UNIVERSIDADE EDUARDO MONDLANE

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE MATEMÁTICA E INFORMÁTICA

Trabalho de Licenciatura

Migração de Sistemas com Recurso à Arquitectura Thin Client/Server

Estudo de caso: EMOSE

Arlindo Bernardo Paulo Nhabomba

Março de 2010

UNIVERSIDADE EDUARDO MONDLANE

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE MATEMÁTICA E INFORMÁTICA

Trabalho de Licenciatura

Migração de Sistemas com Recurso à Arquitectura Thin Client/Server

Estudo de caso: EMOSE

Autor: Arlindo Bernardo Paulo Nhabomba

Supervisor: Dr. José António Nhavoto

Co-Supervisor: dr. Flávio Pinto Matsolo

Data: Março de 2010

Declaração de honra

Declaro por minha honra, que este trabalho é fruto da minha investigação, e que o mesmo foi realizado para ser submetido apenas como Trabalho de Licenciatura em Informática na Universidade Eduardo Mondlane.

Maputo, Março de 2010

_____

(Arlindo Bernardo Paulo Nhabomba)

Agradecimentos

A Deus, o todo poderoso, pela capacidade intelectual e acompanhamento, principalmente nos momentos difíceis.

Aos meus pais pela educação, dedicação e carinho.

Meu muito obrigado aos meus supervisores, o Dr. José António Nhavoto e o dr. Flávio Pinto Matsolo, pela sua simplicidade, simpatia e sugestões demonstradas e ao Prof. Doutor José Leopoldo Nhampossa, pelo contributo que deu durante a realização deste trabalho.

Ao dr. Silvestre Sechene pela motivação e força que sempre me transmitiu durante a minha formação.

Aos meus colegas e amigos pelo apoio e encorajamento transmitidos ao longo do curso.

E a todos os professores e funcionários do DMI com os quais tive a oportunidade de lidar durante o curso,

Muito Obrigado.

Dedicatória

Aos meus pais **Bernardo Paulo Nhabomba** e **Laura Júlio Bié**.

Resumo

A maior parte das organizações baseadas em transacções e geograficamente distribuídas depende muito da utilização de sistemas de informação nas suas actividades. Acredita-se que por este caminho, todos os problemas da empresa estarão minimamente resolvidos e viriam também ganhos consideráveis de produtividade.

Tem-se constatado, entretanto, que após a implantação dos sistemas de informação e mesmo com o seu uso adequado, estes não conseguem produzir os resultados esperados. E, em muitos casos, os problemas que surgem estão associados a questões administrativas dos recursos informáticos, largura de banda para as transacções, segurança e desempenho.

Diante destes factos está a EMOSE, uma empresa pública e a mais antiga companhia de seguros em Moçambique, actualmente com uma arquitectura baseada em dois ambientes computacionais, *Unix* e *Windows*, utilizando clientes gordos (PCs normais) em todas as estações, o que nem sempre é necessário, tendo em conta as actividades do dia a dia.

A principal contribuição deste trabalho é a demonstração de que é possível conseguir os resultados esperados após a implantação de uma arquitectura de sistemas de informação adequada ao negócio da empresa e tem como objectivo principal conceber uma arquitectura de sistemas de informação para uma empresa que efectua transacções financeiras em tempo real e a escala nacional com maior segurança, baixos custos de administração e alto desempenho.

Para o alcance dos objectivos deste trabalho recorreu-se à revisão bibliográfica e a pesquisas na *Internet*, para além dos conhecimentos adquiridos ao longo do curso, e foram utilizadas entrevistas para recolher informação. Para a realização das actividades foi adaptado o modelo em cascata, que é usado para projectos de desenvolvimento de *software*.

Foram utilizados o material bibliográfico consultado e uma análise baseada em dados estatísticos para a escolha da melhor opção possível a implementar na arquitectura proposta (*thin client/server*).

Conteúdo

Lista de Figuras

Abreviaturas

API	Interface de Programação das Aplicações (*Application Programming Interface*)
BD	Base de Dados
BI	Interconexão de negócios (*Business Interconection*)
CPD	Centro de Processamento de Dados
DOM	Disco no Módulo (*Disk On Module*)
DOS	Sistema Operativo em Disco (*Disk Operating System*)
EMOSE	Empresa Moçambicana de Seguros
ERP	Planeamento de Recursos Empresariais (*Enterprise Resourse Planning*)
IBM	Máquinas para Negócios Internacionais (*International Business Machines*)
ICA	Arquitectura de Computação Independente (Independent Computing Architecture)
IDS	Servidor Dinâmico Informix (*Informix Dynamic Server*)
NT	Nova Tecnologia (*New Technology*)
OLAP	Processamento Analítico em tempo Real (*Online Analytical Processing*)
OLTP	Processamento de transacções em tempo Real (*OnLine Transaction processing*)
OS	Sistema Operativo (*Operating System*)
PC	Computador Pessoal (*Personal Computer*)
RAM	Memória de Acesso Aleatório (*Random Access Memory*)
SAN	Área de Armazenamento em Rede (*Storage Area Network*)
SGBD	Sistema de Gestão de Bases de Dados
SIGES	Sistema Integrado de Gestão Empresarial de Seguros
SSH	Linha de Comando Segura (*Secure Shell*)
TI	Tecnologias de Informação
TIC	Tecnologias de Informação e Comunicação
UEM	Universidade Eduardo Mondlane

Capítulo

1 Introdução

1.1 Enquadramento

Nas últimas décadas, a globalização e a evolução tecnológica têm contribuído, em grande escala, na introdução de novos serviços sobre demanda, e desta forma aumentando a dependência destas no auxílio prestado pelas Tecnologias de Informação e Comunicação (TICs) visando sobretudo minimizar os tempos de resposta, erros ou omissões no processamento e transmissão de dados.

Essa necessidade de fazer algo diferente e melhorar o mais depressa possível o funcionamento das organizações, aumenta a competitividade das mesmas, e as TICs assumem um papel fundamental para as organizações que pretendam sobreviver nesta sociedade global, onde há elevadas exigências no acesso à informação com qualidade e com mínimos tempos de espera.

Essas exigências poderão ser solucionadas através da definição de arquitecturas informáticas alinhadas aos processos de negócio, o que exige uma descrição rigorosa do modo como a organização funciona, para que os sistemas possam satisfazer plenamente as suas actividades. Após esta avaliação fica mais claro como a informatização será feita e que tipo de tecnologias e arquitecturas se adequam à organização.

Organizações orientadas a transacções financeiras e geograficamente distribuídas neces-

sitam de sistemas que possam proporcionar soluções eficazes e inteligentes que ofereçam condições favoráveis para uma redução considerável de tempos de resposta. Entretanto, a preocupação das organizações não se deve prender somente na performance das aplicações, mas também na redução de custos de *hardware* e de gestão associada.

Tendo em conta a evolução tecnológica visando dar resposta a novos requisitos de negócio nas organizações, verifica-se a evolução das arquitecturas de sistemas de informação, desde a arquitectura baseada em *mainframes* e a arquitectura cliente/servidor, visando maior funcionalidade na prestação de serviços de administração de recursos informáticos.

É nesta perspectiva que se propõe neste trabalho a implementação da arquitectura *thin client/server* para empresas baseadas em transacções e geograficamente distribuídas, como o caso da EMOSE.

1.2 Breve historial da EMOSE

A Empresa Moçambicana de Seguros (EMOSE) nasceu em 1977 da fusão das companhias de seguros sedeadas em Moçambique nomeadamente a Lusitana, Tranquilidade de Moçambique e a Nauticus, assim, tornando-se na maior e única companhia de seguros (EMOSE 2007).

De seguida foram adoptadas estratégias de formação profissional interna e externa para assegurar o funcionamento da empresa e garantir que fossem alcançados os objectivos para os quais ela fora criada (EMOSE 2007).

O mercado dos seguros se abriu dando origem a novas companhias de seguros o que trouxe novos desafios à EMOSE, nomeadamente na melhoria dos seus sistemas de informação e dos processos de negócio de seguros visando servir da melhor maneira os seus clientes e a atracção de potenciais clientes. Actualmente possui representações em todas as capitais provinciais e em todas as fronteiras internacionais.

1.3 Definição do problema

Actualmente, a nível mundial, as empresas com um número grande de utilizadores e sobretudo geograficamente dispersos, como é o caso da EMOSE, se debatem com problemas de custos elevados de aquisição, licenciamento, administração e manutenção dos recursos informáticos, aliado ao facto de o tempo de vida útil de equipamento variar entre 3 a 5 anos, o que obriga as empresas a reinvestir em curto espaço de tempo, disperdiçando assim recursos financeiros que são geralmente escassos.

A situação acima referida agrava-se mais em arquitecturas cliente/servidor, como é o caso da arquitectura usada pela empresa EMOSE, em virtude de debater-se com problemas de insuficiência de largura de banda para correr todas as suas transacções em simultâneo, além de que quando há um problema na rede, por exemplo, causado por vírus, é de difícil localização, pois cada computador conectado à rede está exposto a ataques contra a sua segurança.

Assim, as equipas de TI (Tecnologias de Informação)são obrigadas a intervir não só nos *datacenters*, mas também em todas as estações de trabalho com muita frequência. Isso faz com que se aumente o número de deslocações e de intervenientes e, por consequência, muito esforço e custos elevados.

1.4 Objectivos

1.4.1 Objectivo geral

O objectivo geral do presente trabalho de licenciatura é conceber um modelo de arquitectura de sistemas de informação para uma empresa que efectua transacções financeiras em tempo real e a escala nacional, com baixos custos de administração, maior segurança e alto desempenho.

1.4.2 Objectivos específicos

São objectivos específicos os seguintes:

- Estudar o modelo actual de arquitectura do sistema de informação da EMOSE para identificar os constrangimentos característicos;
- Estudar a literatura de referência sobre sistemas legados, sistemas baseados em transacções e arquitecturas de sistemas de informação para identificar e compreender os conceitos chave;
- Fazer um estudo comparativo das arquitecturas *thin client/server*, centralizada, descentralizada e cliente/servidor.
- Desenhar o modelo de conversão da arquitectura actual do sistema de informação para uma arquitectura mais adequada ao negócio da empresa;

1.5 Conteúdo da tese

O presente trabalho está dividido em 5 capítulos. O primeiro apresenta a introdução, onde são destacados o enquadramento do trabalho, o breve historial da EMOSE, os objectivos do trabalho e é definido o problema. No segundo capítulo é apresentada a metodologia utilizada na realização deste trabalho.

No terceiro capítulo é feita a revisão de literatura sobre vários conceitos incluindo os de sistemas legados, sistemas baseados em transacções e arquitecturas de sistemas de informação e no quarto são apresentadas as descrições das arquitecturas actual do sistema em estudo e a que apresenta as soluções viáveis propostas pelo autor.

Finalmente, no quinto capítulo, são apresentadas as conclusões, recomendações e limitações do trabalho.

Capítulo

2 Metodologia

2.1 Introdução

A metodologia tem um papel muito importante na vida do homem pois permite a eficácia na solução dos problemas que o homem encontra na vida, fornecendo uma variedade de métodos, ferramentas, procedimentos e paradigmas para os resolver. Cada método não é melhor do que outro, cada um tem suas vantagens e desvantagens. Podem haver situações em que um é mais adequado do que o outro, e vice versa.

Na realização deste trabalho, por exemplo, foram utilizados métodos e materiais que acreditou-se que poderiam ser eficientes e produtivos para gerar uma solução eficaz no projecto.

Assim, neste capítulo, serão apresentados os métodos e materiais usados para a consecução dos objectivos traçados, com destaque para a revisão bibliográfica e as entrevistas efectuadas, e o processo de desenvolvimento do projecto, com destaque para o modelo em cascata.

2.2 Métodos e material usados

Para alcançar o segundo e terceiro objectivos específicos, o autor recorreu à revisão bibliográfica, utilizando os livros disponíveis em algumas bibliotecas espalhadas pela cidade de Maputo, incluindo a biblioteca central Brazão Mazula da UEM, e pesquisas e leituras de artigos e apostilas na *Internet*, para além dos conhecimentos adquiridos ao longo do curso.

Com vista a atingir o primeiro e o quarto objectivos específicos, foram efectuadas entrevistas ao pessoal sénior da EMOSE (ver anexo III) a fim de compreender o funcionamento do seu sistema de informação e da sua arquitectura actual. Também foram efectuadas outras entrevistas visando colher dados sobre os componentes da arquitectura actual e sua interligação, a colaboração de cada órgão do CPD na gestão do sistema, suas dificuldades e perspectivas de solução para minimizar essas dificuldades.

As entrevistas mencionadas foram:

Não estruturada: por ser objectiva e dar uma visão geral do problema pesquisado, é quase uma conversa. Foi a primeira forma de entrevista usada.

Entrevista por pautas: esta explora, no decorrer da entrevista, os pontos ou pautas, que são ordenados e devem ter uma certa relação entre si. Foram feitas poucas perguntas directas, tendo se deixado o entrevistado falar livremente enquanto se referia às pautas assinaladas. Foi necessário intervir quando o assunto fugisse ao da pauta.

2.3 Processo de desenvolvimento do projecto

Segundo (Pfleeger 2004) um processo é uma série de etapas que envolvem actividades, restrições e recursos para alcançar a saída desejada.

Os processos dão consistência, estrutura e orientação a um conjunto de actividades, permitindo examinar, entender e controlar o seu desenvolvimento. Os processos são aplicados em diversas áreas de desenvolvimento de projectos e são fundamentais para se obter produtos de qualidade e cumprir correctamente as metas dos projectos.

A espectativa dos desenvolvedores de qualquer projecto é encontrar um modelo que melhore a produtividade e qualidade. Mas cada projecto pode ser adequado para um determinado modelo.

Para o desenvolvimento de projectos de *software* destacam-se os seguintes modelos: em cascata, evolutivo, transformação e em espiral.

Para o desenvolvimento do presente trabalho foi utilizado o modelo em cascata, o qual foi concebido para o desenvolvimento de projectos de *software*. Assim, o modelo em cascata foi adaptado para o desenvolvimento do presente trabalho por ser simples para disciplinar e sistematizar o processo e, segundo Pfleeger (2004), apresenta uma visão de muito alto nível do que acontece durante o desenvolvimnento e sugere aos desenvolvedores a sequência de eventos que eles podem encontrar.

2.3.1 O modelo em cascata

Segundo Pfleeger (2004) o modelo em cascata é aquele em que os estágios são apresentados em sequência, como em uma cascata. O desenvolvimento de um estágio deve terminar antes do próximo começar. Desse modo, só quando todos os requisitos forem enunciados pelo cliente, tiverem sido cumpridos e documentados num documento de especificação, é que a equipe de desenvolvimento pode realizar as suas actividades.

As fases deste modelo destacadas, são as seguintes: estudo de viabilidade, identificação de requisitos, análise detalhada, desenho, implantação e testes e manutenção.

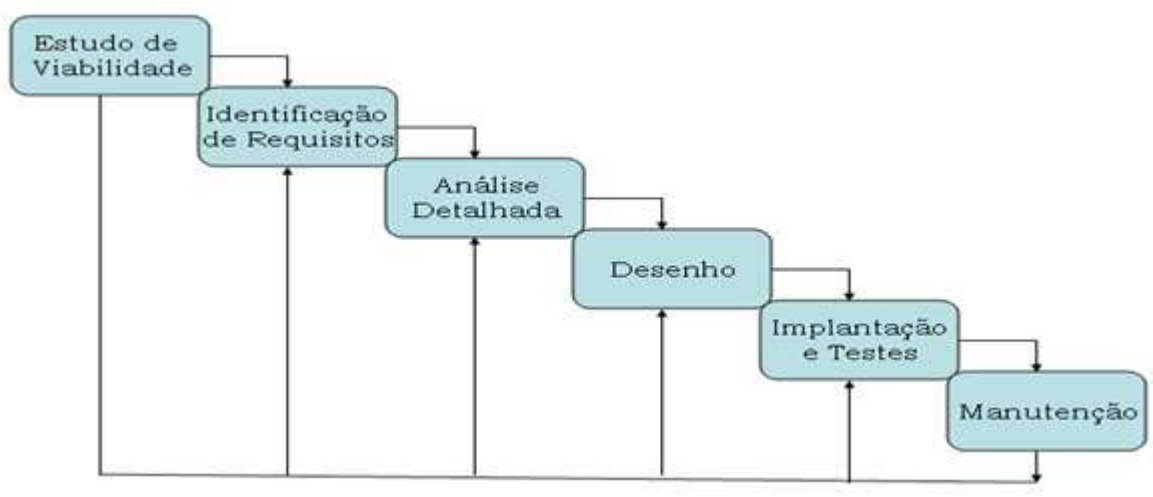

Figura 2.1: As fases do ciclo de vida do modelo em cascata - Adaptado: Lopes et al. (2005)

Cada fase tem objectivos bem definidos, nomeadamente:

Estudo de viabilidade: consiste em analisar o problema existente e de uma forma breve apontar soluções alternativas, resultando uma proposta descrita em termos técnicos, operacionais e económicos que poderá ou não ser aceite. Neste último caso o processo pára.

Identificação de requisitos: consiste em fazer uma recolha profunda de informação sobre o projecto a desenvolver: requisitos, restrições a que é preciso obedecer, problemas e falhas existentes.

Análise detalhada: consiste em desenvolver uma especificação dos requisitos levantados na fase anterior, construindo modelos consistentes.

Desenho: tendo como base a fase anterior, desenvolve a arquitectura do sistema, especificando os seus componentes.

Implantação e testes: consiste em executar testes ao produto resultante, em definir a forma de conversão na organização do sistema antigo para o novo e fazer a formação dos utilizadores.

Neste modelo, segundo Lopes et al. (2005), cada fase tem objectivos bem definidos e dá possibilidade de iteração entre fases. Assim, se houver algo numa determinada fase que afecte as fases anteriores poder-se-à voltar a essas fases para fazer as correcções necessárias.

Algumas críticas estão associadas a este modelo, como, segundo Lopes et al. (2005), o tempo associado com o progresso das actividades que, por vezes, na altura da entrega do projecto, os requisitos podem ter mudado; e a ausência do envolvimento do utilizador no processo de desenvolvimento.

Existem várias versões do modelo em cascata, diferindo normalmente no número, nome e descrição de cada fase, existência de iteração entre fases e existência de validação em cada fase. Existem também muitas variantes deste modelo de desenvolvimento adaptadas a diferentes tipos de projectos. No entanto, eles têm uma característica comum que é o fluxo sequencial de actividades.

2.4 Resumo

Foram neste capítulo destacados o material e métodos e o processo de desenvolvimento do projecto utilizados para o alcance dos objectivos traçados.

Viu-se, portanto, que foram efectuadas a revisão bibliográfica e entrevistas e adaptou-se o modelo em cascata, que foi concebido para projectos de *software*, para desenvolver a arquitectura de sistema de informação tida como solução dos problemas identificados na arquitectura actual.

Capítulo

3 Revisão de literatura

3.1 Introdução

O impacto provocado pelos sistemas de informação em todo o mundo leva as organizações a dependerem do auxílio prestado por estes na satisfação dos seus interesses. Actualmente, os sistemas de informação e suas arquitecturas desenvolvidas anteriormente são reavaliados e até substituídos por novos modelos que melhor explicam e ajustam a nova realidade, visando a satisfação dos seus clientes e consequente sucesso organizacional.

No presente capítulo, mostra-se a forma como as arquitecturas de sistemas de informação e as novas tecnologias estão a definir os negócios nas organizações, as vantagens da sua implementação, particularmente o seu uso em empresas baseadas em transacções e geograficamente distribuídas.

Assim, na primeira secção serão abordados os sistemas legados com destaque para a sua estrutura e sua avaliação. Na segunda secção serão abordados os sistemas baseados em transacções onde se destacam os conceitos de estado de uma transacção e o de transacções OLTP. E na terceira e última secção serão abordadas as arquitecturas de sistemas de informação com destaque para as arquitecturas centralizada, descentralizada, cliente/servidor e *thin client/server* e o servidor citrix XOpen.

3.2 Sistemas legados

As empresas gastam muito dinheiro em sistemas de informação, e para que elas obtenham um retorno desse investimento, o software deve ser utilizado por vários anos. Muitos dos sistemas antigos ainda são fundamentais para as empresas, isto é, as empresas dependem dos serviços fornecidos pelo sistema, e qualquer falha desses serviços teria um sério efeito no dia a dia da empresa.

Segundo Sommerville (2003), sistemas legados são sistemas que apesar de serem antigos eles ainda são fundamentais e fornecem serviços essenciais às organizações.

Os sistemas legados não são os que foram originalmente construídos. Factores internos e externos às empresas introduzem mudanças que geram novos requisitos e, assim, todos sistemas de informação são modificados quando as empresas passam por essas mudanças.

Normalmente são de difícil manutenção e, pelo grau de custo para a sua modernização, continuam activos. Por falta de documentação e com a saída do pessoal técnico que participou no seu desenvolvimento podem apresentar problemas como dificuldade de compreensão das regras de negócio neles implementadas, obsolescência das ferramentas de desenvolvimento e impossibilidade de reaproveitamento dos equipamentos nos quais são executados.

As empresas substituem seus equipamentos e até mesmo suas arquitecturas por sistemas modernos. Contudo, substituir sistemas legados por sistemas modernos envolve riscos significativos para as empresas. E mantê-los em uso evita esses riscos, mas ao agir assim podem advir outros problemas como por exemplo a desactualização e como consequência a empresa pode correr o risco de não responder cabalmente com a demanda do mercado bem como dos seus clientes.

3.2.1 Estrutura dos sistemas legados

Os sistemas legados são sistemas baseados em computadores; assim, eles incluem *software*, *hardware*, dados e processos corporativos (Sommerville 2003).

A figura abaixo ilustra as diferentes partes lógicas de um sistema legado e suas relações:

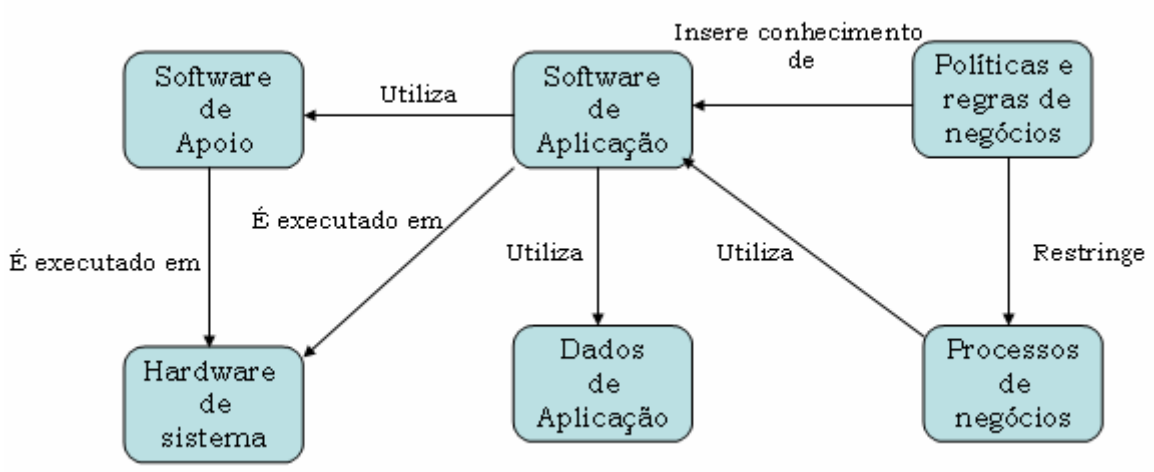

Figura 3.1: Componentes de sistemas legados (Sommerville 2003)

***Hardware* de sistema:** em muitos casos, sistemas legados foram escritos para *hardware* de computadores antigos, que já não está disponível, têm manutenção dispendiosa e pode não ser compatível com as actuais políticas organizacionais de compra de tecnologias de informação.

***Software* de apoio:** o sistema legado depende de diferentes produtos de *software* de apoio fornecidos pelo fabricante de *hardware*. Novamente, esses produtos de *software* podem estar obsoletos e não ter mais a assistência técnica dos seus fornecedores originais.

***Software* de aplicação:** o sistema de aplicação que fornece os serviços de negócios é, em geral, composto de vários programas separados, desenvolvidos em épocas diferentes.

Dados de aplicação: são os dados processados pelo sistema de aplicação. Em muitos sistemas legados, grande volume de dados se acumula durante o tempo de duração do sistema. Esses dados podem ser inconsistentes e podem estar duplicados em diferentes arquivos.

Processos de negócios: são os processos utilizados nas empresas a fim de atingir algum objectivo de negócios. Um exemplo de processo de negócios numa empresa de seguros seria emitir uma apólice de seguro; numa empresa manufactureira, um processo de negócios seria aceitar um pedido de produto e definir o processo de fabricação correspondente.

Políticas e regras de negócios: são as definições de como a empresa deve ser conduzida e as restrições às quais ela deve se submeter. O uso de um sistema legado de aplicações pode estar inscrito nessas políticas e regras.

Uma maneira alternativa de considerer esses diferentes componentes de um sistema legado é consirerá-los como uma série de camadas, como mostra a figura abaixo (Sommerville 2003):

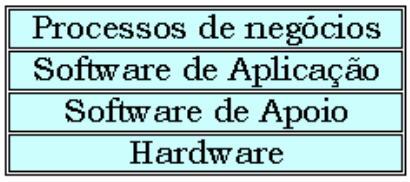

Figura 3.2: Modelo em camadas de um sistema legado (Sommerville 2003)

Cada camada depende da camada imediatamente inferior e faz *interface* com essa camada. Se as *interfaces* forem mantidas, será possível fazer alterações numa camada sem afectar qualquer uma das camadas adjacentes.

Algumas vezes, esse simples encapsulamento não funciona, e mudanças numa camada

do sistema podem requerer consequentes mudanças nas camadas que estão tanto acima como abaixo da camada alterada.

3.2.2 Avaliação dos sistemas legados

As empresas que dependem de sistemas legados e têm um orçamento limitado para a manutenção e actualização desses sistemas precisam decidir como obter o melhor retorno do seu investimento. Isso significa que elas devem fazer uma avaliação real dos seus sistemas legados e, então, decidir sobre qual é a estratégia mais apropriada para que esses sistemas evoluam. Para tal existem quatro opções estratégicas, segundo (Sommerville 2003):

Descartar completamente o sistema: quando o sistema não estiver a prestar uma contribuição efectiva aos processos de negócios. Isso ocorre quando os processos corporativos foram modificados, desde que o sistema foi instalado, e não são mais inteiramente dependentes do sistema. Essa situação é muito comum quando terminais de *mainframe* são substituídos por PCs (Personal Computers) e quando produtos de *software*, nessas máquinas, são adaptados para fornecer o apoio computacional de que o processo de negócios precisa.

Continuar a manter o sistema: essa opção deve ser escolhida quando o sistema ainda é necessário, excepto quando ele for bastante estável e os usuários não pediram um grande número de modificações no sistema.

Transformar o sistema para melhorar sua facilidade de manutenção: quando a qualidade do sistema for degradada por modificações regulares e quando essas modificações aindo forem exigidas.

Substituir o sistema por um novo: quando outros factores, como *hardware* novo, significarem que o sistema antigo não pode continuar em operação ou quando sistemas novos estiverem disponíveis e permitirem que o novo sistema seja desenvolvido por um custo razoável.

Naturalmente, essas opções não são exclusivas: assim, quando um sistema é composto por vários programas diferentes, diferentes opções podem ser aplicadas em diferentes partes do sistema.

Entretanto, a mudança, em todos os níveis, só pode ser efectiva se os planos envolvidos forem satisfatórios aos interessados no processo e têm oportunidade de visualizar o valor da nova estrutura do sistema (Elliman & Coakes 1999).

3.3 Sistemas baseados em transacções

Para competir, de uma forma eficiente, as organizações têm de encontrar um *software* que possa facilitar e gerir Bases de Dados (BDs) de forma estratégica e que possua uma arquitectura que facilite a operacionalização das suas actividades de modo a que se cumpram os seus objectivos.

Hoje em dia, os negócios se tornam, cada vez mais, em processos de negócio baseados em transacções orientadas a serviços, bastante úteis nas organizações para fornecer serviços cada vez mais satisfatórios aos clientes. Daí interessar muito, no ambiente das organizações, o conceito de transacção.

3.3.1 Transacção

Segundo Pereira (1998), uma transacção é um conjunto de operações (que desempenham uma função lógica dentro de uma aplicação do sistema de BDs) sobre a BD, perfeitamente delimitado que exibe algumas características importantes:

Atomicidade: O conjunto de operações que constituem uma transacção forma um grupo indivisível (atómico), no sentido em que todas elas são executadas com sucesso ou nenhuma é executada. Por outras palavras, uma transacção ou termina com sucesso (faz o *commit*) ou, então, todas as suas acções sobre a BD são desfeitas (faz o

rollback), dando a ilusão que nunca existiram;

Integridade: Uma transacção, se envolver actualização de dados, deve transportar a BD de um estado de integridade para outro estado também de integridade. Durante a execução duma transacção, integridade da BD pode ser nomeadamente violada, contudo, quando esta termina, a integridade deve ser assegurada;

Isolamento: Embora várias transacções possam executar concorrentemente, cada transacção tem de desconhecer que outras estão a executar duma forma concorrente. Resultados intermédios da transacção tem de ser escondidos das outras transacções concorrentes. Ou seja, numa situação em que várias transacções concorrentes acedem aos mesmos dados, o sistema deve evitar que estas interfiram entre si, garantindo que o resultado final seja o mesmo que se as transacções executassem em série;

Persistência: O sistema deve assegurar que todos os efeitos provocados por uma transacção bem sucedida se tornem persistentes na BD e visíveis para as outras transacções. Ou seja, após uma transacção terminar com sucesso (*commit*), as suas actualizações sobre a BD passam a ser efectivas, sobrevivendo a todo o tipo de falhas que, eventualmente, possam ocorrer. Desta forma, os seus efeitos sobre a BD não poderão ser desfeitos, a não ser por transacções posteriores.

Exemplo duma transacção: para transferir 600 Mts da conta C(a) para a conta C(b)

1. ler [C(a)]

2. C(a):= C(a) - 600

3. escrever [C(a)]

4. ler [C(b)]

5. C(b) := C(b)+ 600

6. escrever[C(b)]

Para esta transacção as principais características são:

Requisito da atomicidade: se a transacção falha após o passo 3 ou antes do passo 6, o sistema deve assegurar que as actualizações não se reflectem na BD; senão, a BD ficará inconsistente.

Requisito da integridade: a soma de C(a) e C(b) não pode mudar pela execução da transacção.

Requisito do Isolamento: se entre os passos 3 e 6, a uma outra transacção é permitido o acesso a uma BD parcialmente actualizada, ela encontrará uma BD inconsistente (a soma C(a) + C(b) será menor que o que deve ser). Pode ser assegurado por correr transacções em série, uma após a outra. Contudo, executar várias transacções concorrentemente tem benefícios significativos.

Requisito da persistência: uma vez que o utilizador foi notificado que a transacção terminou (a transferência de 600 Mts teve lugar), as actualizações da BD feitas pela transacção devem persistir apesar das falhas.

3.3.2 Estado duma transacção

Normalmente, uma transacção completa-se quando o cliente executar e terminar o pedido. Se a transacção tiver progredido normalmente o estado duma transacção é *commited*, isto é, constitui uma garantia para o cliente de que todas as alterações requeridas na transacção foram permanentemente registradas e que qualquer transacção futura que irá aceder os mesmos dados verá os resultados de todas as alterações feitas durante a transacção (Coulouris et al. 2001).

Alternativamente, a transacção poderá ir ao aborto (estado de abortada) por uma das várias razões relacionadas com a natureza da transacção, como por exemplo, por causa de conflitos com outra transacção ou processo de computador. Quando uma transacção é abortada as partes envolvidas devem assegurar que nenhum dos efeitos é visível para futuras transacções, nem nos objectos ou nas suas cópias no armazenamento permanente. Uma transacção ou é sucedida ou é abortada em uma das duas maneiras: o cliente aborta-a ou o servidor aborta-a. Referimo-nos a uma transacção como falhada em ambos os casos (Coulouris et al. 2001).

3.3.3 Transacções OLTP

No mundo das transacções *online* executadas pelas aplicações podemos encontrar os sistemas OLTP (*OnLine Transaction Processing*) que se têm tornado bastante úteis nos dias de hoje para as organizações nas áreas operacionais em que são adequados.

Os sistemas OLTP são sistemas que se encarregam de registrar todas as transacções contidas numa determinada operação organizacional. Por exemplo: o sistema de transacções bancárias registra todas as operações efectuadas num banco (Wikipedia 2009*a*).

Os OLTP também são adequados para sistemas transaccionais de supermercados e sistemas de seguros. Os ERP (*Enteprise Resource Planning*) também são sistemas que se enquadram nesse tipo de transacções.

Requisitos

OLTP requer suporte para transacções em rede. Por isso, as actuais aplicações OLTP utilizam processamento cliente/servidor e aplicações que permitem que as transacções corram em diferentes plataformas de computadores numa rede. Para uma maior descentralização de sistemas de BD, aplicações OLTP intermediárias podem distribuir o processamento de transacções por diversos computadores numa rede de computadores (Wikipedia 2009*a*).

Vantagens dos OLTP

Entre muitas vantagens que OLTP apresenta destacam-se a sua simplicidade e eficiência (Wikipedia 2009*a*):

Simplicidade: Redução de documentos e rapidez no cálculo de retornos e despesas são exemplos sobre como OLTP simplifica os negócios. Ele também serve como base para o estabelecimento de uma organização estável, por causa da actualização constante. Outro factor de simplicidade é o que permite aos consumidores a escolha de como eles querem pagar, tornando muito mais fácil concretizar uma transacção.

Eficiência: OLTP torna as bases de uma organização muito dinâmicas, os processamentos individuais são mais rápidos e estão disponíveis durante muito tempo. Desta forma pode-se evitar perda de tempo e de recursos no tratamento de vários processos de uma organização.

Desvantagens dos OLTP

OLTP é uma excelente ferramenta para qualquer organização, mas ao utilizá-lo existem algumas contrariedades, as questões de segurança e custos (Wikipedia 2009*a*):

Segurança: Um dos problemas do OLTP é também uma grande ameaça de segurança pois a disponibilidade plena das informações que esses sistemas propiciam também deixa os dados acessíveis a *crackers* e intrusos, podendo por isso permitir o roubo ou a falsificação da informação.

Custos: Nestes sistemas, a menor falha tem o potencial para causar uma série de problemas, causando perda de tempo e dinheiro. Outro custo a ser observado tem a ver com a falha potencial de servidores. Isto pode causar demora na recuperação ou até mesmo perda de uma quantidade enorme de dados.

Para além dos OLTP, no mundo das transacções *online*, podemos encontrar os sistemas **OLAP** (*Online Analytical Processing*), que constituem-se de uma capacidade para manipular e analisar um elevado volume de dados sob múltiplas perspectivas, introduzindo o conceito de BI (*Business Interconnection*). As aplicações OLAP são usadas pelos gestores em qualquer nível da organização para lhes permitir análises comparativas que facilitem a sua tomada de decisões diária.

OLAP versus OLTP

Enquanto o OLTP opera com dados que movimentam o negócio em tempo real, suportando operações quotidianas de negócio empresariais por meio do seu processo operacional, OLAP trabalha com dados históricos no sentido de analisar informações.

O OLTP tem como função alimentar a BD que compõem o OLAP, o OLTP é uma ferramenta relacional, orientada para processos, trabalhando com dados do presente e processando um registro de cada vez, não sendo multidimensional como o OLAP (Wikipedia 2009*a*).

A finalidade do OLTP é fazer com que uma grande quantidade de pequenas informações não se perca, processando milhares de informações por dia, que contém em cada uma delas uma pequena porção de dados. Os usuários de OLTP freqüentemente lidam com um registro de cada vez, o que faz com que a mesma tarefa seja executada inúmeras vezes, pois a maioria dos seus relatórios são feitos numa tabela inteira. Logo (Wikipedia 2009*a*).

Embora muitas vezes citado como referência para BD, o OLTP geralmente pode ser utilizado para descrever um ambiente de processamento de transacções e assim, agilizar o ambiente de consulta além de apoiar o OLAP.

3.4 Arquitecturas de sistemas de informação

Segundo Kim & Everest (1994), uma arquitectura pode ser definida como um plano para construção de alguma coisa, no qual todas as partes são reunidas num todo, de modo a satisfazer determinadas necessidades funcionais ou artísticas. Arquitectura de sistemas de informação pode se designar como a conjugação de todos os elementos que o constituem para estabelecer o fluxo da informação numa organização, tendo como principal objectivo o suporte do negócio.

Nos primórdios, o ambiente informático de uma organização era limitado a uma só máquina de grandes dimensões e com enorme poder computacional, o *mainframe*. Neste ambiente, o administrador tinha controlo total sobre a organização, podendo gerir toda informação que flui na organização.

Entretanto, a evolução tecnológica levou ao desaparecimento da arquitectura baseada em *mainframes*, surgindo outras, visando melhorar o seu desempenho para a satisfação dos requisitos de negócio das empresas, destacando-se as arquitecturas centralizada, descentralizada, cliente/servidor e *thin client/server*.

3.4.1 Arquitectura centralizada

Geralmente, fazer tudo a partir de um ponto central simplifica a administração, permitindo a uma única equipa de TI resolver todas as questões de manutenção e *update* a partir de um só local. Porém, a centralização de serviços pode resultar num desempenho de rede muito lento para as estações de trabalho, em alguns casos.

Segundo (Camargo et al. 2005), os sistemas centralizados são aqueles que armazenam todas as informações e documentos num servidor central. Todos os usuários do sistema acedem, consultam, obtêm e carregam arquivos de documentos neste servidor. O computador do usuário apenas fornece a *interface* ao sistema e, eventualmente, alguma funcionalidade

local para manipular arquivos ou pré-processar informações a serem enviadas ao servidor. O uso de sistemas centralizados só é possível quando o usuário está conectado ao sistema.

Nesta arquitectura somente o servidor central controla todas as estações de trabalho conectadas. É normal usar-se um conjunto de servidores no tratamento das actividades do sistema e todos esses servidores se encontram no mesmo espaço físico, o *datacenter*, como mostra a figura abaixo:

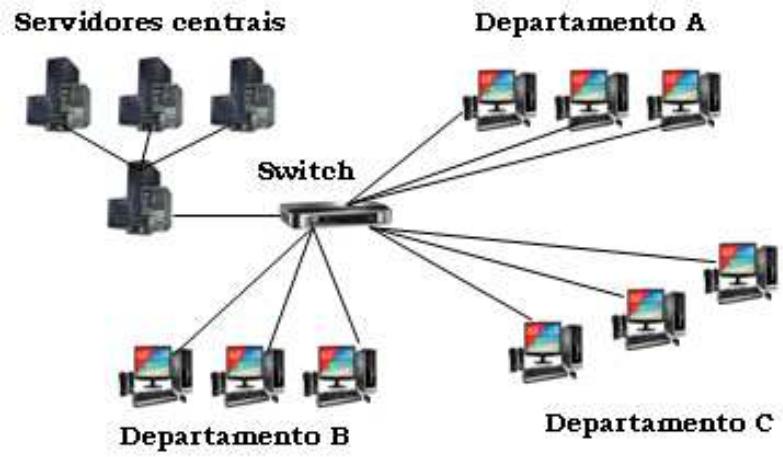

Figura 3.3: Arquitectura centralizada

Vantagens da arquitectura centralizada

A arquitectura centralizada e mostrada na figura 3.3 é extremamente útil para sistemas de informação tendo como base duas carcterísticas, a simplicadade e a flexibilidade, duas características desejáveis em qualquer sistema, sendo actualmente bastante utilizada em muitas empresas. Isso ocorre em razão, entre outras, das seguintes vantagens:

- Redução de custos e do esforço físico na administração da rede pois os servidores encontram-se no mesmo espaço físico, criando excelentes níveis de produtividade e segurança;

- Baixo custo de licenciamento das máquinas pois o licenciamento não é feito em todas as máquinas;

- Baixo custo de aquisição pois os equipamentos das estações não precisam de ter recursos poderosos;
- Não exige nenhuma capacidade especial da infra-estrutura de rede pois todas as conexões são ponto a ponto;

Desvantagens da arquitectura centralizada

Apesar de possuir grandes vantagens nos sistemas de informação, esta arquitectura ainda possui desvantagens, dentre as quais destacamos as seguintes:

- No caso de algum problema no *datacenter*, a falta de redundância noutro local físico pode causar uma paragem total da rede;
- Geralmente requer servidores especiais, que são recursos caros e limitados;
- Gera mais tráfego na rede pois o tráfego apenas é roteado pelo servidor podendo provocar um congestionamento na rede;
- A centralização baseada em acesso por terminais remotos implica custos de comunicação significativos, porque não tira partido da possibilidade do trabalho local (Marques & Guedes 1999).

3.4.2 Arquitectura descentralizada

As empresas podem recorrer a outras arquitecturas para minimzar alguns dos problemas associados à arquitectura centralizada. Por exemplo, recorrendo à arquitectura descentralizada tira-se partido do trabalho desenvolvido localmente devido à não dependência total dos servidores centrais podendo facilitar a evolução dos sistemas e evitar que algum problema no *datacenter* cause uma paragem total da rede.

Segundo Camargo et al. (2005), os sistemas descentralizados fazem uso de um conceito diferente: utilizam BDs sincronizáveis, o que significa que o usuário pode fazer uso de

parte do sistema mesmo estando desconectado. Quando o mesmo se conectar ao sistema, as informações alteradas pelos usuários na sua BD local serão enviadas para um servidor central que se encarregará de sincronizar tais alterações com as dos demais usuários. Igualmente, o usuário terá sua BD local sincronizada com as alterações feitas pelos demais usuários. Evidentemente que a funcionalidade total de um sistema descentralizado só é possível quando o mesmo está conectado ao servidor da aplicação.

Nesta arquitectura, basicamente cada unidade da organização possui os próprios servidores (ver Figura 3.4) e os componentes localizados na rede comunicam e coordenam as suas acções através do envio e recepção de mensagens.

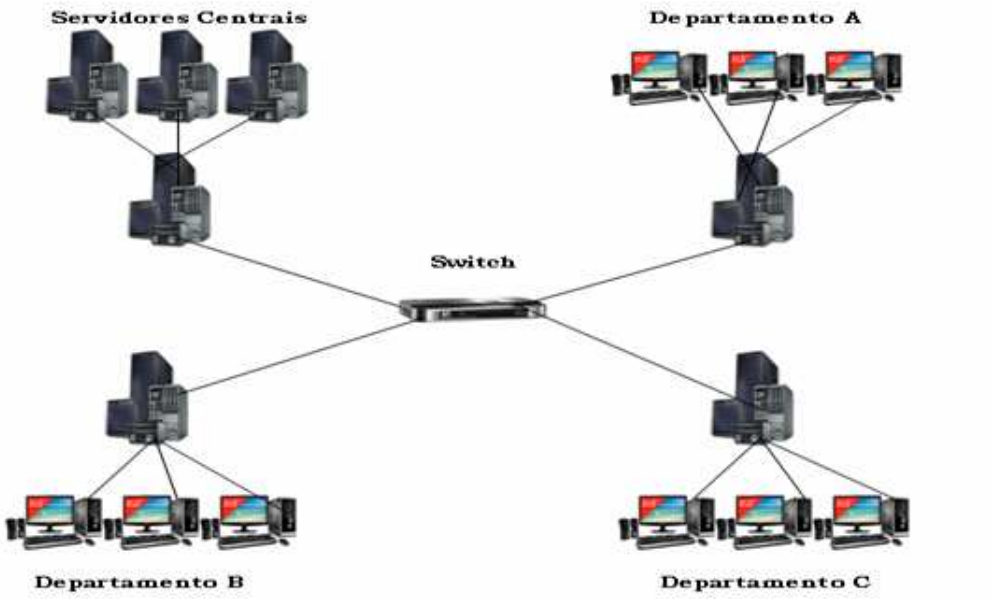

Figura 3.4: Arquitectura descentralizada

Vantagens da arquitectura descentralizada

O principal objectivo dos sistemas descentralizados é de conectar usuários e recursos nas melhores condições do seu negócio, podendo se associar às seguintes vantagens:

- Se uma máquina apresenta algum problema, ainda que seja um dos servidores, parte

do sistema continua activa;

- O poder computacional pode ser expandido gradualmente, conforme a necessidade e torna-se fácil modernizar o sistema quando necessário;
- Pode-se misturar várias plataformas para melhor atender às necessidades individuais de diversos sectores e usuários;
- Maior rapidez pois em muitos casos as operações são processadas localmente sem necessidade de contactar o servidor, como é o caso do processamento *offline*;

Desvantagens da arquitectura descentralizada

Algumas inconveniências tornam esta arquitectura menos opcional em alguns casos. Destacam-se as seguintes desvantagens:

- O custo do licenciamento é elevado pois geralmente devem ser licenciadas todas as máquinas;
- Dificuldades de garantir segurança e detecção de falhas pois os sistemas com arquitectura descentralizada são geralmente complexos;
- Maior probabilidade de ocorrência de falhas devido ao maior número de componentes;

3.4.3 Arquitectura cliente/servidor

É uma arquitectura baseada no conceito de prestação de serviço e define um diálogo de pedido e resposta entre um computador cliente que tem necessidade de serviços fornecidos pelo outro computador, o servidor.

Nesta arquitectura, o termo servidor aplica-se para qualquer programa que oferece um serviço que pode ser disponibilizado numa rede. O servidor aceita um pedido na rede, executa o seu serviço e devolve o resultado do requerente (Comer 1995).

Ambas partes devem estar conectadas entre si através de uma rede. Uma representação gráfica deste tipo de arquitectura seria a seguinte:

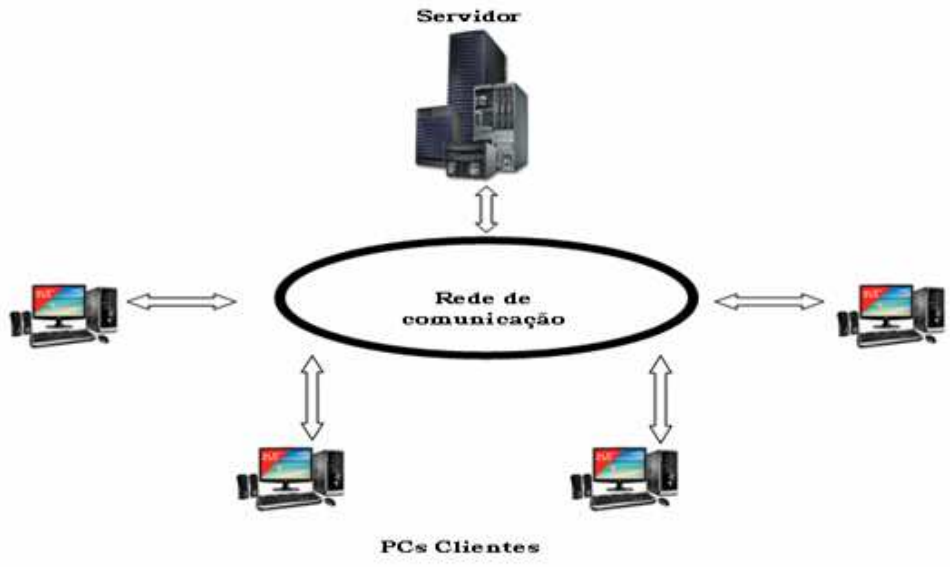

Figura 3.5: Arquitectura cliente/servidor (VolHost 2007).

Funcionamento

Esta arquitetura necessita de três tipos de *software* para o seu correcto funcionamento (VolHost 2009):

***Software* de gestão de dados:** este *software* se encarrega da manipulação e gestão de dados armazenados e requeridos pelas aplicações. Normalmente este *software* se hospeda no servidor;

***Software* de desenvolvimento:** este tipo de *software* se hospeda nos clientes e só naqueles que se dedicam ao desenvolvimento de aplicações;

***Software* de interacção com os usuários:** também reside nos clientes e é a aplicação gráfica de usuário para a manipulação de dados, sempre é claro, a nível de usuário (consultas principalmente).

Para além destes existem mais aplicações *software* para o correcto funcionamento desta arquitectura, mas já estão condicionados pelo tipo de sistema operativo instalado, o tipo de rede na qual se encontra.

Vantagens da arquitectura cliente/servidor

Para diversos tipos de aplicações com BD, a arquitectura cliente/servidor oferece várias vantagens, incluindo as seguintes:

- De acordo com Tanenbaum (1992), todos os servidores correm como processos do cliente e não têm acesso directo ao *hardware*. Como consequência, se uma falha ocorre no servidor do ficheiro, dificilmente irá paralisar toda máquina;
- Esta arquitectura adapta-se facilmemte aos sistemas distribuídos pois facilmente um cliente troca mensagens remotamente com um servidor (Tanenbaum 1992);
- As operações são mais confiáveis, porque um único servidor de BD interage com os dados, ao invés de várias cópias de BD;
- Tem capacidade de responder ao aumento da procura de serviços sem degradar a performance (Ribeiro 1998).

Desvantagens da arquitectura cliente/servidor

Entre outras desvantagens que caracterizam esta arquitectura, apresentam-se as seguintes:

- Os pacotes das mensagens trocadas podem se perder na rede, havendo possibilidade de recuperá-los. Porém, é extremamente difícil recuperar um pacote perdido na rede (Tanenbaum 1992);
- Tarefas com alta demanda do servidor e grandes volumes de transferências de dados ocasionam aumento do tempo ou paragem total de respostas, poder de processamento do cliente não utilizado e volume de recursos para construir servidores potentes;

- A troca de dados pode ser ineficiente devido à complexidade dos sistemas que utilizam este tipo de arquitectura.

3.4.4 Arquitectura *thin client/server*

A arquitetura *thin client /server* é aquela em que as aplicações são executadas nos servidores, e a estação de trabalho oferece a função de interface com o usuário e permite uma forma de implementar e gerir aplicações com alta performance (SAMURAI 2008).

Nesta arquitectura, geralmente o computador servidor é um multiprocessador ou computador *cluster* correndo uma versão de um processador de um sistema operativo como *Unix* ou *Windows* (Coulouris et al. 2001).

Ao se conceber qualquer arquitectura, deve-se decidir sobre que partes das tarefas devem ser executadas no cliente e quais no servidor. Esta decisão pode afectar o custo de clientes e servidores, a robustez, segurança e a flexibilidade do sistema para uma modificação ou migração para uma outra arquitectura. Dependendo do resultado destas decisões, pode-se dizer que tipo de cliente deverá ser usado (magro ou gordo ou uma mistura de ambos).

3.4.5 *Thin client*

Um *thin client* é um computador cliente numa rede de arquitectura cliente/servidor o qual tem poucos ou nenhum *software* instalados, de modo que depende completamente de um servidor central para o processamento de tarefas (ITCENTER 2008). Enquanto que um cliente gordo executa tanto processamento quanto possível e passa ao servidor somente dados necessários para serem processados nele.

Um *thin client* conta com um servidor de aplicações para as tarefas mais relevantes da sua lógica interna, tendo um mínimo de *hardware* e *software* presentes na máquina cliente

(Wikipedia 2009*b*).

Um *thin client* é projectado para fornecer apenas aquelas funções que são úteis para programas de *interface* de usuário. O usuário precisaria apenas de um monitor, teclado, um dispositivo apontador e capacidade de processamento suficiente para lidar com a exibição de imagens e as comunicações. Esta circunstância é adequada para usuários dos sistemas baseados em transacções, como por exemplo os bancos e empresas de seguros, pois eles não precisam tanto dos recursos computacionais locais.

Qualquer que seja o tamanho da empresa, o *thin client* permite acesso aos usuários da sua rede às aplicações e dados que estão nos seus servidores.

Aplicação dos *thin clients*

Os *thin clients*, para além dos sistemas baseados em transacções, podem também ser aplicados noutros ambientes de virtualização de clientes, como por exemplo: na tecnologia móvel permitem, entre outros benefícios, o suporte organizacional.

Os *thin client* também são adequados para os laboratórios de investigação tecnológica, devido às facilidades na sua instalação, compatibilidade, funcionalidade, capacidade multimédia, baixo custo, e facilidade de padronização (Huwo 2009).

Incorporando a tecnologia *thin client* numa infraestrutura de rede de um campus universitário pode aumentar a disponibilidade tecnológica sem aumentar custos (CarrBy & Bair 2007).

Funcionamento da arquitectura *thin client/server*

De acordo com ITCENTER (2008), a arquitetura *thin clien/server* é composta basicamente por *hardware* (cliente e servidor) e *software* (cliente e servidor).

Hardware

Servidor: Os servidores constituem a parte mais importante desta arquitectura. Todas as aplicações e dados são instalados, administrados e mantidos neles. Não é obrigatória a instalação das aplicações e dados num único servidor. Múltiplos servidores podem estar ligados numa arquitectura distribuída, onde aplicação e BD estão instaladas e sendo executadas em servidores distintos. Neste caso o servidor onde a aplicação servidora *thin client* é executada é chamado de servidor gráfico.

Cliente: O *hardware thin client* é basicamente um terminal gráfico. O *thin client* não possui disco duro, e pode inicializar a partir de um DOM (*Disk On Module*) ou a partir da rede, fazendo *download* de todo o sistema operativo ou de partes dele armazenados num servidor de *boot* na rede. O processador deve ser definido em função da necessidade imposta para processar o protocolo de comunicação entre o servidor e o cliente. A memória deve ser suficiente para carregar o sistema operativo, a parte gráfica e o cliente.

Software

Servidor: O servidor cria um *desktop* no qual as aplicações desenham a sua *interface* segundo uma API (*Application Programming Interface*) qualquer. Esse *desktop* pode ou não ser virtual. Se o servidor se limita a ler imagens da placa gráfica, trata-se de um *desktop* real. O servidor detecta as alterações que as aplicações realizam sobre o *desktop* e envia-as ao cliente. Por outro lado o servidor recebe do cliente os códigos das teclas digitadas e as movimentações do *mouse*.

Cliente: O cliente é uma aplicação gráfica muito simples (justificando a designação *thin client*) que reproduz o *desktop* residente no servidor e lhe envia os códigos das teclas e movimentações do *mouse* sobre a sua janela.

Comunicação

Existem vários protocolos que permitem o acesso a um ambiente gráfico remoto. Trata-se de implementações independentes das aplicações que transmitem imagens de um *desktop* virtual ou não. O protocolo é a base da comunicação entre o cliente e servidor *thin client* (ITCENTER 2008).

A figura abaixo representa de forma simplificada a interação destes componentes numa ligação entre máquina cliente e máquina servidora.

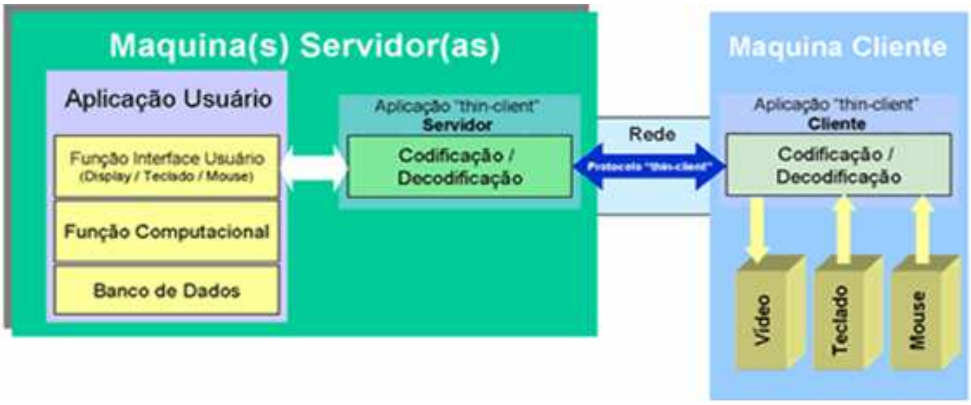

Figura 3.6: Funcionamento da arquitectura *thin client/server* (ITCENTER 2008)

Modelo de aplicação da arquitectura *thin client/server*

Não existe padrão na metodologia para aplicação da arquitectura *thin client/server*. A arquitectura pode ser híbrida, de modo a tirar proveito dos benefícios das arquitecturas centralizada e descentralizada.

O objectivo da adopção de uma arquitectura baseada na arquitectura *thin client/server* está na possibilidade do uso de servidores dedicados para execução das aplicações, de *hardware* que seja robusto e barato no cliente, características da arquitectura centralizada, aliada aos recursos avançados de processamento especializado e *interface* gráfica elaborada, disponíveis na arquitectura descentralizada (ITCENTER 2008).

Segundo ITCENTER (2008), o modelo pode ser uma mistura de:

- Arquitetura centralizada;
- Arquitetura descentralizada;
- Arquitetura cliente/servidor tradicional;
- Arquitetura *Internet* com cliente baseado em *browser*.

Vantagens da arquitectuira thin client/server

Entre outras vantagens desta arquitectura, podemos destacar as seguintes:

- Menor custo de investimento na aquisição, pois o valor de um *thin client* é menor em relação ao de um PC e não precisa de ser substituído após *upgrades* de *softwares* ou podemos transformar computadores com baixo processamento em *thin clients*;
- Diminuição de custos de manutenção, dado que os *thin clients*, apresentam um tempo de vida superior ao PC e escapam na inevitável obsolescência tecnológica e raramente precisam de actualizações;
- Os *thin clients* reduzem o tempo e as visitas da equipa de TI aos computadores dos usuários para a manutenção dos equipamentos porque as aplicações pesadas se encontram nos servidores centrais, aos quais os clientes estão conectados (ITCENTER 2008);
- A administração remota de recursos de tecnologia de informação directamente do servidor facilita a gestão de novas versões de aplicações, além de controle de níveis de segurança e redução dos custos de manutenção e administração (ITCENTER 2008);
- Não permite que os utilizadores façam configurações, instalações de software e alterações não autorizadas e cópias de dados da empresa;

- Gera menos tráfego na rede pois, segundo José Pinheiro (2004), somente o teclado, as telas e o *mouse* trafegam pelo meio de comunicação ou pela rede, tendo como resultado imediato, uma melhor performance das aplicações, independente da largura de banda do meio de comunicação.

Desvantagens da arquitectuira thin client/server

Destacam-se as seguintes desvantagens desta arquitectura:

- Requer mais servidores e poderosos pois todo processamento é efectuado ao nível dos servidores;
- Qualquer falha dos servidores pode paralisar toda a rede, afectando a disponibilização dos serviços aos usuários;
- Empresas que pensam em migração para esta arquitectura devem encontrar resistência dos usuários pois estes estão habituados a um ambiente livre de gestão, onde os computadores são configurados de acordo com suas preferências pessoais.

3.4.6 O servidor *citrix XOpen*

É uma tecnologia baseada em servidor, optimizada para trabalhar mesmo em conexões de baixa velocidade, possibilitando que as aplicações sejam executadas de forma eficiente sobre qualquer infra-estrutura de comunicações (Pinheiro 2004).

Esta tecnologia é uma combinação de *hardware* e *software* base que permite a disponibilização de aplicações desenvolvidas para plataformas *Windows*, com uma alta performance, oferecendo às organizações uma maneira de expandir seus recursos computacionais, simplificar o desenvolvimento e a manutenção dos seus sistemas, para além de diminuir o custo de propriedade dos mesmos (CitrixMetaframeXp 2008).

Características do servidor *citrix XOpen*

Destacam-se as seguintes características, do *citrix XOpen* (CitrixMetaframeXp 2008):

- Os dados são acedidos a partir do servidor de ficheiros e a apresentação dos mesmos é feita na tela do usuário;
- O processamento está alinhado com a tecnologia de processamento de aplicações, alinhado com o conceito de *thin clients*;
- Possui uma arquitectura baseada em servidor e retém todo o tráfego gerado na rede. Somente as modificações das telas, os movimentos do *mouse* e os caracteres digitados no teclado trafegam pela rede e meios de comunicação;
- É optimizado para trabalhar com conexões de baixa velocidade como 14.4 Kbps, permitindo a execução de forma muito rápida de aplicações sobre qualquer infra-estrutura de comunicação incluindo roteadores, nós remotos, linhas discadas, entre outros;
- Funciona com qualquer tipo de equipamento que a organização possui. Permite a utilização de terminais não inteligentes e de qualquer equipamento cliente que utilize sistemas operativos DOS, UNIX, OS/2 Warp, Mac OS e Java;
- Permite gerir todo o ambiente corporativo, e suportar um ambiente de processamento distribuído com centenas ou até milhares de usuários operando simultaneamente;
- As aplicações podem ser administradas através de vários servidores a partir de um ponto central aumentando a flexibilidade e facilidade de suporte e manutenção de aplicações. E os dados vitais da empresa estarão completamente protegidos porque não serão transmitidos para fora da empresa pelas linhas de comunicação.

Funcionamento do servidor *citrix XOpen*

Segundo Pinheiro (2004), o servidor *citrix XOpen* está baseado no modelo de computação

baseada em servidor. A sua arquitectura permite reter todo o tráfego gerado no *backbone* da rede local. Assim, somente as modificações das telas, os movimentos do *mouse* e os caracteres digitados no teclado trafegam pela rede e meios de comunicação, resultando numa pequena fracção de todo o tráfego gerado pela aplicação, como mostra a figura abaixo:

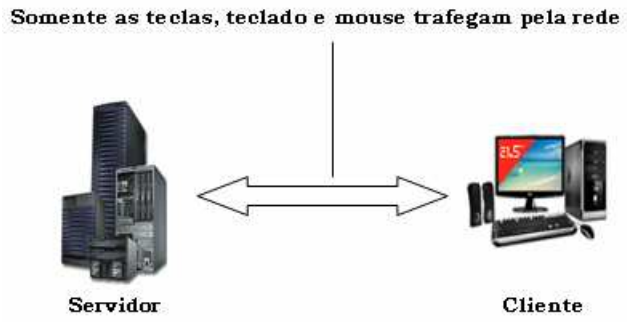

Figura 3.7: tráfego com o servidor *citrix XOpen* (Pinheiro 2004)

Isso permite que os usuários locais ou remotos tenham um excelente tempo de resposta nas aplicações, mesmo em redes congestionadas.

Por utilizar o processo de envio de tela, *mouse* e teclado ao PC cliente, tanto a carga do programa executável, quanto a abertura do arquivo de dados ocorrerá no PC virtual de forma muito rápida na rede local. Somente a imagem da tela, o *mouse* e teclado é que serão transmitidos ao usuário. Por serem poucos dados, o tempo de resposta é imediato. O usuário remoto terá a sensação de estar a operar um PC local, dada a facilidade e rapidez do processo. O *citrix XOpen* permite que usuários de sistemas possam receber e operar sistemas desenvolvidos para a plataforma *Windows*, através da instalação de um programa cliente no PC remoto.

As modificações e novas versões dos sistemas são implantados de uma única vez para todos os usuários no servidor.

Segurança no *citrix XOpen*

O *citrix XOpen* aumenta a segurança dos sistemas de informação porque mantém todas

as informações vitais na rede. Isso significa que usuários localizados em qualquer parte da organização (no país ou no exterior) podem aceder a mesma informação centralizada e actualizada, ao mesmo tempo em que a organização tem condições de reduzir os custos operacionais, aumentar a segurança dos dados e aplicações, aumentar o controle e disponibilidade de BD (CitrixMetaframeXp 2008).

O *citrix XOpen* também reduz a quantidade e a possibilidade de que os dados precisem ser recuperados em razão de perda ou roubo de informações. Isso porque somente as telas das aplicações são transmitidas pelas redes ou linhas de comunicações (CitrixMetaframeXp 2008).

Para ajudar a controlar os acessos dos usuários autorizados a aceder as aplicações corporativos, o *citrix XOpen* incorpora a segurança do *Windows-NT*, que registra os acessos a nível de usuário/*password*, de acesso às pastas e arquivos, os *logins* e *logouts*, e às tentativas de quebrar segurança. Se uma segurança adicional for necessária, o administrador pode adicionar os serviços opcionais do *SecureICA* (Independent Computing Architecture) para o servidor *citrix XOpen*, e garantir que todos os dados sejam encriptados com chaves de até 128 *bits*, tanto para o tráfego das telas quanto dos dados (CitrixMetaframeXp 2008).

O usuário, ao dar *login* num servidor de acesso remoto *citrix XOpen*, recebe uma tela de identificação para que o seu perfil possa ser validado. Uma vez identificado ao sistema, o servidor *citrix XOpen* criará um ambiente conhecido como máquina virtual que se assemelha em todas as suas funcionalidades a um computador real (CitrixMetaframeXp 2008).

O perfil e os limites de acesso que o usuário remoto tem sobre o seu PC virtual depende do contexto que ele foi criado no servidor de acesso remoto *citrix XOpen*. A quantidade de memória virtual, os periféricos a que terá direito, os protocolos que serão habilitados e os programas que ele terá acesso, dependerão dessas definições. O usuário no seu PC físico, estará a receber apenas tela, teclado e *mouse* do seu PC virtual. Tudo se passa na

memória do servidor de controle remoto *citrix XOpen* (CitrixMetaframeXp 2008).

Vantagens do uso do servidor citrix XOpen

Ao utilizar este tipo de servidores, podemos ter as seguintes vantagens:

- Reduz-se o tráfego gerado nas redes permitindo que os usuários locais ou remotos tenham um excelente tempo de resposta nas suas aplicações, mesmo em redes congestionadas;
- Permite a renovação do potencial dos recursos informáticos da organização desactualizada, permitindo que os sistemas e aplicações mais avançados e complexos possam ser operados virtualmente a partir de qualquer equipamento *Windows* ou não *Windows*, com o máximo de performance e produtividade;
- Permite a disponibilização das aplicações de automação de escritórios aos usuários locais e remotos, sem necessidade de qualquer modificação no código e no ambiente de rede de comunicação (CitrixMetaframeXp 2008);
- Permite aos gestores de TI actuar de forma rápida e fácil na gestão e no desenvolvimento/actualização das aplicações. Além disso a organização pode desenvolver essas aplicações utilizando qualquer tipo de equipamento de *hardware* disponível (CitrixMetaframeXp 2008);
- Se o número de usuários ou sistema aumentar significativamente que exija ampliação da capacidade dos servidores, o *citrix XOpen* poderá ser reforçado com a adição de mais servidores e a utilização de um produto de gestão de carga chamado *Citrix Load Balance*, que tranaforma o conjunto de servidores num *cluster* com uma única imagem para o administrador e para os usuários (CitrixMetaframeXp 2008);
- O *citrix XOpen* simplifica o desenvolvimento de sistemas de aplicações e facilita a gestão das aplicações por permitir o desenvolvimento, a manutenção e o suporte a partir de um ponto central, de maneira rápida e fácil (CitrixMetaframeXp 2008).

3.5 Resumo

Neste segundo capítulo, foram destacados os sistemas legados, sistemas baseados em transacções e arquitecturas de sistemas de informação.

Viu-se que os sistemas legados são originados por mudanças que surgem nas empresas. Estes sistemas podem ser refrescados recorrendo a novas tecnologias, sendo para tal necessário fazer uma avaliação cuidadosa.

Os negócios têm se tornado em processos de negócio baseados em transacções orientadas a serviços, úteis para satisfazer os clientes e, no mundo das transacções, as organizações podem utilizar os sistemas OLTP e OLAP nas áreas em que cada um é adequado para aumentar a sua competitividade no mercado.

Com a evolução tecnológica surgiram novas arquitecturas de sistemas de informação visando melhorar o seu desempenho e satisfação dos interesses das organizações. Empresas baseadas em transacções e geograficamente distribuídas podem recorrer à arquitectura *thin client/server*, para melhorarem os custos de administração dos seus equipamentos, segurança e desempenho.

Capítulo

4 Arquitecturas actual e proposta

4.1 Introdução

A arquitectura de sistemas de informação é uma abordagem metodológica que garante o alinhamento entre o negócio e as tecnologias de informação que o suportam. Hoje em dia, as arquitecturas de sistemas de informação são alvo de grandes preocupações por parte dos responsáveis dos sistemas e tecnologias de informação, assumindo a sua aplicação e manutenção nas organizações uma atenção especial.

Neste capítulo pretende-se, apresentar a arquitectura actual implementada na EMOSE, na primeira secção, e, na secção seguinte, a arquitectura proposta como solução dos constrangimentos identificados e que estão associados à arquitectura actual.

4.2 Arquitectura actual implementada na EMOSE

A EMOSE possui actualmente uma arquitectura baseada em dois ambientes: *UNIX* e *Windows*. No ambiente *UNIX*, funciona a sua principal aplicação de negócios, o SIGES (Sistema Integrado de Gestão Empresarial de Seguros) cujo SGBD e a aplicação estão assentes no servidor IDS (*Informix Dynamic Server*) da IBM, que é uma plataforma orientada para transacções e com alta disponibilidade.

Os servidores *UNIX*, estão em *cluster*, e estão divididos em servidores de aplicação, cujo sistema operativo é *Linux*, e servidores de BD, com sistema operativo *Solaris*.

No ambiente *Windows* estão implementados, em modo cliente/servidor, todos os serviços complementares de suporte à produção como, por exemplo, o serviço de *e-mail*, *Internet*, *Extranet*, *Intranet* e aplicações para *desktop MS-Office*. A gestão do *datacenter* da EMOSE é centralizada, isto é, não possui servidores nas agências, o que simplifica a sua administração. Todos os dados da organização são armazenados no SAN (*Storage Area Network*).

Ao nível dos utilizadores, o sistema operativo é *Windows*, e para o acesso ao ambiente *UNIX*, está implementado o emulador de terminal SSH (*Secure Shell*), no *desktop* de cada utilizador. A figura abaixo ilustra a arquitectura actual da EMOSE:

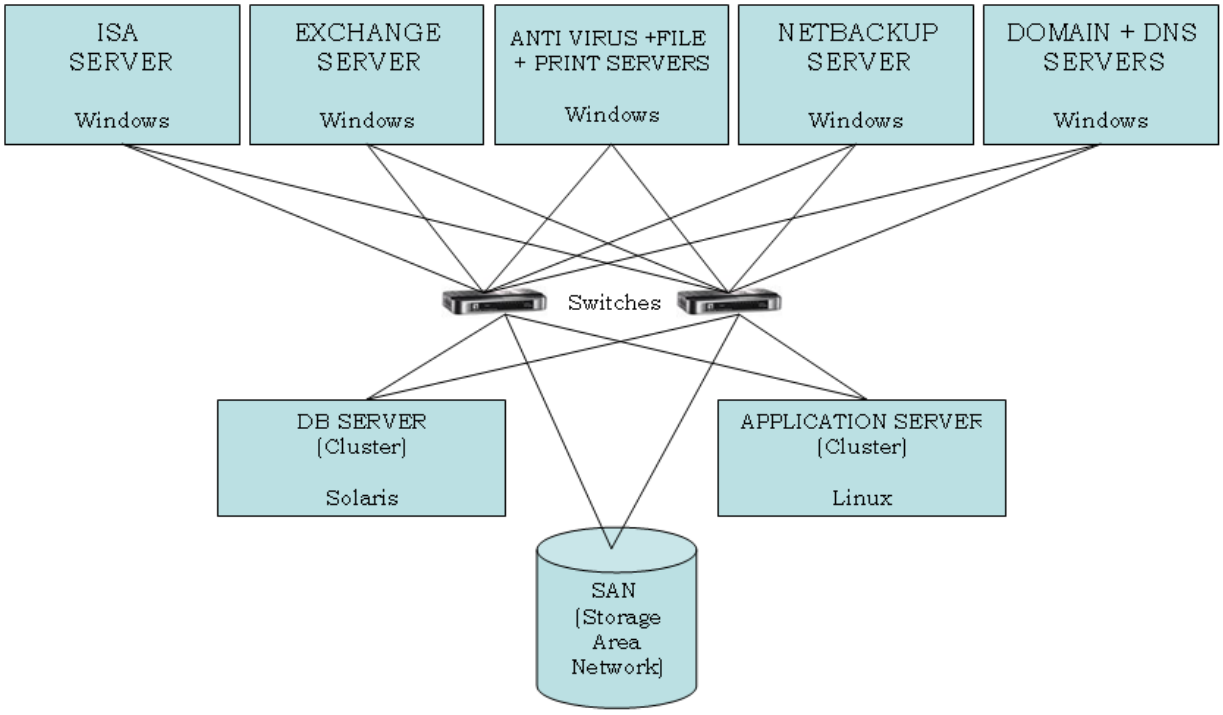

Figura 4.1: Arquitectura actual da EMOSE

Todo o sistema conta com cerca de 250 utilizadores, sendo que cerca de 80 (que corresponde a 32%) destes operam no SIGES e os restantes 170 estão nas aplicações de suporte

(que corresponde a 68%), como mostra a figura seguinte:

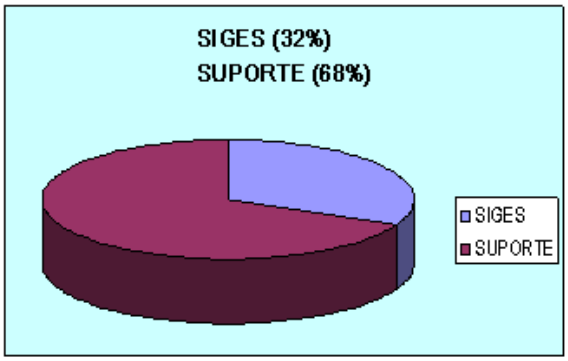

Figura 4.2: Distribuição global de utilizadores no sistema

Nas aplicações de suporte, fazem parte os cerca de 36 utilizadores das áreas administrativas e patrimoniais, como mostra a figura seguinte:

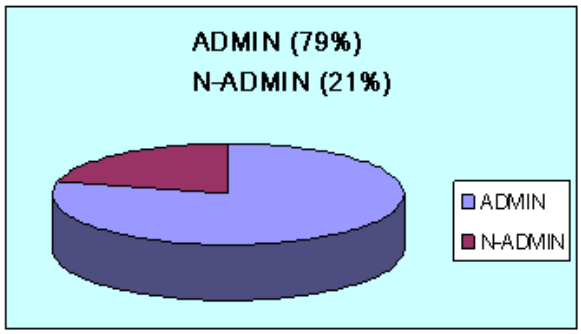

Figura 4.3: Distribuição de utilizadores de suporte

Todos estes utilizadores usam clientes gordos, no entanto a maior parte de utilizadores que usam o SIGES, não usam as tradicionais aplicações de *desktop*, apenas necessitam do SSH, pelo que não necessitam de ter terminais gordos, assim como os das áreas administrativas, cujo uso de computadores não é muito intenso.

Portanto, há necessidade de divisão de clientes em gordos e magros, de acordo com o gráfico seguinte:

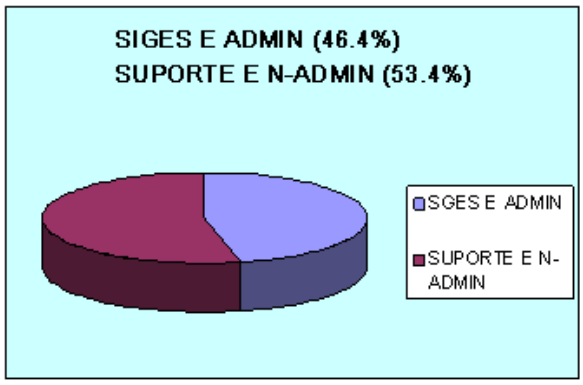

Figura 4.4: Divisão dos utilizadores do sistema da EMOSE

Os utilizadores que operam directamente com o SIGES (80) e os das áreas administrativas e patrimoniais (36) ocupam uma percentagem de 46,4% e os restantes ocupam 53,6%. À partida, vê-se que cerca de 46,4% de utilizadores não precisam de ser gordos. Esta percentagem pode aumentar, como veremos na secção a seguir.

4.3 Arquitectura proposta

Este trabalho propõe a implementação da arquitectura *thin client/server* que, para além dos componentes da arquitectura actual, serão acrescentados quatro servidores *citrix XOpen* em *clusters*. Estes servidores irão efectuar o processamento das aplicações desenvolvidas para plataformas *Windows* e disponibilizarão produtos de *MS-Office* e correio eletrônico aos utilizadores, no ambiente *Windows*, Cada servidor *citrix XOpen* suportará, no máximo, 70 utilizadores. Geralmente, estes servidores suportam no máximo 80 usuários. A arquitectura proposta é ilustrada na figura seguinte:

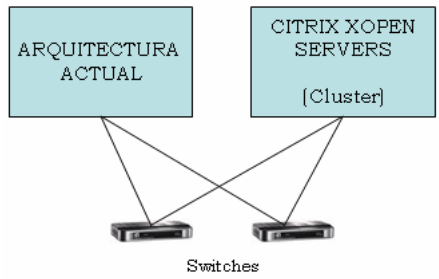

Figura 4.5: Arquitectura proposta

Nesta arquitectura, os utilizadores que operam directamente com o SIGES (80), assim como os das áreas administrativas e patrimoniais (36) serão convertidos para processamento baseado em servidor, podendo ser substituídos por *thin clients*, pois assim exigem as suas actividades diárias. Para além destes, cerca de 119 dos das áreas não administrativas, dos que operam nas aplicações de suporte, também serão convertidos para esse modelo de computação. Desta maneira, na nova arquitectura, serão convertidos 235 PCs (que corresponde a 94%) e teremos apenas 15 utilizadores (que corresponde a 6%) que não necessitam de ser convertidos, como mostra a figura seguinte:

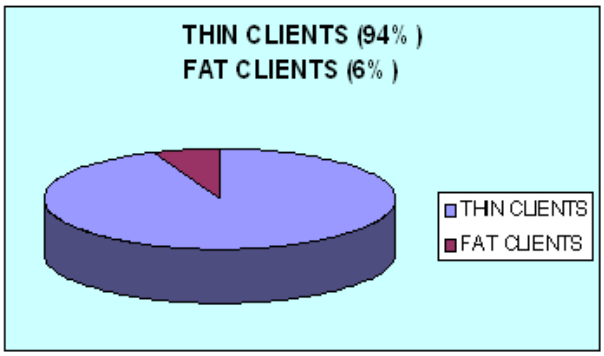

Figura 4.6: Utilizadores magros e gordos na nova arquitectura

Embora a implementação desta arquitectura requerer alto investimento inicial devido ao custo dos servidores, a longo prazo, irá permitir à organização:

- Tirar maior proveito da largura de banda disponível, pelo facto de não se efectuar nenhum processamento nos clientes, gerando menos tráfego na rede;
- Reduzir os custos de licenciamento de produtos, porque o licenciamento só é feito nos servidores, e os custos de aquisição ou *upgrade* de PCs, porque o baixo processamento nas máquinas clientes aumenta o seu tempo de vida;
- Reduzir os problemas relacionados com a manutenção, porque há poucas avarias, uma vez que não serão usados os discos duros;
- Facititar a administração, servindo-se da assistência remota oferecida pelo servidor *citrix XOpen*, reduzindo o esforço da equipa de TI;

- Aproveitar cerca de 94% dos PCs actualmente em uso, convertendo-os para o processamento baseado em servidor, usando os servidores *Citrix XOpen*.

Assim, para a implementação da arquitectura proposta, temos duas opções: a **utilização dos computadores existentes actualmente** e a **aquisição de *thin clients***.

4.3.1 Utilização dos computadores existentes actualmente

Esta opção passa pela utilização dos PCs existentes actualmente, convertendo-os para funcionarem no modelo de computação baseado em servidor, com os servidores *citrix XOpen*. Na EMOSE, os PCs actuais têm cerca de um ano de existência, os custos serão naturalmente mais baixos num período de dois anos, mas após este período, devido à depreciação que é de cerca de 20% por ano será necessário comprar novos PCs. A principal vantagem desta opção é que a sua implementação terá menos custos financeiros em relação à outra opção, uma vez que não serão comprados novos computadores, somente serão comprados os servidores *citrix XOpen*.

4.3.2 Aquisição de *thin clients*

Se se optar pela compra de *thin clients*, será necessário vender-se os cerca de 235 PCs actuais. O valor pelo qual foi comprado cada PC é de cerca de 850 USD. Estima-se que cada PC custe actualmente cerca de 680 USD (tirando o valor da depreciação correspondente ao primeiro ano). Este valor pode participar no valor a ser utilizado para a compra de *thin clients*. Cada *thin client* custa cerca de 800 USD. Neste caso será necessário acrescentar cerca de 18% do valor obtido na venda de PCs actuais para compra de *thin clients*. A principal vantagem na compra de *thin clients* reside no facto de que eles são mais duráveis, confiáveis e económicos em relação aos PCs normais e, para a sua compra não será necessário muito dinheiro.

Neste trabalho, propõe-se a utilização desta última opção, dadas as vantagens que apresenta em relação à primeira.

N.B. Esta análise não inclui o custo dos servidores, uma vez que eles serão comprados, independentemente da opção que for escolhida.

4.4 Resumo

Apresentaram-se neste capítulo as arquitecturas actual implementada na EMOSE e a proposta.

Viu-se que na arquitectura actual o sistema da EMOSE possui dois ambiemtes de computação com PCs gordos, mas que nem todos esses utilizadores precisam de ser gordos, tendo em conta as actividades diárias da empresa. Assim, propôs-se a implementação da arquitectura *thin client/server* com os servidores *citrix XOpen* que serão colocados no ambiente *Windows*, com a substituição da maior parte dos PCs actuais por *thin clients*.

Capítulo 5

Conclusões, recomendações e limitações do trabalho

5.1 Conclusões

Este trabalho foi desenvolvido com o intuito de apresentar uma implementação estratégica da arquitectura *thin client/server* para a EMOSE, que consiste na computação baseada em servidor. Nesta arquitectura pretende-se colocar o servidor *citrix XOpen* e solidificar o seu modelo de computação e tirar partido das vantagens de assistência remota que ele oferece. Mas primeiro, foi necessário através de entrevistas aos responsáveis do CPD da EMOSE identificar os constrangimentos que afligem esta organização. Para superar estes constrangimentos foi proposta uma conversão da arquitectura actual recorrendo a uma outra, que oferece tecnologias que permitirão melhores soluções à empresa. Foi feito um estudo sobre sistemas legados, sistemas baseados em transacções e arquitecturas de sistemas de informação, o que permitiu dar os conceitos chave. E de seguida desenvolveu-se a arquitectura específica proposta para a EMOSE. Analisou-se também uma estratégia sobre a integração dos servidores *citrix XOpen* em *cluster* na arquitectura actual. A partir deste trabalho concluiu-se que:

- Organizações baseadas em transacções e geograficamente distribuídas, como a EMOSE, podem ter uma maior redução dos custos de administração de recursos informáticos, maior segurança e desempenho, com recurso à arquitectura *thin client/server* comparativamente com as outras arquitecturas;
- O investimento tendo em conta a retoma do antigo equipamento, no caso da EMOSE,

é só cerca de 18%, na aquisição de *thin clients* e a EMOSE não muito necessita de usar clientes gordos;

- Para a implementação da arquitectura *thin client/server* é melhor a opção de aquisição de *thin clients* pois, a longo prazo, irá proporcionar mais benefícios económicos e operacionais à empresa do que a manutenção dos PCs actuais;

- A arquitetura *thin client/server* pode ser híbrida, de modo a tirar proveito dos benefícios das arquiteturas centralizada e descentralizada;

- A centralização do sistema num *datacenter*, característico na arquitectura *thin client/server*, permite que o tráfego de rede fique confinado dentro do seu *datacenter*, assim os investimentos em infra-estrutura duma rede serão menores do que capacitar todo o parque para ter uma rede enorme;

- Através das facilidades de administração remota oferecidas pelo servidor *citrix XOpen*, reduzir-se-à o esforço da equipe de TI, ao mesmo tempo em que a organização terá condições de reduzir os custos operacionais e aumentar o controle e disponibilidade da BD.

5.2 Recomendações

Para que a implementação das estratégias da arquitectura *thin client/server* contribua para a solução dos problemas da EMOSE assim como de outras empresas com as mesmas características, recomenda-se que:

- Ao conceberem uma arquitectura de sistema de informação têm que decidir sobre que partes da tarefa devem ser executadas no cliente e quais no servidor pois esta decisão pode afectar o custo de clientes e servidores, a robustez e a segurança do sistema e a sua flexibilidade.

- Empresas que dependem de sistemas legados devem tomar uma decisão sobre como obter o melhor retorno do seu investimento, ao avaliarem os seus sistemas, tendo em

conta que qualquer estratégia escolhida para essa avaliação só pode ser efectiva se os planos envolvidos forem satisfatórios aos interessados no processo.

- Seja feita uma avaliação da aplicação principal do negócio da EMOSE (o SIGES), para determinar se será mantida, transformada ou substituída, visando melhorar o seu desempenho.

- Seja implementada a arquitectura proposta na EMOSE, sejam efectuados testes desta arquitectura e formação dos usuários, para a concretização da fase da implantação e testes.

5.3 Limitações do trabalho

- Não foi possível a realização da última fase (implantação e testes), do modelo em cascata, devido à falta de equipamento.

- A falta de disponibilidade imediata dos técnicos do CPD da EMOSE dificultou a recolha de dados em tempo programado.

- Não foi possível fazer uma análise económica mais detalhada da arquitectura proposta por falta de dados sólidos sobre os custos associados à gestão de equipamento.

- A arquitectura proposta é uma tecnologia nova. Isso limitou a colheita de experiência da sua utilização por parte de empresas ou indivíduos que a possam conhecer. No entanto, uma empresa que a utiliza (o Banco de Moçambique) foi consultada.

Bibliografia

Camargo, A., Khouri, L. H. E. & Giarola, P. C. (2005), 'O uso de sistemas colaborativos na gestão de projetos'. [Online; Acessado em Janeiro de 2010].
URL: *http://imasters.uol.com.br/artigo/5068/webmarketing/sistemas-colaborativos-proprietarios-e-baseados-em-software-livre/*

Carmona, T. & Hexsel, R. A. (2005), *Redes*, Digerati Books.

CarrBy, M. & Bair, B. (2007), 'Using thin client technology to offset costs', *Campus Technology* .

CitrixMetaframeXp (2008), 'Metaframe – metaframe secure access manager'. [Online; Acessado em Janeiro de 2010].
URL: *http://www.polo-informatica.com.br/pages/metaframe.htm*

Comer, D. E. (1995), *Internetworking with TCP/IP: Principles, Protocols and Architecture*, Vol. 1, Prentice-Hall.

Coulouris, G., Dollimore, J. & Kindberg, T. (2001), *Distriduted Systems*, Addison-Wesley.

Elliman, T. & Coakes, E. (1999), 'Focus issue on legacy information systems and business process change: The role of stakeholders in managing change', *Communications Of Association For Information Systems* **2**(4), 2–31.

EMOSE (2007), 'Historial – seguro com garantia'. [Online; Acessado em Fevereiro de 2010].
URL: *http://www.emose.co.mz/por/a-emose/historial*

Huwo, T. K. (2009), 'Thin client, meet the mobile future', *Building Digital Libraries* pp. 22–24.

ITCENTER (2008), 'Thin client – terminal services'. [Online; Acessado em Setembro de 2008].
URL: *http://www.itcenter.com.pt/terminal-service.php*

Kim, Y.-G. & Everest, G. (1994), 'Building an is architecture', *Information & Management* **26**, 1–11.

Lopes, F., Morais, M. & Carvalho, A. (2005), *Desenvolvimento de Sistemas de Informação*, FCA, Lisboa.

Marques, J. A. & Guedes, P. (1999), *Tecnologia de Sistemas Distribuídos*, 2 edn, FCA.

Pereira, J. L. (1998), *Tecnologia de Bases de Dados*, 3 edn, FCA.

Pfleeger, S. L. (2004), *Engenharia de Software*, Prentice Hall, São Paulo.

Pinheiro, J. M. S. (2004), 'Computação baseada em servidor – projeto de redes'. [Online; Acessado em Agosto de 2008].
URL: *http://www.projetoderedes.com.br*

Ribeiro, N. V. (1998), 'Modelo cliente/servidor'. [Online; Acessado em Novembro de 2009].
URL: *http://ltodi.est.ips.pt/nribeiro/Lecturing/CRC-01-02/CRC-Client-Server-Architecture.ppt#265,7,Fat Client vs. Thin Client*

SAMURAI (2008), 'Thin clients – samurai projectos especiais'. [Online; Acessado em Fevereiro de 2010].
URL: *http://www.samurai.com.br/thinclients*

Sommerville, I. (2003), *Engenharia de Software*, Addison Wesley, São Paulo.

Tanenbaum, A. (1992), *Modern Operating Systems*, 1 edn, Prentice Hall, Upper Saddle River.

VolHost (2009), 'Arquitectura cliente-servidor'. [Online; Acessado em Setembro de 2009].
URL: *http://www.criarweb.com/artigos/arquitetura-cliente-servidor.html*

Wikipedia (2009*a*), 'Oltp – wikipedia, a enciclopedia livre'. [Online; Acessado em Novembro de 2009].
URL: *http://pt.wikipedia.org/wiki/OLTP*

Wikipedia (2009*b*), 'Thin client – wikipedia, a enciclopedia livre'. [Online; Acessado em Fevereiro de 2010].
URL: *http://pt.wikipedia.org/wiki/Thin-client*

ANEXOS

Anexo I: Glossário

***Backbone*:** No contexto de redes de computadores, o backbone (ou espinha dorsal) designa o esquema de ligações centrais de um sistema mais amplo, tipicamente de elevado desempenho.

Bases de Dados: Conjunto de registros dispostos em estrutura regular que possibilita a reorganização dos mesmos e produção de informação.

***Browser*:** É um programa de computador que habilita seus usuários a interagirem com documentos virtuais da *Internet*, também conhecidos como páginas HTML, que estão hospedadas num sevidor *Web*.

Cliente gordo: É um computador pessoal usado da maneira tradicional, com aplicações armazenadas no computador e dados no servidor.

***Cluster*:** Um *cluster*, ou aglomerado de computadores, é formado por um conjunto de computadores, que utiliza um tipo especial de sistema operativo classificado como sistema distribuído. Muitas vezes é construído a partir de computadores convencionais (*Personal Computers*), os quais são ligados em rede e comunicam-se através do sistema, trabalhando como se fossem uma única máquina de grande porte.

***Cracker*:** É o termo usado para designar quem pratica a quebra (ou *cracking*) de um sistema de segurança, de forma ilegal ou sem ética.

***Datacenter*:** É o local onde são concentrados os computadores e sistemas confiáveis (*software*) responsáveis pelo processamento de dados de uma empresa ou organização.

Disponibilidade: É a característica que os sistemas informáticos têm de resistir a falhas de software e energia, cujo objectivo é manter os serviços disponibilizados o máximo de tempo possível.

Encriptar: Proteger informações através de códigos ou cifras.

Escalabilidade: Capacidade que determinado equipamento possui para receber implementações evitando que se torne obsoleto ou deixe de atender às necessidades do usuário. Podem incluir, por exemplo, aumento de quantidade de memória, troca de discos ou processador, entre outros.

***Hepl Desk*:** É um termo que designa o serviço de apoio a usuários para suporte e resolução de problemas técnicos em informática, telefonia e tecnologias de informação. Este apoio pode ser tanto dentro de uma empresa (profissionais que cuidam da manutenção de equipamentos e instalações dentro da empresa), quanto externamente (prestação de serviços à usuários).

***Hardware*:** É a parte física do computador, ou seja, é o conjunto de componentes electrônicos, circuitos integrados e placas, que se comunicam através de barramentos.

***Interface*:** O conceito de *Interface* se expressa pela presença de uma ou mais ferramentas para o uso e movimentação de qualquer sistema de informações, seja ele material ou virtual. Pode significar um circuito electrônico que controla a interligação entre dois dispositivos hardwares e os ajuda a trocar dados de maneira confiável.

***Login*:** *Login* ou Palavra-Senha é um conjunto de caracteres solicitado para os usuários que por algum motivo necessitam aceder a algum sistema computacional. Geralmente os sistemas computacionais solicitam um *login* e uma senha para a liberação do acesso.

***Macintosh*:** *Macintosh*, ou *Mac*, é o nome dos computadores pessoais fabricados e comercializados pela Apple Inc desde janeiro de 1984. O nome deriva de Macintosh, um

tipo de maçã apreciado por Jef Raskin.

***Mainframes*:** São computadores de grande porte, dedicados normalmente ao processamento de um volume grande de informações. Os *mainframes* são capazes de oferecer serviços de processamento a milhares de usuários através de milhares de terminais conectados directamente ou através de uma rede. (O termo *mainframe* se refere ao gabinete principal que alojava a unidade central de processamento nos primeiros computadores)

Plataforma: Tecnologia fundamental e básica para a construção de um sistema de computador.

Seguro: Contrato pelo qual uma das partes (segurador) se obriga para com a outra (segurado), mediante o pagamento de um prémio, a indemnizá-lo de prejuízo decorrente de riscos futuros, previstos no contrato.

Sincronização : É a gestão adequada de múltiplas linhas de execução ou processos concorrentes que acedem a um mesmo recurso limitado ou a uma porção de dados, situação conhecida como condição de corrida.

***Software*:** É uma sequência de instruções a serem seguidas e/ou executadas, na manipulação, redireccionamento ou modificação de um dado/informação ou acontecimento.

***Update*:** É um serviço Windows de actualização dos sistemas operativos da Microsoft (a partir do Windows 98) responsável por actualizar o sistema. A cada mês, a Microsoft lança um pacote de actualizações, que pode consistir em correcções de *bugs*, falhas de segurança e outras melhorias. Entretanto, se houver uma falha de segurança muito crítica, a correcção é expedida o mais brevemente possível.

***Winframe*:** É um servidor citrix de aplicação do Windows multiusuário com base no Windows NT 3.51 sob licença da Microsoft que oferece suporte a implantação de aplicaçõess da empresa usando uma arquitectura com cliente magro.

Anexo II: Telnet e SSH

Telnet é um serviço de administração remota que possibilita o acesso a sistemas e arquivos ainda em utilização, disponibilizado pela Microsoft em seus sistemas. É pouco seguro, pois permite que o tráfego das sessões transcorra abertamente pela rede, incluindo *logins* e senhas (Carmona & Hexsel 2005).

Numa rede interna *Windows* bem protegida, contudo, a implementação do telnet pode ser interessante, uma vez que a aplicação ocupa pouca memória RAM (*Random Access Memory*), e sua implementação é bem mais simples, do ponto de vista operacional, do que a de outras ferramentas similares (Carmona & Hexsel 2005).

Além do telnet, existem outros serviços de administração remota. Um desses serviços é o SSH (*Secure Shell*), que apresenta inúmeras vantagens em relação ao telnet, como segurança no sistema de comunicação de dados (todo tráfego é criptografado, em qualquer tipo de linha) e no sistema de compartilhamento de arquivos. No entanto, o SSH só constitui uma solução efectiva para administradores de sistemas *UNIX* (Carmona & Hexsel 2005).

Além dos recursos de administração remota, o SSH possibilita o acesso, em modo gráfico, a recursos e aplicações do sistema por meio de um atalho para o X, o servidor gráfico da maioria dos sistemas *UNIX* (Carmona & Hexsel 2005).

Na EMOSE encontra-se implementado actualmente o SSH.

Anexo III: Guião de entrevistas

Apresentam-se a seguir os pontos destacados durante as entrevistas efectuadas:

1. Actividade principal da EMOSE;

2. Gestão dos vários serviços da EMOSE;

3. Gestão dos seguros e das apólices;

4. Acesso ao SIGES por parte dos utilizadores;

5. O papel do CPD na EMOSE;

6. Arquitectura actual implementada no sistema;

7. Funcionamento do sistema actual;

8. Utilizadores do sistema;

9. Dificuldades enfrentadas no exercício das actividades de gestão do sistema e necessidade de vê-las superadas;

10. Como é que as avarias têm sido superadas;

11. Necessidade de avaliação do seu principal sistema;

12. Integração das várias partes que constituem o sistema;

13. O que se pode esperar da implementação da arquitectura *thin client/server*.