/2002
Analyst:                                       Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          .79430     (Insuf)    .71753     1.5875         87
tallyTempServ2          .88397     (Insuf)    .80456     1.3980         89
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        795
tallyTempServ3          .76981     (Insuf)    .71753     1.4135         89
tallyTempServ4              --         --         --         --          0
tallyTempServ5              --         --         --         --          0
BDServer_R_Q Queue Tim  .01045     .00425     .00000     .17400       1149
tallyTempServ6              --         --         --         --          0
APPServer_R_Q Queue Ti  .01207     (Corr)     .00000     .34800        530

                           DISCRETE-CHANGE VARIABLES
```

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.5113E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.19993	.00729	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.02401	(Insuf)	.00000	2.0000	.00000
# in APPServer_R_Q	.01279	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18444	(Corr)	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.75023	(Insuf)	.71753	1.5875	89
tallyTempServ2	.87610	(Insuf)	.80456	1.2991	88
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	808
tallyTempServ3	.77705	(Insuf)	.71753	1.4135	91
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.00699	.00320	.00000	.17400	1164
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01213	.00560	.00000	.34800	539

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00808	2.6097E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.20254	.00715	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.01626	(Insuf)	.00000	2.0000	.00000
# in APPServer_R_Q	.01308	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18723	.00590	.00000	1.0000	1.0000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	91	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.77174	(Insuf)	.71753	1.5875	92
tallyTempServ2	.86252	(Insuf)	.80456	1.2395	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ3	.75974	(Insuf)	.71753	1.4135	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.00786	(Corr)	.00000	.17400	1182
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00868	.00355	.00000	.34800	545

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00818	2.8887E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.20560	.00697	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.01857	(Insuf)	.00000	2.0000	.00000
# in APPServer_R_Q	.00947	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18966	.00667	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.77951	(Insuf)	.71753	1.5875	86
tallyTempServ2	.86454	(Insuf)	.80456	1.5293	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	.76610	(Insuf)	.71753	1.4135	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.00827	(Corr)	.00000	.17400	1149
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01045	(Corr)	.00000	.34800	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.2553E-04	.00000	3.0000	.00000

# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.19993	.00714	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.01900	(Insuf)	.00000	2.0000	.00000
# in APPServer_R_Q	.01108	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18444	.00524	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.78465	(Insuf)	.71753	1.5875	91
tallyTempServ2	.86385	(Insuf)	.80456	1.2395	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ3	.78826	(Insuf)	.71753	1.4135	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.00976	.00337	.00000	.17400	1174
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01173	.00391	.00000	.34800	542

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00813	1.8219E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.20428	.00560	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.02291	(Insuf)	.00000	2.0000	.00000
# in APPServer_R_Q	.01271	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18862	.00452	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 5.57 minutes.
Simulation run complete.

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 1 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.21114	(Insuf)	.20203	.50653	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	810
tallyTempServ2	.25467	(Insuf)	.24579	.59353	90
tallyTempServ3	.21141	(Insuf)	.20203	.46303	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.00210	(Corr)	.00000	.08700	1170
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01296	4.2308E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.10179	.00330	.00000	1.0000	.00000
# in BDServer_R_Q	.00492	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	90	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	90	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 2 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.22210	(Insuf)	.20203	.50653	87
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	.26500	(Insuf)	.24579	.59353	89
tallyTempServ3	.21022	(Insuf)	.20203	.46303	89
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.00364	.00196	.00000	.08700	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01272	4.2058E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000

```
BDServer_R Busy         .09996     .00336     .00000     1.0000     .00000
# in BDServer_R_Q       .00837     (Insuf)    .00000     2.0000     .00000
ProCliente_R Available  19.000     (Insuf)    19.000     19.000     19.000

                                  COUNTERS

                    Identifier                Count   Limit

                    CountServSaida01             87  Infinite
                    CountServSaida02             89  Infinite
                    CountServSaida03             89  Infinite
                    CountServCheg01              87  Infinite
                    CountServCheg02              89  Infinite
                    CountServCheg03              89  Infinite


                           ARENA Simulation Results
                           Silvia - License #9400000

                       Summary for Replication 3 of 6

Project:                                      Run execution date :  11/19/2002
Analyst:                                      Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                  TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations

tallyTempServ1          .20877     (Insuf)    .20203     .50653         89
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        809
tallyTempServ2          .26545     (Insuf)    .24579     .59353         88
tallyTempServ3          .21794     (Insuf)    .20203     .46303         92
tallyTempServ4             --         --         --         --          0
BDServer_R_Q Queue Tim  .00325     (Corr)     .00000     .08700       1166
tallyTempServ5             --         --         --         --          0
tallyTempServ6             --         --         --         --          0

                              DISCRETE-CHANGE VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Final Value

ProCliente_R Busy       .01294     4.7920E-04 .00000     3.0000     .00000
BDServer_R Available    1.0000     (Insuf)    1.0000     1.0000     1.0000
# in ProCliente_R_Q     .00000     (Insuf)    .00000     .00000     .00000
BDServer_R Busy         .10136     .00389     .00000     1.0000     1.0000
# in BDServer_R_Q       .00759     (Insuf)    .00000     2.0000     .00000
ProCliente_R Available  19.000     (Insuf)    19.000     19.000     19.000

                                  COUNTERS

                    Identifier                Count   Limit

                    CountServSaida01             89  Infinite
                    CountServSaida02             88  Infinite
                    CountServSaida03             92  Infinite
                    CountServCheg01              90  Infinite
                    CountServCheg02              88  Infinite
                    CountServCheg03              92  Infinite


                           ARENA Simulation Results
                           Silvia - License #9400000

                       Summary for Replication 4 of 6

Project:                                      Run execution date :  11/19/2002
Analyst:                                      Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                  TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations

tallyTempServ1          .21628     (Insuf)    .20203     .50653         92
```

ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	819
tallyTempServ2	.25952	(Insuf)	.24579	.59353	91
tallyTempServ3	.21234	(Insuf)	.20203	.46303	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.00295	(Corr)	.00000	.08700	1183
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01310	5.0588E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.10292	.00400	.00000	1.0000	.00000
# in BDServer_R_Q	.00698	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	90	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.21857	(Insuf)	.20203	.50653	86
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	.26514	(Insuf)	.24579	.59353	89
tallyTempServ3	.21860	(Insuf)	.20203	.46303	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.00403	(Corr)	.00000	.08700	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01272	3.7797E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.09996	.00301	.00000	1.0000	.00000
# in BDServer_R_Q	.00927	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.21790	(Insuf)	.20203	.50653	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ2	.25826	(Insuf)	.24579	.59353	90
tallyTempServ3	.21732	(Insuf)	.20203	.46303	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.00336	.00190	.00000	.08700	1174
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01301	3.3529E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.10214	.00284	.00000	1.0000	.00000
# in BDServer_R_Q	.00788	(Insuf)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 10.72 minutes.
Simulation run complete.

```
attProcAPP=0.174
ProcBDServer = attProcAPP/4
                          ARENA Simulation Results
                          Silvia - License #9400000

                       Summary for Replication 1 of 6

Project:                                       Run execution date :  11/19/2002
Analyst:                                       Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                  TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          .56036     (Insuf)    .54353     1.0630         90
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        810
tallyTempServ2          .60682     (Insuf)    .58706     .75528         90
tallyTempServ3          .56251     (Insuf)    .54353     .88900         90
tallyTempServ4              --         --         --         --          0
BDServer_R_Q Queue Tim  4.5483E-04 5.6010E-04 .00000     .04350       1170
tallyTempServ5              --         --         --         --          0
tallyTempServ6              --         --         --         --          0
APPServer_R_Q Queue Ti  .00828     .00406     .00000     .34800        540

                             DISCRETE-CHANGE VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Final Value
-------------------------------------------------------------------------------

ProCliente_R Busy       .00810     2.5307E-04 .00000     3.0000     .00000
BDServer_R Available    1.0000     (Insuf)    1.0000     1.0000    1.0000
# in ProCliente_R_Q     .00000     (Insuf)    .00000     .00000     .00000
BDServer_R Busy         .10179     (Corr)     .00000     1.0000     .00000
APPServer_R Available   1.0000     (Insuf)    1.0000     1.0000    1.0000
# in BDServer_R_Q       .00106     (Insuf)    .00000     1.0000     .00000
# in APPServer_R_Q      .00894     (Insuf)    .00000     2.0000     .00000
APPServer_R Busy        .18792     .00555     .00000     1.0000     .00000
ProCliente_R Available  19.000     (Insuf)    19.000     19.000    19.000

                                  COUNTERS

                    Identifier                Count   Limit
                    ------------------------------------------

                    CountServSaida01             90  Infinite
                    CountServSaida02             90  Infinite
                    CountServSaida03             90  Infinite
                    CountServCheg01              90  Infinite
                    CountServCheg02              90  Infinite
                    CountServCheg03              90  Infinite


                          ARENA Simulation Results
                          Silvia - License #9400000

                       Summary for Replication 2 of 6

Project:                                       Run execution date :  11/19/2002
Analyst:                                       Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                  TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          .56913     (Insuf)    .54353     1.0630         87
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        795
tallyTempServ2          .61225     (Insuf)    .58706     .87603         89
tallyTempServ3          .55891     (Insuf)    .54353     .88900         89
tallyTempServ4              --         --         --         --          0
BDServer_R_Q Queue Tim  6.4399E-04 4.2934E-04 .00000     .04350       1149
tallyTempServ5              --         --         --         --          0
tallyTempServ6              --         --         --         --          0
APPServer_R_Q Queue Ti  .00962     .00433     .00000     .34800        530

                             DISCRETE-CHANGE VARIABLES
```

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.5285E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.09996	.00365	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.00148	(Insuf)	.00000	1.0000	.00000
# in APPServer_R_Q	.01019	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18444	.00628	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.56437	(Insuf)	.54353	1.0630	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	809
tallyTempServ2	.61131	(Insuf)	.58706	.75878	88
tallyTempServ3	.56663	(Insuf)	.54353	.88900	92
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	4.4859E-04	(Corr)	.00000	.04350	1164
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01038	(Corr)	.00000	.34800	539

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00809	2.6590E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.10127	.00365	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.00104	(Insuf)	.00000	1.0000	.00000
# in APPServer_R_Q	.01118	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18732	.00626	.00000	1.0000	1.0000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	92	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.55851	(Insuf)	.54353	1.0630	92
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ2	.60838	(Insuf)	.58706	.75708	91
tallyTempServ3	.56654	(Insuf)	.54353	.88900	89
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	3.7926E-04	3.8994E-04	.00000	.04350	1183
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00901	(Corr)	.00000	.34800	546

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00818	2.9480E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.10292	.00361	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	8.9734E-04	(Insuf)	.00000	1.0000	.00000
# in APPServer_R_Q	.00984	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18977	.00659	.00000	1.0000	1.0000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.56962	(Insuf)	.54353	1.0630	86
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	.60794	(Insuf)	.58706	.83082	89
tallyTempServ3	.57276	(Insuf)	.54353	.88900	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	5.3018E-04	(Corr)	.00000	.04350	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01155	(Corr)	.00000	.34800	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.4039E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000

# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.09996	.00340	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.00122	(Insuf)	.00000	1.0000	.00000
# in APPServer_R_Q	.01225	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18444	.00568	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.57179	(Insuf)	.54353	1.0630	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ2	.59596	(Insuf)	.58706	.75178	90
tallyTempServ3	.56633	(Insuf)	.54353	.88900	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	5.7908E-04	4.2932E-04	.00000	.04350	1174
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00875	.00382	.00000	.34800	542

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00813	1.9897E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.10214	(Corr)	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.00136	(Insuf)	.00000	1.0000	.00000
# in APPServer_R_Q	.00949	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18862	.00487	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 13.28 minutes.
Simulation run complete.

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 1 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.89616	(Insuf)	.72403	1.9420	90
tallyTempServ2	1.0908	(Insuf)	.89829	2.2900	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	810
tallyTempServ3	.88483	(Insuf)	.72403	2.0706	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04042	.01455	.00000	.34800	1170
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01296	3.4267E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40716	.01004	.00000	1.0000	.00000
# in BDServer_R_Q	.09459	.03254	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	90	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	90	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 2 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.95563	(Insuf)	.72403	1.9420	87
tallyTempServ2	1.1597	(Insuf)	.89829	2.2900	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	.90692	(Insuf)	.72403	1.8762	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.05195	(Corr)	.00000	.34800	1149
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01272	3.5496E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000

BDServer_R Busy	.39985	.01264	.00000	1.0000	.00000
# in BDServer_R_Q	.11939	.04010	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.87453	(Insuf)	.72403	1.9420	89
tallyTempServ2	1.1451	(Insuf)	.89829	2.2900	88
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	808
tallyTempServ3	.92022	(Insuf)	.72403	1.7680	91
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04551	(Corr)	.00000	.34800	1164
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01293	4.0169E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40507	.01255	.00000	1.0000	1.0000
# in BDServer_R_Q	.10603	(Corr)	.00000	2.0000	1.0000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	91	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.89895	(Insuf)	.72403	1.9420	92

tallyTempServ2	1.1054	(Insuf)	.89829	2.2900	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ3	.87139	(Insuf)	.72403	1.7680	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04066	(Corr)	.00000	.34800	1182
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01309	4.4708E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.41109	.01355	.00000	1.0000	1.0000
# in BDServer_R_Q	.09611	.02843	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.93752	(Insuf)	.72403	2.0152	86
tallyTempServ2	1.1126	(Insuf)	.89829	2.2900	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	.87935	(Insuf)	.72403	1.9681	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04475	(Corr)	.00000	.34800	1149
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01272	3.3874E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.39985	.01302	.00000	1.0000	.00000
# in BDServer_R_Q	.10284	(Corr)	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.94622	(Insuf)	.72403	1.9420	91
tallyTempServ2	1.1150	(Insuf)	.89829	2.2900	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ3	.93027	(Insuf)	.72403	1.9007	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04965	.01245	.00000	.34800	1174
tallyTempServ6	--	--	--	--	0

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.01301	2.7589E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40855	.01061	.00000	1.0000	.00000
# in BDServer_R_Q	.11658	.03169	.00000	2.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 36.78 minutes.
Simulation run complete.

```
attProcAPP=0.174
ProcBDServer = attProcAPP
                          ARENA Simulation Results
                          Silvia - License #9400000

                       Summary for Replication 1 of 6

Project:                                       Run execution date :  11/19/2002
Analyst:                                       Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                  TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          1.2486     (Insuf)    1.0655     2.6315         90
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        810
tallyTempServ2          1.4492     (Insuf)    1.2395     2.4575         90
tallyTempServ3          1.2360     (Insuf)    1.0655     2.4118         90
tallyTempServ4              --         --         --         --          0
BDServer_R_Q Queue Tim  .03929     .01393     .00000     .34800       1170
tallyTempServ5              --         --         --         --          0
tallyTempServ6              --         --         --         --          0
APPServer_R_Q Queue Ti  .00875     .00335     .00000     .34800        540

                             DISCRETE-CHANGE VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Final Value
-------------------------------------------------------------------------------

ProCliente_R Busy       .00810     1.9953E-04 .00000     3.0000     .00000
BDServer_R Available    1.0000     (Insuf)    1.0000     1.0000    1.0000
# in ProCliente_R_Q     .00000     (Insuf)    .00000     .00000     .00000
BDServer_R Busy         .40716     .01011     .00000     1.0000     .00000
APPServer_R Available   1.0000     (Insuf)    1.0000     1.0000    1.0000
# in BDServer_R_Q       .09195     .03155     .00000     2.0000     .00000
# in APPServer_R_Q      .00945     (Insuf)    .00000     2.0000     .00000
APPServer_R Busy        .18792     .00433     .00000     1.0000     .00000
ProCliente_R Available  19.000     (Insuf)    19.000     19.000     19.000

                                  COUNTERS

                    Identifier                Count   Limit
                    ------------------------------------------

                    CountServSaida01             90  Infinite
                    CountServSaida02             90  Infinite
                    CountServSaida03             90  Infinite
                    CountServCheg01              90  Infinite
                    CountServCheg02              90  Infinite
                    CountServCheg03              90  Infinite


                          ARENA Simulation Results
                          Silvia - License #9400000

                       Summary for Replication 2 of 6

Project:                                       Run execution date :  11/19/2002
Analyst:                                       Model revision date:  11/19/2002

Replication ended at time      : 500.0

                                  TALLY VARIABLES

Identifier              Average   Half Width  Minimum    Maximum   Observations
-------------------------------------------------------------------------------

tallyTempServ1          1.2982     (Insuf)    1.0655     2.6315         87
ProCliente_R_Q Queue T  .00000     .00000     .00000     .00000        795
tallyTempServ2          1.5048     (Insuf)    1.2395     2.6160         89
tallyTempServ3          1.2580     (Insuf)    1.0655     2.2835         89
tallyTempServ4              --         --         --         --          0
BDServer_R_Q Queue Tim  .04846     (Corr)     .00000     .34800       1149
tallyTempServ5              --         --         --         --          0
tallyTempServ6              --         --         --         --          0
APPServer_R_Q Queue Ti  .01003     .00377     .00000     .34800        530

                             DISCRETE-CHANGE VARIABLES
```

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.2185E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.39985	.01210	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11136	.03752	.00000	2.0000	.00000
# in APPServer_R_Q	.01063	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18444	.00477	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.2274	(Insuf)	1.0655	2.6315	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	808
tallyTempServ2	1.4894	(Insuf)	1.2395	2.4575	88
tallyTempServ3	1.2718	(Insuf)	1.0655	2.2835	91
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04230	(Corr)	.00000	.34800	1164
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01102	.00438	.00000	.34800	538

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00808	2.5020E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.40473	.01255	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09848	(Corr)	.00000	2.0000	.00000
# in APPServer_R_Q	.01186	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18698	.00543	.00000	1.0000	1.0000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	91	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.2531	(Insuf)	1.0655	2.6315	92
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ2	1.4562	(Insuf)	1.2395	2.4575	91
tallyTempServ3	1.2223	(Insuf)	1.0655	2.2835	89
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.03845	(Corr)	.00000	.34800	1181
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01013	(Corr)	.00000	.34800	545

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00818	2.4454E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.41075	.01302	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09083	.02664	.00000	2.0000	.00000
# in APPServer_R_Q	.01104	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18966	.00494	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.2897	(Insuf)	1.0655	2.6315	86
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	1.4661	(Insuf)	1.2395	2.5733	89
tallyTempServ3	1.2222	(Insuf)	1.0655	2.3094	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04145	(Corr)	.00000	.34800	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.01119	.00405	.00000	.34800	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.0334E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000

# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.39985	.01322	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09525	(Corr)	.00000	2.0000	.00000
# in APPServer_R_Q	.01186	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18444	.00502	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/19/2002
Analyst: Model revision date: 11/19/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.2996	(Insuf)	1.0655	2.6315	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ2	1.4633	(Insuf)	1.2395	2.4575	90
tallyTempServ3	1.2822	(Insuf)	1.0655	2.2835	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04780	.01194	.00000	.34800	1174
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00892	.00347	.00000	.34800	542

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00813	1.8219E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.40855	.01074	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11223	.03051	.00000	2.0000	.00000
# in APPServer_R_Q	.00967	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.18862	.00416	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 23.93 minutes.
Simulation run complete.

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 1 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.0717	(Insuf)	.89153	2.2835	90
tallyTempServ2	1.2623	(Insuf)	1.0655	2.4575	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	810
tallyTempServ3	1.0577	(Insuf)	.89153	2.2378	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04045	.01457	.00000	.34800	1170
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00289	.00125	.00000	.17400	540

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00810	2.1618E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40716	.01004	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09465	.03251	.00000	2.0000	.00000
# in APPServer_R_Q	.00312	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.09396	.00236	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	90	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	90	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 2 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.1281	(Insuf)	.89153	2.2835	87
tallyTempServ2	1.3298	(Insuf)	1.0655	2.4575	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	1.0807	(Insuf)	.89153	2.1095	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.05143	(Corr)	.00000	.34800	1149
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00350	.00158	.00000	.17400	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value

ProCliente_R Busy	.00795	2.1369E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.39985	.01248	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11819	.03979	.00000	2.0000	.00000
# in APPServer_R_Q	.00371	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.09222	.00243	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.0497	(Insuf)	.89153	2.2835	89
tallyTempServ2	1.3153	(Insuf)	1.0655	2.4575	88
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	808
tallyTempServ3	1.0917	(Insuf)	.89153	2.1095	91
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04519	(Corr)	.00000	.34800	1164
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00312	.00153	.00000	.17400	538

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00808	2.5425E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40490	.01248	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.10520	(Corr)	.00000	2.0000	.00000
# in APPServer_R_Q	.00336	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.09354	.00280	.00000	1.0000	1.0000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	91	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.0738	(Insuf)	.89153	2.2835	92
tallyTempServ2	1.2742	(Insuf)	1.0655	2.4575	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ3	1.0410	(Insuf)	.89153	2.1095	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04014	(Corr)	.00000	.34800	1181
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00307	.00135	.00000	.17400	545

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00818	2.6051E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.41092	.01335	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09481	.02850	.00000	2.0000	.00000
# in APPServer_R_Q	.00334	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.09483	.00286	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.1140	(Insuf)	.89153	2.2835	86
tallyTempServ2	1.2809	(Insuf)	1.0655	2.4575	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	1.0521	(Insuf)	.89153	2.1354	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04418	(Corr)	.00000	.34800	1149
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00377	.00154	.00000	.17400	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.0755E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.39985	.01307	.00000	1.0000	.00000

APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.10153	(Corr)	.00000	2.0000	.00000
# in APPServer_R_Q	.00399	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.09222	.00250	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.1219	(Insuf)	.89153	2.2835	91
tallyTempServ2	1.2808	(Insuf)	1.0655	2.4575	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ3	1.1013	(Insuf)	.89153	2.1095	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04915	.01241	.00000	.34800	1174
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00282	.00147	.00000	.17400	542

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00813	1.7243E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40855	.01074	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11541	.03150	.00000	2.0000	.00000
# in APPServer_R_Q	.00306	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.09431	.00205	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 6.90 minutes.
Simulation run complete.

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 1 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.97915	(Insuf)	.80453	2.1965	90
tallyTempServ2	1.1743	(Insuf)	.97856	2.3705	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	810
tallyTempServ3	.96807	(Insuf)	.80453	2.1508	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04077	.01477	.00000	.34800	1170
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	6.5165E-04	(Corr)	.00000	.08700	540

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00810	2.1618E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40716	.01004	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09541	.03314	.00000	2.0000	.00000
# in APPServer_R_Q	7.0378E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.04698	.00118	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	90	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	90	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 2 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.0401	(Insuf)	.80453	2.1965	87
tallyTempServ2	1.2387	(Insuf)	.97856	2.3705	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	.99226	(Insuf)	.80453	2.0225	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.05204	(Corr)	.00000	.34800	1149
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	.00108	6.6740E-04	.00000	.08700	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value

ProCliente_R Busy	.00795	2.2185E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.39985	.01256	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11959	.04013	.00000	2.0000	.00000
# in APPServer_R_Q	.00114	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.04611	.00120	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.95745	(Insuf)	.80453	2.1965	89
tallyTempServ2	1.2260	(Insuf)	.97856	2.3705	88
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	808
tallyTempServ3	1.0034	(Insuf)	.80453	2.0225	91
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04551	(Corr)	.00000	.34800	1164
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	9.5230E-04	(Corr)	.00000	.08700	538

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00808	2.5425E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40499	.01253	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.10596	(Corr)	.00000	2.0000	1.0000
# in APPServer_R_Q	.00102	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.04681	.00139	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	91	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.98183	(Insuf)	.80453	2.1965	92
tallyTempServ2	1.1865	(Insuf)	.97856	2.3705	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ3	.95409	(Insuf)	.80453	2.0225	89
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04076	(Corr)	.00000	.34800	1182
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	6.7301E-04	(Corr)	.00000	.08700	545

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00818	2.7189E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.41101	.01351	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09637	.02873	.00000	2.0000	.00000
# in APPServer_R_Q	7.3358E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.04741	.00152	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.0203	(Insuf)	.80453	2.1965	86
tallyTempServ2	1.1938	(Insuf)	.97856	2.3705	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ3	.96231	(Insuf)	.80453	2.0484	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04477	(Corr)	.00000	.34800	1149
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	9.0353E-04	(Corr)	.00000	.08700	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.0962E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.39985	.01314	.00000	1.0000	.00000

APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.10288	(Corr)	.00000	2.0000	.00000
# in APPServer_R_Q	9.5774E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.04611	.00125	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	1.0295	(Insuf)	.80453	2.1965	91
tallyTempServ2	1.1961	(Insuf)	.97856	2.3705	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ3	1.0139	(Insuf)	.80453	2.0225	90
tallyTempServ4	--	--	--	--	0
tallyTempServ5	--	--	--	--	0
BDServer_R_Q Queue Tim	.04977	.01260	.00000	.34800	1174
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	8.8732E-04	(Corr)	.00000	.08700	542

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00813	1.7738E-04	.00000	3.0000	.00000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
BDServer_R Busy	.40855	.01068	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11685	.03193	.00000	2.0000	.00000
# in APPServer_R_Q	9.6186E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.04715	9.9707E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 8.33 minutes.
Simulation run complete.

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 1 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.92636	(Insuf)	.75233	2.1443	90
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	810
tallyTempServ2	1.1190	(Insuf)	.92636	2.3183	90
tallyTempServ3	.91504	(Insuf)	.75233	2.0986	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04067	.01482	.00000	.34800	1170
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	1.1236E-04	(Corr)	.00000	.03480	540

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00810	2.1618E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.40716	.01004	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09518	.03321	.00000	2.0000	.00000
# in APPServer_R_Q	1.2135E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.01879	4.7345E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	90	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	90	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 2 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.98590	(Insuf)	.75233	2.1443	87
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	1.1860	(Insuf)	.92636	2.3183	89
tallyTempServ3	.93713	(Insuf)	.75233	1.9703	89
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.05204	(Corr)	.00000	.34800	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	1.7958E-04	(Corr)	.00000	.03480	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value

ProCliente_R Busy	.00795	2.2185E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.39985	.01261	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11958	.04056	.00000	2.0000	.00000
# in APPServer_R_Q	1.9036E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.01844	4.7761E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	87	Infinite
CountServSaida02	89	Infinite
CountServSaida03	89	Infinite
CountServCheg01	87	Infinite
CountServCheg02	89	Infinite
CountServCheg03	89	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 3 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.90489	(Insuf)	.75233	2.1443	89
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	808
tallyTempServ2	1.1733	(Insuf)	.92636	2.3183	88
tallyTempServ3	.95039	(Insuf)	.75233	1.9703	91
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04574	(Corr)	.00000	.34800	1164
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	1.7744E-04	(Corr)	.00000	.03480	538

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00808	2.5425E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.40504	.01254	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.10654	(Corr)	.00000	2.0000	1.0000
# in APPServer_R_Q	1.9093E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.01872	5.4940E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	89	Infinite
CountServSaida02	88	Infinite
CountServSaida03	91	Infinite
CountServCheg01	90	Infinite
CountServCheg02	88	Infinite
CountServCheg03	92	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 4 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.92933	(Insuf)	.75233	2.1443	92
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	818
tallyTempServ2	1.1335	(Insuf)	.92636	2.3183	91
tallyTempServ3	.90162	(Insuf)	.75233	1.9703	89
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04089	(Corr)	.00000	.34800	1182
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	1.6935E-04	(Corr)	.00000	.03480	545

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00818	2.7974E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.41106	.01353	.00000	1.0000	1.0000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.09667	.02891	.00000	2.0000	.00000
# in APPServer_R_Q	1.8460E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.01897	6.2975E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	92	Infinite
CountServSaida02	91	Infinite
CountServSaida03	89	Infinite
CountServCheg01	92	Infinite
CountServCheg02	91	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 5 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.96810	(Insuf)	.75233	2.1443	86
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	795
tallyTempServ2	1.1409	(Insuf)	.92636	2.3183	89
tallyTempServ3	.90967	(Insuf)	.75233	1.9962	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04501	(Corr)	.00000	.34800	1149
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	1.7844E-04	(Corr)	.00000	.03480	530

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00795	2.1371E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.39985	.01306	.00000	1.0000	.00000

APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.10344	(Corr)	.00000	2.0000	.00000
# in APPServer_R_Q	1.8914E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.01844	5.1107E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	86	Infinite
CountServSaida02	89	Infinite
CountServSaida03	90	Infinite
CountServCheg01	86	Infinite
CountServCheg02	89	Infinite
CountServCheg03	90	Infinite

ARENA Simulation Results
Silvia - License #9400000

Summary for Replication 6 of 6

Project: Run execution date : 11/22/2002
Analyst: Model revision date: 11/22/2002

Replication ended at time : 500.0

TALLY VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Observations
tallyTempServ1	.97650	(Insuf)	.75233	2.1443	91
ProCliente_R_Q Queue T	.00000	.00000	.00000	.00000	813
tallyTempServ2	1.1434	(Insuf)	.92636	2.3183	90
tallyTempServ3	.96061	(Insuf)	.75233	1.9703	90
tallyTempServ4	--	--	--	--	0
BDServer_R_Q Queue Tim	.04989	.01267	.00000	.34800	1174
tallyTempServ5	--	--	--	--	0
tallyTempServ6	--	--	--	--	0
APPServer_R_Q Queue Ti	1.9641E-04	(Corr)	.00000	.03480	542

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half Width	Minimum	Maximum	Final Value
ProCliente_R Busy	.00813	1.7243E-04	.00000	3.0000	.00000
BDServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in ProCliente_R_Q	.00000	(Insuf)	.00000	.00000	.00000
BDServer_R Busy	.40855	.01063	.00000	1.0000	.00000
APPServer_R Available	1.0000	(Insuf)	1.0000	1.0000	1.0000
# in BDServer_R_Q	.11715	.03212	.00000	2.0000	.00000
# in APPServer_R_Q	2.1290E-04	(Insuf)	.00000	2.0000	.00000
APPServer_R Busy	.01886	3.9221E-04	.00000	1.0000	.00000
ProCliente_R Available	19.000	(Insuf)	19.000	19.000	19.000

COUNTERS

Identifier	Count	Limit
CountServSaida01	91	Infinite
CountServSaida02	90	Infinite
CountServSaida03	90	Infinite
CountServCheg01	91	Infinite
CountServCheg02	90	Infinite
CountServCheg03	90	Infinite

Simulation run time: 21.87 minutes.
Simulation run complete.

Referências Bibliográficas

ABDUL-FATAH, ISTABRAK; MAJUMDAR, SHIKHARESH. **Performance of CORBA-Based Client-Server Architectures.** In: IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 13, NO. 2, FEBRUARY 2002. p 2-9.

AHUJA, SANJAY P.; QUINTAO, RENATO. **Performance Evaluation of Java RMI: A Distributed Object Architecture for Internet Based Applications.** In: SYMPOSIUM ON MODELING, ANALYSIS AND SIMULATION OF COMPUTER AND TELECOMMUNICATION SYSTEMS, 8, 2000, California, United States. p. 1-5.

BANKS, Jerry; CARSON, John S; NELSON, Barry L. **Discrete-Event System Simulation.** 2. ed. New Jersey: Prentice-Hall International Series In Industrial And Systems Engineering. 1999. 548 p.

BARBETTA, Pedro Alberto. **Estatística Aplicada às Ciências Sociais.** 2 ed. Florianópolis: Editora da UFSC, 1998. 276 p.

BELLETTINI, C.; SERAZZI, G.; BORGHESE, P.. **Measurement of the Quality of Service of Web sites.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-7.

BOCHENSKI, Barbara. **Implementando Sistemas Cliente/Servidor de Qualidade.** 1 ed. São Paulo: Makron Books, 1995. 566 p.

CRAIN, PAT; HANSON, CRAIG. **Web Application Tuning.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 3-9.

CRABB, ROGER. **Evolution - Backend to Frontend - Part 2 Monitoring Distributed Systems Performance**. In: UNITED KINGDOM COMPUTER MEASUREMENT GROUP

2000 INTERNATIONAL CONFERENCE, 2000, United Kingdom Region, United Kingdom. p. 6-14.

DENATRAN. **Departamento Nacional de Trânsito - Código Nacional de Trânsito.** Capítulo II, parágrafo 19 itens VIII e IX. Última atualização: 21 de dezembro de 2001. Disponível em: http://www.denatran.gov.br/pg151_2.jsp. Acesso: nov/2001

DHILLON, PREET. **Preparing IT Infrastructure for ERP Implementation**. In: UNITED KINGDOM COMPUTER MEASUREMENT GROUP 2000 INTERNATIONAL CONFERENCE, 2000, United Kingdom Region, United Kingdom. p. 4-5.

EDWARDS, Jeri . **3-Tier Client/Server At Work.** 1. ed. New York: John Wiley & Sons, Inc. 1999. 336 p.

EDWARD, SIMON. **IBM Servers: Family Values**. In: UNITED KINGDOM COMPUTER MEASUREMENT GROUP 2000 INTERNATIONAL CONFERENCE, 2000, United Kingdom Region, United Kingdom. p. 18-23.

TINDALL, Paul. **Desenvolvendo Aplicações Corporativas com Visual Basic, MTS, IIS. SQL Server e XML.** 1. ed. São Paulo: Editora Campus, 2000. 416 p.

GIMARC, RICHARD L.; SPELLMANN, AMY. **Redefining Response Time in an Asynchronous World.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 2-4.

GOMES, RITA DE CÁSSIA CERQUEIRA. **Avaliação de Desempenho de Ambientes de Videoconferência: Um Estudo de Caso**. 2002. 110f.. Dissertação (Mestrado em Ciência da Computação) - Universidade de Santa Catarina, Florianópolis.

GREIS, WOLFRAM. **OS/390 und e-Business**. In: COMPUTER MEASUREMENT GROUP AUSTRALIA 2000 INTERNATIONAL CONFERENCE, 2000, Australia Region, Australia. p. 7-14.

HANSON, CRAIG D.;CRAIN, PAT. **Building or Buying a Scalable Software System - A Handbook**. In: COMPUTER MEASUREMENT GROUP 2000 INTERNATIONAL CONFERENCE, 26, 2000, Orlando, United States. p. 6-8.

HANSON, CRAIG; CRAIN, PAT; WIGGINTON, STEVE. **User And Computer Performance Optimization - A NewModel For Efficiency.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-5.

HELLERSTEIN, JOSEPH L.; JAYRAM, T. S.; SQUILLANTE, MARK S. **Analysis of Large-Scale Distributed Information Systems.** In: INTERNATIONAL SYMPOSIUM ON MODELING, ANALYSIS AND SIMULATION OF COMPUTER AND TELECOMMUNICATION SYSTEMS, 8, 2000, California, United States. p. 1-4.

HULL, CHRIS. **Client-Server Application Modeling - A Methodology.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-4.

JAIN, Raj. **The Art of Computer Systems Performance Analysis.** 1. ed. New York: John Wiley & Sons, Inc. 1991. 683 p.

JURIC, MATJAZ B.; ROZAMN, IVAN; HERICKO, MARJAN; STEVENS, ALAN P.; NASH, SIMON. **Java 2 Distributed Object Models Performance Analysis, Comparison and Optimization.** In: INTERNATIONAL CONFERENCE ON PARALLEL AND DISTRIBUTED SYSTEMS, 7, 2000, Iwate, Japan. p. 2-6.

KALM, DENISE P.. **End User Performance Tuning**. In: COMPUTER MEASUREMENT GROUP 2000 INTERNATIONAL CONFERENCE, 26, 2000, Orlando, United States. p. 3-6.

KELTON, W. David;SADOWSKI, Randall P.;SADOWSKI, Deborah A. **Simulation With Arena.** 1. ed. Local: McGraw Hill College Div, 1998. 632 p.

KEMPER, ALFONS; KOSSMANN, DONALD; ZELLER, BERNHARD. **Performance Tuning for SAP R/3.** In: DATA ENGINEERING BULLETIN, VOL. 22, NO. 2, FEBRUARY 1999. p 2-9.

KIM, KWANG-HOON; HAN, DONG-SOO. **Performance and Scalability Analysis on Client-Server Workflow Architecture.** In: INTERNATIONAL CONFERENCE ON PARALLEL AND DISTRIBUTED SYSTEMS, 8, 2001, Kyongju City, Korea. p. 1-4.

KON, Fabio;ROMÁN, Manuel;LIU, Ping;MAO, Jina;YAMANE, Tomonori;MAGALHÃES, Luiz Claudio;CAMPBELL, Roy H.. **Monitory, Security, And Dynamic Configuration With The dynamicTAO Reflective ORB**. Departament Of Computer Science University Illinois. Última atualização: 22 de novembro de 2001. Disponível em: http://choices.cs.uiuc.edu/2k/dynamictao. Acesso: Fev/2002

LEANDER, Rick. **Building Application Servers**. Cambridge: Cambridge University Press, 2000. 405 p.

LINDGREN, Lisa M. **Application Servers for E-Business**. 1. ed. Local: Auerbach Publications, 2001. 288 p.

LOOSLEY, Chris;DOUGLAS, Frank . **High-Performance Client/Server.** 1. ed. New York: John Wiley & Sons, Inc. 1998. 751 p.

LOU, Y; JOHN, I. K **Workload Characterization of Multithreaded Java Servers**. Technical Report TR-010815-01, Department of Electrical and Computer Engineering, University of Texas at Austin. Última atualização: 10 de dezembro de 2002. Dispinível em: http://www.ece.utexas.edu/projects/ece/lca. Acesso: Set/2002

MERTON, JOSEPH K.. **Evolution of Performance Testing in a Distributed Client Server Environment**. In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 2-5.

MENASCÉ, Daniel A.;ALMEIDA, Virgilio A. F. **Capacity Planning For Web Performance Metrics, Models, And Methods.** New Jersey: Prentice Hall PTR, 1998. 350 p.

MENASCÉ , DANIEL A. **A Method for Design and Performance Modeling of Client/Server Systems.** In: IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 26, NO. 12, DECEMBER 2000. p 1-9.

MENASCÉ, Daniel A.; ALMEIDA, Virgilio A.F. **Scaling For E-Business.** 1 ed. New Jersey: Prentice Hall, 2000. 449 p.

NICOLA, MATTHIAS; JARKE, MATTHIAS. **Performance Modeling of Distributed and Replicated Databases.** In: IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 12, NO. 4, JULY/AUGUST 2000. p 1-3.

NIXON, BRIAN A. **Management of Performance Requirements for Information Systems.** In: IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 26, NO. 12, DECEMBER 2000. p 1-7.

PENTAKALOS, ODYSSEAS. **Managing the Performance of Microsoft's Internet Information Server.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-2.

PRADO, Darci. **Usando o Arena em Simulação.** 1. ed. Local: EDG, 1999. 284 p.

SILVA, MARCOS VINICIUS DRISSEN. **Avaliação de Desempenho de uma Plataforma de Comutação Telefônica para Serviços Especializados de Atendimento ao Cliente aso**. 2002. 109f. Dissertação (Mestrado em Ciência da Computação) - Universidade de Santa Catarina, Florianópolis.

TENENBAUM, Andrews S. **Modern Operating Systems.** 2. ed. Local: Prentice Hall, 2001. 976 p.

TENENBAUM, Andrews S. **Redes de Computadores.** 1. ed. Local: Campus, 1997. 948 p.

SCHMIDT, TÂNIA. **Planejamento de Capacidade em Provedores de Serviços Internet**. 2002. 115f. Dissertação (Mestrado em Ciência da Computação) - Universidade de Santa Catarina, Florianópolis.

SCHUMACK, WAYNE. **A Hundred-Day Sprint: Going Live With Sap 4.0b On S/390 And Nt. Lessons Learned And Wisdom Gained.**. In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-2.

SEVITSKY, GARY; PAUW, WIM DE; KONURU, RAVI. **An Information Exploration Tool for Performance Analysis of Java Programs.** In: TECHNOLOGY OF OBJECT-ORIENTED LANGUAGES AND SYSTEMS, 2001, Zurich, Switzerland. p. 4-6.

SMARKUSKY, DEBRA L.; AMMAR, REDA A; HOWARD, A. SHOLL. **A Framework for Designing Performance-Oriented Distributed Systems.** In: IEEE SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS, 6, 2001, Hammamet, Tunisia. p. 1-5.

SMITH, CONNIE U.. **SPE Models for Multi-Tier Client/Server Interactions with MQSeries and Other Middleware.**. In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-4.

SOMIN, YEFIM; GROSS, LEONID. **Adventures in Modeling PeopleSoft**. In: COMPUTER MEASUREMENT GROUP 2000 INTERNATIONAL CONFERENCE, 26, 2000, Orlando, United States. p. 1-4.

SUN MICROSYSTEMS**. ECperf™ Specification**. Última atualização: 16 de abril de 2002. Disponível em: http://java.sun.com/j2ee/ecperf/download.html.
Acesso: jul/2002

THESERVERSIDE. **Full Disclosure Report Of Ecperf Benchmark For Oracle9*i*AS Release 2**. Última atualização: 24 de junho de 2002.
Disponível em:
http://www2.theserverside.com/ecperf/index.jsp?page=results/application_server.
Acesso: jul/2002

WATSON, LARRY E.. **Successful Multi-Tier Benchmarking And Load Testing.** In: COMPUTER MEASUREMENT GROUP 1999 INTERNATIONAL CONFERENCE, 25, 1999, Reno, United States. p. 1-6.