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

Aluno: José Edemar Palubiack Marinho

UMA ANÁLISE SOBRE REPLICAÇÃO DE DADOS EM AMBIENTES DE COMPUTAÇÃO MÓVEL

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

Orientador: Murilo Silva de Camargo

Florianópolis, 03/2002

UMA ANÁLISE SOBRE REPLICAÇÃO DE DADOS EM AMBIENTES DE COMPUTAÇÃO MÓVEIS

Aluno: José Edemar Palubiack Marinho

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

Prof. Dr. Fernando O. Gautier
(Coordenador do Curso)

Banca Examinadora

Prof. Dr. Murilo Silva de Camargo
(Orientador e Presidente da Banca)

Prof. Dr. Vitório Bruno Mazzola

Prof. Dr. Rosvelter João Coelho da Costa

"Minha condição humana me fascina. Conheço o limite de minha existência e ignoro por que estou nesta terra, mas às vezes o pressinto. Pela experiência cotidiana, concreta e intuitiva, eu me descubro vivo para alguns homens, porque o sorriso e a felicidade deles me condicionam inteiramente, mas ainda para outros que, por acaso, descobri terem emoções semelhantes às minhas."

Albert Einstein

OFERECIMENTO

A Deus, pelas oportunidades que me foram concedidas.
Aos meus Pais João e Judith Marinho, pelo esforço dedicado a minha formação.
Aos meus irmãos Frei Everaldo, Francisco, irmãs Nilce, Enerilda e cunhados Nestor e Kaplum, pelo apoio ao meu crescimento pessoal e intelectual.
E em especial para minha esposa Silvana, pela presteza, incentivo e compreensão dedicados durante todos estes anos em que acompanhou a minha trajetória.

AGRADECIMENTOS

Agradeço a todos aqueles que direta ou indiretamente contribuíram para a conclusão deste trabalho.
Aos professores do Curso de Pós-Graduação em Ciências da Computação da UFSC que me deram embasamento teórico durante este curso.
Ao meu Orientador Professor Murilo pelo auxilío e incentivo que me ajudaram a evoluir com este trabalho.
A minha esposa Silvana pela compreensão de minha ausência e pelo apoio oferecido nas horas mais difíceis.
E aos amigos que se fizeram presentes durante este período.

SUMÁRIO

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

RESUMO

Recentes avanços na tecnologia da informação sugerem que usuários tenham acesso a dados por intermédio de uma rede de computadores sem fios, o que nos leva a um novo paradigma na computação, estar conectado a qualquer momento em qualquer lugar. A chamada computação móvel, na qual usuários carregam dispositivos portáteis e possuem acesso a dados e informações independentemente destes estarem fixos ou em um comportamento de movimento, tem despertado bastante interesse em grandes empresas ligadas aos setores de informática e telecomunicação. Tais empresas apostam em uma crescente utilização da mobilidade computacional nos diversos setores da economia global. No trabalho a seguir será apresentada uma contextualização de ambientes computacionais móveis, e abordará o apoio que o SGBD (Sistema Gerenciador de Banco de Dados) oferece para um armazenamento persistente das informações em tais ambientes, analisando a habilidade de tratar a desconexão através de esquemas de replicação. Desta forma, o que se pretende, é analisar o esquema de integração entre instâncias de banco de dados. Para isto deverão ser estudadas estratégias de replicação que permitam esta operacionalização. Será realizado um estudo comparativo entre as principais soluções de Hardware e Software disponíveis para a aplicação em projetos de banco de dados distribuídos móveis, bem como, será apresentada a evolução da computação móvel nos últimos anos e os estudos que estão sendo realizados nesta área.

ABSTRACT

Recent advances in the Information Technology suggest that users have access to data using a wireless network computing, that have engendered a new paradigm: to be connected anytime at anywhere. It's called mobile computing, in which users carrying portable devices have access to data and information services regardless their physical location or movement behavior. This is an attractive market for big IT and telecommunication companies, that are investing in a growing utilization of mobile computing at various sectors of computing environments. This paper will present a composition of mobile computing environments, and will discuss how a DBMS (Data Base Management System) supports data storage, analyzing the capacity of managing the disconnection event throughout a replication structure. Thus, what is proposed is to analyze the integration schema between different instances of the database. For this response strategies will have to be studied that allow this functionality. A comparative study of the main solutions of the available hardware and software to apply in mobile distributed database projects will be involved, as the evolution of the mobile computing in the last years will be presented and the studies that are being carried through in this area.

INTRODUÇÃO

Hoje em dia, computadores portáteis têm se tornado comuns. Juntamente com as tecnologias de comunicação sem fio, estes computadores fornecem uma base para a computação móvel. À medida que a tecnologia de computação móvel amadurece, milhões de pessoas se tornarão usuários móveis, comunicando-se entre si e acessando vários recursos de informação utilizando computadores portáteis, assistentes digitais pessoais, equipamentos de rádio e celular. Em ambientes de negócios, a capacidade de acessar dados críticos independentemente da localização do usuário é ainda mais crucial. Dessa forma, faz-se necessária a disponibilidade das informações corporativas todo o tempo, fazendo com que as aplicações desenvolvidas para as estações de trabalho móveis não parem por falta de dados.

Por outro lado, bancos de dados e sistemas distribuídos estão cada vez mais aderindo a padrões estabelecidos, como CORBA[1], tendo como principal característica a interoperabilidade. Já existem grupos de pesquisa em código móvel, objetos e agentes. Este esforço deve ser ampliado para abordar mobilidade também em sistemas distribuídos e bancos de dados. Alguns tópicos relacionados são: modelos de dados para sistemas móveis de informação, arquiteturas móveis de software para sistemas de computação distribuída, tecnologias orientadas a objetos para computação móvel, gerenciamento de banco de dados móvel, suporte de sistemas operacionais para a computação móvel, migração de dados e processos, replicação e recuperação, agentes móveis inteligentes, entre outros.

Avanços da tecnologia em administração de redes de dados sem fio e aplicações em tecnologia de informação portátil geraram um novo paradigma na computação, chamado Computação Móvel. Neste contexto, é imperativo que usuários que transitam

---

[1] *Common Object Request Broker Architecture* – segundo a OMG (*Object Management Group* – www.omg.org) é um padrão que permite às aplicações comunicarem-se com outras não importando onde elas estão localizadas ou quem as projetou.

com dispositivos portáteis tenham acesso a serviços de informação por intermédio de uma infra-estrutura de rede compartilhada, embora o local físico possua um comportamento de movimento. Tal ambiente sugere transformações técnicas na área de acesso à informação. Técnicas tradicionais para acesso à informação estão baseadas em suposições de que o local dos nodos em sistemas distribuídos não mudam e as conexões entre os nodos também não mudam durante o processo computacional. Porém, num ambiente móvel, estas concepções são raramente válidas ou apropriadas.

1.1 PROBLEMA E MOTIVAÇÃO

A computação móvel é distinta da clássica computação fixa, devido a mobilidade dos usuários e dos seus computadores e o estreitamento dos recursos móveis sem fios limitados a vida da bateria. A mobilidade dos usuários permite dizer que eles podem se conectar de pontos de acesso diferentes através de ligações de dispositivos sem fios e podem querer ficar conectados enquanto se movimentam, apesar de uma possível desconexão intermitente. Redes sem fios são relativamente duas ou três vezes mais lentas que as redes convencionais (com fios). Além disso, as unidades móveis movidas a bateria sofrem limitações, dado o tempo de duração de carga da bateria. Esta limitação gera muito trabalho a ser feito para que a computação móvel esteja completamente habilitada. Isto tem sido uma constante, apesar dos recentes avanços na comunicação de dados através de uma estrutura de rede sem fios e o aparecimento de dispositivos computacionais de mão.

Houve uma recente proliferação de pesquisas focando assuntos de aplicação de sistemas móveis, especialmente para o propósito de acesso móvel à informação, onde a preocupação com a integridade, confiabilidade e disponibilidade da informação que transita em uma rede sem fio, tem sido constante. Os sistemas móveis requisitam informações conectando-se aleatoriamente aos repositórios de dados. Desta forma, os aplicativos disponíveis para computação móvel têm se apoiado cada vez mais em SGBD (Sistema Gerenciador de Banco de Dados), pois necessitam do armazenamento persistente das informações e do controle de integridade relacional dos dados. Uma abordagem que deve ser considerada está ligada ao fato dos sistemas móveis estarem

inaptos para realizar a conexão com o conjunto de dados em um SGBD central, estando desconectados. O conhecimento destas dificuldades nos sistemas móveis, e o desejo de estudar modelos transacionais para a atualização de dados na computação móvel, foram a motivação para o desenvolvimento deste trabalho, onde será abordado o assunto de réplica de dados em ambientes móveis, como uma forma de se tratar a desconexão e manter a integridade das informações.

1.2 OBJETIVOS

1.2.1 OBJETIVO PRINCIPAL

O objetivo principal deste trabalho é realizar uma análise das soluções propostas para utilização de bancos de dados consistentes juntamente com a computação móvel, dando ênfase a troca de informações exigida na aplicabilidade deste modelo.

1.2.2 OBJETIVOS ESPECÍFICOS

- Trabalhar sob o tema da mobilidade computacional em sua forma geral, abordando informações sobre equipamentos, infra-estrutura de rede e aplicações para esta tecnologia.
- Realizar um estudo sobre a arquitetura de bancos de dados distribuídos móveis, descrevendo os modelos de transações móveis e o resumo de alguns estudos relacionados ao tema.
- Analisar as características de bancos de dados móveis devidamente selecionados e estudados.

1.2.3 OUTROS OBJETIVOS

Outros objetivos deste trabalho estão relacionados a obtenção de uma clara visão do ambiente computacional móvel e a relação com a aplicação destes conceitos sobre o

conjunto de necessidades que surge diariamente, de acordo com a velocidade de transformação exigida pela globalização das informações.

1.3 LIMITE DO TRABALHO

Não faz parte do escopo deste trabalho o desenvolvimento de qualquer aplicação com o intuito de provar a utilização de algum modelo abordado durante o estudo.

1.4 ORGANIZAÇÃO DO TRABALHO

Para tratar o assunto em questão, o estudo será dividido em cinco capítulos: O capítulo I versará sobre a mobilidade computacional, fornecendo um breve histórico onde é tratada a evolução tecnológica, também será apresentada características do ambiente móvel, bem como os padrões adotados neste sistema; no capítulo II serão apresentados modelos de distribuição de dados através de um sistema de gerenciamento de banco de dados; O capítulo III abordará o tema de distribuição de dados em ambiente móvel, demonstrando modelos e arquiteturas existentes, neste capítulo também será apresentado um resumo de quatro trabalhos realizados enfocando esse tema; No capítulo IV será desenvolvida uma explanação sobre a problemática da replicação de dados em ambientes móveis e sobre algumas propostas de solução utilizadas em tais ambientes; No capítulo V será analisada a questão da mobilidade em bancos de dados corporativos; e enfim, na conclusão desta pesquisa será realizada uma análise final sobre o tema de replicação de dados em ambientes móveis.

CAPÍTULO I

1 MOBILIDADE NA COMPUTAÇÃO

1.1 EVOLUÇÃO HISTÓRICA

Provavelmente o primeiro aparelho portátil era um pedaço de pedra ou argila com sinais para gravar informações numéricas, que de um certo modo, poderiam afetar a vida de seus usuários. Na verdade, esta informação proporcionava a "função" de contar através da criação de símbolos correspondentes a uma cabeça de gado. Isto teria sido muito útil para indivíduos cuja sociedade ainda não havia inventado o sistema de números. Para os indivíduos de tal estilo de vida, o fácil manuseio, portabilidade, durabilidade e confiança eram essenciais. Provavelmente os símbolos tinham que ser profundos o suficiente para que fossem detectados ao simples toque no aparelho. Uma vez utilizado, este aparelho teria que ser tão portátil que não interferisse em outra atividade, como assustar os predadores, e mais importante que tudo, escapar deles. Durabilidade teria sido importante, já que os usuários não tinham meios de proteger seus dispositivos das variações de temperatura, choques, etc. Para o usuário, este dispositivo poderia ter tido um papel importante para estabelecer credibilidade, respeito e responsabilidade perante a comunidade.

Quando a tecnologia matemática e escrita desenvolveram-se, a civilização progrediu para o papiro e caneta nanquim. Estes aparelhos eram portáteis e podiam expressar informações muito complexas. Os usuários levaram algum tempo para aprender a usá-los (ler e escrever). Até recentemente somente um número limitado de indivíduos estão aptos a usar a tecnologia. A caneta e o papel persistiram por vários anos e ainda é a tecnologia de informação portátil preferida pela maioria da população.

Dois outros aparelhos de informações portáteis, o relógio de bolso e o livro impresso, são invenções relativamente recentes que transformaram a sociedade. O relógio de pulso capacitou o nível da sincronia logística entre indivíduos, advinda de uma necessidade exigida pela industrialização. Os livros impressos também possuem uma certa preferência para o acesso a informações padronizadas em formato portátil.

Portanto, papel e lápis, livros impressos e o relógio de pulso têm sido os aparelhos de informação portátil dominantes desde a revolução industrial (HELAL et al, 1999).

A invenção da tecnologia do semicondutor no início dos anos 60 iniciou a transformação dos aparelhos portáteis. O primeiro aparelho eletrônico portátil mais adotado apareceu nos anos 70 em forma de calculadora. O desenvolvimento desses produtos começou na metade dos anos 60 junto com a exploração e desenvolvimento da tecnologia de transistores. Nos anos 70, havia muitas calculadoras grandes a preços acima de US$ 300. Até 1975, as calculadoras encolheram ao tamanho do bolso e diminuíram ao preço de US$ 20. Neste período, relógios digitais começaram a substituir os relógios mecânicos usados por muitos anos.

Até o início dos anos 80, as câmeras de vídeo portáteis venderam mais de um milhão de unidades em todo o mundo (HELAL et al, 1999) e os eletrônicos portáteis entraram de vez para o dia-a-dia do consumidor. Esta rapidez de penetração foi forçada pela necessidade de adquirir e guardar imagens em movimento. Este mercado foi mais acelerado pela introdução dos modelos 8 mm que foram largamente miniaturizados.

Organizadores pessoais (agendas eletrônicas de bolso), foram também introduzidos nestes moldes e fizeram mais sucesso no Japão, onde o uso de PCs (Computadores Pessoais) era algo em atraso. Na América do Norte, os PCs eram populares entre técnicos, mas em geral, desapontavam os indivíduos que tinham experimentado o ***Desktop***, já que havia pouca compatibilidade entre organizadores pessoais e ***Desktops***[2].

Até o fim dos anos 80, mais de 10 milhões de telefones celulares foram vendidos (HELAL et al, 1999) e tornaram-se uma necessidade para muitos e símbolo de status para outros. No início dos anos 90, mais de 1 milhão de ***Notebooks*** foram vendidos no mercado mundial (HELAL et al, 1999). Os primeiros modelos tinham LCDs (Liquid Cristal Display) monocromáticos. Esses sistemas eram muito adequados para o processamento de textos e planilhas, por isto foram adotados por profissionais viajantes. A transferência de dados era feita por discos magnéticos.

Vários produtores experimentaram os PDAs (***Personal Digital Assistants***) como conceito de produto. Eles tentaram cobrir a lacuna entre o organizador pessoal e o

Notebook. Esses produtos conciliaram a miniaturização dos organizadores e as funcionalidades dos ***Notebooks***. Mais tarde, eram desvalorizados devido a pouca capacidade de reconhecimento da escrita. O mais interessante é que esses produtos, ao invés de complementarem os ***Desktops*** ou ***Notebooks***, competiam com eles. Vários fabricantes tentaram acrescentar comunicação sem fio aos seus PDAs , mas a falta de integração com o ***Desktop*** e as limitações de banda na comunicação sem fio, desvalorizaram esses produtos. A infra-estrutura oferecida era somente com fio e a capacidade de transmissão de dados através do celular era baixa para produtos portáteis.

Os PDAs entraram em cena de vez a partir de 1993. A exuberância inicial foi ampla e severa. Agora, com força de novos produtos e de nova geração, esta mesma indústria está tentando outro ataque, desta vez com alvo na aplicação vertical específica. Os PDAs emergiram para a organização de pontos simples de dados, vulgarização e comunicação instantânea e um novo paradigma de operação usando interfaces gráficas e reconhecimento de escrita. A maioria dessas características não alcançou as expectativas dos consumidores, pois as aplicavam erroneamente e havia a incompatibilidade de estruturas. Também em 1993, a indústria de PCs introduziu sua mais recente linha de laptops que incluía capacidade computacional e armazenamento. Rivalizaram com os mais poderosos ***Desktops***, mesmo que a capacidade computacional e de armazenamento dobrasse a cada ano. O resultado foi a percepção da pouca capacidade de desenvolvimento e operação no ambiente do PDA. O custo era alto e a venda era baixa. Conseqüentemente, nos primeiros 2 anos, o volume vendas era tão baixo que nunca atingiram o custo benefício. Estava claro: consumidores demandavam utilidade e a primeira linha de PDAs simplesmente não era viável, pois somente um baixo percentual de aparelhos era realmente usado.

Do início dos anos 60 até os dias atuais, os avanços nos aparelhos de informação portáteis foram impressionantes. Dentro da limitação de um notebook portátil, o poder de computação e qualidade de imagem vêm crescendo continuamente. A comunicação sem fio via celular tornou-se comum. Os consumidores gravam milhões de horas de informação em vídeo todo ano usando filmadoras portáteis. Estes produtos levaram ao

---

[2] Computador de mesa utilizado em escritórios

desenvolvimento de novas tecnologias. A integração do silício desenvolveu-se dos discretos aparelhos de transistores para um único chip contendo mais de 6 milhões de transistores.

Hoje em dia, esta imagem mudou bastante. Os PDAs são vistos agora com ceticismo e suspeita. Mas ainda há um batalhão de produtos entrando nesta guerra de fabricação. A batalha está acirrada na tentativa de reduzir os modelos de aparelhos e, em alguns casos, os fabricantes estão tentando se distanciar do passado evitando o nome PDA, escolhendo nomes do tipo: organizador de bolso e gerente de informações pessoais (PIM).

Apesar da redução da variedade, as expectativas do consumidor com os PDAs continuam a crescer devido ao desempenho do ***Laptop***, que continua a duplicar todo ano. Este, juntamente com uma maior capacidade de resolução, aumentou as funções de multimídia, melhorou a ergonomia e mais aplicações impuseram um novo padrão aos PDAs. Uma visão detalhada na oferta atual de PDAs revela que eles estão crescendo além dos conceitos de PCs, utilizando os mesmos componentes e infra-estrutura de fabricação que otimizam o suporte de ***Desktops*** e laptops. A integração do silício, imagens, tamanho do componente, aplicações de software e densidades de substrato desta infra-estrutura levou o PDA para a escolha de uma direção: ou para um produto inteiramente funcional que é muito grande para ser prático, ou para um produto que atende as necessidades ergonômicas do paradigma, mas que limita funcionalidade e desempenho .

Apesar dos avanços tecnológicos, muitos desses aparelhos portáteis pareciam tão estáticos quanto um livro impresso. Para se conectar ao mundo lá fora, utilizavam um meio através de canal de voz com uma largura de banda pequena, cuja confiabilidade para transferência de dados era baixa. Há alguns anos a Internet apareceu criando novas expectativas sobre acesso à informação. Sem acesso móvel à informação os aparelhos portáteis não sobreviveriam. É possível que a rede de conexão sem fio possa ultrapassar a conexão com fio em termos de desempenho para o futuro. A melhor estratégia para desenvolver aparelhos de informação portáteis é criar produtos que proporcionam funções úteis e independentes como uma câmera eletrônica, ou que complementem as

plataformas de rede com fio. O mercado emergente de PDAs é uma grande descoberta em termos de habilidade complementar ao computador ***Desktop*** e a mobilidade computacional.

Hoje, o mercado de computadores portáteis começa a proporcionar uma maior mobilidade aos usuários destes equipamentos. A capacidade de armazenamento e performance tem acompanhado os ***Desktops*** e cada vez mais começam a ser usados no mercado corporativo como extensão do sistema de gestão na mão de seus profissionais. A seguir estão relacionados alguns equipamentos para computação móvel de última geração:

1.1.1 PALM PC (P/PC)

Nesta categoria temos equipamentos de mão sem teclado (veja a figura 1) e com dispositivo de reconhecimento de escrita. Estes equipamentos possuem sincronia com desktop para troca de informações e também podem realizar conexões através de modem, telefones celulares ou dispositivos de infravermelho.

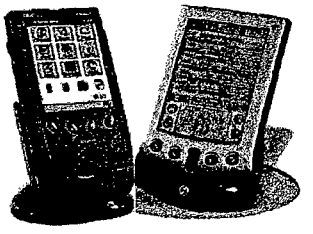


Figura 1 - Palm PC

1.1.2 HANDHELD PC (H/PC)

A categoria de ***Handheld PC*** é semelhante a categoria de Palm PC nos aspectos de funcionalidade e mobilidade. A diferença está na interação com teclado, além da caneta (veja a figura 2).

Figura 2 - Handheld PC

1.1.3 NOTEBOOKS

Os ***Notebooks*** têm feito muito sucesso como extensão portátil dos ***Desktops*** (veja a figura 3). A capacidade de armazenamento e processamento evoluiu muito nesta categoria, tanto que neste sentido podem ser comparados aos ***Desktops***.

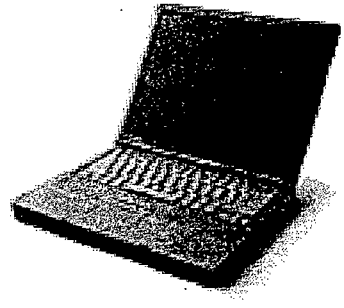


Figura 3 - Notebook

1.1.4 SUB-NOTEBOOKS

Os ***Sub-notebooks*** têm sido uma opção a mais para a portabilidade computacional. Mesmo com capacidade de processamento e armazenamento menor que os Notebooks,

não deixam a desejar na sua funcionalidade. Uma grande oferta de acessórios permite que esta categoria possa se equipar tanto quanto um Notebook, com a vantagem de possuir dimensões menores (veja a figura 4).

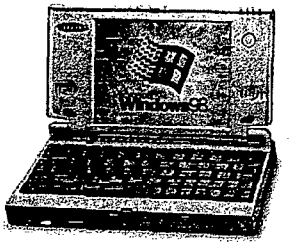


Figura 4 - Sub-Notebook

1.2 COMPARATIVO ENTRE DISPOSITIVOS DE ALTA MOBILIDADE

A seguir serão comparados alguns equipamentos que podem ser utilizados na computação móvel. O que eles possuem em comum é a facilidade de transporte destes equipamentos. A tabela 1 mostra as características técnicas de cada equipamento, enquanto a tabela 2 oferece a visualização destes equipamentos:

Modelo	Fabricante	Sistema Operacional	Tela Colorida	Peso (g)	Dimensões (cm)	Memória (Mb)	Processador (MHz)	Autonomia de Bateria
IPAQ H3650	Compaq	WindowsCE	Sim	182	8,39 x 13,0	32	206	12 horas
IPAQ H3670	Compaq	WindowsCE	Sim	182	8,39 x 13,0	64	206	12 horas
SPT 1800	Symbol	PalmOS	Não	288	9,2 x 17,8	4 / 8	33	8 horas
PPT 2800	Symbol	WindowsCE	Sim	288*	9,2 x 17,8	32 / 64	206	8 horas
PocketPC Revo	Psion	EPOC 32	Não	200	7,9 x 15,7	8	36	14 horas
PocketPC 5mx	Psion	EPOC 32	Não	354	9,0 x 17,0	16	36	8 horas
Jornada 545	HP	WindowsCE	Sim	260	8,0 x 13,5	16	206	8 horas
Jornada 720	HP	WindowsCE	Sim	510	9,5 x 18,9	32	206	9 horas
Palm IIIxe	Palm	PalmOS	Não	170	8,3 x 12,2	8	20	14 dias
Palm Vx	Palm	PalmOS	Não	100	7,78 x 11,29	8	20	14 dias

Tabela 1 – Equipamentos para Computação Móvel – Características

* Com leitor óptico

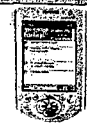
Modelo	Fabricante	Visualização
IPAQ H3650	Compaq	
IPAQ H3670	Compaq	
SPT 1800	Symbol	
PPT 2800	Symbol	
PocketPC Revo	Psion	
PocketPC 5mx	Psion	
Jornada 545	HP	
Jornada 720	HP	
Palm IIIxe	Palm	
Palm Vx	Palm	

Tabela 2 – Equipamentos para Computação Móvel – Visualização

1.3 AMBIENTE MÓVEL

Computação Móvel é o último estágio do desenvolvimento da computação pessoal. Em 1946, ***Illinois Bell Telephone Company*** introduziu um serviço de telefonia móvel que permitia usuários na direção de veículos, comunicarem-se com o sistema telefônico. Foi a primeira iniciativa de permitir comunicação bidirecional sem fio. Nos nossos dias temos a popularização do telefone celular, que garante mobilidade dos usuários do sistema telefônico (BATES & GREGORY, 1997).

São muitos os argumentos que nos levam a desenvolver um ambiente de computação móvel. Devemos pensar, em primeiro lugar, que nem todas as pessoas que necessitam do computador estão presentes no escritório. Vendedores são exemplos de trabalhadores com alta mobilidade que necessitam acessar bases de dados remotas e executam operações diversas como: emissão de pedidos; requisição de mercadorias; etc. Mesmo os habituais usuários de computadores de um escritório podem passar por momentos em que terão um difícil acesso ao escritório, como em viagens, cursos, congressos, etc.

Podemos enumerar diversas vantagens que um sistema móvel pode oferecer: conforto para utilização em qualquer ambiente; flexibilidade para utilização em diversas aplicações que exijam movimento; disponibilidade independente da localização do usuário. Existem também algumas exigências que devem ser obedecidas, como: a portabilidade facilitando o transporte; autonomia de energia para garantir o funcionamento onde não existe disponibilidade de energia; e desempenho comparável às estações fixas.

A exemplo dos computadores pessoais fixos, deseja-se garantir capacidades de comunicação aos computadores móveis por uma estrutura de rede ***Wireless*** (sem fio). Esta comunicação deve ser feita através de um sistema de ondas de rádio, utilizando-se de antenas para transmissão e recepção. Pela portabilidade do sistema, espera-se que as antenas não sejam muito grandes e nem mesmo muito potentes, para não ferir a autonomia de energia do sistema.

Um ambiente de computação móvel envolve computadores portáteis interligados em rede através de sistema de ondas de rádio. Para facilitar, podemos chamar de

Unidade Móvel (UM) ao elemento interligado à rede de computação móvel através de um computador portátil. As Unidades Móveis são ligadas em uma rede, que pode estar dividida em sub-redes, cada qual mantida por um grupo de antenas (infra-estrutura fixa), as quais são chamadas de Ponto de Acesso. Não é estritamente necessária a presença de Pontos de Acesso. Um ambiente de computação móvel pode ser desenvolvido de forma independente de infra-estrutura fixa, o que chamamos de redes AD-HOC, onde eles comunicam entre si diretamente através das antenas.

A princípio podemos utilizar o próprio sistema de telefonia celular para fazer a interconexão destes computadores em rede. Porém, algumas limitações estão presentes neste tipo de acesso, especialmente no que diz respeito à baixa velocidade de acesso obtida neste sistema. Para este sistema, a Unidade Móvel é um computador portátil utilizando um modem e um telefone celular para se conectar a um servidor de acesso remoto. As estações de rádio são as estações do sistemas de telefonia celular e todas as operações de mobilidade são gerenciadas pela camada física deste sistema, provida pelo sistema de telefonia celular.

Outras formas de comunicação podem também ser usadas, como por exemplo, a ***Wavelan***[3] que permite a utilização de taxas de transmissão mais satisfatórias, entre 1 e 2 Mbps (BATES & GREGORY, 1997).

A mobilidade, porém, sempre implica em algumas condições típicas do ambiente, que devem ser consideradas independente do sistema de acesso (MATEUS & LOUREIRO, 1998):

- Capacidade de comunicação limitada com largura de banda variável e alta taxa de erros;
- Autonomia de energia limitada por baterias com limite de consumo, de forma que deve-se dispender o mínimo de energia com processamento e dispositivos de apoio ao sistema;
- Limites físicos de hardware para garantia de portabilidade, limitando também o poder de processamento e dispositivos;

[3] É uma rede de comunicação sem fio particular de alta performance baseada em rádio transmissão.

- Problemas de roteamento de pacotes quando há variação da sub-rede onde está presente a UM;
- Perda temporária de comunicação quando o deslocamento entre as áreas são mantidas por diferentes estações de rádio e exige renegociação de características de acesso.

Para contornar estes problemas, diversas soluções têm sido propostas, algumas das quais já em operação, outras estão em pesquisa.

1.4 APLICAÇÕES PARA A COMPUTAÇÃO MÓVEL

Em muitas áreas de trabalho, o fato de estar em processo de movimento faz parte do negócio. Nessas áreas, o acesso à informação de forma móvel pode representar um diferencial competitivo. A seguir estão descritas algumas dessas áreas onde poderia-se utilizar a computação móvel:

- Corretores de imóveis: estes profissionais podem trabalhar em casa ou ao ar livre. Utilizando computadores móveis eles podem ser mais produtivos, à medida que conseguem obter informações atualizadas sobre imóveis que possuem para comercializar. Eles podem fornecer para os clientes uma avaliação imediata que considera casas específicas ou bairros, e facilitar a aprovação de negócios, realizando o cadastramento de um novo cliente, além da possibilidade de consultar informações sobre o cliente a partir do local onde se encontra.
- Serviços de emergência: a rapidez no recebimento de informações pode ser vital para salvar vidas quando serviços de emergência estão envolvidos. Podem receber informações sobre o paciente que será atendido (dados cadastrais, dados clínicos), além de outros detalhes do incidente ocorrido. Neste caso, a solicitação pode ser espalhada para diversas unidades móveis. Assim, aquela que se encontrar disponível e mais próxima do local poderá realizar um atendimento de forma mais eficaz.

- Em tribunais: tanto a Defesa como a Acusação podem ter acesso a um banco de dados legal para consultas durante a participação em um tribunal, além da possibilidade de ser auxiliado pelo seu escritório remotamente.

- Em companhias: gerentes podem utilizar computadores móveis para dar continuidade aos seus trabalhos mesmo estando distantes de suas mesas. Podem receber informações sobre novos produtos, tabelas de preços, informações sobre vendas e informações sobre clientes. Podem trabalhar de forma desconectada enquanto se movimentam, ou de forma conectada para o recebimento de mensagens e informações sobre sua agenda.

- Setores de Expedição: em ambientes onde o acesso à rede é limitado (a armazéns de fábrica, por exemplo), a utilização de computadores móveis com bancos de dados locais facilita o trabalho do selecionador[4]. Desta forma, as informações referentes aos despachos podem ser recebidas e enviadas de forma seletiva. Neste trabalho o selecionador ao invés de imprimir uma listagem dos ítens constantes em um determinado pedido, receberá esta informação em seu equipamento portátil..

- Automação Bancária: neste setor a computação móvel poderá contribuir para o atendimento de clientes em uma agência, através de um caixa móvel que tenha autonomia para realizar transações rápidas para pessoas que se encontram em uma fila de atendimento. Estas transações poderão ser: pré-atendimentos que facilitarão o trabalho final do caixa; ou atendimentos efetivos que contribuirão para a redução da fila. Outros trabalhos que poderiam ser realizados de forma móvel durante uma visita externa: é o pré-cadastramento de clientes; consultas aos dados do cliente; além de realizar operações bancárias no próprio local de visita.

- Coletas de uma Transportadora: através de uma central de chamadas, podem ser acionadas as unidades móveis (caminhões, furgões, motocicletas, etc) para a realização de novas coletas, mesmo que estas estejam em processo de

[4] Pessoa que seleciona os produtos a serem enviados a um determinado cliente

entrega ou coleta. Isto evita que estas unidades móveis retornem ao local de origem antes de realizar uma nova coleta.

- Correio Eletrônico: o uso de uma unidade móvel para enviar e receber mensagens através de um correio eletrônico pode ser recurso bastante útil para o indivíduo que necessita manter-se em contato constante com parceiros de negócio, clientes ou fornecedores.

- Seguradoras: as seguradoras possuem a necessidade de realizar visitas em dois casos: quando se avalia um bem para um novo contrato de seguro; ou quando acontece um sinistro. Em ambos os casos, a computação móvel pode ser utilizada, pois evitaria retrabalho na coleta (preenchimento de formulários, fotos, etc) e digitação das informações, uma vez que estes dados podem entrar diretamente no sistema através da coleta eletrônica.

- Comanda Eletrônica: Quem não gostaria de ser prontamente atendido quando solicita o fechamento das despesas em restaurante? Esta tarefa poderia ser realizada através da computação móvel, evitando perda de tempo para o cliente e acelerando o processo de venda nos restaurantes.

- Hospitais e Clínicas: neste caso a utilização de equipamentos móveis pode auxiliar no tratamento dos pacientes internados, uma vez que as enfermeiras e os médicos podem acompanhar o estado clínico de cada paciente, além de receberem informações quanto ao horário de medicação, solicitação de atendimentos ou alertas de emergências, independentemente do local em que estes profissionais estejam localizados.

1.5 REDE DE TELECOMUNICAÇÃO SEM FIO

O fato da comunicação entre um elemento móvel e uma Estação Rádio Base utilizar uma rede sem fio o leva aos seguintes problemas (MATEUS & LOUREIRO, 1998):

- A comunicação sem fio provê uma largura de banda menor e com uma latência maior do que a comunicação através de cabos (conectividade fraca e intermitente);
- A qualidade da conexão sem fio pode variar abruptamente. Por exemplo, devido a interferências, devido a distância para a ERB (Estação Rádio Base) e/ou devido ao compartilhamento de ERBs por vários elementos móveis (conectividade variável);
- Os serviços e aplicações distribuídas tradicionais dependem fortemente da rede, podendo inclusive ter sua execução interrompida na presença de falhas na mesma. Por exemplo, um sistema de arquivos distribuído pode ficar bloqueado esperando por outros servidores. Falhas na rede são comuns em computação móvel, já que a comunicação sem fio é muito susceptível a desconexões. Quanto mais autônomo for um computador móvel, maior será sua tolerância a estas desconexões freqüentes;
- Como é fácil acessar um link sem fio, aspectos de segurança tornam-se ainda mais importantes;
- Computadores móveis, ao contrário de computadores convencionais, precisam lidar com redes sem fio heterogêneas, à medida que se movem para diferentes lugares. É comum a necessidade da troca de interface devido a mobilidade. Por exemplo, interfaces que utilizam infravermelho não podem ser utilizadas em ambientes externos devido a luz do sol, gerando a necessidade de mudarmos para uma interface que utilize outra tecnologia como, por exemplo, rádio-freqüência ou microondas.

1.6 IEEE 802.11

O padrão IEEE 802.11, definido pelo IEEE (***Institute of Electrical & Electronics Engineers***) em 1997, prevê uma rede local sem fio, conhecida por *Wireless LAN,* em atendimento à padronização dos equipamentos já em uso para este tipo de rede. Estão

definidas neste padrão: a camada física; a subcamada de acesso ao meio (MAC) como qualquer padrão 802.X; e infra-estrutura de rede (veja a figura 5).

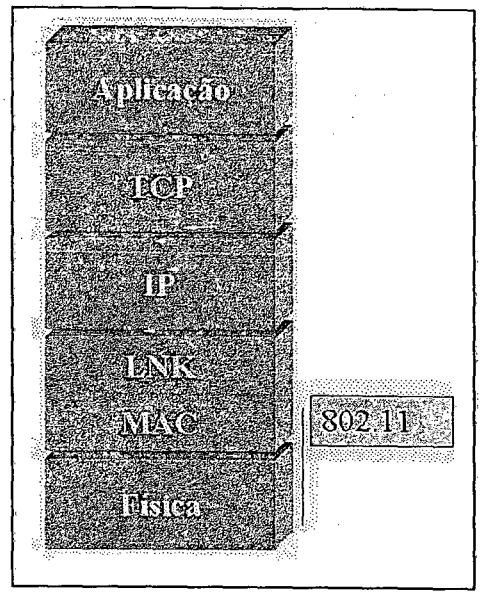


Figura 5 - Infra-estrutura de Rede Local sem Fio

Uma rede IEEE 802.11 é baseada na arquitetura celular, onde o sistema é subdividido em células. Cada célula, chamada Conjunto de Serviços Básicos ou *BSS* ***(Basic Service Set)***, é controlada por uma estação base, chamada "Ponto de Acesso".

Embora uma rede local sem fio possa ser formada por uma simples célula, com um único ponto de acesso e até mesmo sem nenhum ponto de acesso, a maioria das instalações será formada por muitas células, onde os pontos de acesso são conectados para o mesmo tipo de ***Backbone***[5], conhecido como sistema de distribuição. A interconectividade da rede local sem fio é garantida por uma rede 802 simples nas camadas superiores do modelo *OSI* ***(Open System Interconnection)***. A esta interconectividade damos o nome de Conjunto de Serviços Estendidos ou *ESS* ***(Extended Service Set)***. A figura 6 ilustra uma rede celular padrão IEEE 802.11.

Quando uma estação é ligada ou entra na área de cobertura de um ponto de acesso, temos o processo de associação. Ao se deslocar de uma região coberta de um

[5] "Backbone" é um provedor de serviço de acesso a uma rede de computadores

ponto de acesso para outro ponto de acesso, o processo realizado é de reassociação. Uma única célula pode cobrir a área de 5000 metros quadrados.

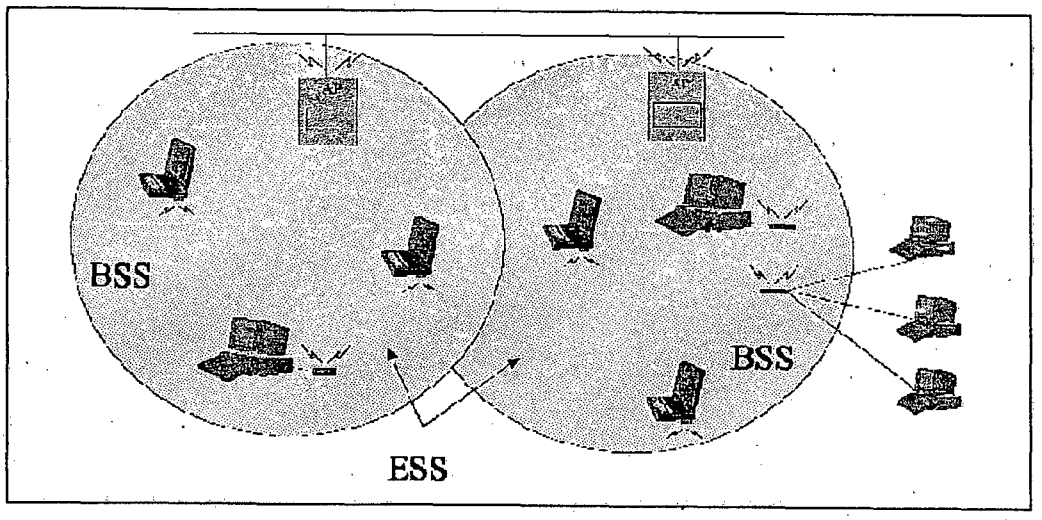


Figura 6 - Infra-estrutura de Rede 802.11

Na camada física temos três possibilidades de acesso:

- Infravermelho;
- Direct Sequenc Spread Spectrum (DSSS) a 2,4GHz com acesso direto, e
- Frequency Hopping Spread Spectrum a 2,4GHz com chaveamento de freqüência entre 79 canais de 1 MHz, com objetivo de restringir acesso aos dados.

Na camada MAC (***Medium Access Control***), o acesso ao meio é feito através do método CSMA/CA (***Carrier-sense Medium Access / Collision avoidance)***. CSMA é bem conhecido no mercado por ser usado em ***Ethernet***. Porém, nesta rede o método é conhecido por CSMA/CD (CD para ***Collision Detect)***.

CSMA trabalha da seguinte maneira: uma estação solicitada para transmitir escuta o meio. Se o meio está ocupado com a transmissão de outra estação, então a estação atrasa sua transmissão por um determinado período de tempo. Se o meio está livre, a estação transmite. No 802.11, através do uso de pacotes do tipo RTS (***Request to Send)***,

CTS(***Clear to Send***) e ACK(Confirmação), as estações são capazes de solicitar o uso do meio antes de transmitir, evitando assim colisões.

Não há limites teóricos para o número de usuários de um mesmo ponto de acesso. Na prática, cerca de 120 usuários dividem o uso de um ponto de acesso sem problemas (PERKINS, 1997).

1.7 INFRA-ESTRUTURA FIXA X AD-HOC

Estão previstas dentro do padrão IEEE 802.11 duas formas de conexão entre as redes de acesso móvel: as redes de infra-estrutura fixa; e as redes independentes ou AD-HOC.

1.7.1 REDES DE INFRA-ESTRUTURA FIXA

Neste tipo de rede o ponto de acesso é utilizado para comunicação entre as unidades móveis, de forma que uma unidade móvel sempre se comunica com outra somente através de um ponto de acesso (veja figura 7). Assim, a rede fixa dá suporte à mobilidade e auxilia em tarefas como roteamento, processamento distribuído, redução de tráfego, adaptabilidade, etc.

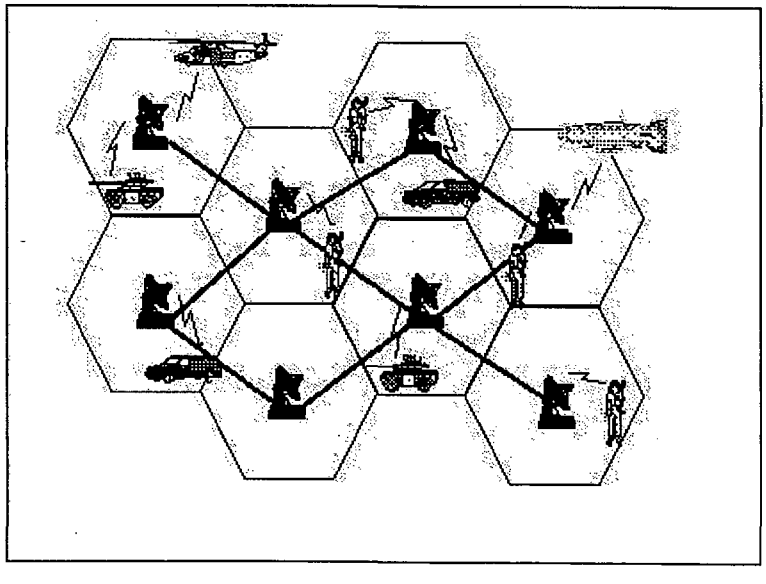


Figura 7 - Infra-estrutura de Rede Celular Fixa

1.7.2 REDES INDEPENDENTES OU AD-HOC

Nestas redes, não é prevista a existência de qualquer infra-estrutura fixa e, quando existe, é conhecida pela rede como qualquer outra unidade móvel. As unidades móveis comunicam-se diretamente através do meio dispensando infra-estruturas fixas, que podem encarecer o projeto (veja a figura 8). Se uma unidade deseja se comunicar com outra que não está dentro de seu alcance, ela o faz através de outras unidades móveis, que retransmitem os pacotes até que ele alcance seu destino. A maior dificuldade neste caso é o roteamento dos pacotes até o destino.

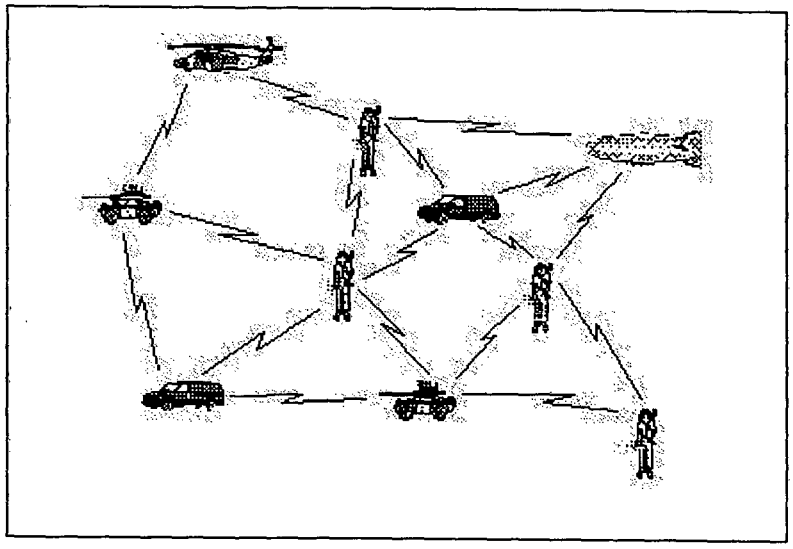


Figura 8 - Infra-estrutura de Rede AD-HOC

1.8 SEGURANÇA EM REDES SEM FIO

As estruturas de redes sem fio são usadas para interconectar quaisquer sistemas móveis. Esses sistemas, devido à possibilidade de sua movimentação, permitem criar topologias arbitrárias e temporárias. Neste sentido, são identificadas algumas características relevantes neste tipo de rede: os sistemas móveis podem ter recursos escassos; a informação propaga-se no ar em um espaço muito menos confinado do que um fio ou fibra das redes convencionais com cabeamento; como os sistemas podem se movimentar, as rotas estão constantemente sendo alteradas. Portanto, a segurança em uma infra-estrutura de rede móvel deve ser considerada em três níveis: físico; enlace e aplicação (REVISTA NETWORK COMPUTING BRASIL, 2001).

No nível físico as tecnologias de rede que dão suporte às redes sem fio utilizam técnicas especiais de transmissão de sinal para garantir confidencialidade, não interferência e não interceptação do sinal. Essas técnicas são denominadas:

- FH (Frequence Hoping): Utilizando-se desta técnica as estações vão alterando a freqüência de transmissão segundo uma seqüência pré-negociada. Como a freqüência é alterada, torna-se mais difícil interceptar o sinal.
- DS (Direct Sequence): Nesta técnica o sinal transmitido é multiplicado por um código equivalente a uma chave e ele só poderá ser recuperado pelos receptores que o conhecerem. O sinal gerado, após a multiplicação pelo código, tem baixa amplitude e é, portanto, de difícil detecção.

No caso do nível de enlace, normalmente empregam-se técnicas de criptografia simétricas baseadas em uma ou mais chaves, usualmente de 40 bits ou menos. A criptografia simétrica de 40 bits é considerada fraca. Contudo, considerando os recursos limitados dos equipamentos móveis, é normalmente inviável ter sistemas mais sofisticados. A solução paliativa para isso é empregar diversas chaves, que vão sendo trocadas periodicamente.

Por último, temos as aplicações. Neste caso, existe uma dificuldade adicional, decorrente de não podermos ter uma única autoridade certificadora. Como as redes são móveis, podem ser inseridos ou retirados os componentes, inclusive a estação que assume o papel de certificadora. Neste caso, então, cria-se o conceito de domínio de confiança, que é constituído de diversas estações que assumem a função de uma autoridade certificadora distribuída (REVISTA NETWORK COMPUTING BRASIL, 2001), podendo então, ser empregados os mesmos mecanismos e protocolos de segurança já existentes para aplicações em redes convencionais.

Portanto, o fato de termos uma rede sem fio interconectando sistemas com poucos recursos e aplicativos que podem entrar ou sair da rede, cria novos paradigmas de segurança. As soluções tradicionais baseadas em mecanismos de criptografia forte, focados em algoritmos assimétricos, e a utilização de autoridades certificadoras centralizadas não valem mais. Impera então a importância da segurança em nível físico e da criação do conceito de autoridade certificadora distribuída.

1.9 INTERNET MÓVEL

Após estabelecimento de mecanismos nas camadas inferiores que possibilitam o uso de computação móvel, devemos adaptar também as camadas superiores para permitir o seu uso de forma mais ampla. O primeiro passo é na camada de rede, para permitir o uso de computação móvel na Internet.

Espera-se que uma unidade móvel seja reconhecida na Internet independente da sub-rede que se encontre. Sabemos porém, que o endereço de uma estação depende de sua sub-rede. Esta sub-rede pode ser modificada com muita facilidade, de forma que para a configuração da sub-rede presente, não é cômodo que o endereço seja constantemente modificado. Um mecanismo transparente ao usuário deve ser utilizado com esta finalidade.

Para servidores móveis o problema é mais crítico ainda, pois deve ser reconhecido na Internet com um endereço fixo, independente da sub-rede em que se encontra. Espera-se também que a estação fixa em comunicação com a unidade móvel não veja diferenças entre esta unidade e outra estação qualquer. Para resolver estes problemas, foi proposto pelo IETF (***Internet Engineering Task Force***) o IP Móvel (PERKINS, 1997), que é uma adaptação do protocolo IP convencional para redes móveis.

1.10 ROTEAMENTO

Seja uma UM presente em uma rede, do tipo Internet, com seu endereço IP único capaz de garantir a capacidade de entrega correta de pacotes a ela destinada. Esta UM utiliza serviços de um roteador presente na sua sub-rede para que possa comunicar-se com elementos presentes em outras sub-redes.

Em um ambiente de computação móvel, uma sub-rede é mantida por um ou mais pontos de acesso que mantém comunicação com a unidade móvel. Para garantir total mobilidade, devemos prever a possibilidade de locomoção da unidade móvel entre áreas mantidas por diferentes pontos de acesso, possivelmente mantidas em diferentes sub-redes.

Essa possibilidade de locomoção entre diferentes sub-redes deve ser transparente para o usuário, de forma a não lhe causar incômodo. Sob o ponto de vista do protocolo de rede, isto implica na mudança do endereço de rede da unidade móvel, pois seus pacotes passam a ser enviados e recebidos por outra rota. Para tratamento desta abordagem para o protocolo IP, temos a opção do IP Móvel, proposto pelo IETF (PERKINS, 1997).

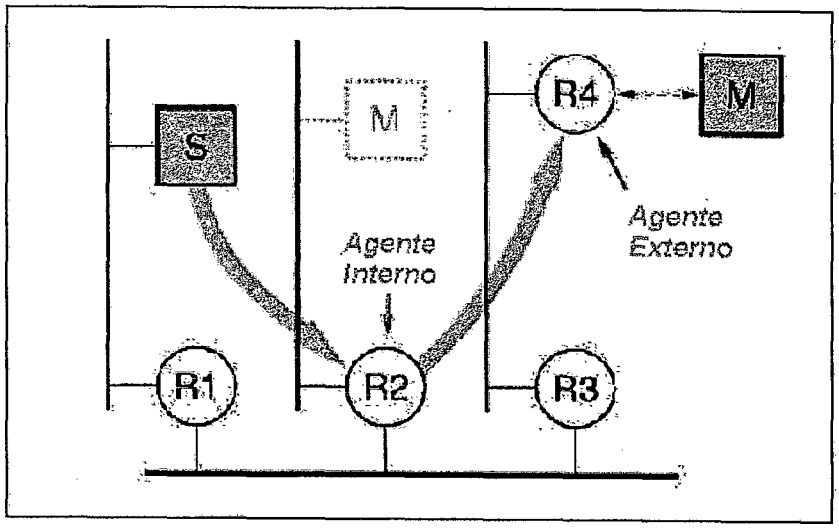


Figura 9 - Arquitetura para IP móvel

Uma unidade móvel presente em uma sub-rede diferente, chamada estrangeira, da sua sub-rede de origem é chamada unidade móvel visitante. O IP Móvel prevê basicamente a associação de um número IP dinâmico e provisório para a unidade móvel visitante, a partir de um servidor de suporte à mobilidade presente na sub-rede estrangeira.

Na sub-rede de origem da unidade móvel, deve estar presente um agente de mobilidade, configurado junto ao roteador da sub-rede de origem. Ao entrar em uma nova sub-rede, a unidade móvel recebe um novo endereço e o envia para o seu agente de mobilidade. Este agente de mobilidade é responsável por: escutar todos os pacotes que são enviados para a unidade móvel quando esta não estiver presente na sub-rede de origem; encapsular estes pacotes; e enviá-los para a sub-rede onde está presente a

unidade móvel neste momento, endereçados ao novo endereço da unidade móvel (veja a figura 9). Este processo de encapsulamento é baseado em ***IP-in-IP*** (PERKINS, 1996).

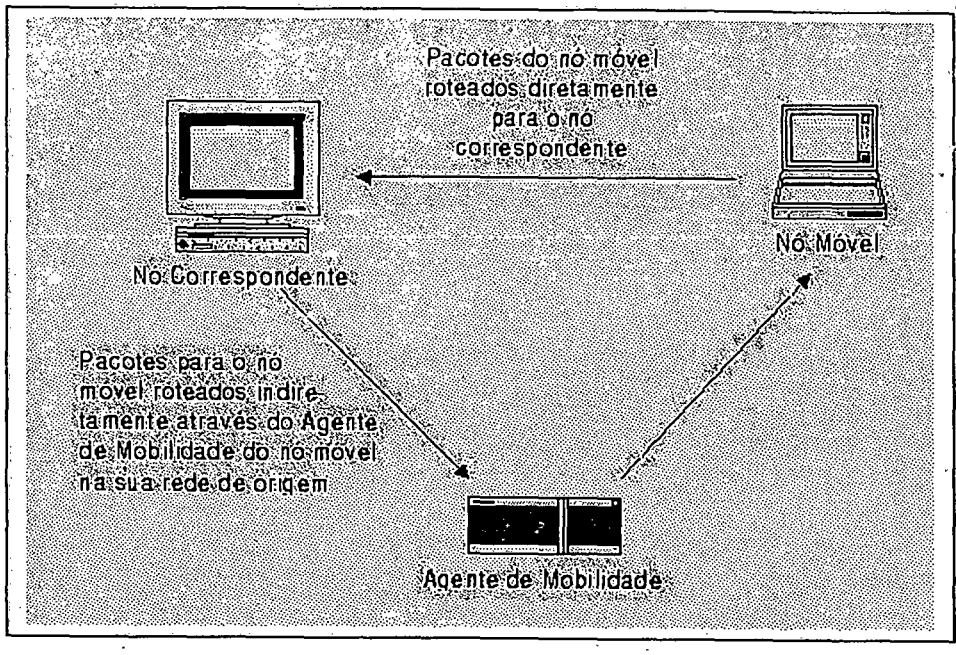


Figura 10 - Operação em um Esquema Móvel

Os pacotes a partir da unidade móvel para outro endereço qualquer da rede seguem diretamente para o destino, mas os pacotes com destino para a unidade móvel sempre devem passar pelo agente de mobilidade (veja a figura 10), por serem endereçados para a sub-rede de origem da unidade móvel.

O IP Móvel prevê ainda, caso a rede de comunicação suporte mobilidade, que ela seja capaz de identificar, através de uma primeira comunicação com o agente de mobilidade, o endereço IP utilizado pela unidade móvel na sub-rede estrangeira, e permita a comunicação do nó correspondente diretamente com a unidade móvel na sua nova sub-rede. Isto implica, porém, na manutenção de tabelas de conversão de endereços espalhadas pela rede, de forma a permitir o roteamento correto destes pacotes.

1.11 ADAPTABILIDADE

Espera-se que uma unidade móvel seja capaz de acessar a todos os recursos disponíveis em uma estação ligada a uma rede através de cabeamento, contornando todos os problemas deste ambiente. Lança-se mão, então, de técnicas para maior aproveitamento do canal de comunicação, da energia do sistema e do processamento, de forma a limitar as tarefas executadas na UM. Podemos verificar algumas abordagens interessantes:

- Alguns elementos, como estações fixas de suporte à mobilidade, podem aumentar consideravelmente a capacidade do sistema em termos de comunicação e processamento (MATEUS & LOUREIRO, 1998). A estas estações são reservadas determinadas tarefas como pré-processamento do tráfego de comunicação, com o objetivo de reduzir a largura de banda necessária e adaptação dos serviços disponíveis para as unidades móveis.
- Protocolos nas diversas camadas podem ser adaptados para resolver problemas específicos. Podemos citar IP móvel para garantir capacidades de roteamento em ambientes móveis, e outras adaptações, especialmente na camada de aplicação, com o objetivo de reduzir a largura de banda necessária ou aumentar a confiabilidade.

Devemos adaptar a aplicação, o protocolo utilizado ou mesmo a pilha de protocolos utilizados no ambiente de computação móvel para garantir os mesmos parâmetros de QoS (***Quality of Service*** - Qualidade de Serviço) que seriam garantidos em uma aplicação semelhante em uma rede fixa.

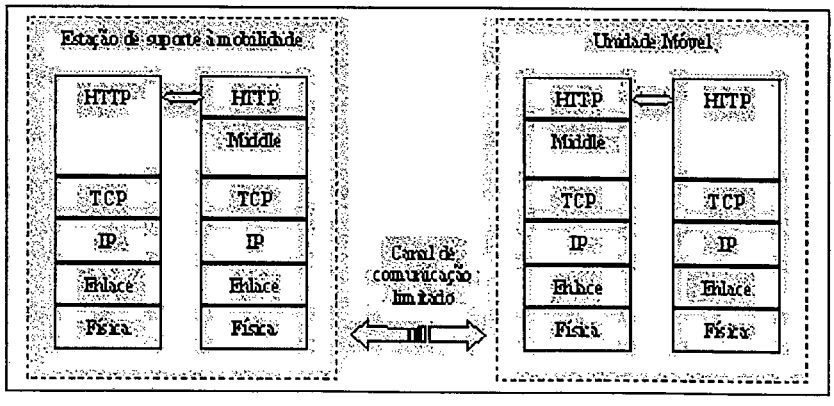


Figura 11 - Pilha de Protocolo HTTP

Podemos alterar a pilha de protocolos entre a UM e a estação fixa, de forma a criar uma camada intermediária que agregue as características necessárias. A figura 11 apresenta um exemplo desta abordagem em relação ao protocolo HTTP.

No exemplo mostrado, a camada ***Middle*** é responsável por fazer uma adaptação do protocolo HTTP para o meio, de forma a economizar a banda e possivelmente o processamento. O resultado pode ser transparente para o usuário ou não.

Uma possibilidade é o uso de compactação na camada ***Middle***. Isto reduz a banda de transmissão de forma transparente para o usuário. Esta função está prevista em camadas da pilha de protocolos do modelo OSI. Além da compactação, serviços de criptografia podem ser adicionados, de forma a garantir restrições de segurança à rede móvel, que está mais sujeita a invasões devido à ausência de privacidade do meio.

Outra abordagem poderia ser a realização de um tratamento prévio sobre serviços oferecidos ao usuário. Objetos HTTP poderiam receber um tratamento prévio com o objetivo de reduzir o tamanho na transmissão. Por exemplo, as figuras poderiam ser inicialmente transmitidas e mostradas em preto-e-branco e somente as figuras necessárias e requisitadas sob a ação do usuário seriam então retransmitidas coloridas. Este tratamento por sua vez não é transparente para o usuário.

Ao vislumbrar agora o serviço de Telefonia IP, desejamos adaptá-lo de forma a possibilitar seu uso em um ambiente de computação móvel. Vamos verificar as características deste serviço, para propor algumas adaptações:

- Tráfego de tempo real, de forma que um pacote só será útil se for recebido no tempo certo. Retransmissão é desnecessária, pois esta característica é garantida pelo protocolo de transporte RTP;
- Exigência de largura de banda constante. Uma queda de comunicação significa fim da conversação. Esta largura de banda é determinada pelo algoritmo de codificação da voz, compressão e supressão de silêncio;
- Serviço muito sensível a erros. A perda de um pacote pode implicar em ruídos na saída de voz;

- Altas exigências de QoS, especialmente no que diz respeito ao tempo de atraso dos pacotes.

O uso de diversos algoritmos de codificação de voz influi na largura de banda necessária para o serviço e na qualidade da reprodução da voz no outro extremo da conversação, de forma a ter possivelmente uma reprodução de voz mais mecanizada, com perda de alguns sons da fala.

A adaptabilidade a ser trabalhada para telefonia IP pode ser feita sobre estes algoritmos de codificação e decodificação, definidos pela ITU-T ***(International Telecommunication Union – Telecom Standardization)***, além da possibilidade de compactação e supressão de silêncio. Assim, a estação fixa pode alterar dinamicamente estes métodos, atuando diretamente na voz, de acordo com a disponibilidade de largura de banda.

Outras propostas estão sendo estudadas para garantir esta adaptação:

- Transmissão de pacotes maiores, contendo não só o conteúdo de voz atual, mas também frações dos pacotes anteriores, permitindo assim uma maior recuperação em caso de erros. Esta abordagem já foi utilizada pela *LUCENT* (LUCENT TECHNOLOGIES, 2001) em redes fixas e aumenta a largura de banda utilizada por este serviço;
- Retransmissão constante de pacotes, com o objetivo de substituir pacotes perdidos. Podemos observar que esta abordagem deve aumentar a utilização de largura de banda para este serviço, em tantas vezes quanto for feita a retransmissão.

1.12 MOBILIDADE EM SOFTWARE

O uso de computadores portáteis tem se tornado comum nos dias de hoje, mesmo assim a utilização de sistemas de informação em ambientes móveis tem evoluído a um passo lento. Isto se deve ao fato das várias dificuldades encontradas neste tipo de ambiente. A qualidade da conexão sem fio é quase sempre inferior à qualidade da

conexão na rede fixa. Adicionalmente, quando o computador portátil se encontra em movimento, a qualidade do canal de comunicação tende a variar consideravelmente. Como conseqüência, protocolos de comunicação, sistemas operacionais e aplicações para este tipo de equipamento começam a se preparar para lidar com estas flutuações na qualidade da conexão. Este esforço é realizado para minimizar o impacto de tais dificuldades na operacionalização dos sistemas por parte do usuário.

O uso de computadores móveis para tarefas diversas tais como edição, correio eletrônico e processamento remoto, tem se difundido rapidamente. De fato, o mercado de computadores móveis é o que mais cresce na indústria de equipamentos de computação (TANIN et al, 1996). Mesmo assim, a combinação de computadores móveis com sistemas de informação tem sido pouco explorada. Computadores móveis são usados quase sempre como sistemas autônomos desconectados da rede fixa. Se for observado, os aplicativos para sistemas de informação disponíveis para computadores móveis são basicamente os mesmos aplicativos disponíveis na rede fixa.

São muitos os benefícios que um sistema de informação móvel pode oferecer. Basta observar a explosiva demanda pela telefonia celular e serviços de mensagem eletrônica (***Paging***) (LYN, 1996). Os usuários podem se locomover livremente de um lugar para outro mantendo-se atualizado durante todo o tempo. Desta forma estes usuários poderiam participar de congressos, convenções, reuniões de negócios, ou até mesmo, realizar visitas a clientes e fornecedores, tendo acesso a uma base de dados remota, respondendo a correio eletrônico, participando de teleconferências e efetuando, normalmente, suas tarefas computacionais diárias, mesmo quando distantes de sua residência ou local de trabalho.

CAPÍTULO II

2 ARQUITETURA DO SGBD DISTRIBUÍDO

A tecnologia para desenvolvimento de sistemas distribuídos tem sido tópico de intenso estudo desde a década de oitenta, sendo que já existem diversos produtos comerciais de primeira geração disponíveis no mercado. Esta tendência baseia-se, principalmente, no fato da maioria das aplicações atuais serem naturalmente distribuídas, gerando assim uma grande demanda por sistemas nos quais os componentes estão repartidos e alocados em diferentes locais, sendo interligados por uma rede de computadores. De acordo com (MATTOSO, 1994), os Sistemas de Bancos de Dados Distribuídos (SBDD) apresentam uma série de vantagens em relação aos Sistemas de Gerência de Banco de Dados (SGBD) centralizados. Como exemplo, tem-se o aumento do desempenho, maior confiabilidade e disponibilidade dos dados, além do aproveitamento da natureza distribuída de algumas aplicações.

Contudo, para modelar e implementar sistemas distribuídos é necessário uma tecnologia mais complexa do que a exigida nos sistemas centralizados. Para que as vantagens destes sistemas sejam alcançadas, o SBDD deve prover algumas funcionalidades adicionais às presentes nos sistemas centralizados, como controle e armazenamento da distribuição e localização dos dados, bem como da garantia da consistência destes mesmos dados.

2.1 SISTEMA CLIENTE/SERVIDOR

As arquiteturas Cliente/Servidor apresentadas em (BERSON, 1992) vem sendo desenvolvidas para servir como modelo na construção de ambientes computacionais distribuídos. Resumidamente, o modelo de processamento Cliente/Servidor visa o

compartilhamento de recursos em um ambiente corporativo (BERSON, 1992), através da interação entre clientes (requisitores de recursos) e servidores (provedores de recursos).

2.1.1 PROCESSAMENTO CLIENTE/SERVIDOR

O modelo de processamento cliente/servidor surgiu como um processo de compartilhamento de periféricos típico de redes locais(LAN). Em um ambiente de compartilhamento de periféricos de uma LAN, um recurso comum pode ser compartilhado pelos computadores pessoais (PCs) conectados à rede. Um arquivo de um disco rígido ou uma impressora são exemplos comuns. Na terminologia de redes locais estes periféricos compartilhados são chamados servidores (um servidor de arquivos e um servidor de impressão no exemplo). O nome servidor foi utilizado porque estes periféricos recebem requisições de serviços dos PCs da rede. O modelo de processamento cliente/servidor é uma extensão natural do processo de compartilhamento de periféricos.

As redes cresceram e os servidores tornaram-se mais poderosos. Os serviços de impressão e de acesso a disco em grandes redes locais passaram a representar uma pequena fração do tempo de um a aplicação típica. Com isso surgiu a possibilidade de compartilhamento de parte da funcionalidade das aplicações entre várias máquinas. Neste modelo o processamento de uma aplicação pode ser dividido entre o cliente e o servidor. O cliente inicia o processo e o controla parcialmente. Tanto o cliente como o servidor trabalham em cooperação para a execução de uma aplicação.

Em resumo temos cinco requisitos necessários em uma arquitetura cliente/servidor:

- Sistema de comunicação robusto entre os clientes e os servidores;
- Interação cooperativa entre o cliente e o servidor, iniciada pelo cliente;
- Distribuição do processo entre o cliente e o servidor;
- Controle do servidor sobre os dados ou os serviços que um cliente pode pedir;
- Solução pelo servidor dos conflitos nos requisitos dos clientes.

A arquitetura cliente/servidor, através da distribuição dos componentes de uma aplicação, busca solucionar os problemas descritos nos modelos anteriores. Como existem algumas diferentes possibilidades de divisão, é necessário entender as diferentes arquiteturas e para que tipo de aplicações elas são mais apropriadas.

2.1.2 MODELO DE PROCESSAMENTO CLIENTE/SERVIDOR EM DUAS CAMADAS (TWO TIERED)

A maioria das configurações de cliente/servidor utilizadas atualmente são do modelo de duas camadas, consistindo em um cliente que requisita serviços de um servidor. No mínimo, a arquitetura cliente servidor assume que as camadas de serviços de apresentação e de lógica de apresentação estão no cliente. Isto soluciona o problema das interfaces inadequadas dos terminais, possibilitando uma gerência local de interface para fornecer uma interface gráfica amistosa. Vamos agora analisar as possíveis divisões das outras camadas.

Primeiramente, vamos colocar somente os serviços operacionais e de dados no servidor, e os serviços de apresentação, lógica de apresentação, lógica da aplicação e lógica dos dados no cliente. Esta arquitetura será chamada de "SGBD remoto". Devido à pequena carga sobre o servidor, este modelo é o de melhor escalabilidade. Contudo, para aplicações complexas, envolvendo muita interação com a base de dados, pode acarretar sobrecarga, tanto no tráfego da rede como no processamento do cliente. Todo o resultado de uma sentença SQL, no caso de SGBD relacionais, deve ser enviado ao cliente para ser processado, porque a decisão lógica, que poderia eliminar parte dos dados, reside somente no cliente (veja a figura 12). O modelo de SGBD remoto também aumenta o esforço de manutenção das aplicações, já que todo o código reside em cada cliente separadamente. Cada alteração de código obriga a alteração em todos os clientes.

Pode-se reduzir a carga na rede e nos clientes movendo-se a parte da lógica da aplicação para o servidor, com isso tem-se o modelo "lógico distribuído". O próximo passo é mover toda a lógica da aplicação para o servidor, deixando somente a lógica e o serviço de apresentação no cliente, diminuindo ainda mais a carga sobre o cliente, que é o modelo da "apresentação remota".

Na decisão de escolha entre esses três modelos, o projetista de sistemas deve determinar o local mais apropriado para colocar a lógica da aplicação. O mecanismo mais comum para se colocar a lógica da aplicação no servidor são os procedimentos armazenados (stored procedures), que são programas de código procedural com sentenças SQL embutidas para acessar a base de dados. Estes procedimentos aumentam a performance da aplicação por serem compilados, ao contrário das sentenças SQL enviadas pelo cliente, que não são compiladas. Outro aspecto é a carga na rede. Ao invés de enviar sentenças SQL do cliente para o servidor e retornar resultados intermediários, todo o processo e decisão pode ser feita no servidor com uma simples chamada a um procedimento armazenado. Esses procedimentos também reforçam a integridade da aplicação e da base de dados. Como os procedimentos armazenados são compartilhados, eles garantem a consistência das operações e cálculos. Isto resulta em maior produtividade, porque os procedimentos são escritos e depurados em um único local. Os procedimentos armazenados aumentam ainda a segurança da base de dados, porque os dados serão acessados indiretamente via procedimentos e não diretamente.

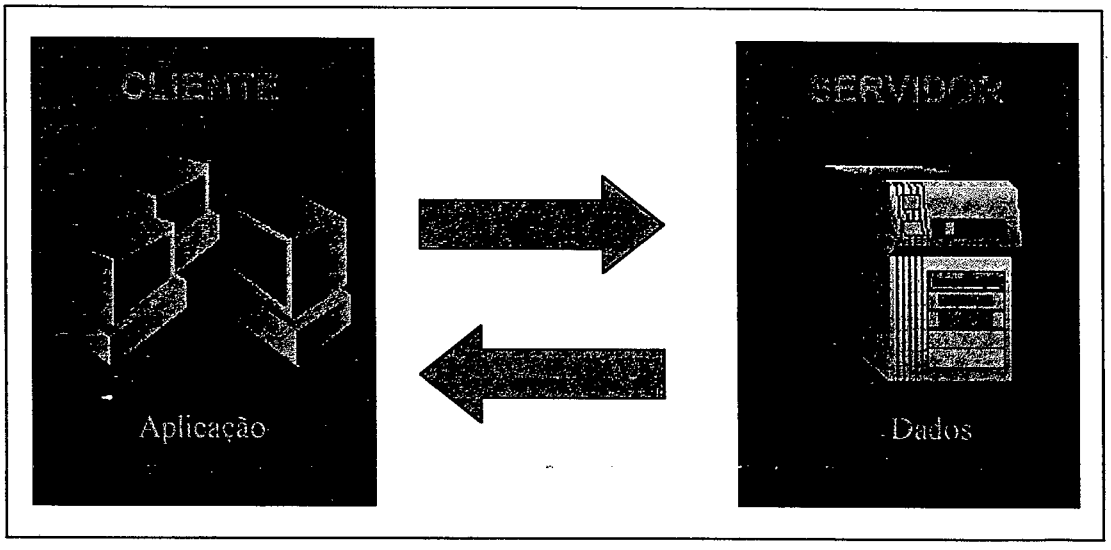


Figura 12 - Arquitetura Cliente/Servidor de duas camadas

O modelo com duas camadas enfrenta problemas em ambientes com aplicações complexas com muitos clientes, lógica intrincada, base de dados heterogênea e entradas heterogêneas na base de dados. O primeiro problema surge da carga de administrar um

grande número de clientes. Colocando a lógica da aplicação no cliente (junto com o serviço e lógica de apresentação), significa que para atualizar uma aplicação ou consertar um erro, deve-se fazê-lo para todos os clientes. Cada aplicação deverá ser enviada, instalada e testada para cada cliente. Levando-se em consideração as prováveis diferenças de configurações, isto culminará em grande esforço.

A movimentação da lógica da aplicação para fora do cliente minimiza o problema. O segundo problema surge da utilização de procedimentos armazenados para implementar complexas lógicas de aplicação. Infelizmente, as linguagens procedurais utilizadas para implementar esses procedimentos não são tão poderosas como as linguaguens procedurais padrão. Outro fator relevante é que toda implementação desses procedimentos armazenados utiliza uma linguagem proprietária, associada a um SGBD específico. Atualmente os procedimentos armazenados não são portáteis entre SGBDs. Isto restringe a flexibilidade de mudar de um SGBD para outro sem o ônus de se reescrever todas as aplicações, bem como conectar clientes em base de dados heterogêneas sem escrever os procedimentos comuns para cada SGBD.

O uso de procedimentos armazenados para implementar a lógica da aplicação também pode reduzir um dos maiores benefícios da arquitetura cliente/servidor, a escalabilidade. Quanto maior o número de procedimentos rodando no servidor, maior deve ser o seu poder de processamento. O excesso de código procedural no servidor acabará por reduzir o número de usuários que o ambiente pode acomodar. O modelo de duas camadas ainda possui outras falhas. Por exemplo, o fato do processamento em lote mesmo sendo executado exclusivamente no servidor só liberar o cliente após o fim do processo. Finalmente, existe o problema da manutenção da integridade dos dados em transações que englobem múltiplas base de dados com diferentes SGBDs. O protocolo normalmente utilizado por cada SGBD para garantir a integridade de cada base de dados (two phase commit) não garante a manutenção da integridade global dos dados nas transações que envolvem ambientes heterogêneos e distribuídos.

2.1.3 MODELO DE PROCESSAMENTO CLIENTE/SERVIDOR EM TRÊS CAMADAS (THREE TIERED)

O modelo de processamento cliente/servidor em três camadas visa solucionar as falhas do modelo em duas camadas. No modelo em três camadas, o cliente dedica-se à lógica e aos serviços de apresentação, e possui uma interface de programação de aplicações (API) para chamar a aplicação que reside em uma camada intermediária. Nesta camada intermediária está o servidor de aplicativos (veja a figura 13) executando a lógica da aplicação e dos dados que requisita os serviços de dados da terceira camada. Nesta última camada está o servidor de banco de dados, que é dedicado aos serviços de dados e operacionais.

O servidor de aplicativos pode gerenciar as diversas transações locais a cada base e garantir a integridade dos dados distribuídos, além de gerenciar consultas assíncronas para assegurar a confiabilidade das transações. Ele também centraliza a lógica das aplicações, facilitando a administração e a manutenção.

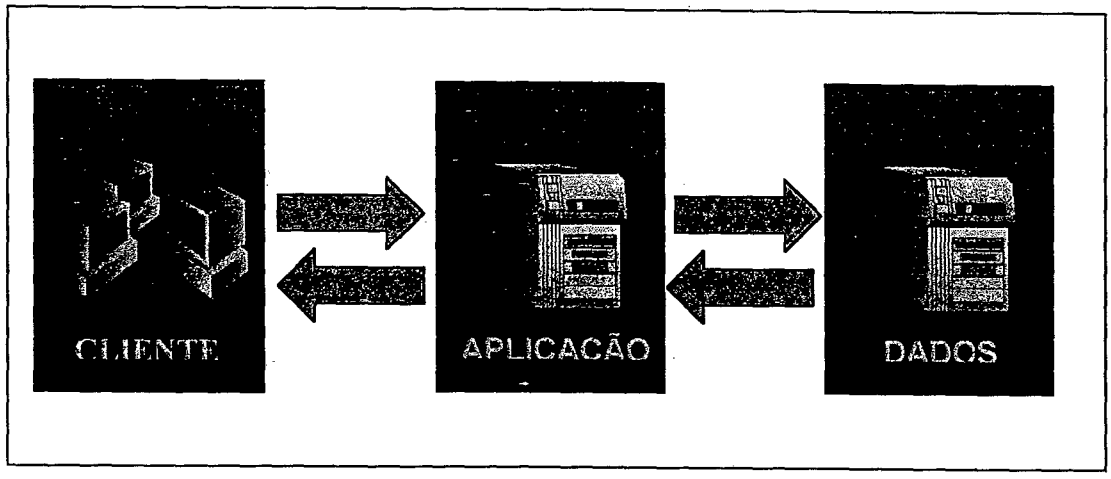


Figura 13 - Arquitetura Cliente/Servidor de três camadas

Devido a flexibilidade das fronteiras entre a lógica da apresentação, a lógica da aplicação e a lógica dos dados, a segunda poderá aparecer nas três camadas. Para identificar onde uma determinada função deve ser colocada, critérios como facilidade de desenvolvimento de teste e administração, escalabilidade dos servidores e performance quanto a processamento e carga na rede, devem ser considerados.

O modelo em três camadas possui outras vantagens no desenvolvimento de aplicações. Primeiro, o desenvolvimento de boas interfaces requer habilidades e treinamento específico. Nem todas as pessoas que implementam bem a lógica de aplicação implementariam bem a lógica da apresentação. Separando as plataformas de execução desses componentes, fica mais fácil organizar a implementação de forma a otimizar os recursos de pessoal da organização. Cabe ressaltar que boas linguagens para construir interfaces podem não o ser para implementar a lógica da aplicação.

2.1.4 COMPONENTES DE UM SISTEMA CLIENTE / SERVIDOR

Para desenvolver um ambiente baseado na arquitetura cliente/servidor são necessários:

- Computadores conectados fisicamente a uma rede;
- Protocolos de comunicação;
- Dispositivos de interligação de redes;
- Sistemas operacionais que se comunicam entre si;
- Ferramentas para desenvolvimento de aplicativos.

As redes que têm a capacidade de suportar aplicações cliente/servidor são as Redes Locais (LANs). O Institute of Electrical and Electronics Engineers (IEEE) define LAN como um sistema de comunicação de dados que permite a comunicação direta de dispositivos independentes entre si, dentro de uma área geográfica reduzida utilizando canais físicos com taxa de transferência moderada.

Protocolos de comunicação existem para permitir a transmissão de dados entre pontos da rede. O problema é que os diferentes sistemas existentes em uma rede possuem diferentes protocolos de comunicação. Dispositivos de interligação de redes como roteadores, ***Bridges***[6] e ***Gateways***[7] permitem a transmissão de dados entre redes de

---

[6] Dispositivo para interligação de duas redes de comunicação independentes.
[7] Dispositivo que restringe os pedidos de acesso de uma rede de comunicação, sejam estes pedidos de entrada ou saída.

acordo com o definido pelos diferentes protocolos. Para implementar a arquitetura cliente/servidor é necessário haver comunicação entre os diferentes sistemas operacionais dos clientes e dos servidores. Padrões para sistemas operacionais começaram a ser aceitos há pouco tempo. UNIX e POSIX são dois dos mais aceitos atualmente. Porém, até para esses "padrões" existem diferentes versões. O Ambiente de Computação Distribuída (DCE) da *Open Software Foundation's* (OSF) foi desenvolvido para gerenciar os múltiplos sistemas operacionais de um ambiente de processamento distribuído. O DCE fornece serviços que ficam disponíveis a todos os pontos da rede, permitindo aplicações acessarem programas e recursos de qualquer local da rede.

As ferramentas para desenvolvimento de aplicativos permitem acesso à base de dados e compreendem APIs, ferramentas de consulta e linguagens de programação. Atualmente a maioria dessas ferramentas permitem o desenvolvimento de sistemas baseado no modelo cliente/servidor de duas camadas com a aplicação localizada no cliente. Essas ferramentas, no geral, são baseadas em banco de dados relacionais e são desenvolvidas para um SGBD específico.

2.2 MODELO DO SGBD DISTRIBUÍDO

De acordo com (ÖZSU & VALDURIEZ, 1994), um sistema de banco de dados distribuído (SBDD) é uma coleção de múltiplos e logicamente interrelacionados bancos de dados, distribuídos sobre uma rede de computadores (veja a figura 14). Em consequência, um sistema de gerência de banco de dados distribuídos (SGBDD), é um sistema de software que executa a gerência do SBDD e torna a distribuição transparente para os usuários. Analisando estas definições temos que:

- Os dados estão armazenados em nós da rede. Considera-se que cada nó consiste logicamente de um único processador. Mesmo que algum nó seja dotado de multiprocessamento, o SGBDD não se preocupa com o armazenamento e gerência de dados nesta máquina paralela;
- Os processadores dos nós estão interconectados por uma rede de computadores, assim não estão em uma configuração de multiprocessadores.

Com isso, a interconexão entre processadores é fraca, cada qual possui seu próprio sistema operacional e opera independentemente;

- SBDD é um banco de dados e não uma coleção de arquivos que podem ser armazenados individualmente em cada nó de uma rede de computadores;
- O sistema possui toda a funcionalidade de sistema de gerência de banco de dados (SGBD).

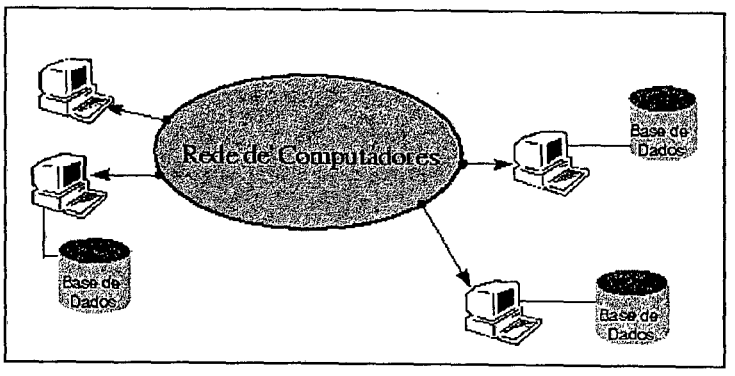


Figura 14 - Arquitetura de um Sistema Distribuído

Em (ÖZSU & VALDURIEZ, 1999) é apresentada uma classificação das alternativas para o projeto de sistemas distribuídos, levando-se em consideração três dimensões (veja a figura 15): autonomia, distribuição e heterogeneidade:

- Autonomia: diz respeito a distribuição do controle e indica o grau de independência com o qual cada SGBD pode operar. Esta dimensão pode ser entendida como a capacidade que cada SGBD possui de manter o controle sobre os seus dados e sobre o processamento dos comandos que recebe (SILVA, 1994). Três tipos de autonomia podem ser considerados: integração forte, semi-autonomia e autonomia total. Em sistemas de integração forte, uma única imagem de todo o banco de dados está disponível para os usuários que desejam compartilhar informações que residam em múltiplas bases. Sistemas semi-autônomos consistem de SGBDs que podem operar independentemente, mas decidiram fazer parte de uma federação para tornar os seus dados locais compartilháveis. Em sistemas com autonomia total,

contudo, os componentes individuais são SGBDs isolados que não se comunicam entre si;

- Distribuição: é a dimensão que classifica os dados. Estes podem estar distribuídos fisicamente pelos nós da rede ou estarem armazenados em um único nó;
- Heterogeneidade: pode ocorrer de várias formas em ambientes distribuídos, variando desde heterogeneidades de hardware e diferenças de protocolos de comunicação, até diferenças entre modelos de SGBDs.

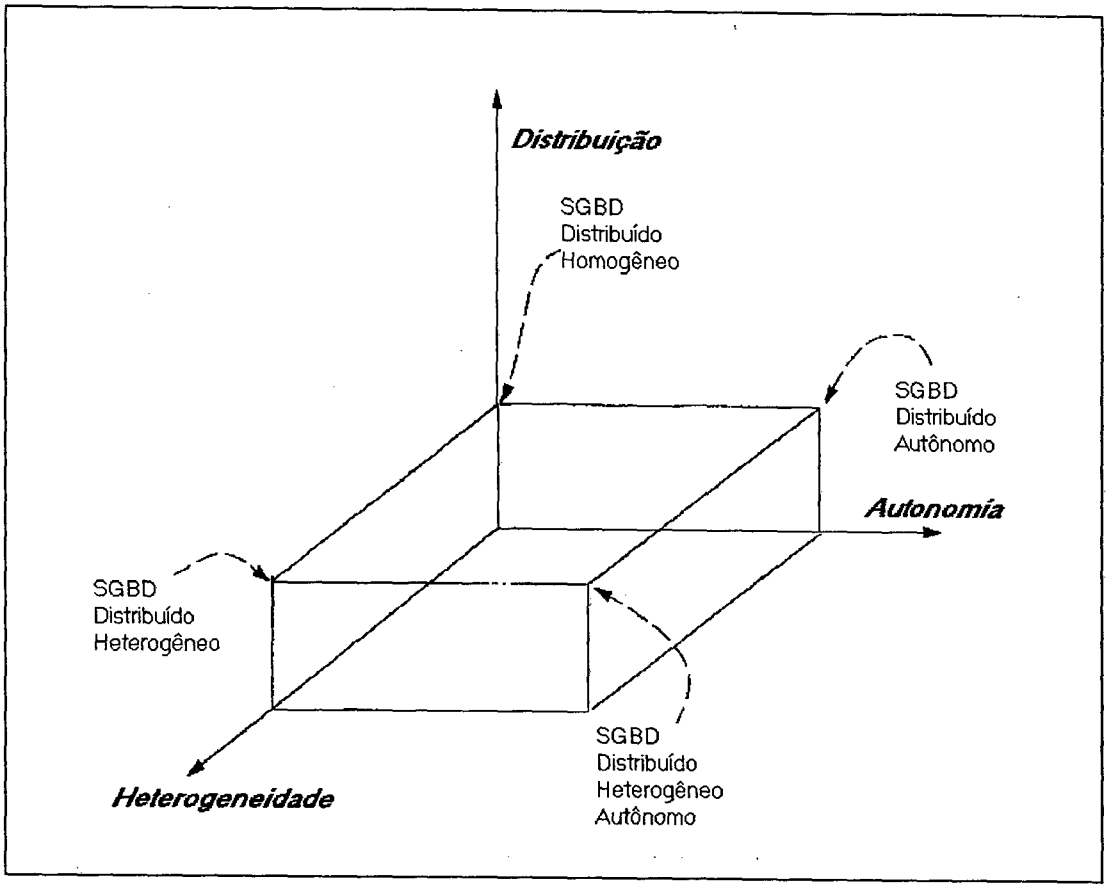


Figura 15 - Alternativas de Implementação do SGBDD

Sistemas de bancos de dados centralizados introduziram o paradigma de processamento de dados no qual as funções de definição e manutenção dos dados passaram a ser de responsabilidade de um servidor denominado SGBD. Esta nova orientação resultou na independência de dados, onde os programas de aplicação estão

imunes a mudanças lógicas ou físicas da organização dos dados e vice-versa. Os SGBDD pretendem estender o conceito de independência de dados para ambientes onde os dados encontrem-se distribuídos e replicados em máquinas conectadas por uma rede de computadores. Isto é alcançado através de várias formas de transparência: transparência de rede (ou distribuição), que encapsula a existência da rede e da distribuição dos dados; transparência de replicação, que mascara a existência de cópias físicas de ítens de dados lógicos e transparência de fragmentação, que evita que o usuário tenha que se preocupar com o particionamento dos dados. O acesso transparente aos dados separa a semântica de mais alto nível das questões de implementação de mais baixo nível. Assim, os usuários do SBDD vêem uma imagem única, integrada logicamente, acessando os dados distribuídos como se eles estivessem centralizados.

Vários módulos do SGBDD cooperam para fornecer as funcionalidades descritas acima (ÖZSU & VALDURIEZ, 1994). Processadores de consultas distribuídas particionam as consultas globais em um conjunto de subconsultas que podem ser executadas em cada nó. Este particionamento é feito transparentemente, utilizando-se de informações do catálogo do banco de dados distribuído. Processadores distribuídos também otimizam as operações de consulta gerando um plano de execução de consultas ótimo para cada fragmento de consulta, através da tomada de decisões quanto a ordem das operações, movimento dos dados entre nós e a escolha dos algoritmos distribuídos e locais para as operações do banco de dados (ÖZSU & VALDURIEZ, 1999). O particionamento de uma consulta global introduz o paralelismo intra-consulta, que contribui para o aumento do desempenho. O paralelismo inerente ao sistema distribuído possibilita que consultas sejam submetidas em vários nós, assim tem-se o paralelismo inter-consulta.

O controle de acesso aos dados é gerenciado por gerentes de transações globais com o auxílio de ***Schedulers***[8] para sincronização e gerentes de recuperação local para confiabilidade. O local e operação dos ***Schedulers*** são dependentes dos algoritmos de controle de concorrência implementados. Os algoritmos mais comuns são alguma variação do ***Two-phase Locking***[9] (2PL). Gerentes de transações são especificamente

[8] Administrador de uma lista de prioridade para tarefas de um sistema computacional (escalonadores).
[9] Algoritmo para restrição de acesso a uma determinada informação contida em um banco de dados.

projetados para suportar um modelo particular de transação como se pode ver em (ÖZSU & VALDURIEZ, 1999) e visam garantir a atomicidade das transações. Os gerentes de recuperação local implementam protocolos de confiabilidade (para garantir a durabilidade) e rotinas de recuperação.

2.3 SISTEMAS DISTRIBUÍDOS PONTO-A-PONTO

Para o desenvolvimento de sistemas distribuídos baseados na comunicação ponto-a-ponto, a arquitetura Cliente/Servidor tem sido utilizada com sucesso em vários projetos de bancos de dados relacionais distribuídos.

Para se projetar este tipo de sistema Cliente/Servidor é necessário analisar várias questões. Primeiramente, a questão do mecanismo de comunicação entre clientes e servidores. Dois paradigmas de comunicação tem sido largamente estudados, *"message passing"* e *"remote procedure call"* (RPC).

O paradigma RPC tem sido mais utilizado por possuir uma semântica mais simples (ÖZSU & VALDURIEZ, 1999). Contudo, a implementação de mecanismos RPC para ambientes heterogêneos de computação não é uma tarefa simples. O problema reside no fato de existirem diferentes implementações do paradigma RPC, sendo que estas implementações normalmente não são interoperáveis. Com isso, pode ser necessário tratar da comunicação com níveis mais altos de abstração para resolver a heterogeneidade, ou em níveis mais baixos (*"message passing"*) para alcançar maior paralelismo.

Uma outra questão consiste em determinar quais funções devem ser fornecidas pelos clientes e pelo servidor. Existem duas alternativas que resultam em dois tipos de arquitetura Cliente/Servidor. Em uma alternativa os clientes requisitam objetos para o servidor, que por sua vez recupera estes objetos da base e os retorna como resposta ao cliente (veja a figura 16). Estes sistemas são chamados servidores de objetos.

Nos servidores de objetos, o servidor é responsável pela maior parte dos serviços de gerência de objetos com o cliente fornecendo, basicamente, o ambiente de execução para as aplicações e algum nível de funcionalidade de gerência de objetos. A otimização

das consultas e a sincronização das transações dos usuários são efetuadas no servidor, sendo que o cliente recebe os objetos resultantes.

O gerenciador de objetos suporta uma série de funções. Primeiro, ele fornece um contexto para a execução de métodos. A replicação do gerente de objetos no cliente e no servidor permite que métodos sejam executados tanto no cliente como no servidor. Um ponto a ser frisado é que a execução de métodos no cliente pode invocar a execução de outros métodos que não foram carregados para o servidor com o objeto.

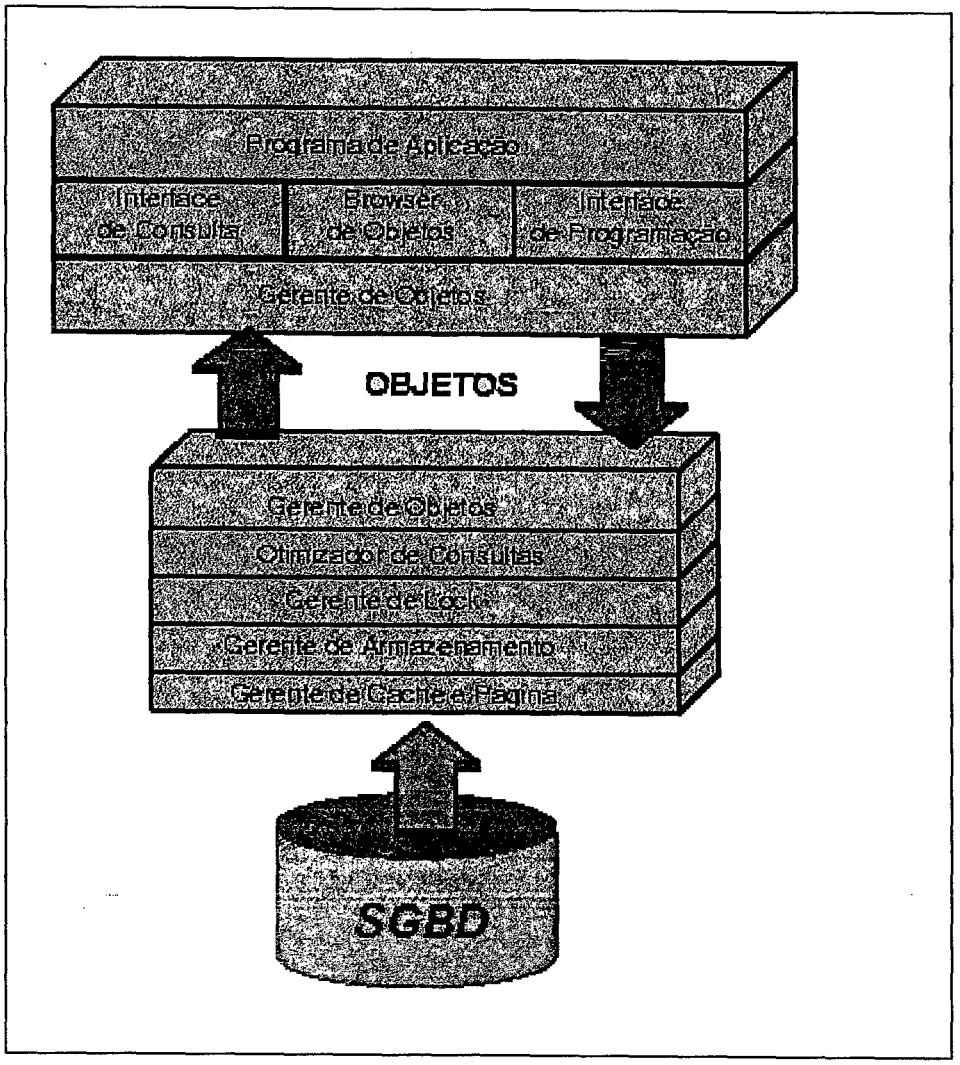


Figura 16 - Arquitetura do Servidor de Objeto

A otimização de execução de métodos deste tipo é um ponto importante que ainda está sendo estudado. O gerenciador de objetos cuida, ainda, da implementação dos

identificadores de objetos e exclusão de objetos, sendo que no servidor fornece suporte para o ***Clustering***[10] e métodos de acesso a objetos.

Uma outra alternativa é a arquitetura Cliente/Servidor com servidor de página, onde a unidade de transferência entre os servidores e clientes são unidades físicas de dados como páginas ou segmentos ao invés de objetos (veja a figura 17).

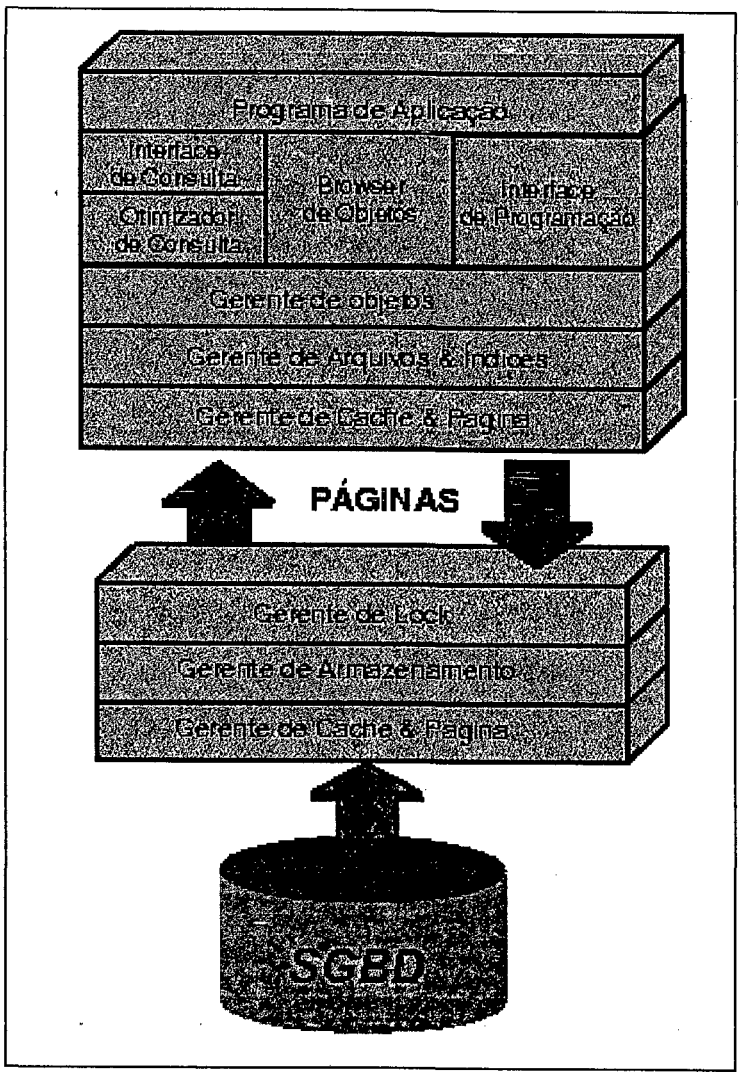


Figura 17 - Arquitetura do Servidor de Página

Arquiteturas de servidor de páginas separam os serviços de processamento de objetos entre os clientes e os servidores. De fato, os servidores não trabalham com objetos, mas sim, como gerentes de conjunto de valores armazenados.

---

[10] Gerenciamento dos dados de uma tabela disposta em mais de um dispositivo de armazenamento

2.4 O PROJETO DE DISTRIBUIÇÃO DE DADOS

A metodologia de um projeto para SBDDs varia de acordo com a arquitetura do sistema. No caso de banco de dados distribuídos fortemente integrados, o projeto é executado de forma descendente a partir da análise dos requisitos e projeto lógico do banco de dados global para o projeto físico de cada banco de dados local.

A atividade (ÖZSU & VALDURIEZ, 1999) começa com a análise dos requisitos que definirá o ambiente do sistema e elicitará tanto as necessidades de dados como de processamento de todos os potenciais usuários do banco de dados. O estudo dos requisitos também especifica quais são os objetivos do sistema quanto a performance, confiabilidade, economia e expansibilidade.

De posse dos requisitos do sistema iniciam-se duas atividades paralelas: projeto das visões e o projeto conceitual. O projeto das visões tem como objetivo definir as interfaces para os usuários finais. O projeto conceitual, por sua vez, consiste em examinar a organização para definir os tipos de entidades e os relacionamentos existentes entre estas entidades. O projeto conceitual pode ser visto como sendo a integração das visões do usuário. A partir do projeto conceitual é definido o esquema conceitual. O esquema conceitual global e as informações dos padrões de acesso coletadas como resultado do projeto de visões são os dados de entrada para o próximo passo, o projeto de distribuição. Nesta etapa, são projetados os esquemas conceituais locais através da distribuição das entidades pelos nós do sistema distribuído. O último estágio deste processo é o projeto físico, o qual mapeia os esquemas conceituais locais em dispositivos de armazenamento físico disponíveis nos nós do sistema.

Existem dois aspectos fundamentais no projeto descendente: fragmentação e alocação. A fragmentação e alocação são as duas etapas que compõem o projeto de distribuição. Em SBDD convencionais, a fragmentação consiste no particionamento de cada relação global em um conjunto de fragmentos de relação, enquanto que a alocação concentra-se na distribuição (e possível replicação) destas relações locais entre os nós do sistema distribuído.

CAPÍTULO III

3 ARQUITETURA DO SGBD DISTRIBUÍDO MÓVEL

Em ambientes de computação móvel, pequenos computadores portáteis com capacidade de conexão sem fio, conhecidos como unidades móveis, são carregados por usuários que podem se locomover livremente para qualquer lugar. Apesar de que isto dá aos usuários grande comodidade em termo de mobilidade, as redes sem fio possuem um alto custo, tem largura de banda limitada e fornecem conexão de baixa qualidade e maior interferência. Consequentemente, desconexões mesmo não intencionais ocorrem ocasionalmente. Isto demonstra o quanto é limitado o acesso aos dados por intermédio de uma unidade móvel.

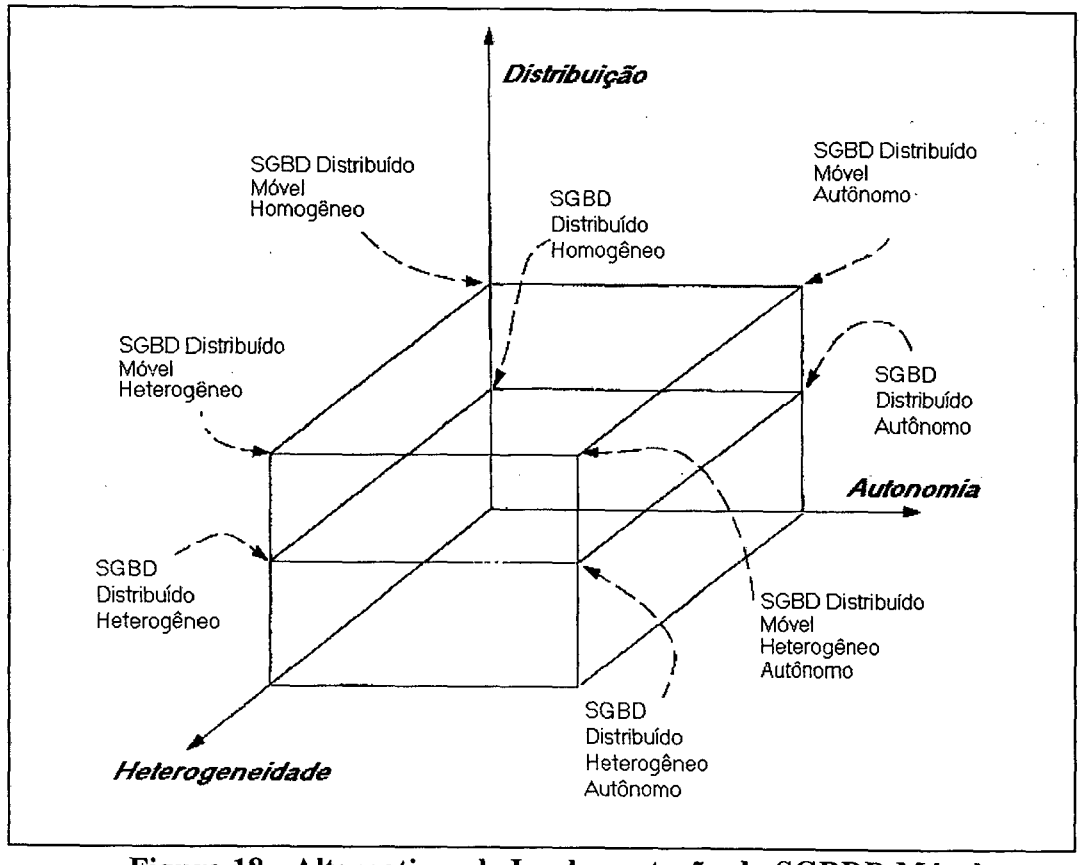


Figura 18 - Alternativas de Implementação do SGBDD Móvel

Um SGBD para o ambiente móvel nada mais é do que a extensão de um SGBD Distribuído (SGBDD). Em (ÖZSU & VALDURIEZ, 1999) é realizada uma classificação excelente para SGBDD baseado nas características de sistema de autonomia, distribuição e heterogeneidade. Em (DUNHAM & HELAL, 1995) é estendida esta classificação para incluir o SGBDD móvel, pois um sistema de computação móvel deve incluir uma rede fixa que é um sistema distribuído (veja a figura 18).

Um sistema de computação móvel pode ser visto como um tipo dinâmico de sistema distribuído, que une unidades móveis independentemente da variação do meio de comunicação de dados utilizado. Estas variações representam a conexão entre as unidades móveis e as estações rádio base, com as quais elas são conectadas. Nós poderíamos ter um sistema móvel concebivelmente sem autonomia ou heterogeneidade entre os componentes do SGBDD baseado nesta categorização. Porém, isto é altamente improvável. O que se busca é um modelo de transação do sistema mais completo: um SGBDD Móvel Heterogêneo Autônomo.

3.1 MODELO CLIENTE/SERVIDOR ESTENDIDO

Outra maneira para caracterizar a computação cliente/servidor em ambientes móveis é examinar o efeito da mobilidade no modelo de computação cliente/servidor. Em um sistema de informações cliente/servidor, um servidor é qualquer máquina que mantém uma cópia completa de um ou mais bancos de dados. Um cliente é capaz de acessar dados residentes em qualquer servidor com o qual pode se comunicar. Sistemas cliente/servidor clássicos assumem que a localização do cliente e do servidor não muda e a conexão entre eles também não. Como resultado, a funcionalidade entre cliente e servidor é estaticamente particionada. Em um ambiente móvel, entretanto, a distinção entre clientes e servidores pode ser temporariamente confusa, resultando em um modelo cliente/servidor estendido mostrado na figura 19. As limitações de recursos de clientes podem requerer que certas operações, normalmente executadas em clientes, sejam executadas em servidores ricos em recursos. De modo oposto, a necessidade de lidar com conectividade incerta requer que clientes às vezes emulem as funções de um servidor. Um caso extremo é chamado de arquitetura cliente leve (***Client Thin***) que

expõe principalmente a funcionalidade e lógica de aplicações de clientes para servidores estáticos. Na arquitetura cliente leve, aplicações nos servidores estáticos são normalmente compatíveis e otimizadas para dispositivos clientes móveis. A arquitetura cliente leve é especialmente adequada para aplicações do tipo terminal burro ou para PDA com pouca capacidade de memória. O outro caso extremo é a arquitetura cliente completo (***Client Full***). A arquitetura cliente completo emula funções do servidor nos dispositivos do cliente e então, é capaz de minimizar a incerteza das desconexões durante as comunicações.

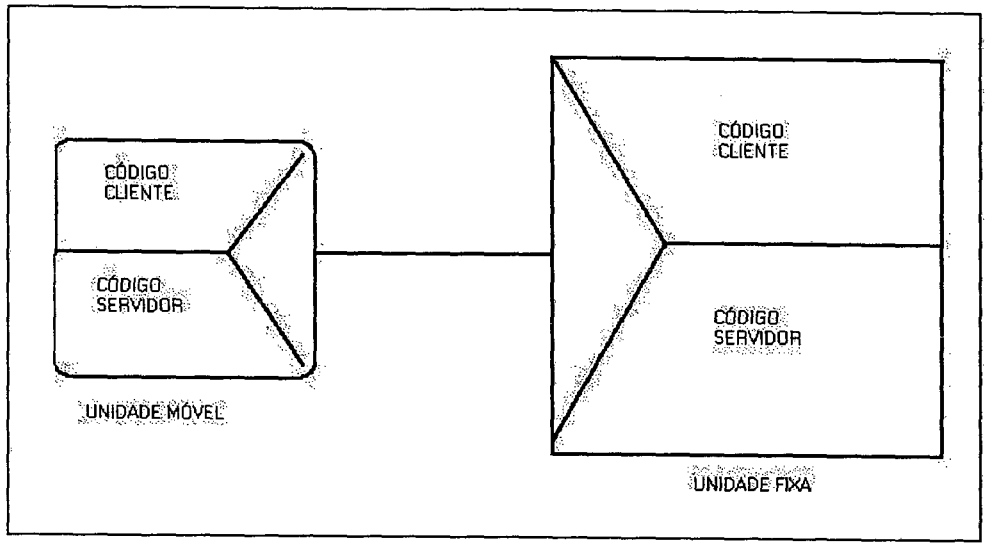


Figura 19 - Modelo Cliente/Servidor Estendido

3.1.1 ARQUITETURA CLIENTE LEVE

O projeto InfoPad (LE et al, 1994) demonstra o conceito de arquitetura cliente leve. O sistema InfoPad é composto de quatro camadas mostradas na figura 20:

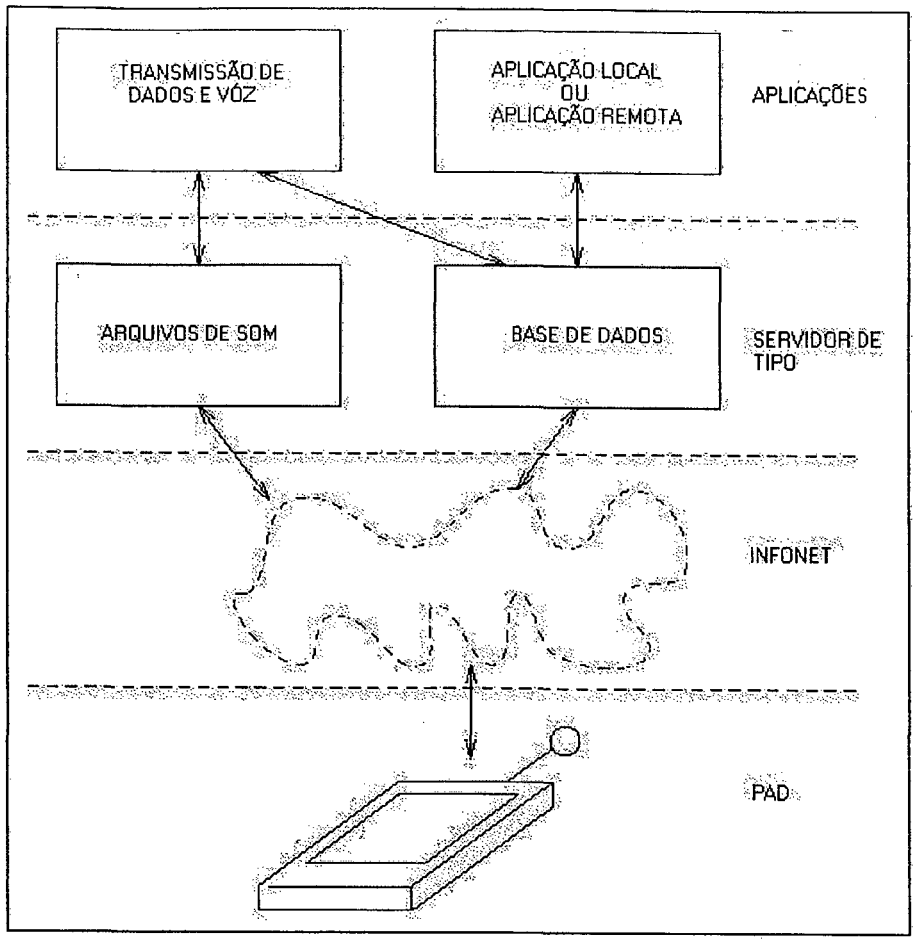


Figura 20 - Camadas do Sistema InfoPad

- Pad: É um terminal de multimídia portátil de pouca potência que é capaz de exibir texto e gráficos, tocar áudio e vídeo compactado, gravar áudio e capturar entrada de dados com toques no visor através de caneta.
- InfoNet: Apresenta a camada de software acima da camada Pad com uma abstração na qual cada Pad aparece como um terminal multimídia, conectado a rede e estático. Esta camada contém os algoritmos de roteamento para dar mobilidade e as rotinas para gerenciar os recursos de rede sem fio (por exemplo - alocação ou freqüências para o Pad).
- Servidores de tipo: Fazem o Pad mostrar-se como uma estação de trabalho típica para aplicações. Isto permite compatibilidade com software da estação de trabalho comum. Os servidores de tipo tornam transparente para as aplicações: os ambientes móveis e os terminais comuns. Entretanto, as aplicações são otimizadas para uso no Pad. A otimização aproveita-se de

características especiais do Pad, como o tamanho pequeno da tela, falta de teclado e suporte para letra de mão e reconhecimento de voz.

- Aplicações: no sistema InfoPad são customizadas para as características do Pad, elas não contém código que as permita adaptar-se dinamicamente a mudanças na rede móvel. Em vez disso, a adaptação da capacidade móvel é largamente executada na camada InfoNet e na camada de servidores de tipo. Neste sentido, a adaptação no sistema InfoPad pode ser caracterizada como adaptação de transparência na aplicação em vez de capacidade de adaptação na aplicação.

A arquitetura cliente leve da ***CITRIX Corporation*** permite que uma variedade de computadores remotos, independentemente de suas plataformas, conectem-se a um servidor de terminal ***Windows NT*** para emular remotamente um ***Desktop*** de mesa e suas aplicações (CITRIX, 2001). Um servidor chamado ***MetaFrame*** roda sob ***Windows NT*** no computador de mesa e comunica-se com os clientes leves executando nos computadores remotos usando o protocolo ICA (Independent Computing Architecture - Arquitetura de Computação Independente). O cliente ICA e o servidor ***MetaFrame*** colaboram para exibir o ***Desktop*** virtual na tela do computador remoto. Eles também colaboram para processar eventos do mouse, do teclado, executar programas e observar os dados armazenados no servidor. Todas as execuções são remotas e nenhuma ocorre no computador portátil cliente. Um projeto de pesquisa na Motorola (DURAN & LAUBACH, 1999) estendeu a arquitetura cliente leve de forma que fosse otimizada no ambiente sem fio. O trabalho apontou que a limitação da largura da banda não é tão prejudicial quanto à latência da rede para execução cliente leve. Isto porque o uso da largura da banda para clientes leves é limitado (JING et al, 1999).

3.1.2 ARQUITETURA CLIENTE COMPLETO

Clientes móveis precisam estar aptos a usar redes com menos características desagradáveis: intermitência, pequena largura de banda, alta latência ou alto custo. A conectividade com uma ou mais destas propriedades é referido como conectividade fraca. No caso extremo, clientes móveis serão forçados a trabalhar no modo

desconectado. A habilidade de operar em modo desconectado pode ser útil mesmo quando a conectividade está disponível. Por exemplo, operações desconectadas podem aumentar o aproveitamento da bateria evitando transmissão e recepção sem fio (JING et al, 1999). Isto pode reduzir a sobrecarga na rede, uma importante característica quando as taxas de sobrecarga são altas. Isto permite que haja silêncio no rádio (situação em que ninguém transmite), uma habilidade vital em aplicações militares. Uma arquitetura cliente completo pode ser usada para efetivamente apoiar os clientes desconectados ou fracamente conectados. Comparada com a arquitetura cliente leve, a arquitetura cliente completo está no extremo oposto da escala de modelos cliente/servidor estendidos (JING et al, 1999). A arquitetura cliente completo sustenta a emulação de funções de servidores no cliente, de forma que as aplicações podem ser executadas sem existir conexão com servidores remotos. A emulação pode ser sustentada através de um agente ou servidor "leve" residindo em hosts clientes. Por exemplo, um agente poderia emular algumas operações de bancos de dados para permitir que usuários móveis trabalhem em modo desconectado.

3.2 ARQUITETURA CLIENTE/SERVIDOR FLEXÍVEL

Arquitetura cliente/servidor flexível aplica, de forma geral, as arquiteturas cliente completo e leve e com isso, as regras de clientes e servidores e lógica de aplicação podem ser dinamicamente mudadas e executadas em diferentes equipamentos móveis e estáticos como ilustrado pela figura 21. Na arquitetura flexível, a distinção entre clientes e servidores pode ser temporariamente confusa por propósitos de execução e disponibilidade. Além disto, a conexão entre clientes e servidores pode ser estabelecida dinamicamente durante a execução de aplicações (por exemplo, para o serviço de ***handoff***[11]).

[11] É o processo de passagem de uma célula para outra em uma infra-estrutura de rede celular (reintegração).

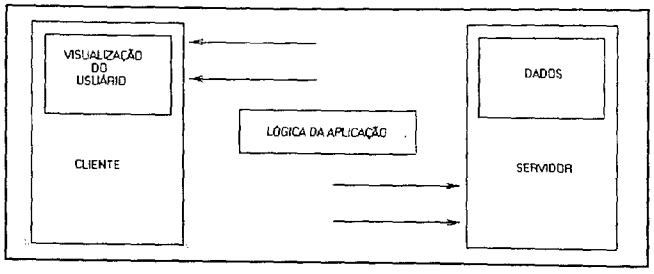


Figura 21 - Arquitetura Cliente/Servidor Flexível

3.2.1 GRUPOS COLABORATIVOS

Um caso que normalmente ocorre pode ser diversos usuários desconectados do resto do sistema enquanto ativamente cooperando; um bom exemplo pode ser um grupo de profissionais fazendo uma viagem de negócios juntos. Mais que dar aos membros deste grupo de trabalho acesso desconectado apenas aos dados que eles tiveram a intenção inicial de copiar para suas máquinas pessoais, uma arquitetura de grupo colaborativo permite a qualquer membro do grupo acesso a quaisquer dados que estiverem disponíveis ao grupo. A arquitetura é baseada em uma divisão de funcionalidade entre servidores que armazenam os dados e clientes que lêem e gravam dados gerenciados pelos servidores. Um servidor é qualquer máquina (por exemplo, uma unidade móvel) que mantém uma cópia completa de um ou mais bancos de dados. Um cliente é capaz de acessar dados residentes de qualquer servidor com o qual ele pode se comunicar. Qualquer máquina que mantém uma cópia de um banco de dados, incluindo ***Laptops*** pessoais, deveria estar pronta para atender solicitações de leitura e gravação de outras máquinas próximas. Nesta arquitetura, computadores portáteis podem ser servidores para alguns bancos de dados e clientes para outros (JING et al, 1999).

3.2.2 MOBILIDADE VIRTUAL DE SERVIDORES

Em um sistema de informações sem fio, servidores de dados são conectados via redes fixas para fornecer serviço de informação para usuários móveis. A replicação dos serviços de informação poderia ajudar a reduzir a latência das operações remotas e balancear a carga de trabalho de servidores de dados em um ambiente de rede distribuído e múltiplo. A replicação de serviços de informação em diferentes redes pode sustentar os ***handoffs*** de serviço de aplicação para clientes móveis (JING et al, 1999).

Em um ambiente sem fio, o cliente se desloca talvez por lugares que nunca tenha visitado antes. Uma vez em um novo local (ou um novo ambiente de rede), ele irá negociar com algumas máquinas próximas para continuamente fornecer serviços para suas operações. O movimento de unidades móveis pode resultar em um longo caminho de comunicações em redes fixas, porque a distância física pode não refletir necessariamente distância de rede e serviço entre o cliente e o servidor, especialmente quando o movimento cruza a fronteira de diferentes redes. O longo caminho da comunicação em redes fixas aumentará o tráfego e latência de serviços transacionais. Se o novo coordenador puder sempre usar um site de serviço de informação local ou próximo como servidor de dados do cliente, o tráfego e a latência em redes fixas pode ser reduzido para a contínua interação entre o cliente e o servidor (JING et al, 1999). Então, a mobilidade do cliente introduz os conceitos de mobilidade virtual de servidores e ***handoffs*** de serviços de aplicações.

Um aspecto de sustentação de mobilidade virtual de servidores é minimizar o tempo de ***handoffs*** de serviços de aplicações para operações multi-servidoras síncronas. O trabalho em (TAIT & DUCHAMP, 1991, 1992) investiga como manter réplicas em um sistema de arquivos distribuídos que sustenta clientes móveis. Este trabalho assume que:

1. Movimentos de clientes não podem ser compelidos, apesar que padrões de movimentos podem existir;
2. A latência de operações remotas aumentam de acordo com o aumento da distância entre os equipamentos;
3. A carga de trabalho compartilhada sequencial não é incomum, mas a carga de trabalho compartilhada simultânea (exceto leitura-leitura) é raro; e
4. Um ***Cache*** de arquivo de tamanho modesto é mantido por cada cliente. Os objetos do projeto neste trabalho incluem: minimizar operações multi-servidoras síncronas; facilitar a adição e exclusão de servidores; e considerar replicação incompleta.

Para atingir estes objetivos, uma arquitetura hierárquica de servidor primário-secundário é proposta (JING et al, 1999), como mostrado na figura 22. Tipicamente, o

cliente precisa não contatar qualquer servidor (remoto), mas comunicar-se sincronamente com o servidor primário local apenas. Todas as alterações são armazenadas no ***Cache*** do cliente. O servidor primário faz buscas temporárias dos clientes que ele está servindo no momento e propaga as atualizações de volta ao servidor secundário assincronamente. Esta estratégia de busca permite que operações de gravação dos clientes retornem imediatamente após o novo valor ser posto em seu ***Cache***.

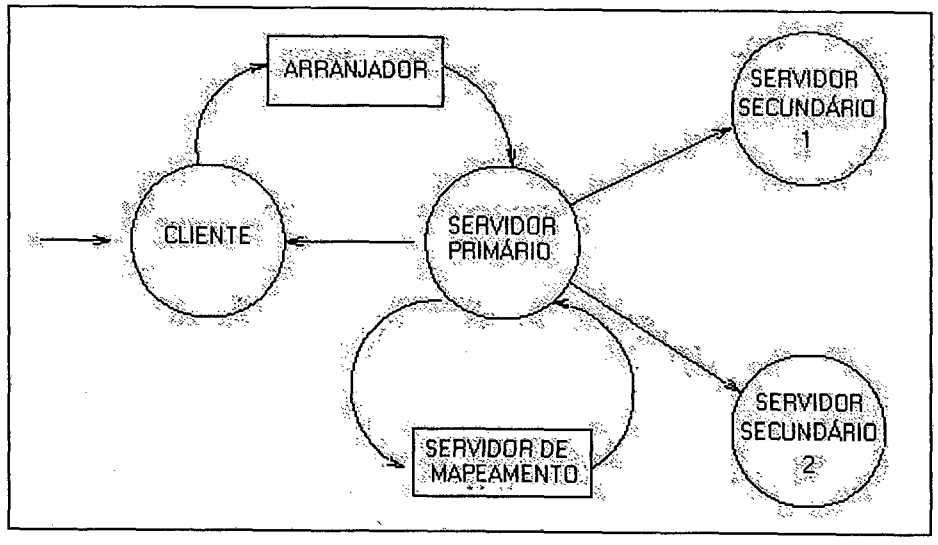


Figura 22 - Hierarquia de Servidores Primários e Secundários

Os servidores primários são usados como intermediários entre clientes e servidores secundários. Devido ao fato de o sistema sustentar adição/deleção de réplicas e replicação incompleta, um servidor primário precisa aprender como mapear do sistema de arquivos para os servidores secundários. O mapeamento é fornecido chamando uma função específica do servidor. Por exemplo, alguém de Nova York que está visitando Seattle chamaria o arranjador Seattle para obter um servidor primário local. O servidor primário local escolhido então chama o servidor de mapeamento em Nova York para alocar arquivos que o cliente deseja acessar. Após o ***handoff*** de servidores primários, o novo servidor primário pode lentamente copiar o arquivo do cliente do ***Cache*** antigo no servidor primário anterior. Então, este método sempre conecta o cliente móvel ao servidor primário local para acessar informação em um sistema de arquivos distribuído (JING et al, 1999).

Em (KRISHNAKUMAR & JAIN, 1994) é apresentada uma arquitetura de sistema e protocolos de ***handoff*** de serviço de aplicação para servidores de mobilidade virtual. A arquitetura é baseada em servidores distribuídos replicados conectados a usuários via uma rede de serviço de comunicação pessoal (PCS – Personal Communication Services). Nesta arquitetura, de acordo como o usuário move-se de uma área de serviço para outra, o servidor local na nova área de serviço assume o comando fornecendo o serviço. Este ***handoff*** de serviço para a mobilidade virtual do servidor é, de um modo geral, análogo ao procedimento de ***handoff*** de chamada de PCS e também requer que a continuação do serviço seja transparente sem interrupções. Para os ***handoffs*** de serviços de aplicação, um coordenador de serviços, quando recebe a informação que um usuário se moveu, inicia a configuração de uma chamada de conferência entre o servidor atual, o host móvel e o novo servidor, de forma que o serviço possa ser transferido transparentemente para o novo servidor. O coordenador pode então terminar a chamada com o servidor antigo. Antes que o novo servidor assuma o comando durante um serviço de ***handoff***, ele tem que saber qual o contexto do usuário com relação ao serviço. A informação de contexto é associada com um usuário e um serviço de modo que o usuário pode acessar diferentes servidores transparentemente. Parte do contexto é estático, incluindo senha e direitos de acesso que não mudam conforme o usuário acessa a informação. Entretanto, o contexto também inclui informação dinâmica de dados específicos da sessão, como por exemplo, quantos dados foram lidos ou modificados pelo usuário e se as mudanças são intencionadas a ser transacionais, se o usuário mantém qualquer bloqueio de acesso aos dados, e muitas outras. Em (ELMAGARMID et al, 1995), um conceito chamado ***Reservation Algorithm*** (RA) é proposto para permitir transferência de log[12] de transações e o uso de protocolos de ***Commit*** global.

3.2.3 CLIENTE/PROXY/SERVIDOR

Nesta arquitetura, um componente de software denominado ***Proxy*** (ou agente) é executado em algum host da rede fixa e age como representante do elemento móvel e intermediário em qualquer comunicação entre o cliente (no elemento móvel) e o servidor

[12] Arquivo que demonstra o resultado de uma seqüência de comandos computacionais

(na rede fixa). Desta forma, a comunicação cliente/servidor é quebrada em uma parte sem fio (cliente ~ ***Proxy***) e uma parte na rede fixa (***Proxy*** ~ servidor) como demonstrado na figura 23. O ***Proxy*** tem como principal tarefa amenizar a discrepância entre a qualidade e a eficiência da transmissão nos dois meios.

Um conjunto básico de funções do Proxy incluem a conversão/tradução de mensagens entre o meio com e sem fio, a buferização e eventualmente compactação de dados. Em alguns sistemas, o proxy também assume um papel mais ativo, notificando o cliente de eventos específicos da aplicação ou efetuando as adaptações necessárias (por exemplo, nos protocolos para o meio sem fio) para compensar as mudanças percebidas no ambiente de execução.

Proxies podem ser específicos para um serviço (p.ex. acesso WWW) ou um protocolo mais básico (IP, TCP). Neste último caso, diz-se que o proxy é genérico, ou independente de serviço.

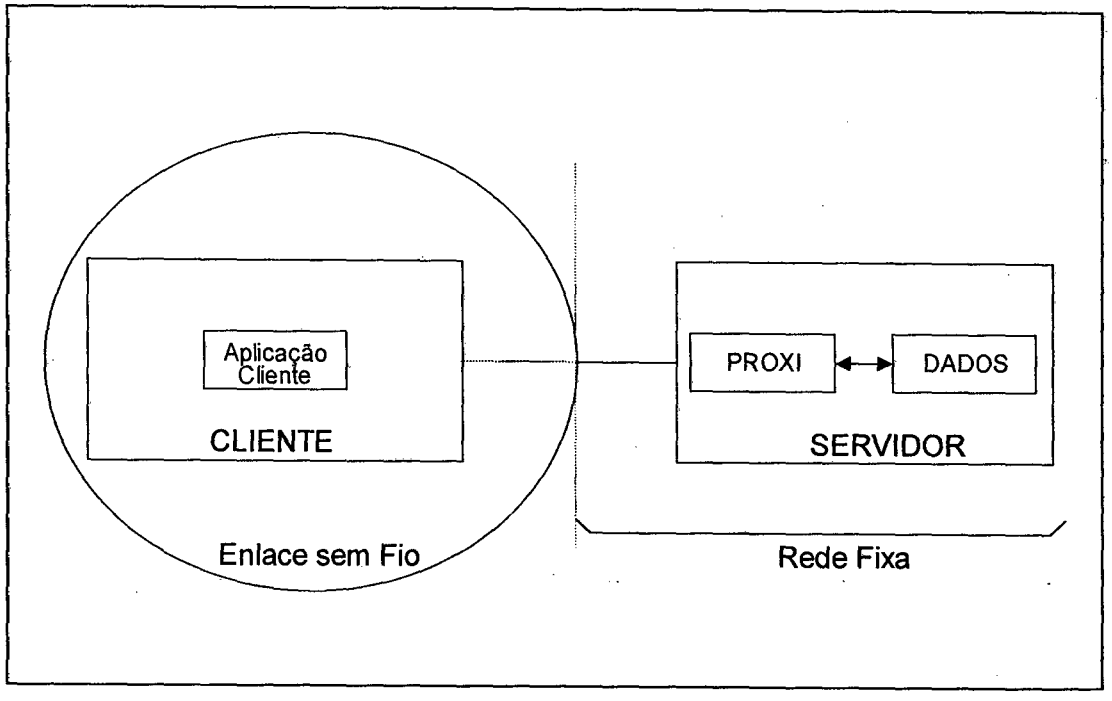


Figura 23 - Arquitetura Cliente/Proxi/Servidor

Esta arquitetura se adequa melhor para dispositivos móveis com poucos recursos, uma vez que boa parte das tarefas de acesso à rede fixa são delegadas ao proxy.

3.2.4 CLIENTE/INTERCEPTADOR/SERVIDOR

Esta arquitetura complementa a arquitetura anterior no sentido de propor a existência de um proxy (ou agente cliente) também no elemento móvel. Este agente é encarregado de interceptar todas as requisições da aplicação cliente e, juntamente com o proxy na rede fixa (o agente servidor), efetuar a conversão de mensagens, a buferização de requisições e respostas e executar as adaptações necessárias devido às mudanças na qualidade e eficiência da transmissão sem fio, este processo é ilustrado pela figura 24.

Devido a existência de dois agentes interceptadores, esta arquitetura permite melhorar a compressão de dados, otimizar a comunicação sem fio e tratar de forma melhor os períodos de desconexão. Por exemplo, pode-se manter um ***Cache*** local no agente cliente, que desta forma pode permitir à aplicação cliente continuar executando durante um período de desconexão e assim dar a esta a ilusão de uma operação conectada.

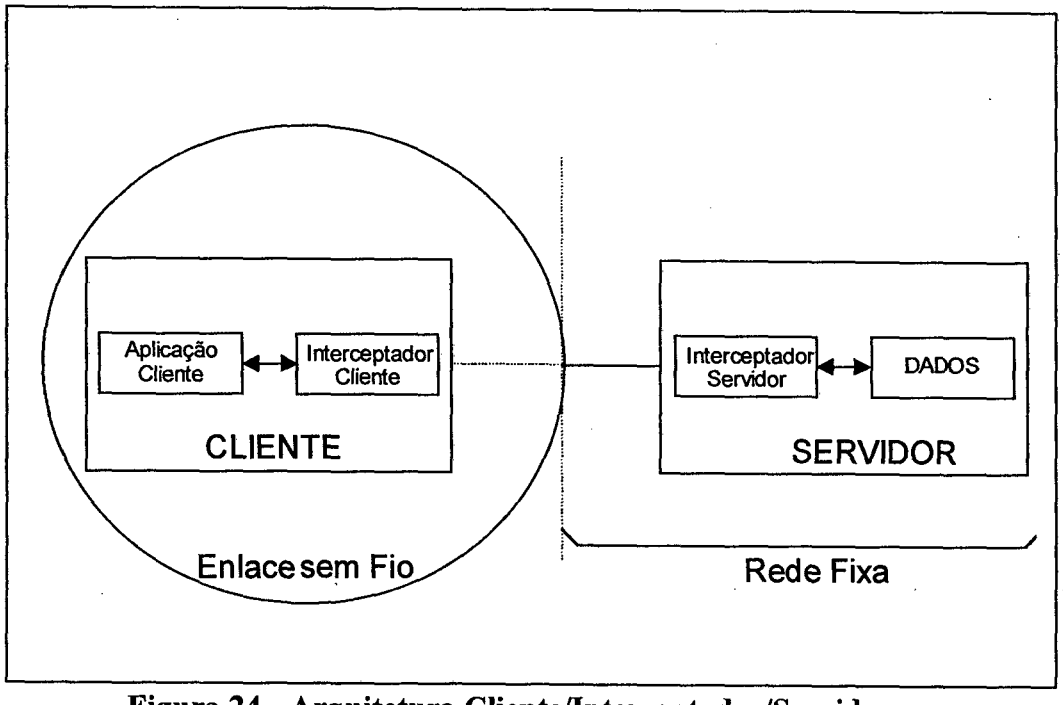


Figura 24 - Arquitetura Cliente/Interceptador/Servidor

Este modelo é mais adequando para dispositivos móveis com maior capacidade de processamento, armazenamento e maior autonomia de energia. O principal problema é o custo de desenvolvimento, pois é necessário implementar um par de agentes para cada aplicação ou serviço que deva ser usado em uma rede celular móvel. Tal desenvolvimento torna-se ainda mais custoso devido às particularidades dos dispositivos

móveis e à diversidade de plataformas de software para estes equipamentos (por exemplo: WindowsCE, PalmOS, GEOS/Nokia).

3.3 AGENTES MÓVEIS

Agentes Móveis (CHESS et al, 1995) são processos (ou objetos ativos) que são capazes de migrar entre equipamentos de uma rede durante a sua execução, levando consigo o seu estado de execução.

Uma arquitetura de software baseada em agentes móveis tem como principal vantagem permitir que elementos móveis enviem agentes para a rede fixa a fim de que estes executem determinada tarefa em seu lugar, de forma mais eficiente, sem gastar os próprios recursos (energia) e independente das variações de conectividade. Neste sentido, uma arquitetura deste tipo é a ideal para satisfazer o requisito de migração de funcionalidade. No entanto, muitos outros problemas relacionados com esta arquitetura continuam abertos, sendo que uma das principais é a segurança. Algumas das questões são:

- Como evitar o acesso não autorizado, a inibição, ou a destruição de um agente móvel executando em nome de um elemento móvel?
- Como evitar ataques do tipo "denial of service"[13] nos hosts da rede fixa?
- Como autenticar a identidade do usuário dono do agente e comprovar que esta não foi falsificada?

Apesar de já existirem muitas plataformas experimentais (e até comerciais) para a programação de agentes móveis (a maioria baseada em Java), por enquanto os problemas de segurança têm inibido o uso desta tecnologia para o desenvolvimento de serviços e aplicações distribuídas em larga escala.

[13] É um tipo de ataque que indisponibiliza parte ou totalmente o servidor para o acesso de seus usuários

3.4 ACESSO A DADOS MÓVEIS

Acesso a dados móveis permite a entrega de dados do servidor e a manutenção de consistência dos dados em um ambiente cliente/servidor móvel e sem fio. Acesso a dados eficiente e consistente em ambientes móveis é uma área de pesquisa desafiadora por causa da fraca conectividade e restrições de recursos. As estratégias de acesso aos dados em um sistema de informações móvel podem ser caracterizadas pelo modo de distribuição, organização dos dados, requerimentos de consistência, etc. O modo de distribuição de dados no servidor pode ser ***Server-push***, ***Client-pull***, ou híbrido (JING et al, 1999). A distribuição ***Server-push*** é iniciada por funções do servidor que inicia o envio de dados do servidor para os clientes. A distribuição ***Client-pull*** é iniciada por funções do cliente as quais enviam requisições a um servidor e "puxam" dados do servidor para fornecer dados para aplicações rodando localmente. A distribuição híbrida usa distribuição ***Server-push*** e ***Client-pull***. A organização dos dados inclui mobilidade específica como fragmentos de banco de dados móvel armazenados no servidor para posterior propagação. Os requerimentos de consistência abrangem desde consistência fraca a consistência forte. A figura 25 ilustra o novo paradigma de acesso a dados móveis.

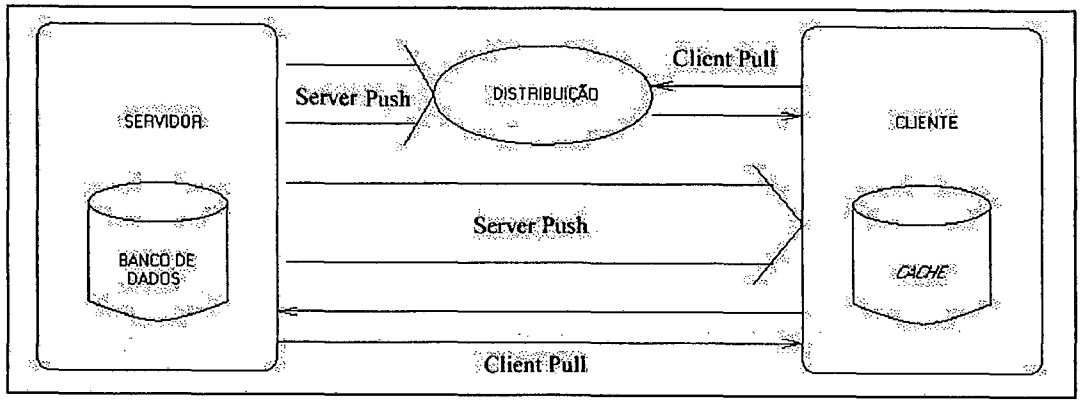


Figura 25 - Modelo de Acesso a Dados Móveis

3.5 TRANSAÇÕES MÓVEIS

Algumas pesquisas enfocaram os problemas de processamento de informações móveis e a restauração de transações ACID (Atomicidade, Consistência, Isolação,

Durabilidade) para processamento de informações móveis. A seguir será apresentada uma análise concisa de trabalhos relevantes nesta área.

- ***Kangaroo Transaction***: o modelo de transação Kangaroo é o primeiro e único estudo que apresenta um modelo de transação para redes móveis que inclui movimento em sua estratégia de execução (DUNHAM et al, 1997). O modelo é construído assumindo uma abordagem com múltiplos bancos de dados, onde o gerenciamento de transações é sempre executado em uma ERB (Estação Rádio Base) (ou nodo específico na rede fixa). À medida que a unidade móvel se movimenta, o gerenciamento da transação acompanha este movimento. Cada transação é dividida em subtransações, as quais são executadas independentemente em sistemas SGBD na rede fixa. Dois tipos de modos de processamento são aceitos, um assegurando atomicidade completa requerendo transações compensadoras ao nível de subtransações. Enquanto a transação kangaroo é solicitada em uma unidade móvel ela somente pode ser processada nos nodos SGBD na rede fixa. Ela não pode ser executada numa unidade móvel.

- ***Reporting and Co-transactions***: em (CHRYSANTHIS, 1993) é apresentado um esquema para estruturar transações do tipo fluxo de tarefas (***Workflow***) (por exemplo, declarações de garantia de processo e políticas de garantia de vendas) a serem executadas em uma plataforma móvel. Ele propõe uma transação mista usando a idéia de "transação de reportagem" e "co-transação". Uma transação mãe (***Workflow***) é representada em termos de reportagem e co-transações, as quais podem ser executadas em qualquer lugar (unidade móvel ou ERB). Uma transação de reportagem pode compartilhar seus resultados parciais com a transação mãe a qualquer hora e pode ser aceita (realizar ***Commit***) independentemente. Uma co-transação é uma classe especial de transação de reportagem a qual pode ser forçada a esperar (suspendida) por outra transação que não seja uma transação de reportagem, depois de enviar seu resultado pode continuar a executar o aceite (realizar ***Commit***). Estes componentes podem ser atômicos, similar aos dois modos de processamento com transações kangaroo. Estes componentes

podem executar em múltiplos sites (unidade móvel ou ERB) de forma paralela ou serial. Esta maneira de gerenciar transações workflow minimiza significativamente o custo de comunicação com fio e sem fio (CHRYSANTHIS, 1993).

- ***Clustering***: em um ambiente cliente/servidor, ***Clustering*** (agrupamento) é o processo de usar dois ou mais equipamentos juntos de tal forma que eles se comportam como um único computador. Quando um deles falha, o outro continua executando as tarefas de forma contínua e sem interrupção. O modelo de ***Clustering*** considera um sistema completamente distribuído e o modelo de transação é projetado para manter a consistência do banco de dados (PITOURA & BHARGAVA, 1995). O banco de dados é dividido em grupos. Além disso, uma consistência dos dados em dois níveis é definida e mantida. Neste modelo, uma transação móvel é decomposta em um conjunto de transações fracas e estritas. A decomposição é feita baseada em requerimentos de consistência. As operações de leitura e gravação são também classificadas como fracas e estritas. As operações fracas podem acessar informações em um período de desconexão, a transação móvel pode acessar e alterar os dados na unidade móvel e tornando-se inconsistente com relação aos dados na rede fixa.

- ***Semantics Based***: O esquema de processamento de transações móveis baseados na semântica vê o processamento de transações móveis como um problema de concorrência e coerência de ***Cache*** (WALBORN & CHRYSANTHIS, 1995). O modelo assume uma transação móvel como duradoura caracterizada por atrasos longos da rede e desconexões imprevisíveis. Esta abordagem utiliza a organização de objetos para dividir objetos grandes e complexos em fragmentos gerenciáveis menores. Um servidor de banco de dados estático reparte os fragmentos numa solicitação de uma unidade móvel. Na finalização da transação as unidades móveis retornam os fragmentos para o servidor. Estes fragmentos são agrupados novamente pela operação de consolidação no servidor. Se os fragmentos podem ser recombinados em qualquer ordem, então os objetos são chamados

reordenáveis, desde que um único servidor de banco de dados seja assumido, assim, as propriedades ACID podem ser mantidas.

- ***MDSTPM***: em (YEO & ZASLAVSKY, 1994) foi examinado como transações multi banco de dados poderiam ser submetidas de estações de trabalho móveis. Ele assume que um MDSTPM (Multidatabase Transaction Processing Manager - Gerenciamento de Processo de Transação Multi Banco de Dados) existe em cada unidade móvel. Uma arquitetura em camadas para suportar estas transações móveis é proposta. A abordagem proposta é uma generalização do modelo MDBS (Multidatabase System – Sistemas Multi Banco de Dados), portanto, o controle de concorrência e os algoritmos de recuperação influenciarão na manutenção das propriedades ACID.

A seguir, será realizado um comparativo entre os modelos de transações móveis abordados, observando as propriedade ACID e a localização de processamento:

Modelo	A	C	I	D	Solicitação	Execução
Kangaroo	Talvez	Não	Não	Não	Unidade Móvel	Rede Fixa
Reporting and Co-transaction	Não	Não	Não	Não	Não assume	Não assume
Clustering	Não	Não	Não	Não	Unidade Móvel	Unidade Móvel ou Rede Fixa
Semantics Based	Sim	Sim	Sim	Sim	Unidade Móvel	Servidor Restrito/ Unidade Móvel
MDSTPM	Não	Não	Não	Não	Unidade Móvel	Unidade Móvel ou Rede Fixa

Tabela 3 – Modelos de Transação Móvel – resumo comparativo

Todas as técnicas discutidas são esquemas diferentes para gerenciar a execução de transações em uma plataforma móvel. Elas normalmente assumem que a transação é solicitada em uma unidade móvel. De fato uma transação móvel poderia ser solicitada de

qualquer nodo na rede com ou sem fio e potencialmente executada em qualquer nodo (KUMAR & DUNHAM,1998).

3.6 TRATAMENTO DE EXCEÇÕES NO ACESSO A DADOS MÓVEIS

Em muitas situações, a execução de algoritmos distribuídos clássicos num ambiente de computação móvel, de forma direta, é ineficiente. Isso se deve ao fato de que tais algoritmos não tratam a mobilidade computacional, nem as restrições de recursos dos computadores móveis. Por esse motivo, é necessário aplicar outros princípios de projeto de algoritmos distribuídos (BADRINATH et al, 1994). Alguns dos algoritmos distribuídos têm sido estudados para aplicação na computação móvel: protocolos de comunicação; definição de mecanismos de ordenação de eventos; propagação de informação em uma rede de comunicação; controle de concorrência; coordenação entre processos para acesso a recursos compartilhados; e comunicação em grupo.

Os bancos de dados também são influenciados pela presença de usuários móveis. Novos métodos para transações deverão ser desenvolvidos de forma a tratar usuários que se movimentam e se desconectam durante a realização de uma transação. Deve-se criar mecanismos para o tratamento de consultas quando a unidade móvel se encontra desconectada da rede de comunicação , como exemplo, pode-se citar o uso de ***Cache*** e manutenção da consistência de dados. Deve-se criar consultas que sejam otimizadas, visando a economia de energia e não a quantidade de informação transmitida. Na verdade, esta regra é válida para qualquer tipo de algoritmo. Isto tem levado ao desenvolvimento de esquemas de processamento que permitam a migração de tarefas que consomem uma grande quantidade de energia de unidades móveis para estações fixas, com o resultado retornando posteriormente para a unidade móvel. Normalmente, isto tem sido feito através de agentes móveis. Também têm sido desenvolvidas técnicas para tratamento da falta de energia na unidade móvel, o que permite que dados críticos existentes na memória principal possam ser deslocados para uma região de memória estática quando do término da energia disponível. Este é o caso típico de projeto de um sistema considerando a utilização de ***Hardware*** e ***Software*** simultaneamente.

O trabalho de pesquisa na área de recuperação de falhas num ambiente de computação móvel tem se concentrado na obtenção de estados globais consistentes de aplicações distribuídas. Recuperação de transações para computação móvel ainda é um assunto em aberto. As características de ambiente que têm sido consideradas no projeto de algoritmos para obtenção de estados globais consistentes são:

- Durante uma computação, o computador móvel pode passar de uma célula para outra e deve-se, então, determinar onde o próximo estado local será armazenado;
- O computador móvel possui memória estável usada para armazenar estados consistentes;
- Existe um enlace sem fio disponível para comunicação com outras entidades.

3.6.1 ESTADO GLOBAL CONSISTENTE EM COMPUTAÇÃO MÓVEL

Recuperação de falhas em sistemas distribuídos é baseado no trabalho pioneiro de (CHANDY & LAMPORT, 1985) para obtenção de estados globais consistentes. Os algoritmos propostos armazenam periodicamente o estado da aplicação numa memória estável. Quando ocorre uma falha, a aplicação usa os estados armazenados (***Checkpoint***) ao longo da computação para retornar ao último estado global consistente e reiniciar a computação. Um estado global inclui o estado de cada processo participante na aplicação distribuída e o estado dos canais de comunicação. As duas condições básicas para recuperação de falhas em um sistema distribuído são:

- Consistência
  - Para que um checkpoint global se torne consistente é necessário que para qualquer mensagem "m", a seguinte condição seja satisfeita:

    Se o evento desfaz(m) está incluído ao ***Checkpoint*** global

    então o evento envia(m) também está incluído no ***Checkpoint***

- Recuperação
  - Para evitar perda de mensagens em trânsito, ou seja, mensagens que foram enviadas mas ainda não foram recebidas por nenhum processo, a seguinte condição deverá ser satisfeita:

    Se o ***Checkpoint*** global consistente contém o evento envia(m)

    mas não contém o evento desfaz(m)

    então o protocolo de recuperação deve salvar a mensagem m

Algoritmos de recuperação são normalmente classificados em dois tipos: coordenados e não coordenados. Algoritmos coordenados (CHANDY & LAMPORT, 1985) requerem que os participantes coordenem seus ***Checkpoints*** locais para garantir que seja possível obter um ***Checkpoint*** global consistente. Algoritmos não coordenados (STROM & YEMINI, 1985) (WANG & FUCHS, 1992) permitem que os processos façam o ***Checkpoint*** de seus estados locais de forma independente. Durante o processo de recuperação, deve haver uma coordenação para selecionar um ***Checkpoint*** de cada participante que leve a um ***Checkpoint*** global consistente.

Devido às características do ambiente móvel, estes dois tipos de algoritmos não podem ser aplicados diretamente a um sistema de computação móvel. Algoritmos coordenados enviam mensagens de controle para computadores móveis para fazer a sincronização de ***Checkpoint.***

Esses algoritmos devem considerar três pontos no caso de um ambiente móvel:

- Custo adicional para localizar o dispositivo móvel;
- O fato de o computador poder se deslocar para outra célula antes do processo de ***Checkpoint*** estar terminado;
- Como recuperar um ***Checkpoint*** global consistente quando houverem estações desconectadas.

Algoritmos não coordenados são mais adequados para computação móvel, pois permitem as unidades móveis fazerem o ***Checkpoint*** de seus estados locais sem trocar mensagens. No entanto, para recuperação do ***Checkpoint*** global consistente, é

necessário trocar mensagens entre os participantes, e os três problemas apontados acima aparecem nesse momento.

3.6.2 RECUPERAÇÃO DE FALHAS EM COMPUTAÇÃO MÓVEL

Recentemente, foram propostos alguns algoritmos coordenados (NEVES & FUCHS, 1997) (PRAKASH & SIHGHAL, 1996) e não coordenados (ACHARYA & BADRINATH, 1994) (PRADHAN et al, 1996) para recuperação de falhas em um ambiente de computação móvel. Esses algoritmos podem ser ainda classificados quanto ao grau de adaptabilidade e se a memória estável do computador móvel é considerada um lugar seguro para armazenar o estado local ou não. Isto leva a classificação das falhas em leves e graves. Uma falha leve não danifica permanentemente um computador móvel, como por exemplo, uma falha do sistema operacional ou falta de energia. Uma falha grave causa um sério problema a unidade móvel, como um problema permanente com a unidade de memória. Falhas leves são tratadas por ***Checkpoints*** que são armazenados no computador móvel, enquanto falhas graves devem ser tratadas por ***Checkpoints*** armazenados na rede fixa. ***Checkpoints*** armazenados localmente num computador móvel não precisam ser transmitidos através do enlace sem fio, são fáceis de serem gerados e armazenados, e não impedem que a unidade móvel continue a trabalhar mesmo durante uma desconexão.

Em (ACHARYA & BADRINATH, 1994) e (PRADHAN et al, 1996) são discutidos algoritmos não coordenados, onde a memória em disco do computador móvel é instável e, conseqüentemente, inadequada para armazenamento do estado do processo participante da computação. O algoritmo apresentado em (ACHARYA & BADRINATH, 1994) usa a abordagem de log imediato, onde o computador móvel cria um novo ***Checkpoint*** toda vez que recebe uma mensagem após também ter enviado uma mensagem. Este procedimento define uma "regra de duas fases", que devidamente aplicada, garante sempre a construção de um ***Checkpoint*** global consistente.

É também responsabilidade do computador móvel gerar um ***Checkpoint*** toda vez que muda de célula e antes de uma desconexão. Isto significa que o dispositivo móvel

deve saber continuamente em que célula se encontra e quando o sistema entrará no modo desconectado.

O algoritmo de recuperação de falhas registra todas as mensagens trocadas com outros processos no caso de ser necessário gerar um ***Checkpoint*** global consistente. Tanto o ***Checkpoint*** quanto o log são armazenados na estação base da célula onde o computador móvel se encontra no momento, que age como um agente para o computador móvel. Durante o processo de recuperação, que pode ser depois de uma falha ou desconexão, a estação base corrente, chamada de estação iniciadora, é responsável pelo processo de recuperação. Neste algoritmo, a estação base tem um papel ativo e o computador móvel um papel passivo. O ***Checkpoint*** global é obtido tomando como base o conjunto de ***Checkpoints*** consistentes locais. O algoritmo de recuperação usa um mecanismo para limitar o tamanho do arquivo de ***Log*** que pode crescer bastante, já que o processo de registro pode ser muito freqüente.

Em (PRADHAN et al, 1996) são propostos dois algoritmos não coordenados. O primeiro usa uma abordagem de ***Log*** imediato, onde o computador móvel cria um ***Checkpoint*** toda vez que recebe uma mensagem. O segundo usa uma abordagem de ***Log*** não imediato, onde ***Checkpoints*** são criados periodicamente. Nesse caso, quando o computador móvel cria um ***Checkpoint***, envia para o ***Log*** todas as mensagens recebidas e não registradas ainda. Os dois algoritmos supõem que ***Checkpoints*** e mensagens são armazenados na estação base corrente. Também em (PRADHAN et al, 1996) são propostos três mecanismos de ***Handoff***, usados para determinar onde armazenar e onde achar o ***Checkpoint*** local mais recente. Os dois algoritmos e os três mecanismos de ***handoff***, definem seis combinações de algoritmos de recuperação de falhas, que possuem características de desempenho dependentes das condições do ambiente. Nesse caso, é possível obter vários parâmetros de compromisso entre as diversas combinações.

Em (NEVES & FUCHS, 1997) e (PRAKASH & SIHGHAL, 1996), ao contrário dos algoritmos de (ACHARYA & BADRINATH, 1994) e (PRADHAN et al, 1996), assumem que o computador móvel possui uma memória estável segura e podem participar no processo de recuperação de falhas como se fossem computadores da rede fixa.

Em (IMIELINSKI et al, 1994), é apresentado um algoritmo de recuperação coordenado que ajusta o grau de uso da memória estável de um computador móvel em função das condições da rede ao longo do tempo.

Em (PRAKASH & SIHGHAL, 1996), o algoritmo proposto considera todos os participantes idênticos e trata de forma especial os ***Handoffs*** de estações móveis.

O algoritmo de (NEVES & FUCHS, 1997) é coordenado por um temporizador que, quando se esgota, faz com que o computador móvel crie um ***Checkpoint*** local independente dos outros processos. Os processos participantes da computação tentam fazer com que o momento em que o ***Checkpoint*** é executado seja o mesmo. Isto é feito informando em cada mensagem enviada para o outro processo, o intervalo de tempo que ainda falta para o próximo ***Checkpoint***. A recuperação é obtida registrando no emissor todas as mensagens enviadas mas não confirmadas no momento do ***Checkpoint***. Este algoritmo assume tanto falhas graves quanto leves e ajusta a taxa de ***Checkpoints*** na rede fixa e ***Checkpoints*** no equipamento móvel em função das condições do canal de comunicação. Por exemplo, se as condições do enlace sem fio não são boas, então o algoritmo gera mais ***Checkpoints*** no equipamento móvel antes de um criar um na rede fixa, ou usa os ***Checkpoints*** do equipamento móvel para se recuperar de uma falha leve, no caso do computador móvel estar desconectado.

No algoritmo proposto em (PRAKASH & SIHGHAL, 1996), só participam do processo de ***Checkpoint*** os computadores móveis que afetam direta ou indiretamente (ou seja, transitivamente) o último ***Checkpoint*** consistente. Os outros computadores móveis não participam. O ponto chave deste algoritmo é a informação de dependência entre computadores móveis, codificada como um vetor de ***Bits*** ao ser transmitida numa mensagem do sistema. O aspecto diferente deste algoritmo é que minimiza a comunicação no enlace sem fio, limitando o número de computadores móveis participantes no processo de ***Checkpoint***.

3.6.3 TRATAMENTO DA DESCONEXÃO

Similar a um SGBD distribuído convencional, um SGBD móvel deveria ser capaz de realizar uma recuperação de uma falta de comunicação a partir de uma determinada posição. Apesar do ambiente móvel estar mais propenso a falhas do que o ambiente computacional fixo (tomando suas devidas proporções), algumas das falhas são previsíveis (ou voluntárias), as quais possibilitam que se tome uma ação eficaz contra a falha (PITOURA & BHARGAVA, 1994). Uma ação possível que pode ser tomada prevendo uma desconexão poderia ser a transferência da transação ativa para um outro computador (provavelmente um fixo). Contudo, isto deveria preceder o resultado substancial de mensagem e transferência de dados entre os dois computadores. Em um ambiente móvel conectado deverá ser eleita a transação com maior criticidade para se fazer esta transferência, as outras transações assumem um estado de espera durante o período da desconexão sem interrompê-las. Outro método que possibilita as transações com maior prioridade continuar suas execuções, mesmo quando seu computador for desconectado, é o ***Cache*** de dados remoto, que pode ser acessado localmente por estas transações.

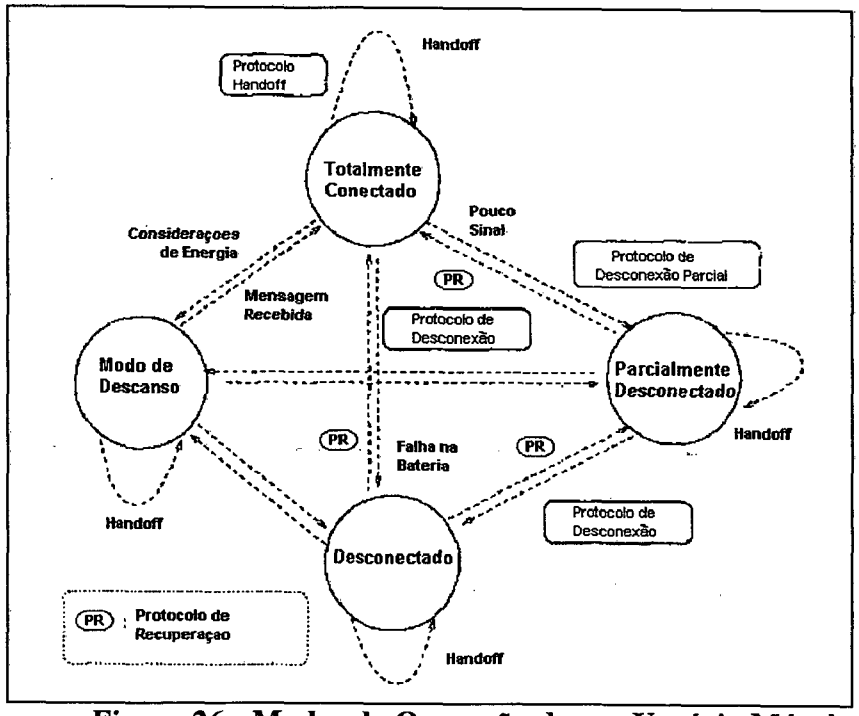


Figura 26 - Modos de Operação de um Usuário Móvel

Fonte: PITOURA, Evaggelia; BHARGAVA, Bharat. Building Information Systems for Mobile Environments, Maryland, ACM Press, Novembro, 1994.

A figura 26 mostra um digrama de estado que ilustra o modo de operação de um usuário móvel. A seguir está descrita a funcionalidade dos protocolos existentes no diagrama:

- Protocolo de desconexão: é executado antes da unidade móvel ser desligada fisicamente da rede fixa. O protocolo deve garantir informações suficientes para que a unidade móvel opere de forma autônoma durante a desconexão.
- Protocolo de desconexão parcial: prepara a unidade móvel para operar de forma que a comunicação com a rede fixa deve ser tão restrita quanto possível. O armazenamento seletivo pode ser usado para guardar dados cuja presença na unidade deverá minimizar o futuro uso da rede.
- Protocolo de recuperação: restabelece a conexão com a rede fixa e prossegue com a operação normal.
- Protocolo de ***Handoff***: refere-se a passagem de uma unidade móvel de uma célula para outra. As informações pertencentes a cada unidade móvel devem ser transferidas para a ERB da nova célula.

É possível também aumentar a reabilitação do ambiente de execução para transações de prioridade alta através da manutenção de seus registros de ***Logs*** locais no computador móvel e também em um ou mais computadores remotos fixos.

Além de transações prioritizadas, outro fator que pode ser levado em consideração em controle de processos de transações ativas é a natureza dos dados implícitos. Por exemplo, em um sistema de telecomunicação existem dois tipos de dados: dados de referência e dados de medição. Os dados de referência representam a configuração dos recursos da rede de telecomunicação. É importante a manutenção da integridade dos dados de referência e é necessário o suporte para transações sobre estes tipos de dados, para o registro e recuperação dos dados. Os dados de medição, ao contrário, são coletados da rede continuamente para identificar problemas. O volume dos dados de medição pode ser muito alto e o processamento de registros de ***Log*** deste tipo de dado pode levar muito tempo, não sendo tolerado em um ambiente móvel conectado. Além do requerimento de integridade destes dados não ser alto como para os dados de referência

e pode ser aceitável a perda de parte destes dados, desde que possa ser coletado dados similares mais tarde. Por esta razão, pode ser possível se ter vários níveis de integridade e estratégias de recuperação baseada nos tipos de dados processados.

3.7 ESTUDOS RELACIONADOS

A seguir, foi realizada uma síntese em alto nível de alguns projetos, que servem como um meio de demonstrar as pesquisas já realizadas sob o paradigma da computação móvel. Os sistemas, denominados BAYOU, ODYSSEY, ROVER e CODA, demonstram alguns conceitos de novos paradigmas para a computação cliente/servidor móvel. O projeto Bayou fornece uma arquitetura cliente/servidor flexível na qual um servidor pode ser qualquer máquina que suporta uma cópia completa de um ou mais bancos de dados. O projeto Odyssey suporta capacidade de adaptação baseada em operações de tipo específico. O projeto Rover fornece um conjunto de ferramentas para computação cliente/servidor baseada em objetos distribuídos. Os objetos relocáveis em Rover habilitam uma arquitetura cliente/servidor flexível para aplicações móveis. O projeto CODA é baseado em computação cliente/servidor utilizando uma arquitetura Cliente Completo através de sistema de arquivo.

3.7.1 BAYOU

BAYOU é um projeto de pesquisa XEROX PARC, e é projetado para aplicações colaborativas em um ambiente de computação móvel contendo máquinas portáteis com conectividade intermitente (DEMERS et al, 1994) (TERRY et al, 1994, 1995). O foco do sistema BAYOU foi explorar mecanismos que permitem que clientes móveis ativamente leiam e gravem dados compartilhados, como agenda de compromissos, bancos de dados bibliográficos, anotações de reuniões, documentos de planos de desenvolvimento, quadros de avisos e notícias, etc.

O projeto BAYOU suporta mecanismos de aplicação específica que detectam e resolvem os conflitos de atualizações, asseguram que réplicas alcancem consistência final e define um protocolo pelo qual a solução de conflito de atualizações pode ser

estabilizada (JING et al, 1999). Neste projeto são incluidos métodos de gerenciamento de consistência para detecção de conflitos, chamados checagem de dependência e solução de conflito por gravação baseada em procedimentos de consolidação determinados pelo cliente. Para garantir a consistência final, os servidores BAYOU são capazes de desfazer os efeitos de gravações executadas anteriormente e refazê-las de acordo com uma ordem de serialização global. Além disto, é permitido que clientes observem os resultados de todas as gravações recebidas por um servidor, incluindo tentativas de gravação cujos conflitos não foram resolvidos.

No sistema BAYOU, cada coleção de dados é replicada integralmente em um determinado número de servidores. Aplicações rodando como clientes interagem com os servidores através da API (Application Programming Interface) BAYOU, a qual é implementada como um alicerse do cliente apoiando a aplicação. Esta API, assim como o protocolo RPC (Remote Procedure Call – Chamada de funções remotas) cliente/servidor, suporta duas operações básicas: leitura e gravação. Operações de leitura permitem consultas sobre uma coleção de dados, enquanto operações de gravação podem inserir, modificar e apagar um número de itens de dados em uma coleção. A figura 27 mostra esses componentes da arquitetura BAYOU. No sistema BAYOU, um cliente e um servidor podem residir em um mesmo equipamento, como seria característico de um laptop ou PDA rodando isoladamente.

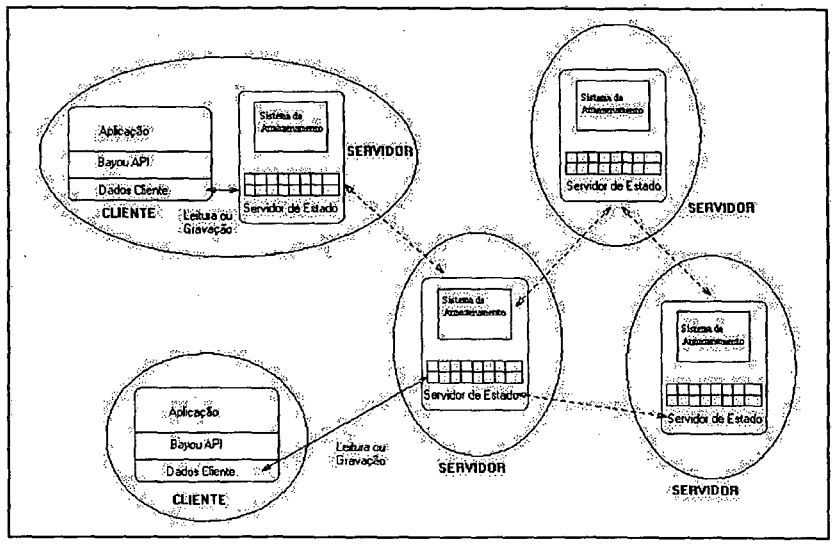


Figura 27 - Modelo do Sistema BAYOU

Enquanto as réplicas mantidas por dois servidores a qualquer momento podem variar em seus conteúdos por terem recebido e processado diferentes gravações, uma

propriedade fundamental do modelo BAYOU é que todos os servidores atingem consistência final (JING et al, 1999). Entretanto, ele não pode forçar amarrações rígidas no prolongamento da propagação de gravações, posto que essas dependências nos fatores de conectividade da rede que não são de controle do sistema Bayou. Duas importantes características do modelo de sistema Bayou permitem que servidores consigam consistência final. Primeira, gravações são executadas na mesma e bem definida ordem para todos os servidores. Segunda, a detecção de conflitos e procedimentos de consolidação são deterministas de modo que os servidores resolvem os mesmos conflitos da mesma maneira.

Quando uma gravação é aceita por um servidor BAYOU vinda de um cliente, ela é inicialmente considerada temporária. Gravações temporárias são solicitadas de acordo com a fatia de tempo designada a elas pelo servidor que as está aceitando. As gravações são solicitadas de acordo com os tempos em que elas executam e antes de qualquer gravação temporária. O único requisito posto em fatia de tempo para gravações temporárias é que elas deveriam ser unicamente incrementadas em cada servidor, de modo que a fatia de tempo e a identificação do servidor que a designou produzisse uma solicitação total em operações de gravação. Servidores BAYOU mantêm relógios lógicos para fatia de tempo para novas gravações. O relógio lógico de um servidor é geralmente sincronizado com seu relógio de sistema de tempo-real, mas, para preservar a ordem causal de operações de gravação, o servidor pode precisar adiantar seu relógio lógico (JING et al, 1999).

Forçar uma ordem global em tentativa, assim como gravações executadas, assegura que um grupo isolado de servidores entrará em concordância na resolução de tentativa de quaisquer conflitos que encontrarem. Isto não é estritamente necessário, posto que clientes precisam estar preparados para lidar com servidores temporariamente inconsistentes em qualquer caso. Além disso, clientes podem esperar que a resolução de tentativa de conflitos dentro de seu grupo corresponderá a sua resolução permanente final, fornecida de tal forma que nenhum outro conflito é introduzido fora do grupo. Isto porque os grupos podem receber gravações de clientes e de outros servidores numa ordem que difere da ordem de execução requisitada, e porque os servidores imediatamente aplicam todas as gravações conhecidas a suas réplicas e precisam estar

aptos a desfazer os efeitos da execução da tentativa anterior de uma operação de gravação e reaplicá-la em uma ordem diferente. O número de vezes que uma determinda operação de gravação é reexecutada depende apenas da ordem na qual gravações chegam diferente da ordem esperada e não da probabilidade de conflitos envolvendo gravações. Conceitualmente, cada servidor mantém um log de todas as operações de gravação. O conteúdo dos dados atuais do servidor são gerados pela execução de todas as gravações na ordem dada.

O projeto BAYOU garante que procedimentos de consolidação, os quais são executados independentemente em cada servidor, produzem atualizações consistentes forçando-as a depender apenas do conteúdo dos dados atuais do servidor e de quaisquer dados fornecidos pelo próprio procedimento de consolidação. Além disso, os procedimentos de consolidação que falham por exceder seus limites usando determinado recurso, precisam falhar deterministicamente. Isso significa que todos os servidores precisam colocar amarrações uniformes nos recursos de CPU e de memória alocados a um procedimento de consolidação e precisam consistentemente reforçar aquelas amarrações durante a execução. Uma vez que essas condições são encontradas, dois servidores que começam com réplicas idênticas irão terminar com réplicas idênticas depois de executar uma gravação.

3.7.2 ODYSSEY

ODYSSEY é um projeto de pesquisa CMU, que se utiliza da capacidade de adaptação para lidar com diversidades de aplicações em ambientes móveis. A capacidade de adaptação é implementada como suporte de operações de tipo específico coordenadas pelo sistema. Ele suporta execução concorrente de diversas aplicações móveis que executam em clientes móveis, mas lêem ou atualizam dados remotos em servidores (KUMAR & SATYANARAYANAN, 1993) (SATYANARAYANAN et al, 1995) (NOBLE et al, 1995, 1997).

No projeto ODYSSEY, o dado acessado por uma aplicação pode estar armazenado em um ou mais repositórios de propósito geral como: um servidor de arquivos; servidores SQL; ou servidores WEB. Pode também estar armazenado em mais

repositórios especializados, como vídeo bibliotecas, bancos de dados de consulta pelo conteúdo da imagem ou sistemas de informação geográfica.

Em condições ideais, um ítem de dados disponível em um cliente móvel deveria ser igual àquele disponível para as aplicações que acessam o servidor que armazena aquele dado. Mas essa correspondência pode ser difícil de preservar, posto que os recursos móveis são limitados e alguma forma de degradação pode ser inevitável. Em ODYSSEY, fidelidade é usada para descrever o grau ao qual dados representados em um cliente está compatível com a cópia de referência no servidor. Fidelidade tem muitas dimensões. Uma bem conhecida dimensão universal é a consistência.

Odyssey implementa uma capacidade de adaptação (JING et al, 1999). O recurso monitor do sistema balanceia, notifica aplicações de alterações relevantes e reforça decisões de alocação de recursos. Cada aplicação independentemente decide a melhor maneira de se adaptar quando notificada. Odyssey incorpora um tipo de aviso via componentes de código especializados, chamados fiscalizadores. Um fiscalizador encapsula o suporte a nível de sistema em um cliente necessário para efetivamente gerenciar um tipo de dado. Para comportar inteiramente um novo tipo de dado, um fiscalizador apropriado tem que ser escrito e incorporado ao Odyssey em cada cliente. Os fiscalizadores são subordinados a um componente do tipo independente, chamado vice-rei, o qual é responsável pelo gerenciamento centralizado (veja a figura 28). A relação colaborativa na capacidade de adaptação é realizada em duas partes. A primeira, entre o vice-rei e seus fiscalizadores, é centralizada nos dados: define o nível de fidelidade para cada tipo de dado e os fatores do gerenciamento de recursos. O segundo, entre aplicações e Odyssey, é centralizado na ação: fornece aplicações com controle sobre a seleção de níveis de fidelidade comportada pelos fiscalizadores.

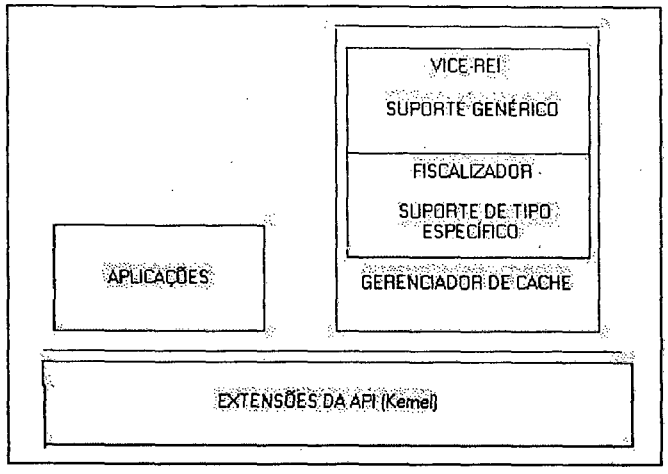


Figura 28 - Arquitetura do Cliente ODYSSEY

3.7.3 ROVER

ROVER é um projeto de pesquisa do MIT. O conjunto de ferramentas Rover oferece um ambiente para comportar tanto capacidade de adaptação quanto transparência de aplicações em ambiente cliente/servidor móvel (JOSEPH & KAASHOEK, 1996) (JOSEPH et al, 1997). A transparência de aplicações é realizada desenvolvendo ***Proxys*** para serviços de sistema que escondem as características móveis do ambiente nas aplicações. A capacidade de adaptação é obtida pelo uso de objetos dinâmicos relocáveis na construção das aplicações de cliente e servidor. O conjunto de ferramentas ROVER fornece um sistema para construir aplicações móveis com arquitetura cliente/servidor flexível. Clientes ROVER são aplicações que rodam normalmente em equipamentos móveis, mas podem também rodar em equipamentos fixos. Servidores ROVER podem ser replicados e normalmente rodam em equipamentos fixos e guardam o estado do sistema. A comunicação entre clientes é limitada a interações ponto-a-ponto entre o equipamento móvel (usando o ***Cache*** de objeto local para compartilhamento) e o servidor, não há suporte para interações ponto-a-ponto de equipamento móvel para equipamento móvel. O conjunto de ferramentas ROVER fornece suporte a comunicação móvel baseado em duas idéias: RDO (Relocatable Dynamic Object - Objeto Dinâmico Relocável); e QRPC (Queued Remote Procedure Call - Chamadas de Funções Remotas Serializadas). Um objeto dinâmico relocável é

composto de código e dados comuns a uma interface que pode ser carregada dinamicamente em um computador cliente a partir de um computador servidor, ou vice-versa, para reduzir solicitações de comunicação cliente/servidor. Chamada de funções remotas serializadas é um sistema de comunicação que permite as aplicações continuarem fazendo chamadas de procedimentos remotos sem bloqueio, mesmo quando o equipamento estiver desconectado – solicitações e respostas são trocadas sob reconexão da rede.

O projeto ROVER fornece controle de aplicações sobre a localização onde a computação será executada. Em um ambiente conectado intermitentemente, a rede freqüentemente separa uma aplicação dos dados sobre os quais é independente. Movendo RDOs pela rede, aplicações podem mover dados e/ou computação do cliente para o servidor e vice-versa (JING et al, 1999).

O uso de RDOs permite que em uma conexão de rede lenta, aplicações para ambientes móveis migrem funcionalidades dinamicamente para um dos dois lados, minimizando o montante de dados transportados pela rede. Fazer ***Cache*** de RDOs reduz a latência e consumo da largura da banda. Funcionalidades da interface podem rodar em velocidade máxima em um equipamento móvel, enquanto manipulações de grande quantidade de dados podem ser executadas no servidor fixo. Todo o código da aplicação e todo dado manipulado pela aplicação são gravados como RDOs. Cada RDO tem um servidor base que mantém a cópia primária. Os clientes importam cópias secundárias dos RDOs em seus ***Caches*** locais e exportam temporariamente RDOs atualizados de volta para seus servidores base. A complexidade dos RDOs pode variar desde ítens de agendas simples com um pequeno grupo de operações, até módulos que encapsulam uma importante parte de uma aplicação (por exemplo, a interface de usuário gráfica para um browser de e-mail).

Clientes ROVER usam QRPC para fazer a busca nos servidores de RDOs (veja a figura 29). Quando uma aplicação produz uma QRPC, ROVER armazena a QRPC em um log fixo local e imediatamente retorna o controle para a aplicação. Se a aplicação registrou uma rotina de retorno, então quando o RDO solicitado tiver chegado, Rover irá executar um comando de retorno para notificar a aplicação. Alternativamente,

aplicações podem simplesmente bloquear para esperar por dados críticos (apesar de que isso é uma ação indesejável, especialmente quando o equipamento móvel está desconectado). Quando o host móvel está conectado, o escalonador de rede ROVER esvazia o log numa área secundária, transferindo qualquer QRPC enfileirado para o servidor. Quando uma aplicação Rover modifica um RDO ***Cache*** localmente, a cópia ***Cache*** é marcada temporariamente como executada. Atualizações são executadas usando QRPCs para lentamente propagar as operações modificadas para o servidor Rover, onde elas são aplicadas às cópias legítimas. Enquanto isso, a aplicação pode optar por usar RDOs executados temporariamente. Isto permite que a aplicação continue a execução mesmo que o host móvel esteja desconectado.

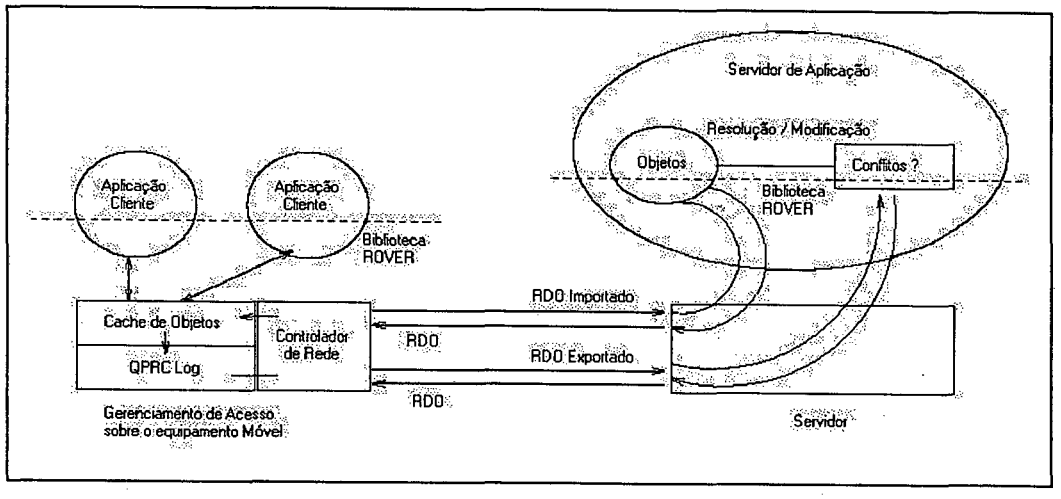


Figura 29 - Arquitetura do RDO ROVER

Conforme mostrado na figura 30, o conjunto de ferramentas Rover consiste de quatro componentes básicos: o gerenciador de acesso, o objeto ***Cache*** (somente cliente), o log de operação e o escalonador de rede. Cada máquina tem um gerenciador de acesso ROVER local, o qual é responsável por manipular todas as interações entre as aplicações cliente/servidor e entre as aplicações clientes.

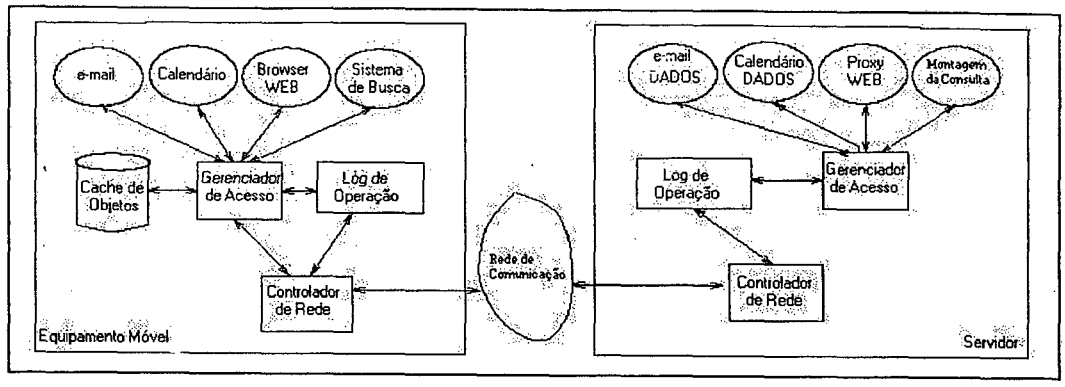


Figura 30 - Arquitetura de Componentes ROVER

O serviço de gerenciamento de acessos solicita RDOs, intermedia o acesso a rede, faz log de modificações para objetos e no cliente gerencia o objeto ***Cache***. Aplicações do cliente comunicam-se com o gerenciador de acessos para importar objetos dos servidores e os armazena em ***Cache*** localmente. Aplicações do servidor são chamadas pelo gerenciador de acessos para tratar solicitações de aplicações dos clientes. Aplicações invocam os métodos fornecidos pelos objetos e, usando o gerenciador de acessos fazem as mudanças visíveis globalmente exportando-as de volta para os servidores. No gerenciador de acessos, RDOs são importados para o objeto ***Cache*** enquanto QRPCs são exportadas para o log de operações. O gerenciador de acessos encaminha solicitações e respostas entre aplicações, o ***Cache*** e o log de operações. O log é esvaziado pelo escalonador de rede, o qual intermedia entre vários protocolos de comunicação e interfaces de rede.

O objeto ***Cache*** fornece armazenamento fixo para cópias locais de objetos importados. Consiste de um ***Cache*** privado local dentro do espaço de endereçamento da aplicação, e um ***Cache*** compartilhado global localizado dentro do espaço de endereçamento do gerenciador de acessos. Aplicações do cliente normalmente não interagem diretamente com o objeto ***Cache***. Quando uma aplicação do cliente solicita uma operação de importação ou exportação, o conjunto de ferramentas satisfaz a solicitação dependendo do objeto encontrar-se num ***Cache*** local e da opção de consistência especificada para o objeto. Uma vez que um objeto foi importado para o espaço de endereçamento local da aplicação do cliente, solicitações do método sem

efeitos de mudança são atendidas localmente pelo objeto. Com o cuidado da aplicação, solicitações do método com efeitos de mudança podem também ser processados localmente, inserindo dados temporários nos objetos ***Cache***. Operações com efeitos de mudança também inserem uma QRPC no log de operações fixo localizado no cliente. Cada inserção é uma ação síncrona. O suporte para conectividade a rede intermitente é efetuado permitindo que o log progressivamente limpe a memória de volta para o servidor. Dessa forma, o cliente irá progredir alcançando um estado consistente.

O escalonador de rede contribui para a otimização da transmissão de log, agrupando operações destinadas ao mesmo servidor para transmissão, selecionando o protocolo de transporte apropriado e a mídia para enviá-los. ROVER é capaz de usar uma variedade de transportes de rede, pois suporta protocolos baseados em conexão (por exemplo, HTTP sobre redes TCP/IP) e protocolos sem conexão (por exemplo, SMTP sobre redes IP ou não IP). O escalonador de rede alavanca o enfileiramento de QRPCs executadas pelo log para aumentar a eficiência da transmissão.

3.7.4 CODA

O projeto CODA serviu como base de vários estudos (JING et al, 1999) relacionados a operações desconectadas. Este sistema trata a transparência de aplicações em ambiente cliente/servidor móvel, utilizando a arquitetura de Cliente Completo em sua definição. Dessa forma, os dados e as aplicações estão contidos no equipamento móvel e é utilizado replicação de dados para atualizar as informações, veja a figura 31. A seguir as características deste projeto:

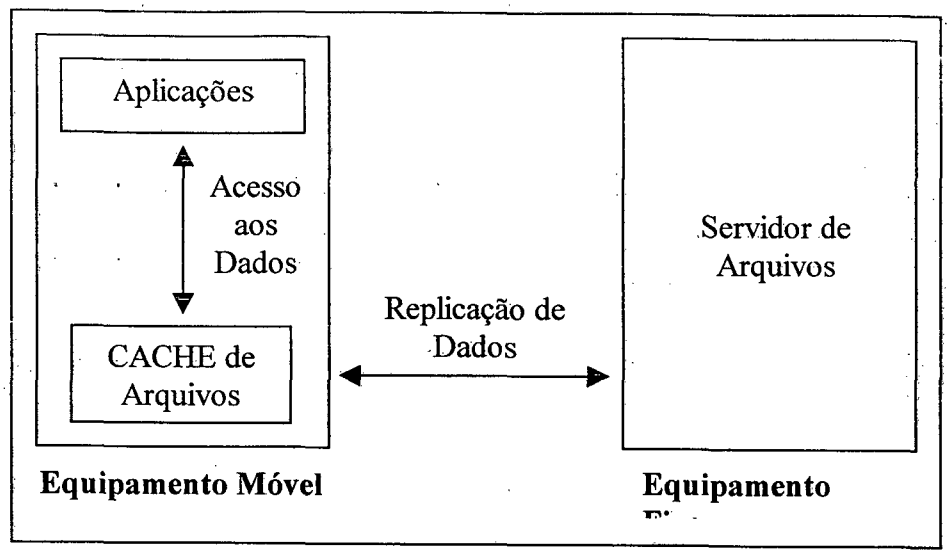


Figura 31 - Modelo de Acesso aos Dados CODA

- Operação desconectada: A operação desconectada é possível, efetiva e aproveitável em sistemas de arquivo Unix distribuídos. Os mecanismos chave para sustentar operação desconectada incluem ***Hoarding*** (gerenciamento de ***Cache*** assistido por usuário), log de atualizações com otimizações extensivas enquanto estiver desconectado e reintegração após reconexão (KLISTER & SATYANARAYANAN , 1992).

- Replicação Otimista: CODA foi um dos primeiros sistemas a demonstrar que uma estratégia de controle de réplica otimista pode ser usada para computação móvel séria e prática. Ele incorpora inúmeros mecanismos novos para tornar esta metodologia viável. Isto inclui resolução de diretório baseado em log, resolução de arquivo para aplicação específica e mecanismos para detecção, detenção e reparo manual de conflitos.

- Suporte para conectividade fraca: A conectividade fraca pode ser explorada para aliviar as limitações de operação desconectada. Os mecanismos necessários para efetuar isto incluem - protocolos de transporte adaptáveis - mecanismos de validação de ***Cache*** rápido - mecanismos de reintegração sequencial para propagação de atualizações - e a utilização de um modelo

baseado em ***Cache*** para a recuperação de falhas (SATYANARAYANAN , 1996).

- Transações ***Isolation-Only***: No contexto de CODA, uma nova abstração chamada transação ***Isolation-Only*** foi desenvolvida para lidar com a detenção e tratamento de conflitos de leitura e gravação durante operação desconectada. Esta abstração seletivamente incorpora conceitos de transações de banco de dados, enquanto faz demanda mínima de clientes móveis com poucos recursos e preserva compatibilidade com aplicações Unix (JING et al, 1999).
- Replicação do Servidor: As replicações do servidor podem ser usadas para complementar a operação desconectada. Apesar de isto não ser particularmente relevante para mobilidade, é um resultado importante em sistemas distribuídos, pois esclarece a relação entre réplicas de primeira-classe (por exemplo, servidor) e réplicas de segunda-classe (por exemplo, ***Caches*** de cliente). Também representa uma das primeiras demonstrações de replicação otimista aplicada a sistemas distribuídos com o modelo cliente/servidor.

3.7.5 RESUMO COMPARATIVO

A seguir é mostrado um resumo comparativo entre os sistemas apresentados nas seções anteriores:

Sistema	Aplicações	Adaptação	Modelo	Dados Móveis
BAYOU	Aplicações colaborativas que não são em tempo real: escalonador de sala de reuniões, banco de dados bibliográfico, agendas de compromissos, documentos de planos de desenvolvimento, quadros de avisos e notícias	Adaptação de aplicação específica a desconexão & conexão intermitente. É permitido que as aplicações façam permuta de consistência dos dados replicados & disponibilidade usando individualmente garantias de sessões selecionáveis	Arquitetura cliente/servidor baseada em grupos flexíveis e colaborativos e arquitetura Cliente Completo.	Suporte do sistema para detecção de conflitos de atualizações, resolução de aplicação específica de conflitos de atualizações, convergência de réplica e garantias de consistência por cliente
ODYSSEY	Acesso de sistema de arquivos, exibir vídeos e ***Browse Web***	Adaptação de aplicação baseado no cliente com o suporte do sistema que fornece monitoração de recursos, notifica aplicações de mudanças de recursos e assegura decisão de alocação de recursos	Arquitetura cliente/servidor clássica	Distribuição de dados baseados em importação pelos clientes (cópias retificadas)
ROVER	e-mail, compromissos, listas de tarefas, lembretes, agendas, páginas e imagens ***Web***	Adaptação de aplicações cliente/servidor com o uso do conjunto de ferramentas ROVER que trata com conexão intermitente, de largura de banda baixa e clientes com poucos recursos. Adaptação através de aplicações usando ***Proxys*** ROVER	Arquitetura cliente/servidor flexível com o uso de RDOs	Importação e exportação de RDOs assíncronos
CODA	Acesso de Sistema de arquivos UNIX	Adaptação de aplicação com suporte a Replicação Otimista, e suporte a Conectividade Fraca	Arquitetura cliente/servidor baseada em no modelo Cliente Completo	Convergência de réplicas

Tabela 4 – Comparativo entre Sistemas Móveis

CAPÍTULO IV

4 REPLICAÇÃO DE DADOS EM SGBD MÓVEL

Usuários de computadores móveis, como ***Palmtops***, ***Notebooks*** e sistemas de comunicação pessoal em breve terão acesso online com um grande número de bases de dados via redes sem fio. O mercado em potencial para esta atividade está estimado em bilhões de dólares anualmente em taxas de acesso e comunicação. Por exemplo, quando em viagem, os passageiros acessarão quadros de horários de linhas aéreas e outros meios de transporte, e informações das condições do tempo. Investidores vão acessar dados de indicadores financeiros, vendedores acessarão dados de estoque, usuários de telefone acessarão dados dependentes da localização (ex.: onde fica o ponto de táxi mais próximo) (BADRINATH & IMIELINSKI, 1992) e carros com computadores que traçam rotas acessarão informações sobre o trânsito.

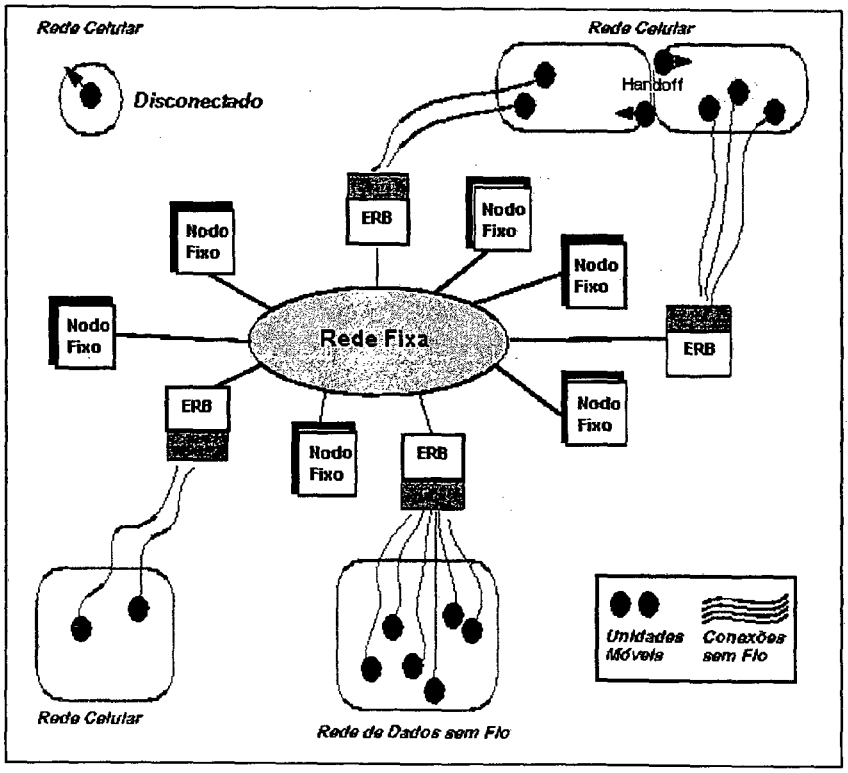


Figura 32 - Arquitetura da Gerência de Sistemas Móveis

Por causa da largura de banda limitada, a comunicação sem fio é mais cara do que a comunicação convencional. Está claro que, para os usuários que fazem centenas de acessos por dia, a comunicação sem fio pode se tornar muito cara.

Portanto, é importante que os computadores móveis acessem as bases de dados locais de maneira a minimizar a comunicação (HUANG & SISTLAWOLFSON, 1994). Isto pode ser obtido por meio de alocação de dados apropriada. Por exemplo, se um usuário freqüentemente lê um item de dado X, e X é atualizado raramente, então é benéfico para o usuário alocar uma cópia de X no seu computador móvel. Em outras palavras, o usuário móvel habilita-se para receber todas as atualizações de X. Deste modo, as leituras acessam a cópia local e não requerem comunicação. As atualizações pouco freqüentes são transmitidas da base de dados online para o computador móvel. Em contrapartida, se o usuário lê X pouco freqüentemente em relação à taxa de atualização, então não deve alocar uma cópia de X para o computador móvel. Em vez disso, o acesso deve ser por demanda; cada pedido de leitura deve ser enviado ao computador fixo que armazena a base de dados online.

Na computação móvel, a área geográfica é geralmente dividida em células, cada uma das quais com um controlador fixo. Nosso computador fixo não deve ser confundido com o controlador fixo. O computador fixo é um nó na rede fixa que é definido por um ítem de dado fornecido e que não se altera quando o computador móvel se move de célula para célula (veja a figura 32).

Os computadores portáteis e os avanços na comunicação sem fio proporcionaram um processamento móvel da informação. O serviço móvel é sensível ao contexto e dependente da localização, considerando as características ambientais e a relação dos usuários móveis com a informação de que necessitam. Isto implica em diferenciar consultas e gravações a um banco de dados baseado, por exemplo, nos objetivos, preferências, conhecimentos e habilidades do usuário, além da localização e equipamento utilizado (onde e quando o dado é usado). Porém, é necessário que se leve em consideração as restrições relativas ao ambiente móvel. Tais considerações devem levar em conta as informações de infra-estrutura disponível, recursos móveis, conectividade, custos, duração das conexões e largura da banda. Resumindo, o contexto móvel envolve:

- Fatores humanos, suas tarefas, regras, outras pessoas;
- Localização (e a mudança desta no momento);
- Hardware e software (nodo móvel e características da rede, equipamentos e ferramentas);
- Informação, características da aplicação (como tipo de mídia, tamanho)

Usuários móveis acessam dados que são também acessados por outros usuários ou por eles mesmos em diferentes locais e equipamentos, respectivamente. Devido a conexões raras, esporádicas e fracas, uma cópia local precisa ser mantida no equipamento móvel para evitar o estabelecimento de uma conexão a cada consulta/gravação, facilitando o trabalho do usuário móvel em modo desconectado (LUBINSKI & HEUER, 2000). Tal cópia incrementa a disponibilidade, diminui o tempo de resposta e também melhora a tolerância a falhas. Replicação de dados é um método muito conhecido em sistemas de banco de dados distribuídos, é uma cópia de controle de fragmentos do banco de dados em diferentes nodos, de tal forma que cada réplica é uma instância do fragmento do banco de dados (de modo geral, existem cópias sem existir qualquer original). O dado replicado tem que ser mantido também no servidor fixo. Entretanto, esta consistência tem que ser certificada, sem conexões freqüentes ao servidor – por razões de custo, entre outros.

A Replicação de dados é um dos problemas chave de sistemas de informação móveis. Um tratamento aplicável de replicação neste novo paradigma da computação precisa de soluções novas que os protocolos de replicação existentes não oferecem (LUBINSKI & HEUER, 2000). Bancos de dados comerciais e sistemas de informação atualmente em uso em ambientes móveis sobrecarregam o usuário com a manutenção e consistência dos dados replicados. Para iniciar um serviço móvel, o DBA (Administrador de Banco de Dados) reúne o dado a ser usado no equipamento móvel e o transfere do servidor. A seguir, o DBA ajusta os parâmetros para especificar quais ações são possíveis sobre o dado em particular tanto no cliente móvel quanto no servidor. O usuário móvel tem que contar com a consistência dos dados enquanto trabalha em num ambiente desconectado. Se novos ou mais dados atuais são requisitados, o usuário móvel precisa conectar-se ao servidor para transferir os dados. Quando consulta o sistema de

informação móvel em um ambiente desconectado, o usuário tem que avaliar a atualização e a consistência do dado sem ajuda do sistema. Checagens de consistência são possíveis quando os dados são transferidos de volta ao servidor depois de terminar o serviço móvel. Qualquer conflito subsequente na consistência precisa ser manualmente resolvido pelo DBA.

A fim de evitar funcionalidade, atualização e consistência restritas, as técnicas de replicação usadas em ambientes móveis precisam atingir os seguintes requisitos (LUBINSKI & HEUER, 2000):

- Alta disponibilidade de dados no equipamento móvel, a despeito de baixos custos e uma consistência de dados aceitável;
- Uso de informação de contexto semântica sobre o usuário, atributos da aplicação e do dado, a fim de executar checagem de consistência significativa a baixos custos;
- Uso de informação de contexto semântica para realizar replicação dinâmica, que é carga transparente de informação a ser atualizada do servidor, a pedido (quando consulta o sistema móvel, por exemplo).

4.1 OPERAÇÃO DISTRIBUÍDA

Em um sistema de banco de dados, os dados armazenados podem estar divididos em um ou mais bancos de dados. Em muitas instalações, parte ou a totalidade dos dados corporativos residem em diferentes computadores, diferentes cidades ou diferentes países. Esta forma de armazenamento pode ser denominada de "Opção Distribuída". A opção distribuída permite que esse cenário se torne realidade. Existe transparência na localização dos dados de tal forma que o usuário quando está acessando alguma informação, não se preocupa onde a mesma está armazenada.

Em um sistema computacional móvel, ter acesso à informação em qualquer lugar e a qualquer momento, pode ser muitas vezes imperativo para que se possa garantir um diferencial competitivo. Esta informação pode ser qualquer dado que seja considerado significativo e que possa ser servido pelo sistema. Em outras palavras, qualquer

informação necessária ao processo operacional ou decisório. Nem sempre é possível realizar uma conexão para se buscar uma determinada informação. Neste caso, a opção distribuída pode ajudar a mobilidade do sistema. A principal vantagem desta implementação é o fato de dispensar a utilização de um meio de comunicação para enviar uma determinada consulta e depois receber os dados de volta. Sendo assim, a consulta é feita na imagem local do banco de dados, o que agiliza a resposta das informações requeridas, aumentando assim a performance e a interoperabilidade do sistema.

A troca de informações entre unidades remotas sugere a descentralização das informações, de acordo com as necessidades e vantagens que se espera deste método de operação. Para se implementar um projeto de distribuição de dados são utilizadas estratégias de replicação. Desta forma, pode-se atualizar uma tabela existente em diferentes instâncias de bancos de dados, além daquele que está sofrendo uma "inserção", "atualização" ou "exclusão" de informações, fazendo com que estas tabelas permaneçam consistentes em todos os Bancos de Dados (ABBEY & COREY, 1997). A atualização das réplicas acontece periodicamente, de acordo com a necessidade de atualização da informação.

Dependendo da característica do projeto e os custo de comunicação envolvidos, poderá ser desejável que tais sistemas trabalhem com operações desconectadas, resultando em um modelo assíncrono de replicação de dados. A atualização das réplicas é sempre permitida, mesmo estando a unidade móvel desconectada da rede de comunicação. Com isto, a idéia é proporcionar ao sistema alta disponibilidade de utilização dos dados, mesmo quando estiver trabalhando de forma desconectada.

4.2 LOCALIZAÇÃO

A mobilidade de usuários e serviços e seu impacto na replicação e migração de dados será um dos principais problemas técnicos a serem resolvidos. Provavelmente a mais desafiadora inovação em sistemas distribuídos do futuro.

O novo elemento distinto que a mobilidade traz às publicações de replicação de dados é a incerteza do sistema a respeito de seu próprio estado (significando em primeiro lugar a localização). Dessa forma o custo de comunicação entre dois usuários móveis depende não somente da distância entre eles, mas também do custo de busca necessário para determinar a localização exata[14] da unidade móvel (BADRINATH & IMIELINSKI, 1992).

Eis algumas questões que surgiram com a mobilidade de usuários e serviços:

- Quais são as condições sob as quais precisamos replicar dados para um nodo móvel?
- As informações a respeito de localização deveriam também ser replicadas e a que nível (localização é um ítem de dados dinamicamente alterado a cada momento)
- Como os movimentos dos usuários afetam o esquema de replicação? Como a cópia deveria seguir o usuário?

4.2.1 ESQUEMAS DE LOCALIZAÇÃO

Nos dias atuais a configuração de sistemas celulares consiste em uma rede de informação fixa estendida com elementos sem fio necessários para acesso sem fio à parte fixa da rede. Estes elementos incluem: unidades móveis sem fio, estações rádio base e comutadores. Cada estação rádio base cobre uma área chamada célula, na qual se

[14] O sistema pode ser fisicamente móvel mas logicamente estático (ou quase estático): isto acontece quando apesar de todos os componentes serem móveis, suas posições serem conhecidas todo (ou quase todo) tempo. Por exemplo, quando dois usuários viajam de avião, apesar de eles estarem em movimento, suas posições relativas são conhecidas e não mudam durante o período do vôo. Em tal caso, na maior parte do tempo o sistema tem uma informação completa sobre a localização de tais usuários.

comunica com unidades móveis sem fio pela largura de banda alocada pelo espectro do rádio (GOODMAND, 1991). Na figura 33 é mostrado que um conjunto L de servidores de localização (comutadores) conectados entre eles mesmos e a estação rádio base por uma rede convencional. Existem cerca de 60-100 estações rádio base sob um servidor de localização. Servidores de localização estabelecem chamadas aos usuários móveis pelas estações rádio base local (MEIER-HELLSTERN & ALONSO, 1991).

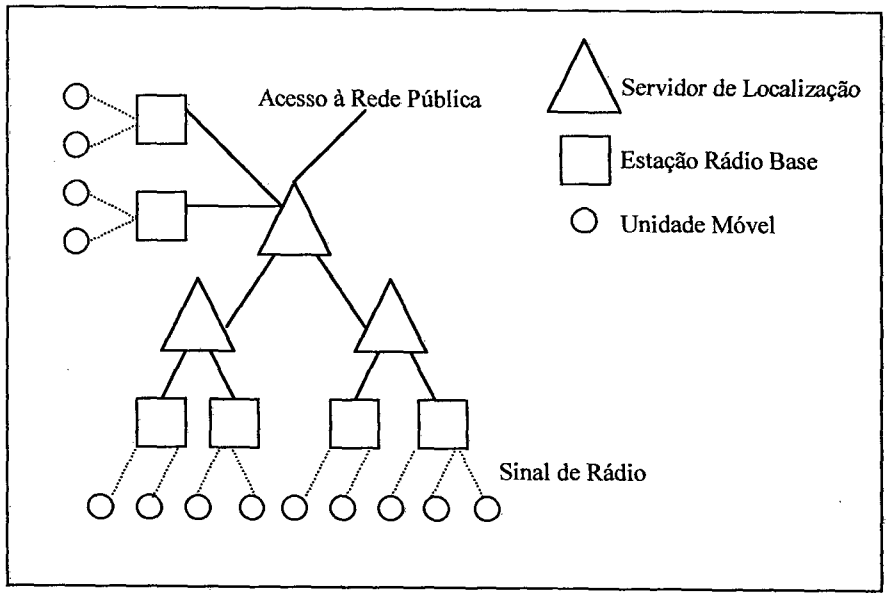


Figura 33 - Arquitetura da Rede Celular

Cada usuário (algumas vezes chamado unidade móvel) será permanentemente registrado sob um dos servidores de localização. A área coberta pelo servidor de localização será chamada área local do usuário; adicionalmente, ele pode também registrar-se como visitante sob algum outro servidor de localização. As alternativas enumeradas a seguir são normalmente consideradas para localizar um usuário (BADRINATH & IMIELINSKI, 1992):

- Paginação (paging): o servidor de localização local transmite a mensagem de paginação para todas as suas estações rádio base. A estação rádio base, sob a qual o usuário está localizado no momento, responde a mensagem de paginação e, usando o canal de rádio especial, envia outra mensagem de paginação para o qual o usuário móvel responde, estabelecendo a conexão definitiva. O custo depende do número de estações rádio base paginadas e não da mobilidade do usuário.

- Endereços de remessa: o custo de busca depende do número de movimentos do usuário, ou seja, o usuário se move para a próxima célula, um endereço de remessa (apontador) é registrado na estação rádio base prévia. A busca então envolve apontadores seguidos de estação rádio base para estação rádio base a fim de localizar o usuário.

- Busca Incremental: o custo depende da diferença entre a posição antiga e a posição nova (quanto mais longe o usuário tenha se movido de sua posição anteriormente conhecida, mais cara é a busca). Por exemplo, busca incremental do servidor de localização local, busca sucessiva de servidores de localização nível mais alto começando com o servidor de localização local ao mais baixo nível.

Além disto, usuários móveis irão causar a si próprios um custo de depender de quem é responsável pela informação da localização (BADRINATH & IMIELINSKI, 1992). As possibilidades são:

- O que está em movimento (a pessoa que o chamador quer alcançar - chamado) informa sua posição atual a um servidor de localização;

- O que está buscando (chamador) tem que descobrir a localização cada vez e o custo é maior;

- O chamado informa apenas uma vez em tantos movimentos, o chamador pode ter que rastrear a posição atual baseado na última localização conhecida.

Tipicamente, o custo de estabelecer a localização e conexão a um usuário móvel dependerá do esquema de endereçamento usado. De maneira geral, entretanto, este custo será bem maior que o custo de acessar o servidor de localização estático da unidade móvel. Por exemplo, se o esquema de paginação é usado (assumindo aproximadamente 60 estações rádio base), o custo de localização de uma unidade móvel será muitas vezes maior que o custo incutido pela comunicação do móvel para o servidor de localização. Esta assimetria do custo de comunicação é causada principalmente pela necessidade de busca enquanto está tentando localizar uma unidade móvel, enquanto nenhum custo de

busca é incutido quando a comunicação é feita na direção oposta. Isto terá um impacto importante na replicação.

Outra arquitetura radicalmente diferente proposta recentemente no projeto Iridium da Motorola consiste de 77 satélites com muitas estações rádio base fornecendo serviços de comunicação celular para centenas de milhares de assinantes simultaneamente. Os satélites podem ser considerados como servidores móveis, movendo-se na órbita em torno da Terra, capazes de fornecer acesso instantâneo a uma rede mundial de usuários.

4.2.2 REPLICAÇÃO DE DADOS UTILIZANDO RECURSOS DE LOCALIZAÇÃO

Como já descrito, a mobilidade tem um impacto importante na replicação quando consideramos replicar sobre uma plataforma móvel. Considerando um servidor S e um cliente C, um item X é gravado pelo servidor S e lido pelo cliente C. Considere uma rede hierárquica simples com N níveis de servidores de localização. Considere um cenário simples de um único servidor e um único cliente que se move inteiramente em seus servidores de localização locais. Além disto, assumiremos que o servidor tem uma cópia dos dados e faz as gravações. O cliente lê uma cópia dos dados. Os lugares possíveis para uma cópia dos dados residir são: no servidor, no cliente ou nos servidores de localização para o cliente ou servidor, como mostrado na figura 34. Também é possível ter uma cópia ***Cache*** dos dados. Em tal caso, o servidor envia apenas mensagens de invalidação para a cópia. O cliente, acessando uma cópia inválida, então lê a cópia válida do servidor.

Quando a cópia reside em plataformas móveis, a performance do esquema de replicação dependerá da mobilidade, do esquema de endereçamento usado e das atividades de leitura e gravação (BADRINATH & IMIELINSKI, 1992). Algumas vezes, o cliente tem que buscar pelo servidor, ou o servidor precisa buscar pelo cliente. Quando a cópia reside num servidor estático, geralmente nenhuma busca é envolvida, já que a posição do servidor estático não muda. Assim, quando cópias são colocadas em usuários, esquemas de replicação precisam levar em consideração esta assimetria no custo de comunicação.

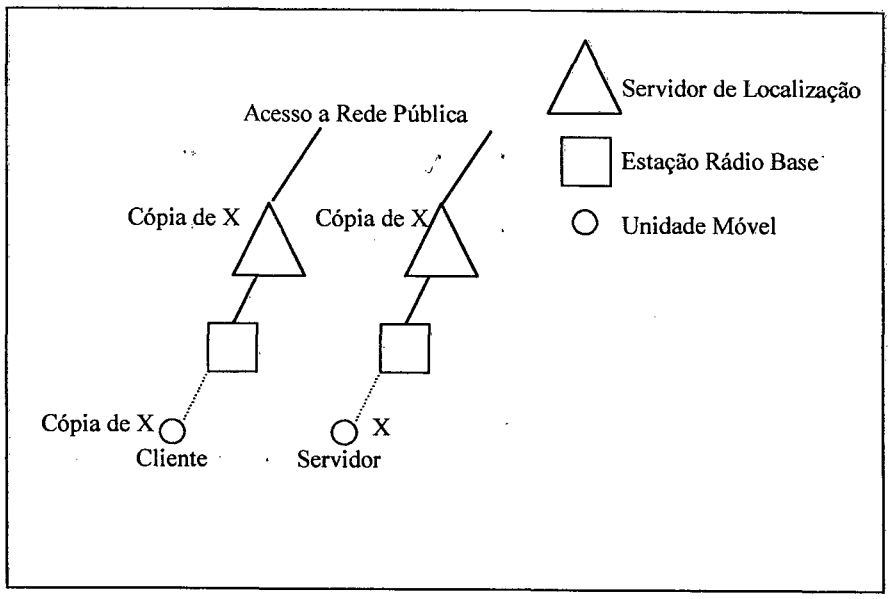


Figura 34 - Possível Localização da Réplica

Assume-se que o cliente e o servidor movem-se apenas na área de seus servidores de localização locais. Dado este modelo, apresentamos agora seis diferentes cenários para colocação de cópias de dados. Nos três primeiros esquemas, nenhuma técnica de ***Cache*** é utilizada. Uma cópia do dado está tanto em qualquer servidor de localização quanto no cliente móvel. Nos três últimos esquemas, uma cópia dos dados fica em ***Cache*** em vários lugares e mensagens de atualização para a cópia são adiadas até que um acesso a um dado previamente invalidado ocorra.

Existem as seguintes opções:

1. O servidor replica a cópia dos dados no cliente móvel. Em cada gravação, o servidor precisa gravar uma cópia no cliente móvel. Para efetuar gravações, é necessário localizar o cliente móvel. A leitura é realizada em uma cópia local no cliente móvel.

2. As cópias replicadas residem no servidor de localização do cliente. Dessa forma, o cliente lê de seu próprio servidor de localização. Aqui, leituras e gravações são sobre cópias estáticas. Entretanto, a cópia está mais próxima ao leitor que ao gravador.

3. O servidor S tem uma cópia dos dados em seu servidor de localização local SL. O cliente lê do SL. Assim, a leitura e a gravação são efetuadas em cópias remotas estáticas.

4. O servidor S tem uma cópia ***Cache*** em seu servidor de localização local SL. A cópia do servidor de localização é invalidada na primeira gravação desde a última solicitação de leitura do cliente. Ler uma cópia ***Cache*** inválida irá requerer a localização do servidor móvel. Entretanto, se a cópia ***Cache*** é válida, então a leitura é feita da cópia do servidor de localização SL.

5. Um ***Cache*** da cópia do servidor existe no servidor de localização do cliente SC. A cópia do servidor de localização torna-se inválida desde a primeira gravação após a última leitura solicitada pelo cliente. Ler uma cópia ***Cache*** inválida irá requerer a localização o servidor móvel. Entretanto, se a cópia ***Cache*** for válida a leitura é feita da cópia do servidor de localização SC.

6. A cópia ***Cache*** é mantida no cliente móvel. Se existiu uma leitura desde a última atualização, o servidor em cada gravação envia mensagem de invalidação ao cliente móvel. Se o cliente quer ler e a cópia ***Cache*** é invalidada, então o servidor móvel é contatado para completar a leitura.

Um usuário pode ficar ao alcance de um servidor de localização local e fazer movimentos locais. Por exemplo, mover-se entre estações rádio base ou fazer movimentos globais. Por exemplo, mover-se de um servidor de localização para outro. Com estes tipos de movimentos, o custo de busca tem dois componentes:

- O custo de buscar a rede celular para o servidor de localização atual e;
- O custo de localizar a estação rádio base atual.

Os parâmetros que afetam estes custos incluem o esquema de endereçamento usado, o padrão e o grau de mobilidade e a distância entre os leitores e os gravadores.

4.3 CACHING

De maneira geral, o uso de ***Cache*** tem como objetivo um acesso mais eficiente aos dados. Mas ele também apresenta outras características desejáveis, como: permitir que os dados sejam tratados de maneira local na máquina cliente, resultando em uma diminuição de carga tanto no servidor quanto no canal de comunicação entre eles. Para que estes resultados sejam alcançados, o ***Cache*** deve oferecer facilidades de gerenciamento de seu conteúdo. Além disso, cada fase de seu funcionamento deve apresentar a melhor adaptação à aplicação e aos dados que a referida aplicação manipula.

Os vários tipos de ***Cache*** conhecidos podem ser classificados de acordo com suas características internas como: estruturas de dados, estratégia de gerenciamento, operações sobre os dados e unidade básica de referência. Diferentes combinações destas características permitem uma melhor adaptação do ***Cache*** à aplicação, resultando em um melhor desempenho tanto no acesso aos dados quanto na sua reutilização.

O funcionamento do ***Cache*** é baseado nas possíveis operações sobre os dados, na natureza de seu conteúdo e na unidade básica de seus dados. Cada uma destas características possuem variações que combinadas definem o contexto de trabalho do ***Cache***. A partir deste contexto são definidas as estruturas de dados e as estratégias de gerenciamento do ***Cache***, principais responsáveis pelo seu bom desempenho. A escolha da combinação a ser utilizada deve atender as características de cada aplicação e ao padrão de acesso aos dados. Idealmente, a aplicação deve ser capaz de reconhecer o contexto de seu funcionamento e se adaptar a ele. Por exemplo, no caso da computação móvel, é desejável que a aplicação saiba tratar de forma diferenciada o cliente móvel de um cliente fixo ou ainda, a sobrecarga do servidor ou sua ociosidade. Porém, dependendo da abrangência e complexidade da aplicação, um modelo de ***Cache*** adaptado a ela é suficiente. A seguir serão descritas as características de um modelo de ***Cache***:

4.3.1 OPERAÇÕES SOBRE OS DADOS

A definição das operações permitidas sobre o conjunto de dados no ***Cache*** influencia diretamente sua funcionalidade e complexidade. Estas operações podem ser divididas em 3 categorias (DESPANDE et al, 1998):

- Dados somente para consulta: O ***Cache*** não permite atualizações locais sobre os dados. Portanto, somente operações de consulta podem ser realizadas sobre o ***Cache***. Assim, os dados não necessitam ser reintegrados ao banco de dados, e o ***Cache*** pode ser estruturado de forma mais simples, sem a necessidade de dados de controle de alteração de seu conteúdo. Este é o tipo mais conhecido de ***Cache***.

- Dados não concorrentes: Neste caso, atualizações locais são permitidas. Mas os dados replicados ficam bloqueados no servidor, evitando atualizações conflitantes.

- Dados concorrentes: Este caso ocorre quando os dados replicados podem ser alterados tanto no servidor quanto no ***Cache***. Para permitir que possíveis conflitos de atualização sejam tratados no servidor, o ***Cache*** deve armazenar informações sobre as alterações ocorridas nos seus dados.

4.3.2 NATUREZA DO CACHE

A natureza do ***Cache*** diz respeito a quando e como o seu conteúdo é definido. Ela pode ser estática ou dinâmica.

- Natureza Estática: O conteúdo do ***Cache*** é definido com antecedência, podendo ser solicitado pela aplicação no início de sua execução. Este tipo de ***Cache*** é encontrado em aplicações onde o comportamento e as necessidades do cliente são bem conhecidos. Nelas o conteúdo do ***Cache*** é mais estável, pois não sofre influência das consultas feitas pela aplicação.

- Natureza Dinâmica: O ***Cache*** define seu conteúdo de forma dinâmica e flexível em tempo de execução. Isso é feito através do armazenamento dos resultados das consultas.

4.3.3 UNIDADE DO CACHE

Unidade de dados do ***Cache*** ou "granulosidade" indica a menor quantidade de dados gerenciada pelo ***Cache***. Ela está diretamente relacionada com a quantidade de informações necessárias para referenciar os dados e com a estratégia de gerenciamento usada pelo ***Cache*** (REN & DUNHAM, 1999), (LEVY et al, 1995).

As unidades mais utilizadas em ***Cache*** são: Arquivos, Tabelas, Objetos, Páginas de dados, Consultas (Query), Tuplas.

A definição da unidade básica do ***Cache*** depende do tipo de flexibilidade que se deseja para o gerenciamento de seu conteúdo. Cada aplicação possui sua própria relação de custo/benefício entre o "***Overhead***" de informações de controle e a flexibilização para a reutilização dos dados.

4.3.4 ESTRUTURA DE DADOS DO CACHE

A estrutura de dados do ***Cache*** deve prover facilidade de gerenciamento dos dados armazenados e também permitir sua fácil manutenção através da inclusão e exclusão dos dados. Para isso, o ***Cache*** deve possuir algumas informações básicas sobre eles (por exemplo, identificação e posicionamento no ***Cache***), além de possuir informações personalizadas de acordo com cada ***Cache***, como: freqüência de acesso, indicadores semânticos, marcas de alteração e de tempo.

Várias estruturas de dados podem ser utilizadas para armazenar os dados no ***Cache***. A escolha da estrutura adequada deve ser baseada em necessidades específicas de gerenciamento tanto dos dados quanto de informações referentes a eles.

4.4 REPLICAÇÃO DE DADOS EM AMBIENTES MÓVEIS

Para que se consiga continuar trabalhando mesmo de forma desconectada, um usuário móvel necessita ter uma cópia dos dados em seu equipamento. Na computação móvel, os usuários deverão ter acesso aos dados independentemente da sua localização física ou até mesmo em processo de movimento (JING et al, 1999), e isto pode ser um problema se no local onde este usuário se encontra, não existir um meio de comunicação pelo qual seja possível acessar um servidor de dados. Neste sentido é que a replicação de dados se torna importante em um ambiente móvel.

A seguir estão relacionados alguns aspectos sobre replicação de dados:

4.4.1 CRITÉRIOS PARA REPLICAÇÃO

- Nenhum: todos os dados deverão ser replicados em todas as réplicas existentes no sistema. Tal situação resulta em excesso de mensagens na rede, estas se fazem necessárias para a propagação das atualizações, elevando o custo para controle de atualizações conflitantes. Num ambiente assíncrono de replicação, tal proposta não é desejada, devido a baixa e instável banda de comunicação dos nós móveis. Uma vez que todas as atualizações sejam propagadas, cada réplica representará o mesmo estado global consistente do banco de dados do sistema (veja a figura 35).

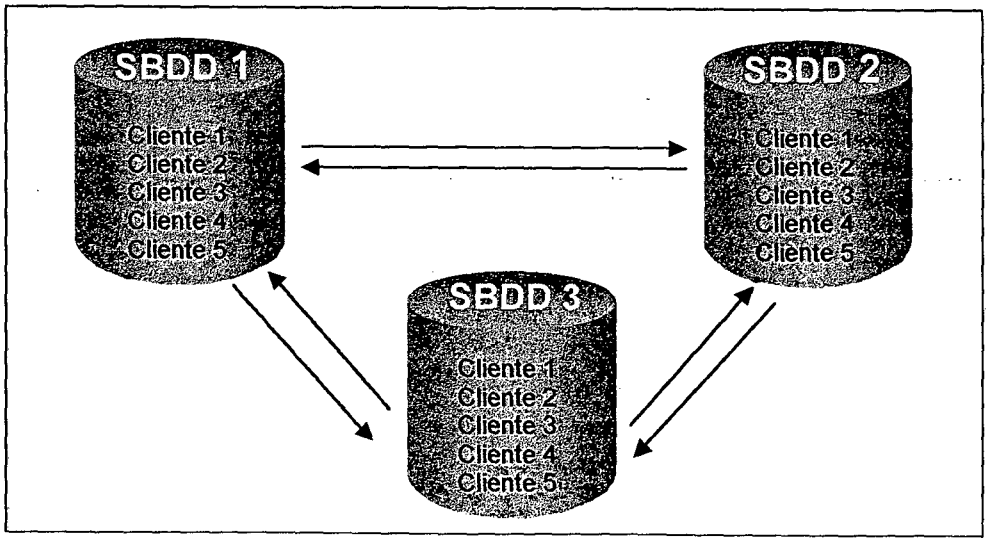


Figura 35 - Replicação Total dos Dados

- Predicado: são critérios que realizam a classificação e escolha dos dados a serem replicados nas diversas instâncias de banco de dados existentes no sistema. A diferenciação entre os dados existentes em tais instâncias é resultado de um predicado, as quais definem as características comuns aos dados pertencentes a uma mesma instância de banco de dados. Diferentes predicados podem ser usados para definir os dados de uma instância. Assim, um mesmo predicado pode ser usado em instâncias de banco de dados diferentes (veja a figura 36). Caso as atualizações conflitantes entre dados pertencentes a instâncias distintas de banco de dados não sejam propagadas pelo sistema, teremos assim, um estado global inconsistente do banco de dados.

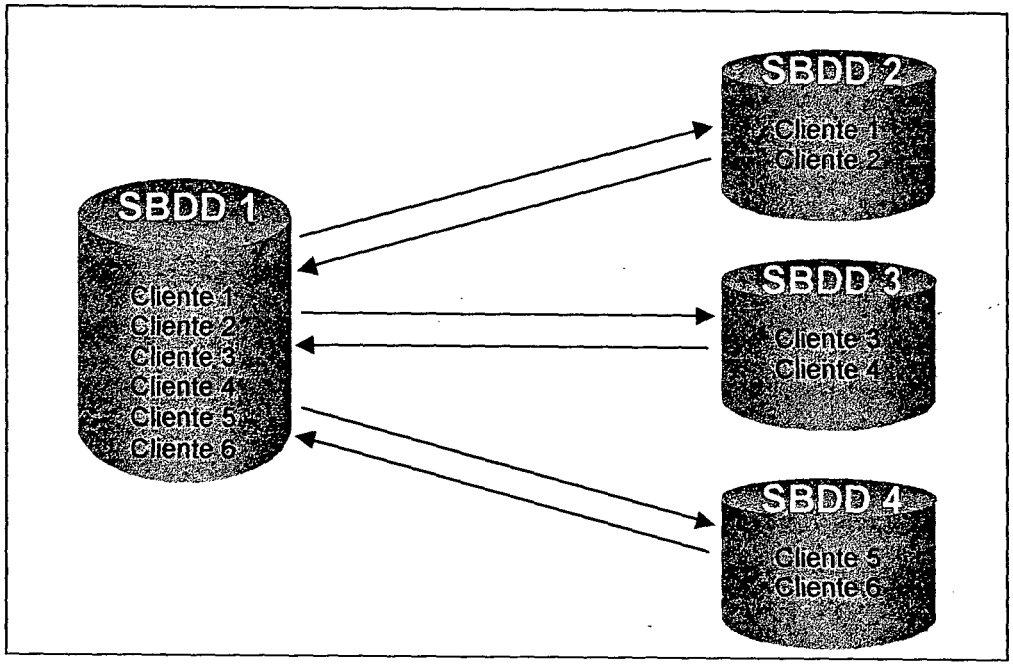


Figura 36 - Replicação Parcial dos Dados

O conceito de localidade está muito relacionado à computação móvel, pois tem-se uma previsão do conjunto de dados envolvidos nas próximas interações do nó móvel no sistema. Por exemplo, ao visitar uma cidade, um vendedor sabe antecipadamente quais são os possíveis clientes envolvidos nas próximas transações comerciais a serem realizadas. Assim, um bom predicado de replicação a ser adotado é a localização do vendedor. Dessa maneira, prevendo-se as cidades que farão parte do roteiro de viagem,

o conjunto de dados relacionados às possíveis transações comerciais serão replicados em sua instância de banco de dados local.

Dependendo dos predicados escolhidos para a escolha dos objetos a serem replicados nas instâncias de banco de dados, teremos dois tipos possíveis de réplicas. São elas:

- Réplicas Disjuntas: Uma réplica pode ser considerada disjunta quando o conjunto de dados representados por ela não participa de nenhuma outra réplica que não seja cópia dela própria, isto é, o resultado do predicado usado para a seleção do conjunto de dados representados pela réplica, não coincide com os demais conjuntos de dados selecionados pelos outros predicados. Assim, um mesmo conjunto de dados só poderá ser representado por réplicas que utilizaram um mesmo predicado. Em algumas situações, dependendo do predicado utilizado, somente teremos a ocorrência da réplica em uma única instância do banco de dados. Neste caso, teremos a forma mais simplificada de atualização de dados, pois como não ocorrem conflitos de atualização, não é necessário a utilização de mecanismos de reconciliação. É desejado que, quando tiver várias réplicas disjuntas de um dado no sistema, seja realizada a propagação das atualizações das demais réplicas;
- Réplicas Sobrepostas: Neste tipo de réplica, diferentes predicados selecionam conjuntos de dados que apresentam partes comuns, isto é, existem intersecções não vazias nos conjuntos de dados selecionados pelos diferentes predicados. Assim, os dados representados pelas réplicas que utilizaram um mesmo predicado podem fazer parte do conjunto de dados representado por outra réplica que utilizou um outro predicado para seleção de dados. Atualizações realizadas sobre réplicas sobrepostas devem ser sincronizadas para manter o banco de dados no estado consistente.

4.4.2 VISIBILIDADE DOS DADOS

- Global: todas as alterações realizadas sobre dados devem ser propagadas a todas as instâncias do banco de dados que armazenam as réplicas desses dados. Tal opção visa proporcionar uma completa disponibilização das atualizações de dados em todo o sistema. O custo relacionado ao processo de propagação é alto, pois mensagens devem ser trocadas entre todas as instâncias do banco de dados, toda vez que inserções e alterações forem realizadas;
- Parcial : não há obrigatoriedade de que as atualizações de dados sejam propagadas a todas as instâncias de banco de dados que armazenam as réplicas dos dados. O custo envolvido na propagação das atualizações é bem menor que o existente no modelo de visibilidade global, pois somente a instância global de banco de dados responsável pela sincronização das atualizações será atualizada.

4.4.3 SINCRONIZAÇÃO DO BANCO DE DADOS

- Central: arquitetura muito utilizada comercialmente. Um nó central é responsável pela sincronização das atualizações realizadas nas instâncias de banco de dados. Tal nó está sempre apto a receber as conexões dos nós móveis. Dependendo do modelo de replicação adotado, o nó central será responsável também pela propagação das atualizações dos dados às demais instâncias de banco de dados envolvidas. Se desejado, armazenará a instância global do banco de dados, pois recebe todas as atualizações realizadas nas demais instâncias de banco de dados do sistema (veja a figura 37);

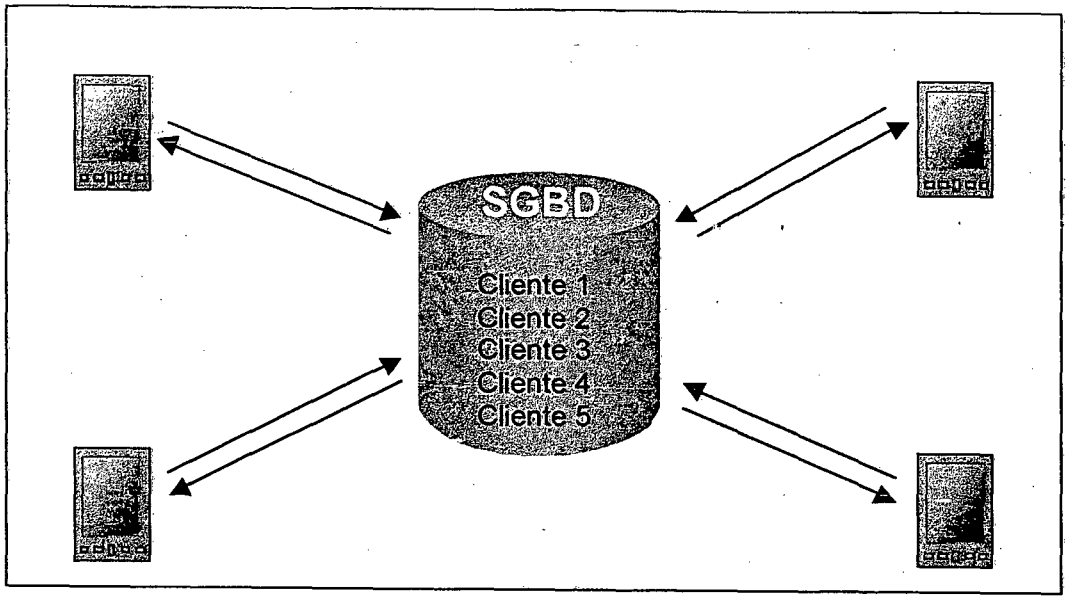


Figura 37 - Modelo de Sincronização Centralizada

- Distribuída: considera que todos os nós têm o mesmo poder para atualizações de dados, devendo estas serem repassadas às demais instâncias de banco de dados nos momentos da reconexão dos nós móveis (veja a figura 38). Tal esquema permite grande disponibilidade de atualização, mas em contra-partida, o custo envolvido para sincronizar as atualizações das réplicas é considerável.

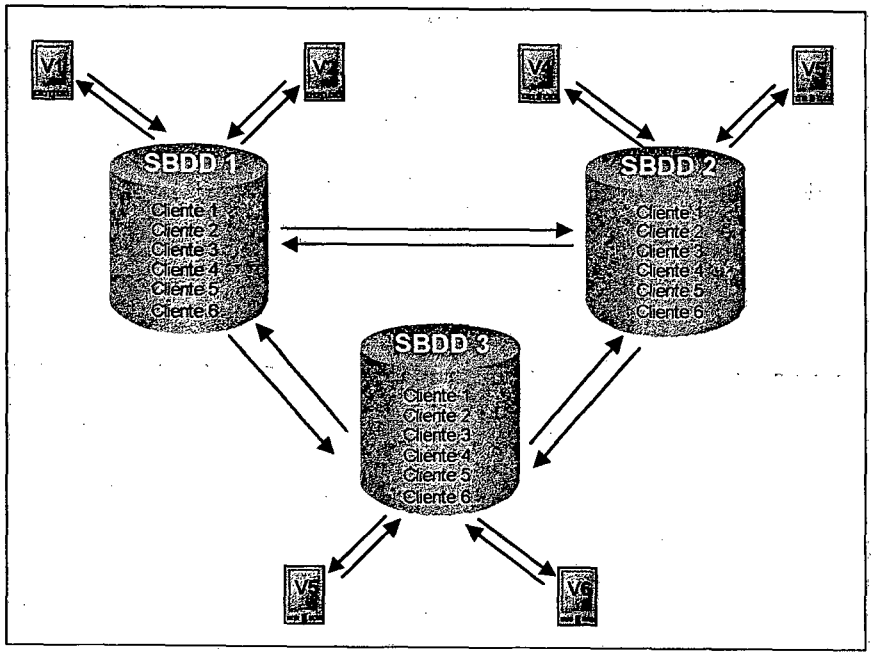


Figura 38 - Modelo de Sincronização Distribuída

4.5 RECONCILIAÇÃO DAS ATUALIZAÇÕES SOBRE OS DADOS

Os Mecanismos de Reconciliação estão relacionados a forma como será realizada a sincronização das atualizações ocorridas nas réplicas durante seus períodos de desconexão. As instâncias do banco de dados serão atualizadas conforme a maneira definida pela estratégia de replicação escolhida, isto é, se as alterações sincronizadas no nó central deverão ser propagadas às demais réplicas do dado. Dependendo da estratégia, poderemos escolher o mecanismo mais adaptável e interessante no momento da integração.

A escolha do mecanismo de reconciliação ocorre no momento da criação da réplica. Uma vez definido o mecanismo de reconciliação e a estratégia de replicação envolvida, as réplicas tratarão de registrar todas as ações realizadas, conforme o modo escolhido, para que posteriormente seja realizada a reconciliação utilizando os registros das alterações.

4.5.1 VERSÃO DE DADOS

Dependendo da estratégia de replicação escolhida, pode-se optar pelo método de multiversão de réplicas. Neste, as atualizações realizadas nos nós móveis, em seus períodos de desconexão, são representadas por versões de dados associadas as modificações ocorridas nas réplicas criadas no nó central. Vários trabalhos (PHATAK & BADRINATH, 1999) têm utilizado tal esquema de reconciliação de atualizações. Estudos realizados demonstram que sua utilização aumenta a probabilidade de sucesso da reconciliação das atualizações, reduzindo assim a ocorrência de conflitos no momento da reconciliação das réplicas.

As réplicas, localizadas nos nós móveis, tratarão de registrar as ações realizadas sobre seus dados. Na reconexão do nó móvel, tais registros deverão ser repassados ao nó central, para a realização da sincronização das atualizações. Algoritmos de multiversão tratarão os registros e decidirão se a propagação das modificações obteve sucesso ou falha. Os nós móveis receberão as notificações das tentativas de propagação realizadas.

4.5.2 ARQUIVO DE LOG

Forma muito utilizada comercialmente pelos SGBDs. Representam o registro das transações ocorridas no banco de dados, com o intuito de garantir a integridade dos dados contra eventuais problemas que venham a ocorrer. Ocasionais problemas com o banco de dados que resultem na inutilização de seus registros podem ser corrigidos através da restauração da situação anterior do banco de dados, por meio de recuperação de um arquivo gravado em fita (backup), e reexecução do log de transações para a recuperação do estado do banco de dados imediatamente anterior ao momento de sua falha.

As réplicas, localizadas nos nós móveis, registrariam em seu arquivo de log local a ocorrência das atualizações sofridas. Na reconexão ao nó central, tais registros seriam repassados ao Objeto Réplica responsável pela sincronização das atualizações da réplica, que trataria de executar as ações registradas no banco de dados global. Eventuais conflitos decorrentes de atualizações seriam tratados e notificações propagadas aos nós móveis envolvidos.

Algoritmos de reconciliação de logs deverão ser implementados para realizar, da melhor forma possível, a reconciliação das atualizações por meio de arquivos de log.

De forma análoga, Bayou (DEMERS et al, 1994) (PETERSEN et al, 1997) propõe o registro de atualizações em logs, para posterior propagação das modificações aos demais repositórios de dados.

4.6 PROTOCOLOS PARA REPLICAÇÃO DE DADOS EM SISTEMAS MÓVEIS

As desconexões e recursos limitados do sistema são restrições que afetam a consistência dos dados em ambientes móveis. Adicionalmente, os ambientes móveis são dinâmicos e se caracterizam pelos três principais objetivos da replicação: prover alta disponibilidade em nodos móveis durante fases de conexão; consistência ;e baixos custos de comunicação. Devido a estas características, os protocolos de replicação comumente utilizados em sistemas distribuídos mostram grandes limitações quando utilizados em

condições de mobilidade. Dessa forma, existem muitos métodos para definir protocolos especiais para tais ambientes. Os métodos oferecem uma gama de conceitos que vão desde restringir o trabalho móvel sobre consistência forte até trabalho móvel reservado (LUBINSKI & HEUER, 2000).

Em (BARBARÁ & MOLINA, 1994) tenta-se resolver os problemas de replicação distinguindo as informações existentes na unidade móvel em: cópias centrais atualizáveis para acesso de leitura e gravação; cópias ***Cache*** somente para leitura; e o diretório para gerenciar tais cópias. A alteração do diretório implica em uma reorganização das cópias centrais, previnindo-se para o caso de desconexão. O artigo enfatizou que a "mobilidade não introduz nenhum problema novo fundamental ou algoritmos para gerenciar dados replicados". É recomendada, além de uma boa fragmentação do banco de dados, a não colocação de cópias centrais em "locais com baixa largura de banda e nodos móveis com poder limitado". O método atinge bons resultados para os objetivos da replicação, mas possui restrições ao trabalho móvel, pois existem aplicações onde usuários estão trabalhando isolados e requerem mais do que cópias ***Cache*** (LUBINSKI & HEUER, 2000).

Estes problemas são tratados em vários artigos, pois o trabalho do usuário não pode ser restringido. Entretanto, as inconsistências devem ser gerenciadas cuidadosamente. Existem muitas experiências com operações de desconexão em contexto com o Coda File System (KLISTER & SATYANARAYANAN , 1992). Geralmente métodos de sistemas de arquivo aplicam uma estratégia otimista para gerenciar cópias ***Cache*** (DUCHAMP & TAIT , 1993). Em (PITOURA , 1996) uma mistura de estratégia otimista e pessimista é proposta. A aplicação ou o sistema de banco de dados determina quais cópias são centrais ou aparentes. Cópias aparentes são consistentes dentro do cluster, enquanto que as cópias centrais são consistentes por todo o banco de dados. Operações de leitura em cópias aparentes resultam em consistência interna do cluster, já as operações de gravação reforçam a consistência do banco de dados. As operações conflitantes entre diferentes combinações de leitura ou gravação são tratadas como inconsistências e são temporariamente possíveis. A estratégia de reconciliação dos dados para operações de gravação é deixada em aberto. A escolha de tal definição é a tarefa do sistema de banco de dados ou da aplicação. Enquanto a

disponibilidade deste método em caso de desconexões é alta, a consistência e os custos dependem dos padrões de leitura/gravação de cópias centrais (LUBINSKI & HEUER, 2000). Usar uma cópia principal num nodo móvel desconectado torna impossível atualizar os outros nodos. Em (BARBARÁ & MOLINA, 1994) é proposto "combinar a estratégia de cópia principal com freqüentes backups para uma cópia ***Cache***". Já (FAIZ et al, 1995) desenvolveu uma estratégia de cópia principal especialmente para este problema, a cópia principal virtual. Neste caso a unidade móvel possui uma cópia principal virtual como representante para a cópia principal desconectada. A consistência é fraca para nodos móveis sem conexão, mas os dados estão disponíveis. Os custos são calculados em média tendo por base as conexões ao nodo móvel principal (LUBINSKI & HEUER, 2000). Em (ZUKUNFT, 1997) foi descrito o uso de sistemas de banco de dados eficazes em ambientes móveis. Regras eficazes são também usadas para replicação. Critérios de validação local controlam o acesso local e facilitam a reintegração após desconexões. Desta forma é possível reconhecer uma conexão forte, uma conexão fraca e um estado de desconexão. Um caminho de propagação de atualização adicional pode constituir uma comunicação direta entre os nodos móveis (por exemplo, através de infravermelho) formando um armazenamento consistente. Devido às limitações de acesso, a disponibilidade é proporcional, enquanto que os custos são baixos (LUBINSKI & HEUER, 2000). Em (RATNER et al, 1999) focou este aspecto especial usando replicação ponto-a-ponto. Os autores usaram um protocolo otimista.

Outros métodos de replicação tratam as falhas mudando de estratégia. Em (KOTTMANN, 1995) uma "replicação modo-misto é apresentada, onde as réplicas controladas de forma pessimista coexistem com réplicas controladas de forma otimista". As operações são agrupadas em três conjuntos de operações: 1 - não podem invalidar outras; 2 - ser invalidada por outras; ou 3 - o conjunto de todas as operações. O autor usou categorias do passado, presente, futuro e otimismo. Este método melhora consistência de cópias móveis (LUBINSKI & HEUER, 2000). Em (BADRINATH & IMIELINSKI, 1992) um cenário básico é considerado: o cliente lê uma cópia dos dados e o servidor grava dados. As cópias podem residir no cliente móvel, no servidor móvel ou em seu servidor de localização. Neste caso foram distinguidas as cópias em: cópias

que possibilitam operações de leitura e gravação; e cópias ***Cache*** que são atualizadas em caso de acesso para leitura. A propagação de atualizações (no tipo não ***Cache***) coincide com a estratégia ROWA (Read On Write All). Um importante aspecto é rever a atividade de leitura/gravação para determinar onde colocar a cópia. Os custos refletidos são referentes a busca do servidor de localização e da estação rádio base. Este ponto de interesse diverge dos estudos atuais sobre replicação, onde conexões ocasionais principalmente estabelecem replicação (LUBINSKI & HEUER, 2000).

CAPÍTULO V

5 ANÁLISE DA MOBILIDADE EM BANCOS DE DADOS

Nos últimos 10 anos muitas mudanças ocorreram na área de tecnologia da informação. A Internet sem dúvida contribuiu para estas mudanças, e ajudou muitas empresas a desprender-se de seus escritórios, permitindo aos seus colaboradores maior flexibilidade na utilização de sistemas corporativos, independentemente do local onde se encontram. Atualmente os sistemas corporativos vêm se estendendo além de ***Web Browsers*** através de dispositivos móveis de computação. Estes equipamentos geraram uma grande revolução na área de processamento de dados, pois as possibilidades de estender as aplicações corporativas para fora da empresa aumentaram, e trouxeram muitas oportunidades para novos empreendimentos. São óbvias as dificuldades de transferir o conhecimento de aplicações tradicionais para estes dispositivos. O tamanho da tela é menor, se comparado a laptops ou desktops, e são limitadas as possibilidades de uso por parte do usuário. Existem fatores menos evidentes que tornam a programação destes dispositivos um grande desafio. O poder de processamento e armazenamento são significativamente menores do que os dispositivos tradicionais. Da mesma maneira, a conexão com a rede pode não ser possível ou limitada a mecanismos que são tipicamente mais lentos que uma rede de dados corporativa. Dessa forma, qualquer sistema que pretende disponibilizar capacidade de mobilidade a seus usuários deverá levar em conta todas estas limitações.

A seguir será apresentada uma análise sobre a capacidade de mobilidade disponível em 5 Sistemas Gerenciadores de Bancos de Dados, que estão entre os mais utilizados em ambientes corporativos a nível mundial segundo o IDG (*INTERNATIONAL DATA GROUP*).

5.1 BANCOS DE DADOS ANÁLISADOS

Fabricante	SGBD
IBM	Informix SE
IBM	DB2 Everyplace
ORACLE	Oracle Lite
Sybase	Adaptive Sever Anywhere
Microsoft	MS SQL Server 2000

Tabela 5 – Bancos de Dados analisados

5.2 INFORMIX SE

O IBM Informix® Standard Engine (SE) é um servidor de banco de dados que pode ser utilizado em diferentes plataformas como: UNIX, Linux e Microsoft. O Informix SE integra-se de forma direta com aplicações desenvolvidas com ferramentas Informix e ferramentas de desenvolvimento de outras empresas que sejam compatíveis com os padrões ODBC e JDBC (INFORMIX SE, 2001). É utilizado comumente para aplicações de médio porte. Na figura 39 é apresentada a arquitetura do INFORMIX SE.

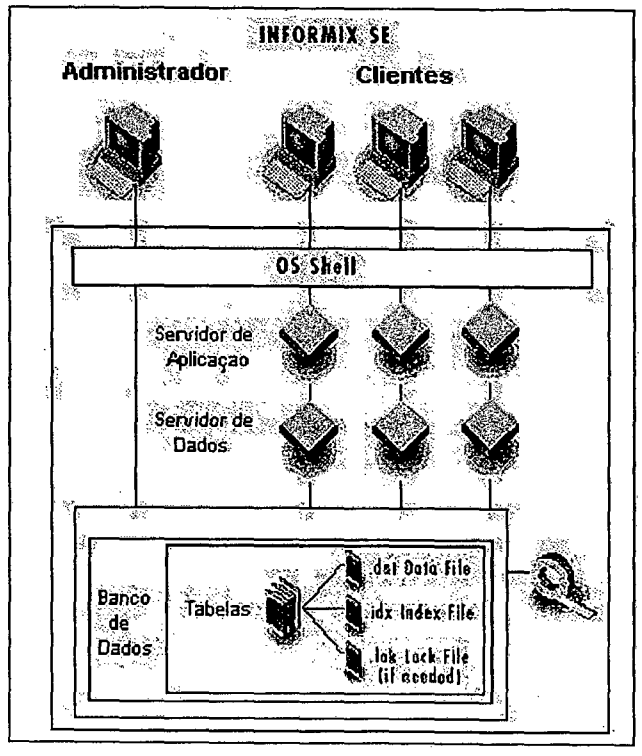


Figura 39 - Arquitetura do SGBD INFORMIX SE

Informix SE provê desempenho excelente, consistência de dados, capacidade de cliente/servidor e pouca intervenção do administrador de banco de dados (INFORMIX SE, 2001).

5.2.1 PRINCIPAIS CARACTERÍSTICA DO INFORMIX SE:

- Desempenho: Possui um otimizador de consultas baseado em custos, combinado com opções de indexações flexíveis, o que permite ao Informix SE manter desempenho excelente até mesmo em consultas complexas. O Informix SE pode trabalhar com um grande volume de dados (até 1Tb) em sistemas operacionais novos que permitem tamanhos de arquivos grandes.
- Facilidade de Administração: O Informix SE tira proveito do sistema operacional UNIX e do sistema de arquivos, enquanto o suporte ao banco de dados e às tabelas nele contidas é semelhante ao suporte oferecido ao próprio sistema operacional UNIX. O Informix SE inclui ferramentas de fácil utilização que permitem a migração de dados para outro banco de dados Informix, provendo alto desempenho e processamento paralelo.
- Connectividade: O Informix SE fornece uma gama extensiva de opções de conectividade. Além das ferramentas tradicionais (como Informix SQL e 4GL[15]), também suporta ferramentas de desenvolvimento com suporte a ODBC e JDBC.
- Integridade de dados: Um ponto forte no Informix SE é a possibilidade de auditar as mudanças ocorridas no banco de dados, enquanto é assegurada a integridade dos dados ao permitir uma rápida e segura restauração do banco de dados quando alguma falha de sistema acontece.
- Durante uma transação: Os dados envolvidos são bloqueados de forma a impedir que outro usuário venha a alterar estes dados. O impacto do

[15] É a contração para "Linguagens de Quarta Geração"

bloqueio dos dados durante uma transação pode ser minimizado através de regras sobre níveis de bloqueios disponíveis no Informix SE.

- Consistência de dados: As diretivas de integridade, os procedimentos armazenados no banco de dados e as instruções disparadas no banco de dados através de eventos, permitem a validação de regras pré-definidas. Estas facilidades asseguram que a informação inserida ou atualizada no banco de dados obedeça as regras de negócios determinadas no Informix SE.
- Segurança de dados: O Informix SE provê controles de segurança para o banco de dados, para as tabelas nele inseridas, suas colunas e para níveis dos procedimento armazenados. Podem ser dadas permissões para procedimentos armazenados de forma a restringir acesso de usuário a funções de negócio em lugar dos dados diretamente, permitindo alta flexibilidade na determinação de direitos do usuário.

5.3 DB2 EVERYPLACE DATABASE

A intenção da IBM ao criar o projeto do DB2 Everyplace Database foi criar um banco de dados relacional para PDAs e ***Handhelds***; permitindo acessar e executar atualizações em um banco de dados de um dispositivo móvel.

De maneira geral, o funcionamento se dá através do armazenamento de uma pequena quantia de dados na unidade móvel que é sincronizada com unidades maiores de Banco de Dados. Dessa forma, as aplicações são instaladas na unidade móvel sendo executadas de forma desconectada utilizando um banco de dados local atualizado de tempos em tempos (DB2 EVERYPLACE, 2001).

A Solução DB2 Everyplace é constituída por três componentes:

- DB2 Everyplace database: sistema de banco de dados relacional projetado para dispositivos móveis;
- DB2 Everyplace Sync Server: servidor de sincronização bidirecional utilizado para atualizar informações no DB2 Everyplace database a partir de dados localizados na empresa e vice-versa;
- DB2 Everyplace Personal Application Builder: é uma ferramenta de desenvolvimento de aplicações para dispositivos móveis.

5.3.1 MECANISMO DE SINCRONIZAÇÃO DOS DADOS

A sincronização dos dados é realizada através do DB2 Everyplace Sync Server, que é um servidor de sincronização bidirecional que utiliza a tecnologia Java Servlet. Na sua arquitetura estão envolvidos três sistemas (veja a figura 40): o dispositivo móvel; o servidor de sincronização; e a fonte de dados.

O DB2 Everyplace Sync Server tem parte da arquitetura construída com base em uma máquina de sincronização chamada SyncML. A SyncML é uma iniciativa de uma indústria patrocinada por várias companhias, como IBM, Lotus, Nokia, Palm, Psion, Motorola, Starfish e Ericsson. Produtos baseados na SyncML são designados para

interoperar um com o outro, usando uma linguagem de padronização e estrutura de sincronização para replicar e sincronizar dados entre aparelhos (DB2 EVERYPLACE, 2001).

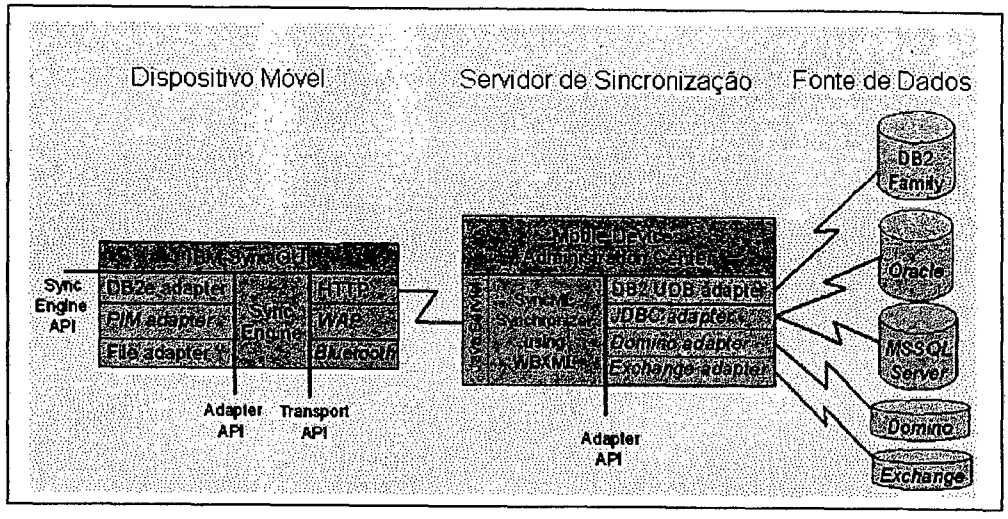


Figura 40 - Modelo de Sincronização do DB2 Everyplace

A seguir será mostrado passo a passo como acontece a sincronização nos dois sentidos (NOETHER, 2001), ou seja, do dispositivo móvel para a fonte de dados e da fonte de dados para o dispositivo móvel. Neste processo estará envolvida a tabela ECCATALOG no dispositivo móvel e no Banco de Dados GPSE que será a fonte de dados.

5.3.2 DO DISPOSITIVO MÓVEL PARA A FONTE DE DADOS

Usuários móveis submetem mudanças que foram feitas em cópias locais dos dados para atualização da fonte de dados.

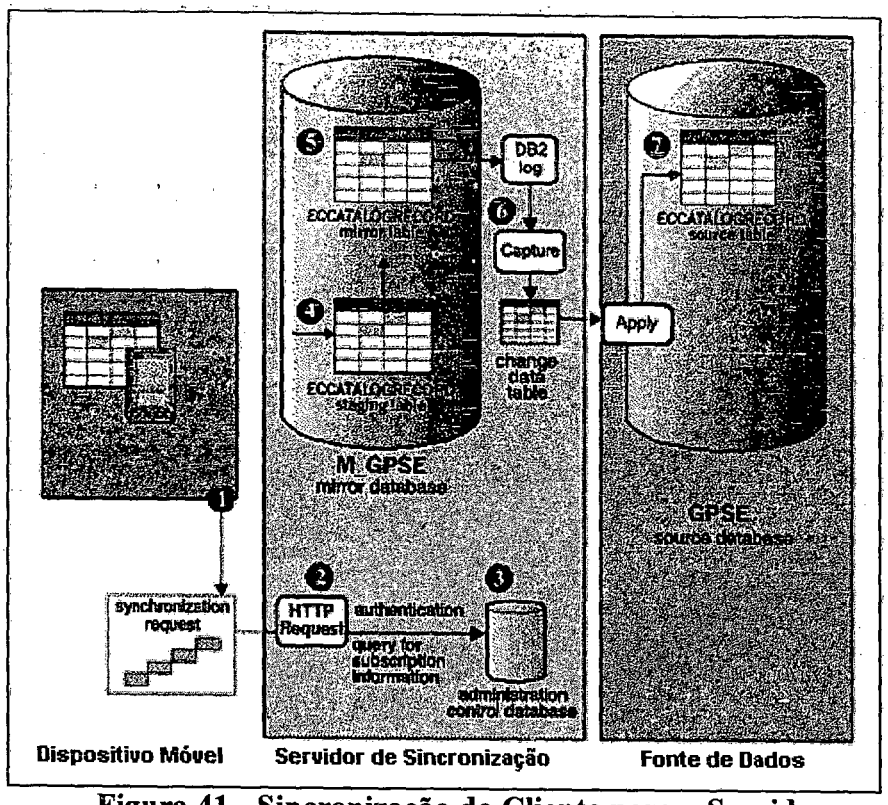


Figura 41 - Sincronização do Cliente para o Servidor

Na figura 41 são mostrados os seguintes passos:

Passo ❶ - É solicitada a sincronização

Passo ❷ - É autenticada a requisição de sincronização, então, é dado entrada na fila do servidor de sincronização

Passo ❸ - O controle de administração do banco de dados verifica se a requisição partiu de um usuário válido para o servidor de sincronização

Passo ❹ - Os dados são colocados em uma tabela temporária em uma cópia do banco de dados (M_GPSE)

Passo ❺ - Os dados são passados da tabela temporária para uma cópia da tabela ECCATALOG (M_ ECCATALOG), e os potenciais conflitos são resolvidos. As mudanças são gravadas no DB2 log

Passo ❻ - O *DB2 DataPropagator Capture Program* trata as mudanças contidas no DB2 log e grava os dados da tabela de Alteração de Dados

Passo ❼ - O *DB2 DataPropagator Apply Program* aplica as mudanças contidas na tabela de Alteração de Dados sobre os dados da tabela ECCATALOG na fonte de dados

5.3.3 DA FONTE DE DADOS PARA DISPOSITIVO MÓVEL

Os usuários móveis recebem mudanças que foram feitas na fonte de dados desde a última vez que os dados foram sincronizados.

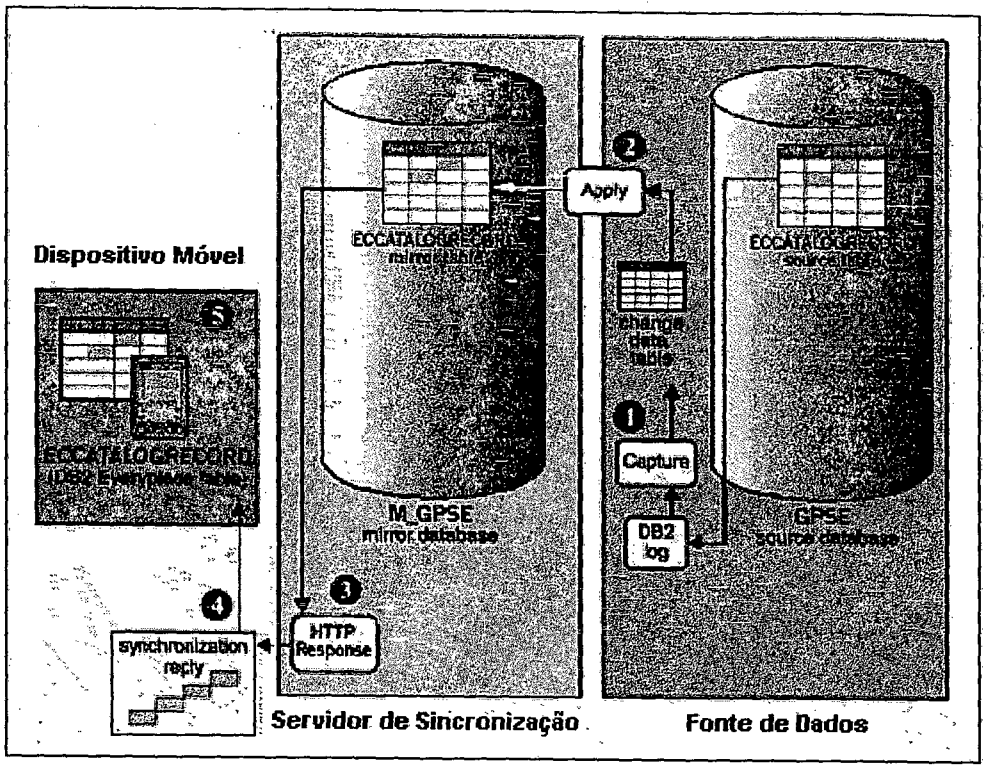


Figura 42 - Sincronização do Servidor para o Cliente

Na figura 42 são mostrados os seguintes passos:

Passo ❶ - O *DB2 DataPropagator Capture Program* é executado continuamente sobre a fonte de dados para tratar as mudanças ocorridas na tabela ECCATALOG contidas no DB2 log e grava os dados da tabela de Alteração de Dados

Passo ❷ - O *DB2 DataPropagator Apply Program* aplica as mudanças contidas na tabela de Alteração de Dados sobre uma cópia da tabela ECCATALOG (M_ ECCATALOG) na fonte de dados

Passo ❸ - Estas mudanças são colocadas em uma fila do servidor de sincronização para uma posterior atualização do dispositivo móvel

Passo ❹ - Durante uma sincronização são passadas as mudanças contidas na fila do servidor de sincronização para o dispositivo móvel

Passo ❺ - Finalmente são aplicadas as mudanças sobre os dados da tabela ECCATALOG no dispositivo móvel

5.4 ORACLE LITE

O Oracle Lite faz parte da família de gerenciadores de banco de dados da Oracle, é um banco de dados destinado para soluções móveis, onde a capacidade de armazenamento e de exècução de programas é reduzida. Por isto que este é um banco de dados compacto, com um tamanho que varia entre 50Kb a 750Kb, podendo ser utilizado em PC's, laptops, e PDA's (ORACLE8 LITE, 2001).

É composto por três componentes principais:

- Oracle Lite: sistema de banco de dados projetado para dispositivos móveis. Possui suporte a Triggers[16], Stored Procedures[17] e Replicação de Dados;
- Oracle Iconnect: É reponsável pela sincronização dos dados através de conexões TCP/IP;
- Oracle Portal-to-Go: Provê o acesso à base de dados do servidor central através da Internet (***Middleware***[18]).

5.4.1 PRINCIPAIS CARACTERÍSTICAS:

- Suporte a ***Stored Procedures*** e ***Triggers***;
- Acesso a classes Java;
- Suporte a múltiplas interfaces: ODBC, JDBC, SQLJ, OKAPI, JAC;
- Possibilidade de utilização de múltiplas ferramentas de desenvolvimento como: C++ Visual, Visual Café, Oracle JDeveloper e Developer, Delphi, Satelite Forms e CodeWarrior, etc;
- Suporte a vários sistemas operacionais como: Windows CE, Windows 95/98/NT, PalmOS, EPOC;
- Compatibilidade com o Oracle Database Servers.

---

[16] É dispositivo utilizado para disparar instruções no banco de dados quando algum evento esperado acontecer.
[17] São procedimentos programados diretamente no banco de dados.
[18] Sistema para intermediar a ligação entre ambientes heterogêneos.

5.4.2 ORACLE iCONNECT

É um conjunto de serviços que viabilizam o sincronismo bilateral dos dados (entre o servidor central e os dispositivos móveis) através de dispositivos de replicação de dados.

O Oracle iConnect possui os seguintes recursos:

- Replicação avançada;
- AQ Lite;
- Consolidator.

5.4.3 REPLICAÇÃO AVANÇADA

Este recurso do Oracle iConnect suporta a replicação bidirecional com integridade de transações, garantindo replicação apenas de dados necessários para cada unidade móvel (ORACLE8 LITE, 2001). Possui detecção e resolução de conflitos que podem ser pré-definidas ou escritas pelo usuário. A replicação avançada suporta sincronização múltipla, protocolos de rede para ***LANs***, Internet, transferência de arquivos e wireless. A seguir são descritos tipos de replicação realizadas no Oracle8i Lite:

- Replicação baseada em conexão: é configurada apenas no cliente. Desta forma o Centro de Replicação Oracle (REPAPI) e seus componentes também são instalados no cliente, exigindo maior quantidade de memória no mesmo.
- Replicação baseada na Internet: Permite ao usuário móvel, enviar as alterações de um nodo Snapshot para uma aplicação Web, HTTP ou protocolos MIME, minimizando o número de conexões.
- Replicação baseada em arquivos: pode ser feita em arquivo ou baseada em disco. Requer configuração, administração e instalação de componentes, tanto no cliente como no servidor.
- Replicação baseada em OMA: Utiliza infraestrutura dos Agentes Móveis Oracle (***Oracle Mobile Agent***) para realizar a replicação bidirecional sobre

uma ***LAN***, ***Dial-up*** ou redes ***Wireless***. Os dados são transmitidos utilizando a estrutura ***Store-and-Forward*** (armazenar e enviar).

5.4.4 AQ LITE

O AQ Lite é um serviço de mensagem para computação móvel que provê a comunicação entre unidades móveis e servidores Oracle. Aplicações que estão sendo executadas em uma unidade móvel podem enviar e receber (***Store-and-Forward***) mensagens de outras aplicações que estão sendo executadas no servidor. Devido a este fato, não há a necessidade de constante conexão entre a unidade móvel e o servidor. Mensagens são armazenadas em seus próprios bancos de dados e enviadas posteriormente para o local de destino (ORACLE8 LITE, 2001).

O AQ Lite apresenta as seguintes características:

- Apresenta mensagem compatível com API, permitindo o desenvolvimento de mensagens baseadas em aplicação;
- Sincronização múltipla e protocolos de rede, tais como: LAN, Internet, wireless e baseada em arquivo.

5.4.5 CONSOLIDATOR

O Consolidator possui suporte a tradução e sincronização de unidades móveis dentro do servidor de banco de dados Oracle. Permite formatar registros de dados para serem replicados, sincronizados e compartilhados com o servidor Oracle. O Consolidator tem arquitetura ***two-tier***, como mostrado na figura 43. De um lado estão os dispositivos móveis representados pela fila 1, e do outro o servidor Oracle representando pela fila 2. A sincronização acontece nos dois lados. O Consolidator também pode ser utilizado em uma arquitetura ***three-tier***. Neste caso, a fila 1 é um banco de dados Oracle Lite e a fila 3 é um servidor Oracle. A função do Consolidator é sincronizar dados entre as filas 1 e 2. O recurso de Replicação Avançada da Oracle se encarregará de sincronizar os dados entre as filas 2 e 3.

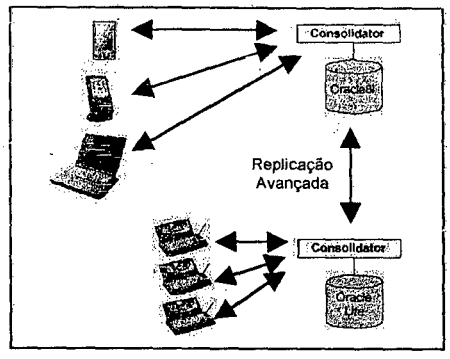


Figura 43 - Funcionamento do Consolidador Oracle

5.4.6 ORACLE PORTAL-TO-GO

O Oracle Portal-to-Go controla a sincronização de dados e a transparência de distribuição de software a usuários sem exigir codificação especial. Desta forma, um único código pode ser executado em diferentes arquiteturas: Internet, aplicação móvel, ou ambiente cliente/servidor. Possui aplicações e dados armazenados em um servidor centralizado, evitando "ilhas de informação" (ORACLE8 LITE, 2001). A aplicação é desenvolvida em HTML ou em Java, sendo disponibilizada pelo ***Oracle Aplication Server*** (OAS) na Internet. Enquanto o usuário estiver em modo conectado, a aplicação irá interagir com o banco de dados do servidor (veja a figura 44), ou seja, as modificações que forem realizadas estarão sendo refletidas diretamente no banco de dados central. Quando o usuário estiver em modo desconectado, a aplicação irá interagir com o banco de dados no dispositivo móvel através do ***Oracle Lite Web Server***, instalado na unidade móvel. Deste modo, a aplicação se apresenta transparente para o usuário, como se estivesse o tempo inteiro conectado.

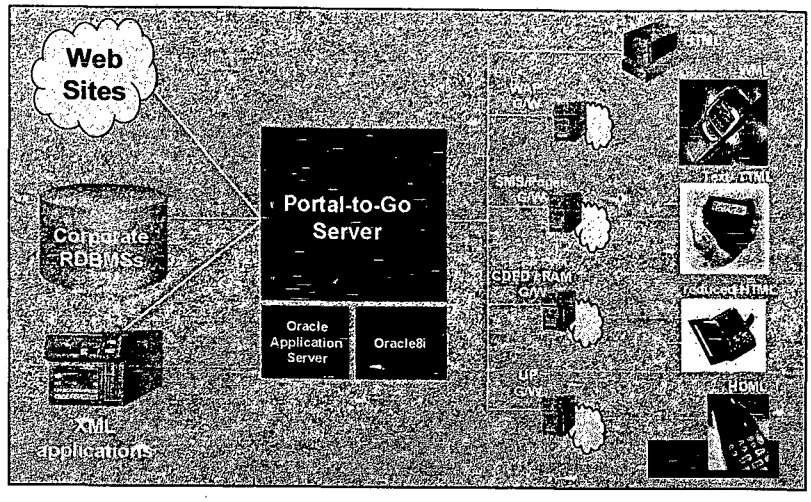


Figura 44 - Arquitetura Oracle Portal-to-Go

5.5 ADAPTIVE SEVER ANYWHERE

O Adaptive Sever Anywhere faz parte de um conjunto de aplicativos (Sybase SQL Anywhere Studio) destinados para o gerenciamento e sincronização de dados que permite a distribuição das informações corporativas para grupos de trabalho, dispositivos móveis e sistemas com bancos de dados embutidos. Neste sentido, o Adaptive Server Anywhere é o banco de dados para aplicações móveis, por um sistema de replicação otimizado para ambientes ocasionalmente conectados e por um conjunto de ferramentas de desenvolvimento e produtividade (SYBASE SQL ANYWHERE, 2001).

A seguir são enumeradas algumas de suas características:

- Banco de dados relacional para cada nodo do sistema, usando Sybase Adaptive Server Anywhere ou banco de dados Ultralite;
- Duas formas de sincronização de dados entre o servidor e as unidades móveis;
- Entrada de dados em qualquer nodo do sistema;
- Operação das unidades móveis em modo desconectado;
- Suporte para um grande número de unidades móveis;

- Administração central, sem necessidade de administração nas unidades móveis.

O Sybase SQL Anywhere Studio é constituído por três componentes principais:

- Adaptive Server Anywhere;
- Adaptive Server Anywhere Ultralite;
- MobileBuilder.

5.5.1 ADAPTIVE SERVER ANYWHERE

O Adaptive Server Anywhere foi projetado para aplicações móveis e distribuídas. A replicação nativa possibilita o acesso a redes corporativas e Intranets, de modo que os dados de uma empresa podem ser replicados em máquinas locais, mesmo quando o usuário estiver acessando os dados fora da empresa.

Os seguintes sistemas operacionais são suportados pelo Adaptive Server Anywhere:

- Windows 95/98, NT, 2000, CE;
- Novell NetWare;
- Solaris/SPARC;
- Solaris/Intel;
- HP-UX;
- IBM AIX;
- Linux;
- UNIX.

5.5.2 ADAPTIVE SERVER ANYWHERE ULTRALITE

O Adaptive Server Anywhere Ultralite é uma tecnologia desenvolvida e patenteada pela Sybase, que gera bancos de dados otimizados para ambientes móveis e embutidos com recursos limitados, para PDA's, ***Handhelds***, entre outros. Permite a sincronização e a replicação do banco de dados em modo desconectado. Tem a capacidade de gerar módulos nas unidades móveis contendo apenas o código necessário para suportar uma aplicação específica, de forma automática. Gera um banco de dados que é acessível através do SQL ANSI. O Ultralite é compilado diretamente no banco de dados. Possui duas tecnologias diferentes, mas complementares de sincronização: SQL Remote e MobiLink.

5.5.3 SQL REMOTE

O SQL Remote é uma tecnologia de sincronização baseada em mensagem. Uma instalação de SQL Remote consiste em um banco de dados central que armazena uma cópia mestra dos dados e de muitos bancos de dados remotos em dispositivos móveis (veja a figura 45). O servidor central pode estar executando no Adaptive Server Anywhere ou Adaptive Server Enterprise Database e nas unidades móveis é executado o Adaptive Server Anywhere. A entrada dos dados pode ocorrer no banco de dados central ou em unidades móveis. A sincronização dos dados acontece baseada em um sistema de mensagem, como e-mail ou transferência de arquivo, através de uma conexão ocasional.

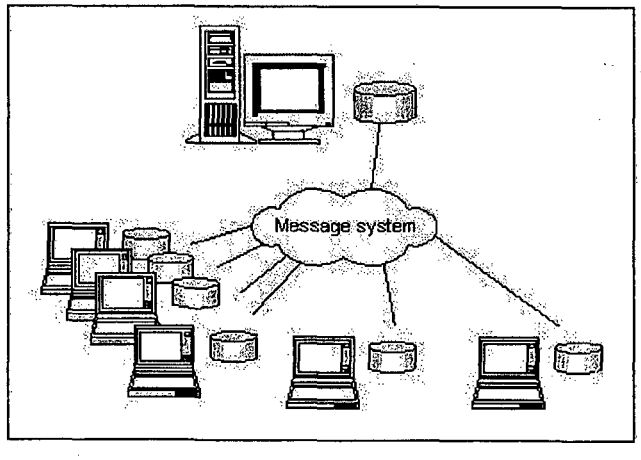


Figura 45 - Arquitetura de Sincronização com o SQL REMOTE

5.5.4 MOBILINK

MobiLink é uma tecnologia de sincronização bidirecional desenvolvido para complementar SQL Remote expandido para sistemas com informações distribuídas. Tem como principais características:

- Consolidar bancos de dados heterogêneos;
- Particionar os dados em subconjuntos;
- Detectar conflitos e solução;
- Encriptação de dados;
- Gerenciar grandes bancos de dados;
- Suporte a ***Handheld*** e dispositivos sem fio.

O Mobilink tem uma forma de sincronização baseada em sessão, como mostra a figura 46, requerendo uma conexão direta através da porta serial utilizando os protocolos TCP/IP, HTTP e ***HotSync*** ou ondas de rádio.

Além do Adaptive Server Anywhere e Adaptive Server Enterprise Database, o Mobilink suporta Oracle, Microsoft SQL Server e IBM DB2.

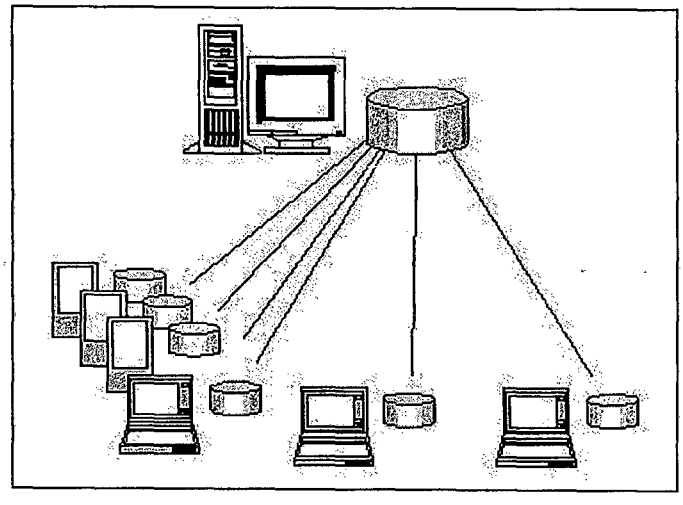


Figura 46 - Arquitetura de Sincronização através do MOBILINK

Grandes empresas podem necessitar de mais do que um banco de dados central. Neste caso, a combinação do SQL remote e Mobilink se torna uma boa solução (SYBASE SQL ANYWHERE, 2001).

5.5.5 MOBILE BUILDER

É uma ferramenta visual para desenvolvimento de aplicações, possui uma tecnologia de integração com o Ultralite e pode ser utilizado por vários sistemas operacionais como:

- Palm OS;
- Windows CE;
- Win32 e Win16;
- DOS & DPMI.

5.6 MS SQL SERVER 2000

O SQL Server 2000 da Microsoft é um gerenciador de banco de dados desenhado exclusivamente para a plataforma Windows. Possui suporte a outros bancos de dados, como Oracle e Sybase. Apresenta-se em cinco versões: ***Standart***, ***Enterprise***, ***Personal***, ***Desktop Edition*** e ***Windows CE Edition***, do qual iremos ter mais detalhes por se tratar de um banco de dados com características de mobilidade.

5.6.1 MS SQL SERVER 2000 WINDOWS CE

O Microsoft SQL Server 2000 Windows CE Edition (SQL Server CE) é um banco de dados compacto, voltado para o desenvolvimento de aplicações baseadas em equipamentos móveis. Apresenta as seguintes características:

- Através do IIS[19] (***Internet Information Services***), o SQL Server CE viabiliza acesso seguro com padrões Internet ao banco de dados SQL Server, nas versões 6.5, 7.0 e 2000;
- Integridade referencial e operações em cascata;
- Manipulação de vários tipos de dados;
- Índices por tabela e Índices multicoluna;
- Suporte a transações;
- Suporte a replicação de dados;
- Comandos SQL (inner/outer join, group by, having);
- Compressão de dados em tempo de sincronização;
- Encriptação de dados.

O SQL Server CE oferece uma série de características de bancos de dados relacionais, como processamento otimizado de consultas e suporte a transações, garantindo um consumo mínimo de recursos valiosos para o sistema operacional

[19] Servidor de Internet fornecido pela Microsoft

Windows CE. O SQL Server CE possui suporte a gerenciamento de dados e possibilidade de conexão, tanto para grandes servidores como para estações de trabalho PC Desktop. No SQL Server CE, o consumo de memória é pequeno, sendo que todas as suas funcionalidades ocupam aproximadamente 1MB de memória (MICROSOFT SQL SERVER, 2001).

5.6.2 ACESSO A DADOS FLEXÍVEL

O SQL Sever CE permite atualizar os dados simultaneamente no dispositivo móvel e no servidor central. Caso o dispositivo móvel estiver fora da área de cobertura de uma rede sem fios ou impossibilitado de trabalhar com uma conexão através da rede local, este dispositivo trabalhará apenas com o banco de dados local. Os dados serão sincronizados mais tarde, quando a conexão entre o dispositivo móvel e o servidor central for restabelecida. O SQL Server CE possui duas opções para sincronização de dados: Merge Replication e Remote Data Access (RDA).

Tanto o Merge Replication como o Remote Data Access utilizam padrões da Internet como: HTTP (***Hiper Text Transfer Protocol***) e SSL (***Secure Sockets Layer***). Estas tecnologias permitem autenticação forte, autorização e criptografia sem sacrificar a flexibilidade que é crítica em muitos ambientes móveis e desconectados (MICROSOFT SQL SERVER, 2001). A comunicação de dados através de HTTP permite ao SQL Server CE trabalhar com sistemas de segurança do tipo ***Firewall*** ou servidor de ***Proxy***.

5.6.3 MERGE REPLICATION

É um processo de distribuição de dados entre um publicador e subscritores, permitindo o publicador e os subscritores realizarem atualizações enquanto estão em modo conectado ou desconectado, fundindo as atualizações entre estes locais quando eles forem reconectados. Os dados poderão ser atualizados por mais de um subscritor ao mesmo tempo, sendo assim a resolução dos conflitos será realizada pelo servidor central ou pelo administrador de dados caso as regras de resolução não tenham sido pré-estabelecidadas (MICROSOFT SQL SERVER, 2001). Para que possa ser utilizado

sincronização de dados através de Merge Replication, será necessária a utilização de um banco de dados SQL Server 2000 no servidor central, conforme mostrado na figura 47.

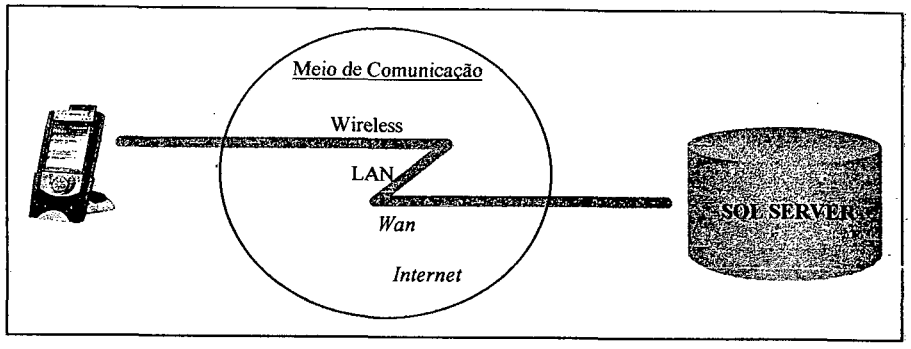


Figura 47 - Arquitetura do MERGE REPLICATION

5.6.4 REMOTE DATA ACCESS

Este modelo provê um modo simples para aplicações baseadas em Windows CE acessarem um servidor de dados central utilizando: SQL Server 6.5; SQL Server 7.0; ou SQL Server 2000. O RDA poderá ser utilizado se o dispositivo móvel estiver continuamente conectado ou intermitentemente conectado. Um benefício da utilização do RDA é a facilidade de utilização, pois poderá ser utilizado sem qualquer configuração especial no servidor de dados central. A figura 48 mostra um exemplo de conexão ao banco de dados central utilizando o RDA.

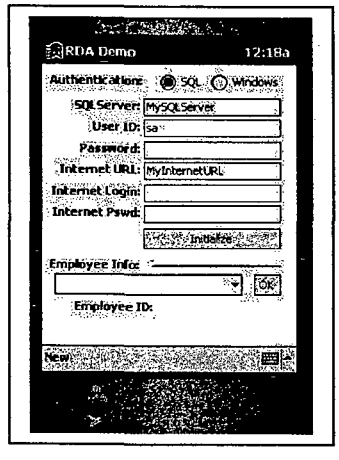


Figura 48 - Emulador para o Pocket PC

5.7 CARACTERÍSTICAS ANALISADAS

Em um projeto de computação móvel a escolha das tecnologias adequadas é sem dúvida o fator predominante de sucesso. A heterogeneidade da computação móvel nos apresenta muitas variâncias, o que mostra a complexidade envolvida em sistemas distribuídos móveis. Segundo (Mateus & Loureiro, 1998) estas variâncias são tratadas em grandes grupos de problemas quanto a: equipamentos; infra-estrutura de rede; sistemas operacionais; bancos de dados; e fatores de segurança. Na maioria dos casos, os recursos disponíveis para a condução de um projeto são escassos, o que inviabilizaria voltar a traz na opção por uma determinada tecnologia. A seguir serão apresentadas algumas justificativas que validam a determinação dos critérios de avaliação dos bancos de dados móveis apresentados, os quais estão contidos nestes grandes grupos de problemas apresentados em (Mateus & Loureiro, 1998) poderão auxiliar a identificar um banco de dados adequado à solução de um determinado problema através da computação móvel:

- O crescente número de equipamentos e sistemas operacionais existentes para a computação móvel, onde, é comum a existência de características proprietárias, conduz a um cenário de incompatibilidade entre determinadas aplicações. Mas a necessidade de utilização de uma mesma aplicação em diferentes equipamentos pode levar a necessidade e reescrever esta aplicação. Isto poderá ser evitado se no inicio do projeto for identificada a necessidade de independência de plataformas, ou pelo menos quais plataformas deverão ser atendidas. Neste sentido a escolha de um banco de dados que contemple esta necessidade é fundamental. A análise neste caso se dará "quanto a plataforma de utilização" e será utilizado para esta comparação os sistemas operacionais de alta portabilidade mais comuns (Palm OS, Windows CE, Psion EPOC, QNX Neutrino, Embedded Linux), os demais sistemas operacionais serão denominados como "outros".

- Os recursos do equipamento também são fatores importantes quando se fala em projeto de computação móvel. Este fator é determinado pela quantidade de informação que se pretende armazenar na estação móvel, e é excludente

quando a escolha do banco de dados depende da quantidade de recursos da plataforma oferecida para operar o aplicativo. Esta análise se dará "quanto ao tamanho do SGBD" o intervalo a ser avaliado terá como extremos os valores "<150 Kb" e ">= 1Mb".

- Uma das principais características observadas em um banco de dados é a forma pela qual se dará o acesso aos dados por ele armazenados. Uma maneira de acessar os dados é de forma nativa, através das ferramentas de desenvolvimento sugeridas pelos fabricantes do banco de dados, outra maneira, é prover acesso através de mecanismos comuns que independam das ferramentas de desenvolvimento (ODBC e JDBC). Alguns fabricantes oferecem também ferramentas especiais para a consulta direta a base de dados, não necessitando a criação de funções para esta finalidade. Na escolha de um banco de dados estas características são importantes para a determinação da ferramenta de desenvolvimento, a qual deverá contemplar pelo menos uma forma de acesso aos dados disponibilizados pelo banco de dados. A análise se dará "quanto a acessibilidade e utilização", sendo que os comparativos serão "flexibilidades em ferramentas de desenvolvimento", "ferramentas de consulta de dados na unidade móvel" e "indexação de múltiplas colunas".

- Outra característica importante é a facilidade do banco de dados móvel em realizar a sincronização de seus dados com diferentes bancos de dados de diferentes fabricantes. Quando a determinação dos bancos de dados que farão parte do ambiente distribuído não pode ser feita no inicio do projeto, seguramente esta independência será fundamental para se evitar uma alteração no aplicativo, o que pode representar um grande retrabalho na maioria dos casos. Esta análise se dará "quanto a sincronização com outros bancos de dados", tendo como comparativo os mesmos cinco bancos de dados já selecionados (Informix, DB2, Oracle, Sybase, MS SQL Server), os demais bancos de dados serão denominados como "outros".

- A forma de atualização das informações no banco de dados da estação móvel de forma transparente para o usuário, como que se estivesse conectado o tempo todo, é o desejo maior quando se trabalha com computação móvel, mas esta característica possui uma importância maior em projetos cuja necessidade de atualização dos dados seja em tempo real. Neste caso o banco de dados deve ter a capacidade de tratar a desconexão e alternar entre o processo assíncrono gravando/acessando informações do banco de dados local e o processo síncrono gravando/acessando informações do banco de dados remoto ao mesmo tempo em que replica as informações para banco de dados local. A análise se dará "quanto a forma de atualização dos dados" levando-se em consideração o modo "síncrono" e o modo "assíncrono".

- No processo de atualização dos dados, o meio de comunicação a ser utilizado pode inviabilizar um projeto de computação móvel se o custo/benefício na sincronização não for adequado. Por isto a capacidade de se utilizar diferentes mecanismos de transferência dos dados, pode ser um grande diferencial na escolha de um banco de dados móvel. A análise se dará "quanto aos mecanismos de transferência de dados" independentemente da utilização de um enlace sem fio ou da rede fixa como meio de comunicação, importando apenas os protocolos que possam ser usados para esta tarefa. No comparativo serão utilizados os protocolos mais comuns (HTTP, FTP, Conexão Serial e e-mail), os demais serão denominados como "outros".

- A mobilidade computacional também introduz novos problemas de segurança e autenticação. Na computação sem fio é mais fácil fazer uma interceptação de mensagens, o que sem dúvida causa sérios problemas de segurança. Neste sentido para minimizar os riscos se faz necessário uma avaliação deste critério quando se analisa uma relação de bancos de dados para um projeto de computação móvel. Esta análise se dará "quanto aos dispositivos de segurança", tendo como comparativos "autenticação do usuário", "transporte de dados em modo seguro", tratamento à "desconexão durante o processo de comunicação" e tratamento à "desconexão durante o acesso aos dados remotos".

5.8 RESULTADOS COMPARATIVOS DESTA ANÁLISE

A seguir serão mostrados os resultados apurados na análise. Os comentários comparativos serão apresentados na conclusão deste trabalho.

5.8.1 QUANTO A PLATAFORMA DE UTILIZAÇÃO

Esta análise mostra a presença do SGBD nas diversas plataformas, dando uma noção de independência do Banco de Dados quanto ao sistema operacional.

SGBD	Palm OS	Windows CE	Psion EPOC	QNX Neutrino	Embedded Linux	Outros
Informix SE	-	-	-	-	-	Sim
DB2 Everyplace	Sim	Sim	Sim	Sim	Sim	Sim
Oracle Lite	Sim	Sim	Sim	-	-	Sim
Adaptive Sever Anywhere	Sim	Sim	-	-	-	Sim
MS SQL Server 2000	-	Sim	-	-	-	Sim

Tabela 6 – Análise dos SGBDs quanto a plataformas utilização

5.8.2 QUANTO AO TAMANHO DO SGBD

As informações dispostas expõem a quantidade mínima de memória necessária no equipamento para suportar o SGBD.

SGBD	< 150 Kb	>= 150 Kb < 300 Kb	>= 300 Kb < 450 Kb	>= 450 Kb < 600 Kb	>= 600 Kb < 1Mb	>= 1Mb
Informix SE	-	-	-	-	-	✔
DB2 Everyplace	✔	-	-	-	-	-
Oracle Lite	-	-	-	-	✔	-
Adaptive Sever Anywhere	✔	-	-	-	-	-
MS SQL Server 2000	-	-	-	-	✔	-

Tabela 7 – Análise dos SGBDs quanto a memória utilizada

5.8.3 QUANTO A ACESSIBILIDADE E UTILIZAÇÃO

Na comparação a seguir, é analisada a forma de acesso aos dados armazenados no SGBD. Neste sentido, são observados os meios de acesso através das ferramentas de programação e também as ferramentas de acesso direto fornecidas pelo fabricante.

SGBD	Flexibilidade em Ferramentas de Desenvolvimento	Ferramenta de Consulta de Dados na Unidade Móvel	Indexação de Múltiplas Colunas
Informix SE	Desenvolvimento utilizando ODBC ou JDBC para acesso aos dados	Informix Client Tools	Possui
DB2 Everyplace	DB2 Personal Application Builder Desenvolvimento utilizando ODBC ou JDBC para acesso aos dados	QBE – Query-By-Example	Não Informado
Oracle Lite	Desenvolvimento utilizando ODBC ou JDBC para acesso aos dados	Não Informado	Não Informado
Adaptive Sever Anywhere	Desenvolvimento utilizando ODBC ou JDBC para acesso aos dados	Não Informado	Possui
MS SQL Server 2000	Desenvolvimento utilizando MS Visual C++ ou MS Visual Basic com instruções restritas para a plataforma Windows CE	Não Informado	Possui

Tabela 8 – Análise dos SGBDs quanto a acessibilidade

5.8.4 QUANTO A SINCRONIZAÇÃO COM OUTROS BANCOS DE DADOS

Esta análise identifica a capacidade do SGBD em interagir com outros bancos de dados durante um processo de atualização dos dados.

SGBD	INFORMIX	DB2	ORACLE	SYBASE	MS SQL SERVER	Outros
Informix SE	✔	-	-	-	-	✔
DB2 Everyplace	✔	✔	✔	✔	✔	-
Oracle Lite	-	-	✔	-	-	
Adaptive Sever Anywhere	-	✔	✔	✔	✔	-
MS SQL Server 2000	-	-	✔	✔	✔	-

Tabela 9 – Análise dos SGBDs quanto a utilização com SGBDs heterogêneos

5.8.5 QUANTO A FORMA DE ATUALIZAÇÃO DOS DADOS

As informações abaixo mostram a natureza do processo de replicação dos dados a partir da unidade móvel.

SGBD	Síncrona	Assíncrona
Informix SE	-	✔
DB2 Everyplace	-	✔
Oracle Lite	-	✔
Adaptive Sever Anywhere	-	✔
MS SQL Server 2000	✔	✔

Tabela 10 – Análise dos SGBDs quanto a forma de atualização dos dados

5.8.6 QUANTO AOS MECANISMOS DE TRANSFERÊNCIA DE DADOS

A análise a seguir mostra quais mecanismos de transferência de dados são suportados pelo SGBD.

SGBD	HTTP	FTP	Conexão Serial	e-mail	Outro
Informix SE	-	-	-	-	✔
DB2 Everyplace	✔	-	-	-	✔
Oracle Lite	✔	✔	✔	-	-
Adaptive Sever Anywhere	✔	✔	✔	✔	-
MS SQL Server 2000	✔	-	-	-	-

Tabela 11 – Análise dos SGBDs quanto ao mecanismo de transferência

5.8.7 QUANTO AOS DISPOSITIVOS DE SEGURANÇA

Esta análise mostra a forma de proteção e a segurança sobre as informações contidas no SGBD.

SGBD	Autenticação do Usuário	Transporte de Dados em Modo Seguro	Desconexão durante o Processo de Sincronização	Desconexão durante o acesso aos dados remotos
Informix SE	✔	-	-	-
DB2 Everyplace	✔	-	-	-
Oracle Lite	✔	✔	✔	-
Adaptive Sever Anywhere	✔	-	✔	-
MS SQL Server 2000	✔	✔	✔	-

Tabela 12 – Análise dos SGBDs quanto a segurança da informação

CAPÍTULO VI

6 CONCLUSÃO

O atual paradigma da computação móvel apresenta algumas suposições fundamentalmente incorretas sobre o papel das tecnologias sem fio no sentido de fomentar a mobilidade entre as pessoas que trabalham com informações. Essas premissas estão enraizadas na visão que iguala o acesso a informações com a conectividade online.

Entretanto, como a tecnologia de armazenamento vem avançando muito mais rapidamente do que a de comunicações sem fio, é mais provável que o acesso à informações de forma desconectada venha a representar um papel cada vez mais importante na promoção da mobilidade das equipes de trabalho. Em vez de construir sistemas que ofereçam acesso à informações em tempo real – um imenso desafio, considerando as leis da física – as organizações inteligentes procurarão desenvolver soluções que permitam que dados importantes, incluindo aplicativos, sejam eficientemente replicados para dispositivos móveis.

Enquanto a propagação síncrona de cada modificação a todos os nodos envolvidos evita conflitos, também aumenta o custo de comunicação de sistemas distribuídos e requer protocolos para reforçar mecanismos de consistência.

6.1 PERSPECTIVAS

Equipar profissionais móveis com assistentes pessoais digitais (PDA) capazes de armazenar gigabytes de informações e uma variedade de novos aplicativos irá introduzir novas oportunidades que podem não ser acessíveis no mundo conectado. Mas embora essa mobilidade abra as portas, ela também desafia as habilidades dos profissionais de computação para implementar sistemas de sincronização de dados que sejam eficientes e seguros. Em alguns aspectos, esses desafios não são nada novos.

A maior parte dos profissionais móveis já porta notebooks com grandes depósitos de informações proprietárias. Não obstante, a elevada freqüência de roubos de computadores portáteis traz para todas as organizações significativos desafios quanto à avaliação de riscos. A indústria de software tem demorado a responder esses desafios, mas as pressões competitivas aumentam a demanda por acesso móvel à informações, e à medida em que essa exigência aumenta indubitavelmente, excelentes oportunidades surgirão para os desenvolvedores que podem satisfazer essas necessidades.

Num futuro próximo, a capacidade de memória dos PDAs tende a aumentar em muito, se comparado aos padrões atuais, tornando o gerenciamento e a sincronização de dados tarefas mais desafiadoras. Atualizações e sincronização por meio da Internet poderão ser um componente para a solução.

A crescente utilização de equipamentos móveis, como telefones celurares e assistentes pessoais, projeta na indústria da computação uma necessidade de prover recursos para atender a necessidade de conectar qualquer pessoa, em qualquer lugar e a qualquer momento.

6.2 CONCLUSÃO DA ANÁLISE

A realização desta análise revela o quanto as pesquisas na área de tecnologia da computação móvel avançaram nos últimos anos e aponta as dificuldades presentes nesta área. A seguir será comentado o resultado de cada comparativo da análise.

6.2.1 QUANTO A PLATAFORMA DE UTILIZAÇÃO

Nesta avaliação observamos que o DB2 Everyplace destaca-se como uma boa opção de banco de dados para ser utilizado em projetos que possam vir a trabalhar com diferentes plataformas de sistemas operacionais. A avaliação nos revela também uma tendência forte em adaptação, pois 80 % dos bancos de dados considerados podem ser utilizados em pelo menos dois sistemas operacionais diferentes, o que reflete o grande esforço em pesquisa para prover a mobilidade independentemente do dispositivo.

6.2.2 QUANTO AO TAMANHO DO SGBD

Os PDAs, por serem equipamentos de alta portabilidade, possuem recursos limitados. Um destes recursos é a ausência de disco rígido, passando a armazenar dados e aplicativos em memória, que geralmente é bastante limitada. O resultado desta análise mostra que dois bancos de dados se destacaram neste critério: o DB2 Everyplace e o Adaptive Server Anywhere. Estes bancos de dados são uma boa saída para projetos onde se deseja utilizar equipamentos com poucos recursos de memória.

6.2.3 QUANTO A ACESSIBILIDADE E UTILIZAÇÃO

As informações deste comparativo mostram uma forte tendência dos bancos de dados em obterem independência das ferramentas de desenvolvimento, fornecendo acesso aos dados através de interfaces ODBC e JDBC. Além disso, os bancos de dados Informix SE e o DB2 Everyplace se destacaram ao apresentarem uma ferramenta de consulta direta aos dados. Outro ponto avaliado e que é muito importante para diminuir o tempo de resposta nas consultas é a indexação de múltiplas colunas. Neste caso o Informix SE, o Adaptive Server Anywhere e o MS SQL Server 2000 possuem esta característica. Enquanto a flexibilidade em ferramentas de desenvolvimento tem grande importância para projetos onde não se deseja trabalhar com tecnologias proprietárias, a indexação de múltiplas colunas oferece um grande diferencial em projetos onde a taxa de crescimento da base de dados é alta.

6.2.4 QUANTO A SINCRONIZAÇÃO COM OUTROS BANCOS DE DADOS

A heterogeneidade, como visto no Capítulo III, é uma das alternativas de implementação de um banco de dados móvel. Neste critério os bancos de dados DB2 Everyplace e o Adaptive Server Anywhere foram os destaques, demonstrando que estão preparados para realizar a troca de informações com diferentes bancos de dados. Esta é

uma característica importante para projetos onde os dados a serem sincronizados podem estar em diferentes bancos de dados na rede fixa.

6.2.5 QUANTO A FORMA DE ATUALIZAÇÃO DOS DADOS

Nesta análise o destaque foi o MS SQL Server 2000, que oferece condições para atualização dos dados em modo síncrono e assíncrono. Esta é uma característica importante para projetos que necessitam manter os dados das estações móveis sempre atualizados.

6.2.6 QUANTO AOS MECANISMOS DE TRANSFERÊNCIA DE DADOS

Neste critério o destaque ficou para os bancos de dados Adaptative Sever Anywhere e o Oracle Lite, que oferecem diferentes protocolos para a troca de informações com outros bancos de dados. Uma tendência que pode ser observada é a utilização do protocolo HTTP pela maioria dos bancos de dados analisados. Isto mostra o quanto está se apostando na Internet como meio de transporte dos dados para a computação móvel. A possibilidade de realizar a troca de informações de diferentes formas pode facilitar a operação de projetos onde a infra-estrutura de rede é carente.

6.2.7 QUANTO AOS DISPOSITIVOS DE SEGURANÇA

Pode-se observar que em todos os casos a autenticação do usuário foi o item de segurança indispensável, mas podemos destacar o Oracle Lite e o MS SQL Server 2000 por oferecerem o transporte de dados de forma segura, o que protege os dados contra uma possível interceptação. Os bancos de dados citados anteriormente, mais o Adaptive Server Anywhere, oferecem mecanismos de proteção dos dados durante as desconexões em processos de sincronização. Mas nenhum dos bancos de dados analisados apresentou uma solução para a consistência dos dados quando o acesso acontece de forma remota. O critério de segurança é indispensável em qualquer projeto de computação móvel que trabalhe com informações confidenciais.

6.3 TRABALHOS FUTUROS

Tendo em vista a grande mudança que a área da computação distribuída móvel tem proporcionado, não resta dúvida que, do ponto de vista acadêmico, as questões de mobilidade computacional são um tanto quanto desafiadoras. Muitos trabalhos podem ser desenvolvidos nesta área, como por exemplo:

- A determinação de uma implementação prática onde poderá ser utilizado um ou mais bancos de dados móveis citados neste trabalho, caracterizando as dificuldades e as soluções adotadas para alcançar o objetivo;
- O estudo e determinação de mecanismos para a sincronização de informações entre bancos de dados heterogêneos;
- Uma pesquisa para a utilização de agentes móveis na recuperação de informações armazenadas em bancos de dados móveis;
- Na área de segurança da informação poderão ser tratadas formas de autenticação das informações provenientes de bancos de dados móveis, a fim de evitar a interceptação;
- Um estudo aprofundado sobre transações móveis, revelando maneiras de garantir o processamento de informações em ambientes de computação móvel.

6.4 CONSIDERAÇÕES FINAIS

A criação de um comparativo com diversas soluções oferecidas por fabricantes de bancos de dados para o ambiente de computação móvel, deverá propiciar através deste trabalho informações para nortear projetos nesta área, sugerindo ou direcionando a aplicação de alguma dessas soluções.

REFERÊNCIAS BIBLIOGRÁFICAS

1. ABBEY Michael; COREY Michael J. **Oracle: Guia do Usuário,** São Paulo – 1997.

2. ACHARYA A.; BADRINATH B. R. **Checkpointing Distributed Applications on Mobile Computers**. In Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems, pages 73-80, Austin, TX, USA, Setembro 1994.

3. BADRINATH B.R.; IMIELINSKI T. **Replication and Mobility**, Proc. Of 2nd Workshop on the Management of Replicated Data (WMRD-II), pp. 9-12, Monterey, CA., Dezembro 1992.

4. BADRINATH B. R.; ACHARYA A.; IMIEIINSKI T. **Structuring Distributed Algorithms for Mobile Hosts**. In Proc. of the 14th International Conference on Distributed Computing Systems, Junho 1994.

5. BARBARÁ D.; MOLINA H. G. **Replicated data management in mobile environments: Anything new under the sun?** In IFIP Conference on Applications in parallel and Distributed Computing, 1994.

6. BATES Regis J.; GREGORY Donald, **Voice and Data Communications Handbook** – McGraw-Hill Series on Computer Communications, 1997

7. BERSON A. **Client/Server Architecture**, McGraw Hill, Estados Unidos, 1992.

8. CHANDY K. M.; LAMPORT L. **Distributed Snapshots: Determining Global States of Distributed Systems**. ACM Transactions on Computer Systems, 3(1):63-75, Fevereiro 1985.

9. CHESS D.; HARRISON C.; KERSHENBAU A. **Mobile agents.. Are they a good idea?** Tedim cal report, IBM Research Division, T.J. Watson Research

Center, Yorktown Heights, New York, mar 1995. URL: http://www.cs.dartmouth.edu/agent/papers/chapter.ps.

10. CHRYSANTHIS P. K. **Transaction Processing in Mobile Computing Environment**. In IEEE Workshop on Advances in Parallel and Distributed Systems, pages 77-82, Outubro 1993.

11. CITRIX. **The CITRIX Metaframe**. http://www.citrix.com, Outubro, 2001

12. CRISTIAN F. **Synchronous and asynchronous**. ACM Press, New York, US, 88-97, 1996

13. DB2 EVERYPLACE. http://www.ibm.com/software/data/db2/everyplace/, Outubro 2001.

14. DEERING S. **Internet Protocol version 6** - RFC 1883 - IEFT - http://www.ietf.org/, 1997.

15. DEMERS A.; PETERSEN K.; SPREITZER M.; TERRY D.; THEIMER M.; WELCH B. **The Bayou architecture: Support for data sharing among mobile users**. Proc. of the IEFE Workshop on Mobile Computing and Applications, pp. 2-7, Dezembro 1994.

16. DESPANDE P.M.; RAMASAMY K.; SKUKLA A.; NAUGTHON J. f.; **Caching Multidimensional Queries Usino Ckunks**. Procedente de SIGMOD, pp. 259-270, 1998.

17. DUCHAMP D.; TAIT C. D. **An interface to support lazy replicated dile service**. In 2nd WS on Management if Replicated Data, 1993.

18. DUNHAM M. H.; HELAL A. **Mobile Computing and Databases: Anything New?**, SIGMOD Record, Volume 24, Número 4, pp. 5-9, Dezembro 1995.

19. DUNHAM M. H.; HELAL A.; BALAKRISHNAN S. **A Mobile Transaction Model That Captures Both the Data and Movement Behavior**. To appear in

ACM/Baltzer Journal on Special Topics in Mobile Networks and Applications, 1997.

20. DURAN J.; LAUBACH A. **Virtual personal computers and the portable network**. In Proceedings of the IEEE Conference on Performance, communication, and Computing (Phoenix, AZ). IEEE Computer Society Press, Los Alamitos, CA, 1999.

21. ELMAGARMID A.; JING J.; BUKHRES O. **An efficient and reliable reservation algorithm for mobile transactions.** In Proceedings of the 1995 International Conference on Information and Knowledge Management (CIKM) Baltimore, MD, 28/Novembro – 02/Dezembro 1995.

22. FAIZ M.; ZASLAVSKY A.; SRINIVASAN B. **Revising strategies for mobile computing environments.** In ECOOP '95 WS on Mobility and Replication, 1995.

23. GOODMAND. J. **Trends in Cellular and Cordless Communications.** IEEE Communications Magazine, Junho, 1991.

24. GRAY J.; HELLAND P.; O'NEIL P. E.; SHASHA D. **The dangers of replication and a solution.** Proceedings of ACM SIGMOD, pages 173-182, Junho 1996.

25. HELAL A.; HASKELL B.; CARTER J. L.; BRICE R.; WOELK D.; RUSINKIEWICZ M. **Any Time, Anywhere Computing,** Kluwer Academic Publishers, EUA, 1999.

26. HELAL A.; HEDDAYA A.; BHARGAVA B. **Replication Techniques in Distributed Systems**, Kluwer Academic Publishers, 1996.

27. HUANG Y.; SISTLA A. P.; WOLFSON O. **Data Replication for Mobile Computers**, In SIGMOD Conference, pp. 13-23, 1994.

28. IMIELINSKI T.; BADDRINATH B. R. **Querying in highly mobile distributed enviroments**, Proc. Of 18th Int'l Conf. On VLDB '92, pp. 41-52.

29. IMIELINSKI T.; VISHNATWAN S.; BADRINATH B. R. **Energy Efficient Indexing on Air**. In Proceedings of the ACM SIGMOD international Conference on Management of Data, pages 25-37, Maio 1994.

30. INFORMIX SE, **http://www-4.ibm.com/software/data/informix/se/**, Outubro 2001.

31. JING J.; HELAL A.; ELMAGARMID A. **Client-Server Computing in Mobile Environments**. ACM Press, Volume 31, Número 2, pp. 125-135, Junho 1999.

32. JOSEPH A. D.; KAASHOEK M. F. **Building reliable mobile-aware applications using the Rover toolkit**. Wireless Networks, pp. 3, 5, 405–419, 1996.

33. JOSEPH A. D.; TAUBER J. A.; KAASHOEK M. F. **Mobile computing with the Rover toolkit**. IEEE Trans. Comput, pp. 46, 3, 337–352, 1997.

34. KLISTER J. J.; SATYANARAYANAN M. **Disconnected operation in the coda file system**. ACM Transactions on Computer System, 1992.

35. KOTTMANN D. A.; **Serializing operations into the past and future: A paradigm for disconnected operations on replicated objects**. In ECCOP '95 WS on Mobility and Replication, 1995.

36. KRISHNAKUMAR N.; JAIN R. **Protocols for maintaining inventory databases and user service profiles in mobile sales applications**. In Proceedings of the Mobidata Work-shop. Rutgers University Press, New Brunswick, NJ, 1994.

37. KUMAR P.; SATYANARAYANAN M. **Supporting application-specific resolution in an optimistically replicated file system**. In Proceedings of the 4th Workshop on Workstation Operating Systems (Napa, CA), Outubro 1993.

38. KUMAR V.; DUNHAM M. H. **Defining Location Data Dependency,Transaction Mobility and Commitment**. TECHNICAL REPORT 98CSE-01, Dallas, Southern Methodist University, Fevereiro 1998.

39. LE M.; SESHAN S.; BURGHARDT F.; RABAEY J. **Software architecture of the InfoPad system**. In Proceedings of the Mobidata Workshop on Mobile and Wireless Information Systems. Rutgers, NJ, Novembro 1994.

40. LEVY A.Y.; MENDELZON A.O.; SAGIV Y.; SRIVASTAVA D. **Answering queries using views, procedente de PODS**, pp. 95-104, 1995.

41. LIN Y. B. **Mobility management for cellular telephony networks**. IEEE Parallel and Distributed Technology, 4(4):65-73, Winter, 1996.

42. LUBINSKI A.; HEUER A. **Configured Replication for Mobile Applications**. Proc. der 12. GI – Workshop "Grundlagen Von Datenbanken", pp. 13-16, Junho 2000.

43. LUCENT TECHNOLOGIES, **Reconstructing Core Transmission**, http://www.lucent.com/livelink/130760_brochure.pdf, Janeiro 2001.

44. MATEUS Geraldo R. e LOUREIRO Antônio A. F. **Introdução à Computação Móvel** – 11ª Escola de Computação - SBC, 1998

45. MATTOSO M. L. Q. **Sistemas de Bancos de Dados Distribuídos e Paralelos**, Jornada de Atualização em Informática, Caxambú, MG, Brasil, julho 1994.

46. MEIER-HELLSTERN K. S.; ALONSO E. **The Use of SS7 and GSM to support high density personal communications**. Technical Report WINLAB-TR-24, Rutgers University, Dezembro 1991.

47. MICROSOFT SQL SERVER. **SQL Server 2000 Windows CE Edition**, Product Guide, Março 2001.

48. NEVES N.; FUCHS W. K. **Adaptive Recovery for Mobile Environments**. Communications of the ACM, 40(1):69-74, Janeiro 1997.

49. NOBLE. B.; PRICE M.; SATYANARAYANAN M. **A programming interface for application-aware adaptation in mobile computing**. In Proceedings of the 2nd USENIX Symposium on Mobile and Location-Independent Computing, 1995.

50. NOBLE B. D.; SATYANARAYANAN M.; NARAYANAN D.; TILTON J. E.; FLINN J.; WALKER K. R. **Agile application-aware adaptation for mobility**. ACM SIGOPS Oper. Syst. Rev. 31, 5, 276–287, 1997.

51. NOETHER Angela. **Extending Enterprise Data to Mobile Devices**. http://www.developer.ibm.com/devcon/novcc00.htm, Outubro 2001.

52. NOETHER Angela. **Introduction to Wireless Technology and the IBM WebSphere Everyplace Suite**. http://www.developer.ibm.com/devcon/marcc01.htm, Outubro 2001.

53. ORACLE8 LITE. **The Internet Platform for Mobile Computing,** http://otn.oracle.com/products/lite/content.html, Outubro 2001.

54. ÖZSU M. T.; VALDURIEZ P. **Distributed DataBase Systems: Where Are We Now?**, in PUCRio DB Workshop on New Database Research Challenges, Rio de Janeiro, Brasil, pp. 74-92, setembro de 1994

55. ÖZSU M. T.; VALDURIEZ P. **Principles of Distributed DataBase System**, Segunda Edição, Prentice Hall, Estados Unidos, 1999.

56. PERKINS Charles, **IP Encapsulation within IP**: RFC 2003, IEFT - http://www.ietf.org/, 1996

57. PERKINS Charles, **IP Mobility Support version 2** - INTERNET DRAFT-IETF - http://www.ietf.org/, 1997

58. PERKINS Charles, **Mobile Networking Through Mobile IP**, IEEE Internet Computing Online, 1997

59. PETERSEN K.; SPREITZER M. J.; TERRY D. B.; THEIMER M. M.; DEMERS A. J. **Flexible update propagation for weakly consistent replication. Proceedings** of the 16th ACM Symposium on Operation Systems Principles, Saint Malo, França, pp 288-301, Outubro 1997.

60. PHATAK S. H.; BADRINATH B. R. **An architecture for mobile databases**, Department of Computer Science Technical report DCS-TR-351, Department of Computer Science, Rutgers University, New Jersey, 1998

61. PHATAK S. H.; BADRINATH B. R. **Conflict resolution and reconciliation in disconnected databases**, Proc. of Mobility in Databases and Distributed Systems (MDDS), Florence, Italy, Setembro 1999.

62. PHATAK S. H.; BADRINATH B. R. **Data partitioning for disconnected client server databases**, Proc. ot International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE'99), Seattle, Washington, Agosto 1999.

63. PHATAK S. H.; BADRINATH B. R. **Multiversion reconciliation for mobile databases**, Proc. of International Conference on Data Engineering (ICDE), Sydney, Australia, pages 582-589, Março 1999.

64. PHATAK S. H..; BADRINATH B. R. **Database server organization for handling mobile clients**, Department of Computer Science Technical report DCS-TR-324, Department of Computer Science, Rutgers University, New Jersey, 1997

65. PITOURA E.; BHARGAVA B. **Maintaining Consistency of Data in Mobile Distributed Environments**. Proceedings of 15th International Conference on Distributed Computing Systems, 1995.

66. PITOURA E. **A replication schema to support weak connectivity in mobile systems**. In 7th International Conference on database and expert Systems Applications (DEXA '96), 1996.

67. PITOURA E.; BHARGAVA B. **Building Information Systems for Mobile Environments**, Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, Maryland, ACM Press. pp. 371-378, Novembro 1994.

68. PRADHAN D. K.; KRISHNA P. P.; VAIDYA N. H. **Recovery in Mobile Wireless Environment: Design and trade-off Analysis**. In Proceedings of the 26th International Symposium on Fault-Tolerance Computing, pages 16-25, Sendai, Japan, Junho 1996.

69. PRAKASH R.; SIHGHAL M. **Low-Cost Checkpointing and Failure Recovery in Mobile Computing Systems**. IEEE Transactions on Parallel and Distributed Systems, 7(10), Outubro 1996.

70. RATNER D. H.; REIHER P. L.; POPEK G. J.; GUY R. G. **Peer Replication with selective control.** In Interactive Conference on Mobile Data Access, 1999.

71. REN Qun; DUNHAM Margaret H. **Using Clustering for effec tive Management of a Semantic *Cache* in Mobile Computing**, procedente de ACM 1999.

72. REVISTA POCKET PC, Fairfield, IA, USA, Thaddeus Computing, Inc., Janeiro 2001.

73. RHODES N.; MCKEEHAM J. **Palm OS Programming: The Developer's Guide**. Segunda Edição, California, O'Reilly & Associates, Inc., Janeiro 2002.

74. SARAIVA A. **Programando em Oracle**, Rio de Janeiro, Infobook, 1999.

75. SATYANARAYANAN M. **Fundamental challenges in mobile computing**. Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing, Philadelphia, Pennsylvania, US, 1996.

76. SATYANARAYANAN M.; NOBLE B.; KUMAR P.; PRICE M. **Application-aware adaptation for mobile computing**. ACM SIGOPS Oper. Syst. Rev. 29, 1, 52–55, Janeiro 1995.

77. SILVA S. D. **Sistema de Banco de Dados Heterogêneos: Modelo de Execução da Gerência de Transações**, Dissertação de Doutorado, Departamento de Informática, PUCRio, Rio de Janeiro, Brasil, 1994.

78. STROM R. E.; YEMINI S. **Optimistic Recovery in Distributed Systems**. ACM Transactions on Computer Systems, 3(3), 1985.

79. SYBASE SQL ANYWHERE, Sincronization Technologies for Mobile and Embedded Computing, http://www.sybase.com/detail/1,6904,1009526,00.html, Outubro 2001.

80. TAIT C. D.; DUCHAMP D. **Service inter-faceand replica consistency algorithm for mo-bile file system clients**. In Proceedings of the First International Conference on Parallel and Distributed Information Systems. Miami Beach, Florida, pp.190–197, 1991.

81. TAIT C. D.; DUCHAMP D. **An efficient variable consistency replicated file service**. In Proceedings of the USENIX File Systems Workshop. USENIX Assoc., Berkeley, CA, pp.111–126, 1992

82. TANIN E.; BEIGEL R.; SHNEIDERMAN B. **Incremental data structures and algorithms for dynamic query interfaces**. SIGMOD Record, 25(4):21-24, 1996.

83. TERRY D. B.; THEIMER M. M.; PETERSEN K.; DEMERS A. J.; SPREITZER M. J.; HAUSER C. H. **Managing update conflicts in Bayou, a weakly connected replicated storage system**. ACM SIGOPS Oper. Syst. Rev. 29, 5, 172–182, Dezembro 1995.

84. TERRY D.; DEMERS A.; PETERSEN K.; SPREITZER M.; THEIMER M.; WELCH B. **Session guarantees for weakly consistent replicated data**. In

Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems (PDIS, Austin, TX), Setembro 1994.

85. WALBORN D. G.; CHRYSANTHIS P. K. **Supporting Semantics Based Transaction Processing in Mobile Database Applications**. Proc. 14th IEEE Symp. on Reliable Distributed Systems, Setembro 1995.

86. WANG Y. M.; FUCHS W. K. **Optimistic Message Logging for independent Checkpointing in Message-Passing Systems**. In Proceedings of the 11th IEEE Symposium Reliable Distributed Systems, pages 147-154, Houston, TX, USA, Outubro 1992.

87. YEO L. H.; ZASLAVSKY A. **Submission of Transactions from Mobile Workstations in a Cooperative Multidatabase Processing Environment**. Proceedings of Distributed Computing Systems, pp. 372-379, 1994.

88. ZUKUNFT O. **Integration mobiler und oktiver Datenbankmechanismen als Basis für die ortsgebundene Vorgangsbearbeitung**. PhD thesis, University of Oldenburg, 1997.