UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Thienne de Melo e Silva Mesquita Johnson

"UMA ARQUITETURA DE COMPUTAÇÃO PERVASIVA PARA TRABALHO DE CAMPO"

ESTE TRABALHO FOI APRESENTADO À PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA UNIVERSIDADE FEDERAL DE PERNAMBUCO COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIA DA COMPUTAÇÃO.

ORIENTADOR: PROFESSOR DJAMEL SADOK

RECIFE, MARÇO DE 2005.

Pós-Graduação em Ciência da Computação

"UMA ARQUITETURA DE COMPUTAÇÃO PERVASIVA PARA TRABALHO DE CAMPO"

Por

Thienne de Melo e Silva Mesquita Johnson

Tese de Doutorado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

Recife, Março de 2005.

Johnson, Thienne de Melo e Silva Mesquita
Uma arquitetura de computação pervasiva para trabalho de campo / Thienne de Melo e Silva Mesquita Johnson. Recife : O Autor, 2005.
xii, 175 folhas : il., fig., gráf., tab.

Tese (doutorado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2005.

Inclui bibliografia e apêndices.

1. Rede de computadores e sistemas distribuídos. 2. Computação pervasiva – Adaptação de conteúdo – Roteamento por contexto. Computação móvel – Localização física. 3. Arquitetura de redes. I. Título.

004.72	**CDU (2.ed.)**	**UFPE**
0045.65	**CDD (22.ed.)**	**BC2005-412**

À minha família

AGRADECIMENTOS

Agradeço a Deus, a quem devemos pela existência de tudo e todos.

Agradeço a minha família, por todo o apoio em todos esses anos de minha formação acadêmica.

Agradeço a todos os professores do Curso de Ciência da Computação da Universidade Federal de Pernambuco e, em especial, ao meu Professor Orientador Djamel Sadok, pela paciência, incentivo e ensinamentos transmitidos ao longo da realização deste trabalho.

À Professora Judith Kelner, pela acolhida na equipe, ajuda e incentivos.

Ao Professor Carlos Kamienski, pelas valiosas dicas e ajuda com este trabalho.

Aos Colegas do GPRT e do CIN, pelas amizades criadas e coleguismo no trabalho.

Ao CNPq, pela bolsa de estudos.

RESUMO

Os ambientes de trabalho de campo possuem diversas restrições, como falta de infra-estrutura e dispositivos de baixa capacidade. Trabalhos atuais sobre redes *ad hoc* e computação pervasiva deixam de considerar diversos aspectos que poderiam melhorar os ambientes de campo. Este trabalho de tese descreve uma arquitetura composta de serviços e um protocolo projetados para dar suporte aos requisitos do trabalho de campo. Os serviços foram projetados para dar suporte aos diversos tipos de dispositivos, com diversos padrões de mobilidade. Os dados de roteamento dentro da área de trabalho consideram aspectos dos tipos dos dispositivos, indicando ao serviço de adaptação de conteúdo os tipos de nós do ambiente. O uso de uma estratégia de atualização de informações de localização física também reduz a carga de dados que é transmitida no ambiente. Também é apresentado um protocolo de roteamento que usa informações de localização física de diversas formas. Essas informações podem gerar sub-áreas de trabalho, limitando o escopo das mensagens que trafegam na rede, sendo utilizadas apenas pelos trabalhadores de uma área. Também permitem a troca de dados entre diferentes locais, através do uso de nós que controlam a borda das áreas de trabalho. O conhecimento dos nós vizinhos permite o roteamento por localização e a troca de informações somente entre os nós próximos. Estas contribuições podem também ser utilizadas por outros sistemas de computação pervasiva que tenham interesse em melhorar seu desempenho.

Palavras-chave: computação pervasiva; roteamento por contexto; localização física; adaptação de conteúdo.

ABSTRACT

Fieldwork environments present diverse restrictions, such as a lack of infrastructure and the use of low capacity devices. Current work on ad hoc networks and pervasive computing do not consider some aspects that could bring improvements to these kinds of environments. This thesis describes an architecture composed of services and a routing protocol to support fieldwork requirements. The services have been projected to support different types of devices, with diverse mobility patterns. The routing data inside of the work area considers aspects of the types of the devices, indicating to the content adaptation service the types of nodes in the environment. The use of a physical localization information update strategy also reduces the load of data that is transmitted over the network. A routing protocol that uses physical localization information in diverse forms is presented; this information can generate work sub-areas, limiting the target of the messages that are passed through the net to workers of a physical sub-area of interest. The physical localization information also allows data exchange between different work areas, through the use of border nodes that control the edges of the work areas. The knowledge of neighboring areas allows routing by localization and the exchange of information between adjacent areas. These contributions can also be used by other pervasive computer systems that have an interest in improving their performance.

Keywords: pervasive computing; context routing; physical localization; content adaptation.

SUMÁRIO

RESUMO ..... **I**

ABSTRACT ..... **II**

SUMÁRIO ..... **III**

LISTA DE FIGURAS ..... **VII**

LISTA DE TABELAS ..... **IX**

LISTA DE GRÁFICOS ..... **X**

LISTA DE ABREVIATURAS ..... **XI**

CAPÍTULO 1 - INTRODUÇÃO **1**

1.1 CONTEXTO E MOTIVAÇÃO **1**
1.2 DEFINIÇÃO DO PROBLEMA E SOLUÇÃO PROPOSTA **5**
1.3 ORGANIZAÇÃO DO TRABALHO **7**

CAPÍTULO 2 - COMPUTAÇÃO PERVASIVA **8**

2.1 COMPUTAÇÃO PERVASIVA **8**
2.1.1 AMBIENTES DE COMPUTAÇÃO PERVASIVA 9
2.1.2 DESAFIOS DOS AMBIENTES DE COMPUTAÇÃO PERVASIVA 10
2.1.2.1 Adaptação de Conteúdo 10
2.1.2.2 Escalabilidade 11
2.1.2.3 Descoberta de Serviços 11
2.1.2.4 Gerenciamento de Mobilidade 12
2.1.2.5 Rastreamento de Localização Física 12
2.1.2.6 Segurança 13
2.1.2.7 Dispositivos de Baixa Capacidade Computacional 13
2.2 TRABALHOS RELACIONADOS **14**
2.2.1 APLICAÇÕES PARA TRABALHO DE CAMPO 14
2.2.1.1 FieldNote 14
2.2.1.2 Stick e-Note 16
2.2.2 COMPUTAÇÃO PERVASIVA PARA REDES AD HOC 17
2.2.2.1 Network Kernel Framework (NFK) 17
2.2.2.2 Protocolo RKF 18
2.2.2.3 Arquitetura M2MI 20
2.2.2.4 iMAC 21
2.2.3 COMPARAÇÃO ENTRE AS ARQUITETURAS 22

2.3 ADAPTAÇÃO DE CONTEÚDO 24
2.3.1 CONTEXTO 25
2.3.2 ESTRATÉGIAS DE ADAPTAÇÃO 26
2.3.2.1 Tipos de Estratégias de Adaptação 26
2.3.2.2 Técnicas de Adaptação de Conteúdo 27
2.3.2.3 Local de Adaptação 30
2.3.3 TRABALHOS RELACIONADOS 32
2.3.3.1 Context-Aware Decision Engine 32
2.3.3.2 Arquitetura $2k^Q$ 34
2.3.3.3 *Middleware* iMash 36
2.3.3.4 Arquitetura Ninja 38
2.3.3.5 *Middleware DaCaPo++* 41
2.3.3.6 Arquitetura da Carnegie-Mellon 43
2.3.4 COMPARAÇÃO ENTRE TRABALHOS RELACIONADOS 44
2.4 CONCLUSÕES 46

CAPÍTULO 3 - TECNOLOGIAS 47

3.1 INTRODUÇÃO 47
3.2 DESCOBERTA DE SERVIÇOS 47
3.2.1 PROTOCOLOS DE DESCOBERTA 48
3.2.1.1 *Service Location Protocol* 48
3.2.1.2 Jini 49
3.2.1.3 *Universal Plug and Play* 49
3.2.1.4 Salutation 50
3.2.2 ARQUITETURAS PARA DESCOBERTA 51
3.2.3 ESCOPO DA DESCOBERTA (ESCOPOS GEOGRÁFICO E ESPACIAL) 52
3.2.4 RESUMO - DESCOBERTA DE SERVIÇOS 53
3.3 TECNOLOGIAS DE LOCALIZAÇÃO FÍSICA 54
3.3.1 LOCALIZAÇÃO FÍSICA EM REDES LOCAIS 55
3.3.2 LOCALIZAÇÃO EM REDES CELULARES BASEADAS EM RF 56
3.3.2.1 Método do Ângulo de Chegada 56
3.3.2.2 Métodos Tempo de Chegada e Diferença de Tempo de Chegada 57
3.3.2.3 Diferença de Tempo Observada 58
3.3.3 GPS 59
3.3.4 GPS ASSISTIDO 62
3.3.5 ESTRATÉGIAS DE RASTREAMENTO DE LOCALIZAÇÃO 63
3.3.5.1 Baseadas ou não em Infra-estrutura 63
3.3.5.2 Estratégias Dinâmicas e Estáticas 64
3.3.6 RESUMO - LOCALIZAÇÃO FÍSICA 67
3.4 ROTEAMENTO EM REDES AD HOC 67
3.4.1 PROTOCOLOS DE ROTEAMENTO CONVENCIONAIS 68
3.4.1.1 DSDV 68
3.4.1.2 DSR 69
3.4.1.3 AODV 70
3.4.1.4 TORA 71
3.4.2 ROTEAMENTO BASEADO EM LOCALIZAÇÃO FÍSICA 73
3.4.2.1 GeoGRID 73
3.4.2.2 GeoTORA 75
3.4.3 DISCUSSÃO DOS PROTOCOLOS DE ROTEAMENTO 75

3.5 CONCLUSÕES 77

CAPÍTULO 4 - PROTOCOLO X-AODV 78

4.1 INTRODUÇÃO 78
4.1.1 DEFINIÇÕES BÁSICAS 78
4.1.2 INFORMAÇÕES DE LOCALIZAÇÃO FÍSICA 80
4.1.3 PREMISSAS 83
4.1.4 FORMATO DOS PACOTES X-AODV 83
4.1.5 TIPOS DE MENSAGENS DA ARQUITETURA 85
4.1.6 GERAÇÃO DE ROTAS DENTRO DE UMA AI 89
4.1.7 GERAÇÃO DE ROTAS ENTRE DIFERENTES AI'S 91
4.1.8 ALGORITMOS X-AODV 93
4.1.8.1 Algoritmos de Roteamento 93
4.1.8.2 Mensagens Recebidas 95
4.2 CONCLUSÕES 98

CAPÍTULO 5 – ARQUITETURA PERVASIVA PARA TRABALHO DE CAMPO.. 99

5.1 INTRODUÇÃO 99
5.2 CAMADA DE CLIENTES E APLICAÇÃO 100
5.2.1 INTEGRAÇÃO NO AMBIENTE 101
5.3 CAMADA DE SERVIÇOS 102
5.3.1 FUNCIONAMENTO DA ARQUITETURA 104
5.4 A CAMADA DE TECNOLOGIAS DE COMUNICAÇÃO 105
5.4.1 TIPOS DE TECNOLOGIAS 105
5.5 UTILIZAÇÃO DE PERFIS 106
5.5.1 FORMATO DOS DADOS ARMAZENADOS NO RP 106
5.6 SERVIÇO DE LOCALIZAÇÃO FÍSICA (SLF) 107
5.6.1 ESTRUTURA DO SLF 107
5.6.2 REPOSITÓRIO DE MOBILIDADE 107
5.6.3 ESTRATÉGIA HÍBRIDA DE ATUALIZAÇÃO DE INFORMAÇÕES DE LOCALIZAÇÃO 108
5.6.4 BUSCA SOB-DEMANDA 111
5.7 SERVIÇO DE ADAPTAÇÃO DE CONTEÚDO (SAC) 112
5.7.1 PROTOCOLO DE NEGOCIAÇÃO DE ADAPTAÇÃO (PNA) 113
5.7.1.1 Escolha da Classes de Adaptação (CA) 115
5.7.1.2 Seleção dos Parâmetros de Dispositivos (PD) 116
5.7.1.3 Negociação dos Parâmetros de Adaptação 118
5.7.2 ESTRUTURA DO SAC 123
5.7.2.1 Funcionamento Geral do SAC 123
5.8 SERVIÇO DE DESCOBERTA DE SERVIÇOS (SDS) 124
5.8.1 PROVEDORES E USOS DE CLASSES 125
5.8.2 SERVIÇOS DE VIZINHOS 125
5.8.3 DIRETÓRIO DE APLICAÇÕES 126
5.8.4 USO DE *GEOCASTING* PARA OFERECER SERVIÇOS 127
5.9 OUTRAS SITUAÇÕES 127
5.9.1 TROCA DE AI 127
5.9.2 BASE FORA DO AR OU DE ALCANCE DE TODOS OS NÓS 128
5.10 CONCLUSÕES 129

CAPÍTULO 6 - AVALIAÇÃO DE DESEMPENHO 130

6.1 INTRODUÇÃO 130
6.2 SIMULAÇÃO 130
6.2.1 TIPOS DE SIMULADORES 130
6.2.2 AMBIENTE DE SIMULAÇÃO - NETWORK SIMULATOR 131
6.2.3 INTERVALO DE CONFIANÇA DA MÉDIA 132
6.3 RESULTADOS DE SIMULAÇÃO 132
6.3.1 VISÃO GERAL DA SIMULAÇÃO 132
6.3.2 METODOLOGIA DE SIMULAÇÃO 133
6.3.3 EXPERIMENTO 1 – DIVISÃO DA ÁREA EM AIS 134
6.3.3.1 Experimento 2 – Efeitos de Mobilidade 136
6.3.4 EXPERIMENTO 3 – ESTRATÉGIAS DE LOCALIZAÇÃO 141
6.3.5 EXPERIMENTO 4 - ROTEAMENTO 143
6.4 MODELAGEM MATEMÁTICA 147
6.4.1 AODV VERSUS X-AODV 147
6.4.1.1 Tempo de Resposta 147
6.4.1.2 Resultados 149
6.4.2 NÓS DE CAPACIDADE HETEROGÊNEA 151
6.4.2.1 Cálculo da Utilização 152
6.4.2.2 Cálculo do Tempo de Serviço de Roteamento 152
6.4.2.3 Resultados 153
6.4.3 TEMPO DE RESPOSTA NA ADAPTAÇÃO DE CONTEÚDO 155
6.4.3.1 Tempo de Resposta *versus* Modalidade de Adaptação 155
6.4.3.2 Tempo de Resposta *versus* Local de Adaptação 156
6.4.3.3 Tempo de Resposta *versus* Uso de Classe de Adaptação máxima 158
6.5 CONCLUSÕES 161

CAPÍTULO 7 163

7.1 CONCLUSÕES GERAIS 163
7.1.1 CONTRIBUIÇÕES 163
7.1.2 TRABALHOS FUTUROS 164

REFERÊNCIAS BIBLIOGRÁFICAS 166

ANEXO A – DIAGRAMA DE ESTADOS 173

ANEXO B – DIAGRAMA DE ATIVIDADES 174

LISTA DE FIGURAS

Figura 1.1: Trabalhador de campo com dispositivo para coleta de dados 1
Figura 1.2: Cenário de trabalho de campo com presença de uma Base..... 2
Figura 1.3: Exemplo de uma Base 2
Figura 1.4: Cenário de trabalho de campo com sincronização via sistema celular 3
Figura 1.5: Cenário de trabalho de campo com utilização de roteamento *ad hoc*..... 4
Figura 1.6: Várias equipes no campo..... 4
Figura 1.7: Divisão da área de trabalho em subáreas de acordo com o contexto..... 6
Figura 2.1: Visão Geral de Computação Pervasiva 9
Figura 2.2: Descoberta automática de serviços [BANA00] 12
Figura 2.3: Arquitetura do sistema FieldNote 15
Figura 2.4: Arquitetura NFK [ROTH02] 17
Figura 2.5: Arquitetura DOC com protocolo RKF [YAU03] 18
Figura 2.6: Arquitetura iMAC [CHEN02] 21
Figura 2.7: Aplicação servindo vários dispositivos heterogêneos..... 24
Figura 2.8: Desafios de Contexto 25
Figura 2.9: Arquitetura com *Decision Engine* [LUM02]..... 33
Figura 2.10: Visão geral da arquitetura $2K^Q$ [XU00]..... 34
Figura 2.11: Visão geral da Arquitetura iMash [PHAN01] 37
Figura 2.12: Handoff de Sessão de Aplicação [PHANT01] 38
Figura 2.13: Visão Geral da Arquitetura Ninja [GRIB00]..... 39
Figura 2.14: Visão Geral da Arquitetura DaCaPo++ [STIL99] 41
Figura 2.15: Diagrama da arquitetura [WEB00] 43
Figura 3.1: Descoberta *Late-biding* [MACB01]..... 52
Figura 3.2: Descoberta Early-biding [MACB01] 52
Figura 3.3: Localização em 802.11 [KISH01] 55
Figura 3.4: Métodos AOA [KHOK01] 56
Figura 3.5: Método Tempo de Chegada [KAIN02]..... 57
Figura 3.6: Método Diferença do Tempo de Chegada [KHOK01] 58
Figura 3.7: Método Diferença de Tempo Observada [EBLI01]..... 59
Figura 3.8: Distribuição de órbita [PUC03] 60
Figura 3.9: Recepção do sinal [DANA00] 61
Figura 3.10: Método de triangulação 61
Figura 3.11: Abordagem GPS rede-assitido com cálculo de localização em uma rede [NARE99]..... 63
Figura 3.12: Esquema Ativo e Esquema Passivo 64
Figura 3.13: Estratégia baseada em tempo..... 65
Figura 3.14: Estratégia baseada em movimentos 65
Figura 3.15: Estratégia baseada em Distância..... 66
Figura 3.16: Rede Ad Hoc [MOHA03] 68
Figura 4.1: Cenário de trabalho de campo 78
Figura 4.2: Exemplo de conjunto de *AI's*..... 80
Figura 4.3: Identificando Vizinhos 81
Figura 4.4: Troca de status para Borda por aproximação de outro Borda 82
Figura 4.5: Potência de transmissão variável de acordo com a distância física..... 83
Figura 4.6: Alcance das mensagens Atualiza Vizinhos 85

Figura 4.7: Geração inicial de enlace*s* no AODV 89
Figura 4.8: Seleção dos Vizinhos para geração de rota 90
Figura 4.9: Sequência de envio de mensagens 90
Figura 4.10: Exemplo de conhecimento dos nós Borda 91
Figura 4.11: Exemplo de roteamento do X-AODV 95
Figura 4.12: Diagrama de Estados simples 97
Figura 5.1: Visão geral da arquitetura 99
Figura 5.2: Interação da arquitetura 101
Figura 5.3: Integração no ambiente 102
Figura 5.4: Serviços e camadas 103
Figura 5.5: *Middleware* nos servidores Base 104
Figura 5.6: *Middleware* nos dispositivos dos usuários 104
Figura 5.7: Registro do Cliente 105
Figura 5.8: Módulos internos do SLF 107
Figura 5.9: Atualização de dados de localização física nos RM 108
Figura 5.10: Diagrama de Funcionamento do SLF 112
Figura 5.11: Funcionamento geral da adaptação 113
Figura 5.12: Template do perfil do UaProf 117
Figura 5.13: Exemplo de Perfil de Dispositivo 118
Figura 5.14: Intermediários entre Fonte e Destino 122
Figura 5.15: Módulos do SAC 123
Figura 5.16: Interceptação de Pacote de Pedido de Serviço Específico 126
Figura 5.17: *Geocasting* para oferecer serviço indispensável 127
Figura 5.18: Troca de AI 128
Figura 6.1: Área total e múltiplas AIs (4 e 9) 134
Figura 6.2: Relação entre as estratégias 143
Figura 6.3: Divisão do Tempo de Resposta [MENA02] 147

LISTA DE TABELAS

Tabela 2-1: Comparação entre as Arquiteturas 22
Tabela 2-2: Comparação das Arquiteturas 44
Tabela 3-1: Comparação dos Protocolos 76
Tabela 4-1: Campo Reserved no AODV e X-AODV 84
Tabela 4-2: Valores do campo Tipo de Mensagem 84
Tabela 5-1: Formato de dados no RP 106
Tabela 5-2: Formato do RM 108
Tabela 5-3: Escolha de Taxa de Atualização de Localização Física 110
Tabela 5-4: Classes de Adaptação de Conteúdo [SMIT99] 115
Tabela 5-5 : Parâmetros de Comparação 120
Tabela 5-6: Mudança dos Parâmetros do Conteúdo 120
Tabela 5-7: Vizinhos 125
Tabela 6-1: Parâmetros de simulação do Cenário 1 134
Tabela 6-2: Parâmetros de simulação do Experimento 2 136
Tabela 6-3: Parâmetros de simulação do Experimento 3 141
Tabela 6-4: Parâmetros de simulação do Experimento 4 144
Tabela 6-5: Tamanho de pacotes para os protocolos utilizados 149
Tabela 6-6: Parâmetros do cálculo de Tempo de Resposta 149
Tabela 6-7: Taxas de serviços para dispositivos variados 153
Tabela 6-8: Parâmetros do Cenário 5 153
Tabela 6-9: Parâmetros de Vídeo e Áudio 155
Tabela 6-10: Parâmetros de Áudio e Vídeo 157
Tabela 6-11: Parâmetros de Áudio e Vídeo para cada Classe de Adaptação 158

LISTA DE GRÁFICOS

Gráfico 6-1: Divisão da área de trabalho – Fração de Pacotes enviados (intervalo de confiança = ± 1) 135
Gráfico 6-2: Divisão da área de trabalho - Atraso fim-a-fim (intervalo de confiança = ± 0,02) 135
Gráfico 6-3: Comparação de carga normalizada de roteamento a)100 e b)50 conexões..... 137
Gráfico 6-4: Comparação de fração de pacotes enviados (a) 100 e (b) 50 conexões..... 138
Gráfico 6-5: Comparação de Atraso fim-a-fim (a) 100 e (b) 50 conexões 138
Gráfico 6-6:Atraso fim-a-fim comparando diversas estratégias 139
Gráfico 6-7: CRN comparando diversas estratégias..... 140
Gráfico 6-8: FPE comparando diversas estratégias 140
Gráfico 6-9: FPE para estratégias variadas (intervalo de confiança = ± 2) 142
Gráfico 6-10: Relação entre pacotes gerados e estratégias (intervalo de confiança = ± 12) . 142
Gráfico 6-11: Atraso fim-a-fim para estratégias variadas (intervalo de confiança = ± 0,04) 143
Gráfico 6-12: Comparação – Carga de Roteamento Normalizada 144
Gráfico 6-13: Comparação Fração de Pacotes Enviados 145
Gráfico 6-14: Comparação – Atraso fim-a-fim 145
Gráfico 6-15: Roteamento – Atraso médio 145
Gráfico 6-16: Roteamento – Fração de Pacotes enviado 146
Gráfico 6-17: Roteamento - Carga de Roteamento Normalizada..... 146
Gráfico 6-18: Número de Datagramas Gerados 150
Gráfico 6-19: Overhead gerado por cada protocolo 150
Gráfico 6-20: Tempo de resposta para largura de bandas diferentes (2 e 11 Mbps)..... 151
Gráfico 6-21: Tempo de resposta para número de salto*s* diferentes (5 e 10)..... 151
Gráfico 6-22: Tempo de Roteamento de cada tipo de CPU variando o tamanho dos dados roteados..... 154
Gráfico 6-23: Utilização de cada tipo de CPU variando o tamanho dos dados roteados 154
Gráfico 6-24: Tempo de Resposta de transmissão de vídeo para três modalidades diferentes 155
Gráfico 6-25: Tempo de Resposta de transmissão de áudio para três modalidades diferentes 156
Gráfico 6-26: : Tempo de Resposta de transmissão de vídeo para locais de adaptação diversos 157
Gráfico 6-27: Tempo de Resposta de transmissão de áudio para locais de adaptação diversos 157
Gráfico 6-28: Tempo de Resposta de transmissão de vídeo de acordo com CAmáx..... 159
Gráfico 6-29: Tempo de Resposta de transmissão de áudio de acordo com CAmáx..... 159
Gráfico 6-30: Tempo de Resposta de roteamento de vídeo de acordo com CAmáx..... 160
Gráfico 6-31: Tempo de Resposta de roteamento de vídeo de acordo com CPU 160
Gráfico 6-32: Tempo de Resposta de roteamento de áudio de acordo com CAmáx..... 161
Gráfico 6-33: Tempo de Resposta de roteamento de áudio de acordo com CPU 161

LISTA DE ABREVIATURAS

AI	Área de Interesse
AOA	Angle of Arrival
AODV	Ad-hoc On-Demand Distance Vector Routing
AP	Atualização Periódica
API	Application Programming Interface
APS	Atualiza Posição Física
APTC	Arquitetura Pervasiva para Trabalho de Campo
AS	Atualiza Sempre
AVZ	Atualiza Vizinhos
BSD	Busca sob-demanda
BVZ	Busca Vizinhos
CA	Classe de Adaptação
CD	Classe de Dispositivo
CLR	Clear
CR	Centro de Reportagem
CRG	Centro de Reportagem Generalizado
DAG	Directed Acyclic Graph
DAG-LBF	DAG Baseado em Localização Física
DAP	Diretório de Aplicações
DE	Decision Engine
DOA	Difference of Arrival
DOC	Distributed Object Computing
DSR	Dynamic Source Routing
EBD	Entre Bordas
EDS	Envio de dados
EVZ	Entre Vizinhos
GBM	Gera Busca na mesma AI
GBO	Gera Busca em outra AI
GEO	Geocasting
GIS	Sistema de Informação Geográfico
GPS	Global Positioning System
HSA	Handoff de Sessão de Aplicação
ILF	Informação de Localização Física
IMEP	Internet MANET Encapsulation Protocol
IR	Infrared
MA	Mobilidade Alta
MB	Mobilidade Baixa
MDD	Middleware
MM	Mobilidade Média
MWS	Middleware Server
NA	Nunca Atualiza
ORB	Object Request Broker
OTD	Observed Time Difference
PD	Parâmetro de Dispositivo
PDA	Personal Digital Assistant
PLS	Pedido de Listagem de Serviços
PRT	Pedido de Rota

PS	Provedor de Serviços
PSE	Pedido de Serviço Específico
QoS	Qualidade de Serviço
QRY	Query
RF	Rádio Freqüência
RM	Repositório de Mobilidade
RP	Repositório de Perfis
RPC	Remote Procedure Call
RSS	Received Signal Strength
SAC	Serviço de Adaptação de Conteúdo
SDS	Serviço de Descoberta de Serviços
SLF	Serviço de Localização Física
SLM	Salutation Manager
SLP	Service Location Protocol
SPS	Standard Positioning System
SSDP	Simple Service Discovery Protocol
TDOA	Time Difference of Arrival
TFR	Transmissor de Radio Freqüência
TOA	Time of Arrival
TORA	Temporally-Ordered Routing Algorithm
UPD	Update
UPnP	Universal Plug and Play
VPRT	Volta de Pedido de Rota

Capítulo 1 - Introdução

1.1 Contexto e Motivação

Em [RYAN99], o trabalho de campo é descrito como uma atividade que envolve freqüentes visitas a um mesmo ambiente de interesse. Consiste em estudos no local, desenhos, medidas de detalhes importantes, manutenção, etc. Alguns ramos de atividade que utilizam o trabalho de campo são mineração, energia (elétrica, alternativa), telecomunicação, transporte, geologia, biologia, arqueologia, dentre outros. Para se ter uma idéia da utilização desse tipo de atividade, o Departamento do Trabalho dos E.U.A. registra que mais de trinta milhões de profissionais americanos prestam serviço distante de seus escritórios [WILS00].

Neste tipo de trabalho, uma visita pode não ser suficiente para coletar todos os dados necessários. Muitas vezes é preciso realizar monitoramento a longo prazo para detectar mudanças ou fazer observações sob uma variedade de condições. No campo, é freqüentemente útil ter acesso às observações das visitas precedentes.

Os dispositivos mais utilizados pelos trabalhadores são os coletores de dados (Figura 1.1). Anteriormente eram dispositivos dedicados, hoje, dispõem de *software* de coleta implementados em computadores de mão *(handhelds)*. Entretanto, atualmente, somente computadores portáteis e *desktops* fornecem funcionalidade real; computadores de mão foram desenvolvidos para fornecer algumas funcionalidades quando um computador portátil não está disponível[1]. Porém, em ambientes de trabalho de campo, a portabilidade que esses dispositivos têm, devido a seus tamanhos pesos pequenos, os tornam uma ferramenta muito útil e difundida.

Figura 1.1: Trabalhador de campo com dispositivo para coleta de dados

Um cenário de trabalho de campo atual é exemplificado na Figura 1.2.

[1] http://www.edgereview.com/features.cfm?Category=Handheld&ID=11

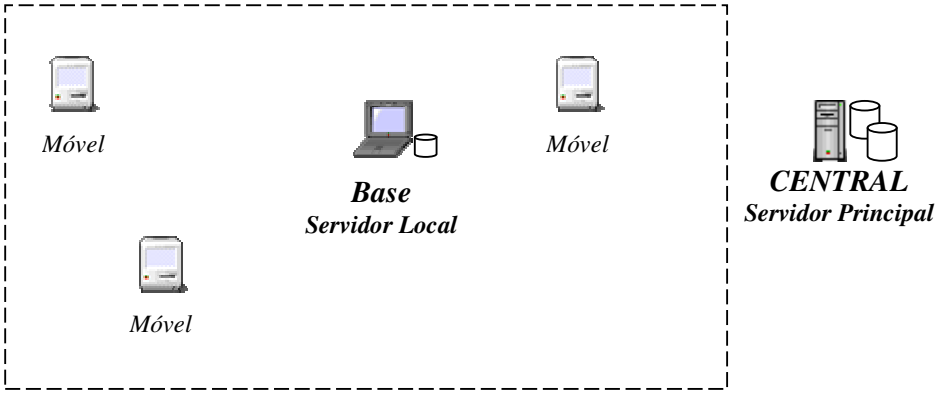


Figura 1.2: Cenário de trabalho de campo com presença de uma Base

Existe um servidor de dados (banco de dados, base de um Sistema de Informação Geográfico, etc) principal, localizado fora da área de trabalho, onde os dados são processados. Este servidor é chamado de Central. Quando um grupo de trabalhadores vai ao campo, uma Base local é montada para lhes dar apoio (exemplo na Figura 1.3).

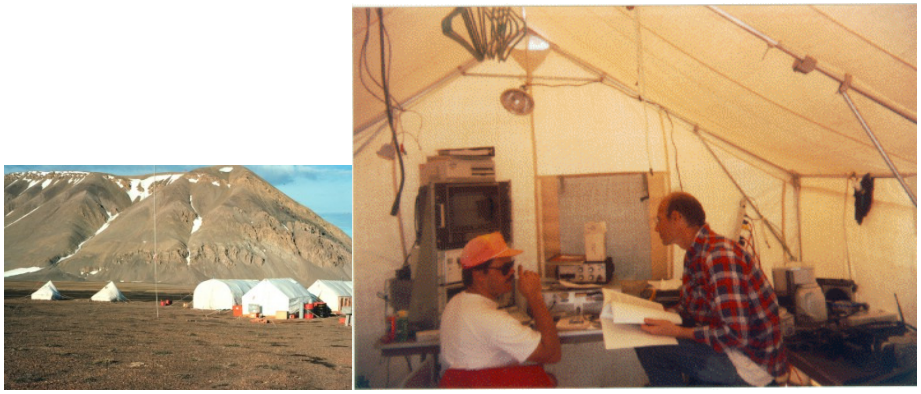


Figura 1.3: Exemplo de uma Base

Geralmente, um computador portátil é instalado na Base onde os dispositivos de cada trabalhador, normalmente computadores de mão , sincronizam os dados antes e depois de ir a campo realizar seu trabalho/pesquisa. O servidor da Base, quando retorna à Central, sincroniza os dados coletados durante a execução do trabalho.

Outro cenário, descrito em [BROW96], permite que os equipamentos móveis (nós) sincronizem com a Base de dados via comunicação de rádio, sem necessitar voltar à Base. Cada nó se comunica diretamente com a Base, portanto, tem que estar ao alcance desta para a sincronização. A Base, por sua vez, comunica-se com a Central via uma conexão *dial-up* Internet ou via telefone celular (Figura 1.4).

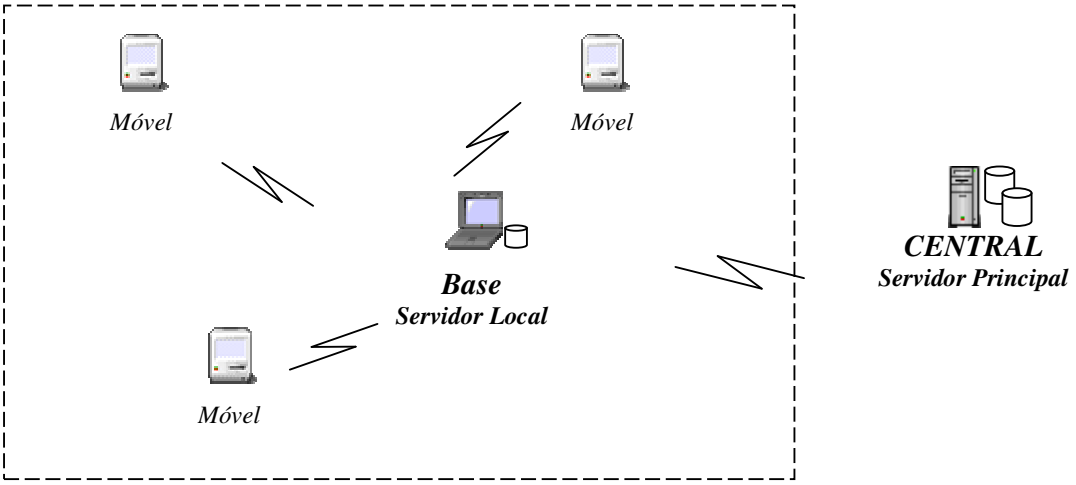


Figura 1.4: Cenário de trabalho de campo com sincronização via sistema celular

Entretanto, em áreas de trabalho muito extensas, a comunicação do móvel com a Base pode não ser possível devido ao pequeno alcance da comunicação. Pode não haver a possibilidade de comunicação via telefone entre a Base e a Central, em locais sem infraestrutura tecnológica (por exemplo, sem cobertura de telefone celular). Apesar da expansão das redes celulares e sem fio, existem diversas áreas em todo o planeta sem acesso a telecomunicação, exceto a comuincação via satélite, que possui alto custo. Dentre essas áreas podem ser citados os desertos, as florestas, os pólos e parte de países do terceiro mundo.

Se os pesquisadores pudessem investigar novos dados, em conjunto com dados anteriores, enquanto eles estão no campo, isso abriria novas estratégias de pesquisas combinando informações recentes com os dados já existentes. As aplicações de GIS móveis seriam beneficiadas significativamente com a interconectividade entre os pesquisadores e a Base, com a habilidade de ter acesso a bancos de dados sob-demanda e conseguir informações atualizadas de outras equipes de trabalhos remotos, em tempo real. Em um projeto multidisciplinar, investigadores individuais poderiam consultar novas informações com especialistas de outras disciplinas e conduzir um estudo no campo guiado remotamente [TRIP04].

Considerando que no local de trabalho de campo não exista infra-estrutura técnica (cobertura de telefonia nem redes de rádio), em um cenário de trabalho ideal como o descrito anteriormente, cada trabalhador móvel poderia se comunicar com a Base, através da utilização dos outros nós (roteamento *multi-hop* em redes Ad Hoc) e a Base poderia também aproveitar os nós que estão localizados entre ela e a Central, como mostra a Figura 1.5.

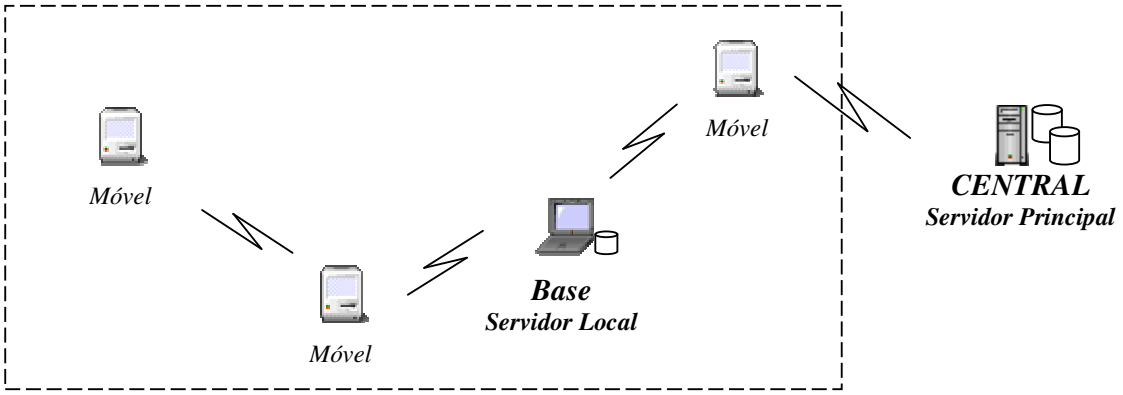


Figura 1.5: Cenário de trabalho de campo com utilização de roteamento *ad hoc*

Neste cenário, em locais com falta de infra-estrutura, a comunicação entre os nós e a Base permite que o alcance do trabalho de campo se estenda e, ainda assim, permita a atualização de informações com menor intervalo de tempo.

Um outro problema é a comunicação entre diversos grupos da equipe de trabalho de campo. Exemplificando, na Figura 1.6 existem 3 equipes de trabalhadores.

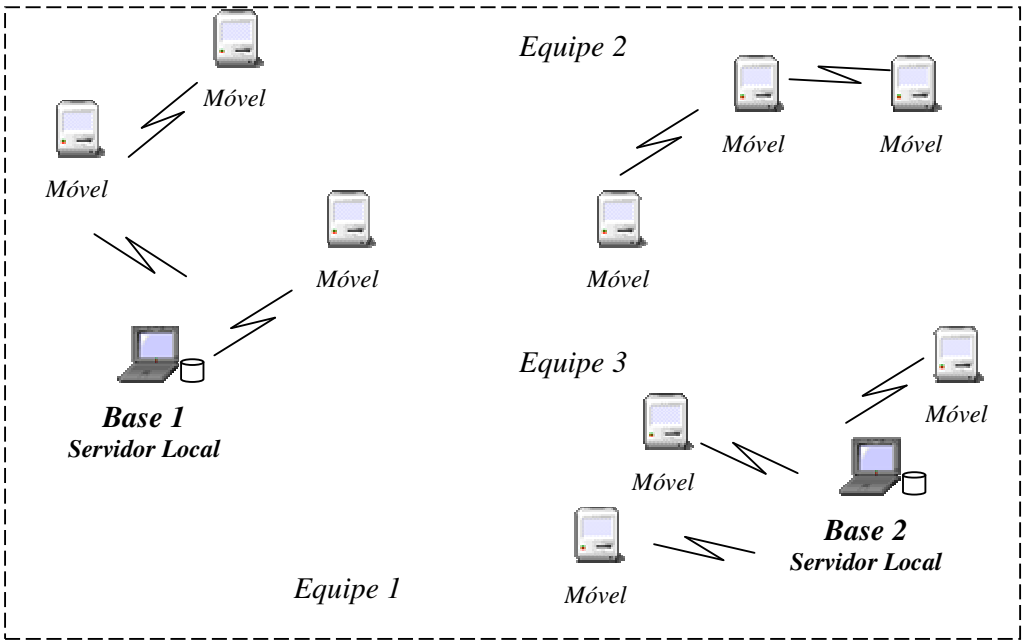


Figura 1.6: Várias equipes no campo

Neste cenário, cada equipe dividiu o trabalho em áreas de contextos diferentes, onde cada área é estudada apenas por integrantes da equipe. As transmissões de dados entre os integrantes devem se restringir apenas aos participantes locais, porque as informações não são interessantes para os trabalhadores de outras áreas.

Apesar disso, os integrantes de uma equipe podem desejar realizar transmissão de dados a participantes de outras equipes. Deve-se ter um método de repassar a informação tanto para outras áreas como para um participante específico localizado fora da área de estudo.

1.2 Definição do Problema e Solução Proposta

As principais características do trabalho de campo são:

1. Grandes áreas físicas: onde são realizados os trabalhos;
2. Grande quantidade de dados: geográficos, estudos, usuários;
3. Heterogeneidade de dispositivos: *notebooks,* computadores portáteis, computadores de mão, coletores de dados;
4. Variedade de usuários: categorias de usuários com diferentes níveis de prioridades e permissões;
5. Mobilidade de usuários: os usuários/dispositivos possuem grande mobilidade por toda a área de trabalho; e
6. Restrição de recursos de energia, comunicação e horas de conectividade como no caso de acesso via satélite.

Devido a essas características, um sistema de apoio ao trabalho de campo possui alguns requisitos básicos:

1. Alta disponibilidade de informações: o acesso à informação deve ser o melhor possível por toda a área de trabalho.
2. Dados de localização geográfica: são utilizados pelo trabalho em si e por toda a infra-estrutura de apoio.
3. Sensibilidade à localização física: deve-se priorizar os dados/aplicações referentes à certas localizações físicas, para evitar envio de informações não necessárias naquele momento
4. Adaptação de Conteúdo: devido à grande variedade de dispositivos presentes na área ed trabalho, deve-se realizar o envio de dados apropriados para cada tipo de equipamento.

Este trabalho mostra uma arquitetura projetada para atender estes requisitos relativos ao trabalho de campo, fornecendo o suporte necessário para atender às necessidade deste tipo de trabalho.

A arquitetura APTC (Arquitetura Pervasiva para Trabalho de Campo) [JOHN05] é formada por uma Camada de Serviços, que atende os requisitos de localização física e adaptação de conteúdo e um protocolo de roteamento, que dissemina as informações pela área de trabalho

Na Camada de Serviços, a utilização de Perfis gera abstrações dos componentes do ambiente para os desenvolvedores. Cada nó possui uma classificação para Classe de Dispositivo (parametrização de tipo de dispositivo e qualidade de comunicação), Classe de Prioridade (que permite realizar transmissões de dados com diferentes níveis de prioridades) e Classe de Mobilidade (depende do comportamento de movimentação dos dispositivos pela área de trabalho).

Os dados de Perfis são utilizados pelas aplicações para Adaptação de Conteúdo. Com o conhecimento do perfil dos nós em relação a sua capacidade física, a adaptação pode ser direcionada para o tipo do dispositivo alvo. No trabalho de campo, a adaptação é necessária devido à natureza de baixa capacidade (de processamento, comunicação, armazenamento, etc.) dos nós. O uso de um protocolo que combine os parâmetros dos dispositivos com as capacidades dos provedores de conteúdo e com parâmetros do conteúdo a ser enviado, também escolhendo o melhor local da adaptação, ajuda reduzir a carga de dados que trafega no ambiente, reduzindo a utilização dos nós de baixa capacidade.

O Serviço de Localização Física é responsável por monitorar a movimentação dos nós pela área de trabalho. Com a utilização da Classe de Mobilidade, cada tipo de nó possui sua própria estratégia de atualização de informações dependendo de sua movimentação.

A divisão da área de trabalho em Áreas de Interesse (áreas físicas divididas por contexto para o trabalhador - Figura 1.7) motivou o projeto de um protocolo que dissemina informações somente para os profissionais daquela área.

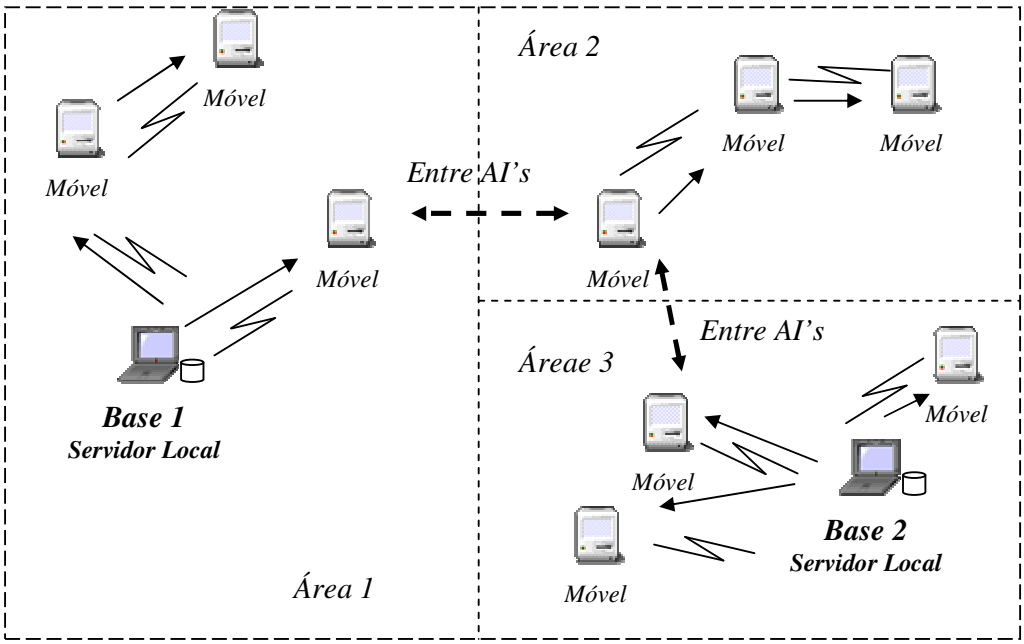


Figura 1.7: Divisão da área de trabalho em subáreas de acordo com o contexto

Para a transferência de dados dentro de uma mesma Área de Interesse, utilizou-se como base o protocolo de roteamento AODV [PERK03] no desenvolvimento do protocolo X-AODV (conteXt-AODV). No momento de geração de rotas, o protocolo adquire a Classe dos Dispositivos para que seja possível evitar o roteamento através de nós de baixa capacidade. Dependendo do contexto dos nós no caminho, a rota mais adequada é escolhida ou o conteúdo transmitido é adaptado de acordo com o perfil da rota.

Os Clientes cooperam entre si para o melhor funcionamento do ambiente. Trocam informações com seus vizinhos: localização física e serviços disponíveis. Através deste pré-conhecimento, facilitam a geração de rotas e a busca de serviços na rede.

1.3 Organização do Trabalho

Este trabalho está organizado em sete Capítulos.

O 1º Capítulo apresenta o contexto, a motivação deste trabalho e um resumo das soluções propostas.

O 2º Capítulo apresenta uma visão geral do estado da arte sobre Computação Pervasiva, incluindo os desafios mais importantes gerados por esse tipo de ambientes, e arquiteturas propostas por outros trabalhos e projetos.

No 3º Capítulo são apresentadas outras tecnologias relacionadas à este trabalho, como descoberta de serviços, tecnologias de localização física e protocolos de roteamento *ad hoc*.

O 4º Capítulo apresenta o protocolo de roteamento por contexto, o X-AODV (conteXt-AODV).

No 5º Capítulo está descrita a Arquitetura de Computação Pervasiva para Trabalho de Campo, com sua divisão em camadas, seus serviços e entidades participantes.

O 6º Capítulo mostra resultados de avaliações de desempenho sobre diversos aspectos da arquitetura.

Finalmente, o 7º Capítulo mostra as contribuições e trabalhos futuros desta Tese.

Capítulo 2 - Computação Pervasiva

"As mais profundas tecnologias são aquelas que desaparecem. Elas se tramam na vida cotidiana até que sejam indistinguíveis dela". Mark Weiser, 1991.

2.1 Computação Pervasiva

Pervasive significa difundido inteiramente por toda parte. Na língua portuguesa ainda não foi criada uma palavra para ser sinônima de *Pervasive Computing*. O termo *Computação Ubíqua* (de *Ubiquitous*, existente ou estando em todo lugar ao mesmo tempo, constantemente encontrado) é o mais utilizado, embora não possuam exatamente o mesmo significado. Neste trabalho, entretanto, será introduzido o termo **Computação Pervasiva** para definir o tipo de computação que está difundido por toda parte.

Este termo abrange o acesso móvel a dados e os mecanismos necessários para dar suporte a uma comunidade de usuários móveis com ênfase em espaços "inteligentes" ou ativos, ciência de contexto e o modo como as pessoas usam dispositivos para interagir com o ambiente [BANA00]. Também descreve uma classe de dispositivos que não se encaixa no perfil dos computadores pessoais tradicionais. Inclui dispositivos flexíveis e móveis, como assistentes pessoais digitais (PDAs) e telefones móveis, que têm ou provêem acesso a uma grande diversidade de aplicações.

A idéia básica da computação pervasiva é disponibilizar uma grande variedade de dispositivos inteligentes nos espaços de trabalho e/ou moradia. Estes dispositivos se coordenam para cada um oferecer aos usuários acesso universal e imediato às informações e dar suporte para os usuários desempenharem suas tarefas [GRIM00]. O aparecimento desses dispositivos e os novos tipos de conectividade criaram uma nova forma de comunicação em rede: redes dinâmicas, não gerenciadas, constituídas por dispositivos que entram e saem espontânea e imprevisivelmente.

A Figura 2.1 resume os conceitos de computação pervasiva. Qualquer dispositivo, utilizando qualquer rede de comunicação e seus serviços, pode ter acesso a quaisquer dados.

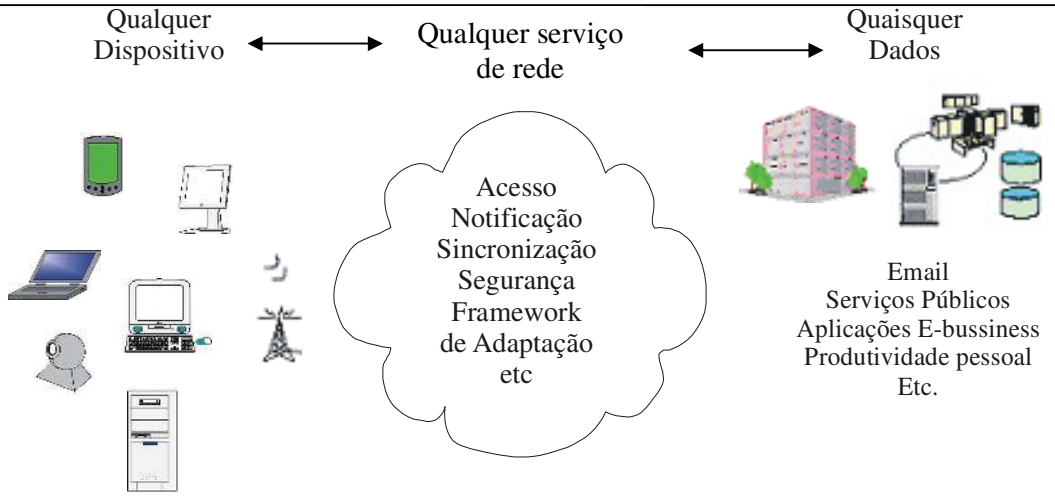


Figura 2.1: Visão Geral de Computação Pervasiva

As próximas sub-seções mostram o conceito de ambiente de computação pervasiva e os desafios identificados para esses ambientes.

2.1.1 Ambientes de Computação Pervasiva

Estes ambientes são compostos de espaços inteligentes, que contém dispositivos móveis sem fio, todos interconectados entre si, sentindo e reagindo inteligentemente a informações do ambiente, acessível a qualquer hora e lugar ("*anywhere and anytime*").

Estes espaços devem [NIST00]:

- Identificar e perceber usuários, suas ações e objetivos;
- Interagir com fontes de informações;
- Habilitar os dispositivos trazidos ao ambiente para se integrarem com outros dispositivos já presentes;
- Fornecer extensiva capacidade adaptativa de apresentação de informações;
- Entender e antecipar as necessidades dos usuários durante a execução de tarefas; e
- Permitir a colaboração local e distribuída, incluindo colaboração com pessoal de campo e trabalhadores móveis.

A infra-estrutura técnica para esses ambientes deve localizar os dispositivos mais apropriados, detectar quando novos dispositivos são adicionados espontaneamente ao sistema, coordenar entidades, ajudar dispositivos fracos (de capacidade simples), adaptar conteúdo quando novos formatos de dados não são compatíveis com os dispositivos de saída e facilitar a comunicação em rede. Esses ambientes devem também considerar a mobilidade e localização atual de usuários e dispositivos.

Como nos ambientes de trabalho de campo a infra-estrutura técnica não está presente, restringindo as possibilidades de se criar um ambiente de computação pervasiva com todas as possibilidades que ela oferece, o conceito de espaços inteligentes se restringe a:

- Permitir que os usuários e dispositivos possam ter acesso à informação, de (quase) qualquer local de trabalho. O ambiente se encarregará de gerir a comunicação de usuários e dispositivos às fontes de informações;
- Habilitar os dispositivos trazidos ao ambiente para se integrarem aos outros dispositivos já presentes;
- Utilizar informações contextuais para auxiliar na disseminação da informação, automatização de configuração, adaptação de qualidade de serviço e de conteúdo, etc.

A próxima Seção mostra alguns dos desafios relacionados aos ambientes de computação pervasiva.

2.1.2 Desafios dos Ambientes de Computação Pervasiva

A computação pervasiva promete um ambiente computacional que dê suporte a usuários e suas tarefas, e torna invisíveis por toda parte os atuais dispositivos computacionais e tecnologias [GRIM00]. Vários problemas relacionados foram identificados por diversas pesquisas em todo o mundo.

2.1.2.1 Adaptação de Conteúdo

A grande variedade de dispositivos computacionais, como PCs, computadores portáteis, PDAs, câmeras de vídeo, projetores, entre outros, cria o problema da heterogeneidade de capacidade de atributos físicos [CUI01]. Podem ocorrer variações de processamento na CPU, tamanho da tela, resolução de imagens, limitação de carga de bateria [KOKR00], conectividade limitada, largura de banda intermitente, que manipulam apenas com um pequeno conjunto de formato de dados e protocolos de redes [GRIB00].

Como dispositivos pequenos normalmente não têm capacidade computacional suficiente para manipular o rico conjunto de tipos de dados que podem trafegar em um ambiente de computação pervasiva, e o crescente conjunto de protocolos desenvolvidos, o ambiente deve adaptar o conteúdo e os protocolos à capacidade de entrada e saída desses dispositivos.

A computação ciente de contexto é o paradigma no qual aplicações podem descobrir e tirar vantagem de informações contextuais (como localização de usuário, hora do dia, pessoas próximas, e atividade de usuários e dispositivos) [CHEN00]. Esse paradigma habilita uma nova classe de aplicações de computação pervasiva. Essas aplicações ajudam, por exemplo, a

navegar por territórios não familiares, encontrar restaurantes mais próximos, receber mensagens de modo mais útil e menos intrusivo, e encontrar pessoas com interesses similares. O uso de informações contextuais por essas aplicações reduz a quantidade de atenção humana que uma aplicação necessita para atender as requisições dos usuários. Como a interação usuário-aplicação pode ser um gargalo, as aplicações oferecem uma oportunidade para aumentar a produtividade humana de modo substancial [EBLI01].

Esses problemas podem ser resolvidos por técnicas de Adaptação de Conteúdo. Tais técnicas podem ser consideradas como um tipo de sistema inteligente que pode tomar decisões de acordo com a versão de resposta de conteúdo ou estratégias de adaptação com ciência de vários tipos de contexto. Detalhes sobre as técnicas e arquiteturas de Adaptação de Conteúdo serão mostradas na Seção 2.3.

2.1.2.2 Escalabilidade

Um grande número de dispositivos computacionais pode estar trafegando pelo ambiente de trabalho. Isso gera um problema de escalabilidade dos sistemas, que deverá dar suporte ao crescimento do número de usuários e dispositivos. Os ambientes devem estar preparados para serem escaláveis e se adaptarem às novas situações de grande número de usuários.

Um serviço deve suportar aumento dos níveis de carga em um curto período e deve continuar disponível (exemplo: grande aumento do número de usuários e dispositivos e conseqüente aumento de tráfego). Entre outras técnicas e algoritmos adotados para simplificar uma arquitetura, a escalabilidade é alcançada através da replicação de clones por múltiplos recursos físicos [GRIB00].

2.1.2.3 Descoberta de Serviços

Espera-se um representativo crescimento de serviços que se tornarão disponíveis em todos os ambientes de computação: acesso à informação, música sob-demanda e serviços que usam a infra-estrutura computacional disponível no ambiente. Com essa tendência, deve-se permitir que os usuários encontrem as aplicações e serviços desejados [BETT00].

Portanto, são necessários mecanismos de descoberta que permitam aos dispositivos identificar e enumerar as aplicações e serviços em sua vizinhança [BANA00]. Como exemplo, a Figura 2.2 mostra um usuário que possui alguns dispositivos detectando, automaticamente, os serviços oferecidos pelo automóvel.

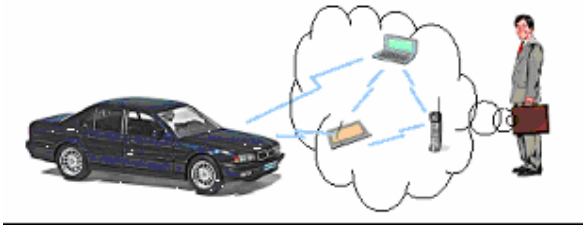


Figura 2.2: Descoberta automática de serviços [BANA00]

Essas tarefas podem ser realizadas pelos protocolos de Descoberta de Serviços, como o *Service Location Protocol* [VEIZ97], *Jini* [SUN99] , *Universal Plug and Play* [UPNP00] e *Salutation* [CHAK00]. Os serviços podem se mostrar, fornecer detalhes sobre suas capacidades e informações de como acessá-los [BETT00]. Portanto, os ambientes de computação pervasiva devem fornecer informações aos usuários e dispositivos sobre como:

- Registrar e divulgar serviços;
- Buscar e examinar serviços;
- Escolher o serviço correto (de acordo com sua necessidade); e
- Utilizar o serviço.

Detalhes sobre Descoberta de Serviços serão mostrados na Seção 3.1.

2.1.2.4 Gerenciamento de Mobilidade

A alta mobilidade de usuários e dispositivos torna necessário um esquema de gerenciamento de informações de mobilidade. Esse serviço pode ajudar a gerenciar recursos de rede (alocação de recursos, controle de fluxo e congestionamento) e permitir realizar previsões sobre a movimentação de usuário/dispositivo.

O gerenciamento de mobilidade é importante porque possibilita economia na alocação de recursos. Por exemplo, ao prever a movimentação de um dispositivo entre redes diferentes, pode-se avisar ao responsável por adaptação de conteúdo, a provável mudança de rede, permitindo a pré-configuração da adaptação da aplicação para o novo tipo de rede de comunicação.

Os problemas de desconexão, largura de banda estreita e alta variação na largura de banda podem ser previstos com o gerenciamento de mobilidade, minimizando a ocorrência desses efeitos.

2.1.2.5 Rastreamento de Localização Física

A mobilidade de usuários e dispositivos, juntamente com a necessidade de se ter Ciência de Contexto, torna o serviço de Rastreamento de Localização Física essencial.

As informações de localização representam o papel mais importante na definição de contexto [BHAT01]. As aplicações disponíveis no ambiente devem adaptar e responder à presença de pessoas e objetos no ambiente, dependendo, portanto, de um serviço de localização fornecido pela infra-estrutura ou por serviços conscientes de contexto [LEON96].

Um ambiente de computação pervasiva deve oferecer rastreamento em tempo real, pois diversas aplicações, como na indústria, saúde, energia e segurança, são dependentes de dados atualizados. A granularidade da atualização (freqüência) depende da aplicação em questão.

2.1.2.6 Segurança

Muitas informações estão disponíveis nos ambientes de computação pervasiva, como informações dos usuários, de serviços, redes e dispositivos. Informações contextuais podem conter dados privados ou pessoais. Para os serviços e aplicações contextuais ganharem mais aceitação e alcançarem o objetivo de reduzir a atenção humana, é necessário estudar os aspectos de segurança envolvidos e oferecer proteção aos usuários dos ambientes [EBLI01], incluindo autenticação e criptografia. Deve-se oferecer, além da segurança de rede, o controle de acesso e a segurança física. A segurança física e o controle de acesso prevêem o uso não autorizado de dispositivos.

2.1.2.7 Dispositivos de Baixa Capacidade Computacional

Utensílios de rede são dispositivos habilitados a se comunicar em rede. Eles possuem menos recursos que computadores pessoais e são desenvolvidos para um número limitado de aplicações [GAED98]. Alguns exemplos são computadores de mão, assistentes pessoais digitais e telefones celulares.

Dispositivos de menor capacidade computacional se diferem dos típicos computadores pessoais. Eles possuem telas pequenas, com baixa resolução e número de cores; poder de processamento e armazenagem (memória) também reduzidos. Usam uma variedade de tipos de comunicação em rede, com diferentes larguras de banda, características e custos.

O acesso à informação por esses dispositivos deve considerar diversas restrições que existem: tamanho de tela, poder de processamento e consumo de energia são relevantes [FOX98].

- Consumo de energia

A vida da bateria e o consumo da energia são problemas básicos que atingem os dispositivos móveis. Devem ser tomados cuidados especiais com aplicações que consumam esse recurso. Por exemplo, deve-se evitar consumo de energia com uso de áudio, definir novos modos de operação como o *sleep-mode* quando não houver atividades no terminal, etc.

- Poder de processamento

Além de possuir menor capacidade computacional do que computadores *desktops*, a capacidade computacional dos dispositivos dessa categoria de baixa capacidade também se diferem. Portanto, o conteúdo que necessite de muito processamento, como vídeo comprimido, deve ser evitado ou simplificado (pré-processado).

- Propriedades de tela

Estes dispositivos possuem telas muito pequenas, comparadas com computadores *desktop*. Resoluções de tela variam de 160x98 a 640x480, tamanho de 3,3x2,1 cm^2 até 13x8 cm^2 e número de cores que variam desde preto e branco até 16 milhões de cores.

- Comunicação

O dispositivo de baixa capacidade computacional também possui pouca largura de banda de comunicação, o que resulta em acesso lento a conteúdos avançados de multimídia. Informações que serão enviadas para esses dispositivos móveis devem ser adaptadas à largura de banda disponível e custos de transmissão. O tráfego que passa por eles deve ser adaptado, pois um nó fraco entre dois nós com maior capacidade estrangula a vazão entre os nós de mais alta capacidade. [SRIV04]. O processamento gasto com o repasse de pacotes gera consumo de energia dos dispositivos. Quanto menor o tamanho dos dados a serem repassados, menor o gasto de processamento e energia.

2.2 Trabalhos Relacionados

Os trabalhos relacionados incluem aplicações desenvolvidas especificamente para trabalho de campo, como o FieldNote [RYAN99] e o Stick e-Note [BROWN96], e arquiteturas de redes *ad hoc*, como NFK [ROTH02] e iMAC [CHEN02].

2.2.1 Aplicações para Trabalho de Campo

Os diversos ramos de trabalho de campo utilizam aplicações dedicadas à coleta de dados. As aplicações que serão mostradas nesta Seção realizam a coleta de notas, uso de informações contextuais e sincronizam com a Base através de enlace*s* de rádio ou de telefone.

2.2.1.1 FieldNote

Em [RYAN99], o trabalho de campo é descrito como uma categoria de trabalho que envolve visitas repetidas a uma mesma área de interesse. Uma visita pode não ser suficiente para coletar todos os dados necessários, ou a intenção pode ser realizar o monitoramento em

longo prazo para detectar mudanças, ou para fazer observações sob uma variedade de condições ambientais, tais como o tempo ou variações devido à estação atual.

No campo, é freqüentemente útil ter acesso às observações das visitas precedentes. Em um caderno convencional do campo, as visitas precedentes podem ser revistas pelo interessado voltando às páginas precedentes no caderno. Um computador de mão pode oferecer sistemas mais flexíveis para procurar e indicar dados das visitas precedentes, ou pode ser utilizado por outros trabalhadores de campo. O sistema FieldNote contém ferramentas para o levantamento de dados de reuso no processo, de maneira pouco intrusiva nas tarefas normais dos trabalhadores.

O sistema FieldNote tem três componentes: uma base de dados para o armazenamento e a recuperação das notas; um computador de mão para o levantamento de dados e acesso à informação contextual no campo; e um *software desktop* para edição, análise e visualização (Figura 2.3). Tais sistemas podem ser combinados de várias maneiras para servir completamente às necessidades de projetos de campo, de levantamento de dados em pequena escala por um indivíduo à colaboração em longo prazo por grupos.

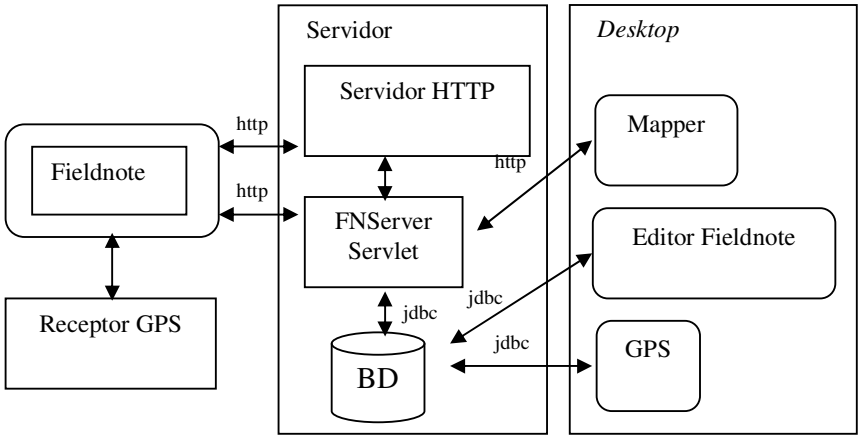


Figura 2.3: Arquitetura do sistema FieldNote

No sistema desenvolvido em 1999, os dados coletados nos computadores de mão são armazenados em um computador portátil, normalmente no horário de almoço e no fim do expediente. Quando uma conexão ao banco de dados da Central estiver disponível, todas as atualizações são transferidas. Isso funciona satisfatoriamente para conexões cabeadas provisórias, porém é menos apropriado para conexões usando enlaces de rádio. Por isso, foram projetadas algumas extensões ao protocolo de comunicação visando garantir que os dados não sejam perdidos quando ocorrerem falhas nos enlaces.

Um desenvolvimento adicional pode permitir que o servidor do campo (o computador portátil) se comunique com a Central (servidor principal) usando o mesmo protocolo

empregado entre o usuário com um computador de mão e o servidor do campo. Isso permitirá espelhar dados novos do campo periodicamente no servidor principal quando um enlace de comunicação, tipicamente uma conexão de linha discada para a Internet, estiver disponível.

Com as implementações portáteis do banco de dados principal, um servidor HTTP é usado para a troca de dados com os computadores de mão. Os vários servidores e os métodos de acesso da base de dados implementados em Apache e servlets Java fornecem o mecanismo da troca de dados, comunicando-se com os computadores de mão por HTTP e pelo servidor da base de dados por JDBC. As comunicações entre computadores de mão e o servidor envolvem um número pequeno dos tipos de mensagem que são formatados como XML.

2.2.1.2 Stick e-Note

BROWN(1996) propõe um sistema eletrônico baseado nas notinhas de papel Post-it (papéis amarelos que possuem cola na parte traseira, para pequenas notas) chamado Stick e-Note. As notas são anexadas a um contexto de ambiente e se tornam ativas quando um usuário entra naquele contexto. Essas notas podem ser anexadas a todos os tipos de contexto, além de posição física, por exemplo, presença de pessoas ou objetos, temperatura, tempo, etc. As notas consistem de conteúdo e contexto e são apresentadas em SGML.

A arquitetura consiste de três tipos de componentes [PASC97]:

- Gatilho - compara o contexto atual do usuário com o contexto das notas carregadas;
- Execução - pode ser qualquer programa existente que executa a nota; e
- Conjunto de sensores - alimenta o módulo de gatilho periodicamente e esconde aspectos específicos dos sensores.

Como exemplo de aplicação, o Assistente de Arqueólogo [RYAN99], que utiliza o Stick e-note como base, dá suporte a exercícios de campo para estudantes em treinamento de arqueologia e ciências ambientais.

A gerência das coleções de notas de campo é realizada em uma máquina *Desktop* convencional. Um banco de dados de notas é necessário para servir como repositório para instruções e exercícios de campo, assim como para os dados coletados. Isto fornece meios diretos de troca de dados com outras bases de dados, GIS e ferramentas analíticas. Os pacotes das tarefas relacionadas aos grupos de estudantes ou projetos de pesquisa são selecionados e então carregados nos dispositivos móveis. No retorno do campo, os resultados do exercício e outros dados gravados são atualizados no sistema *desktop* onde, então, estarão disponíveis para análise e reuso. Para estadias mais longas no campo, pode-se usar um computador

portátil como substituto para a máquina *desktop* e realizar cópias de segurança regulares para os computadores de mão.

Com enlace*s* de dados apropriados, tais como um modem de rádio ou um telefone móvel, formulários podem ser usados como ferramentas remotas de acesso à base de dados ou como uma janela remota em um sistema de informação geográfico hospedado em um servidor remoto.

2.2.2 Computação Pervasiva para Redes Ad Hoc

Nos últimos anos, o crescente uso das comunicações *sem fio* popularizou as redes *ad hoc* [MOHA03]. Diversos estudos foram publicados com o objetivo de propor novas arquiteturas e protocolos para essas redes. Esta Seção mostra algumas desses trabalhos.

2.2.2.1 Network Kernel Framework (NFK)

Descrita em [ROTH02], esse middleware tem por objetivo ser uma plataforma para dispositivos pequenos e sistemas *desktop* tradicionais, com a especialidade de atender novos dispositivos que não possuem suporte em plataformas de *middleware* de domínio público, como o CORBA. O NFK é modular: para um dado dispositivo, um desenvolvedor deve implementar apenas um sub-conjunto de módulos específicos. A Figura 2.4 mostra os detalhes da arquitetura.

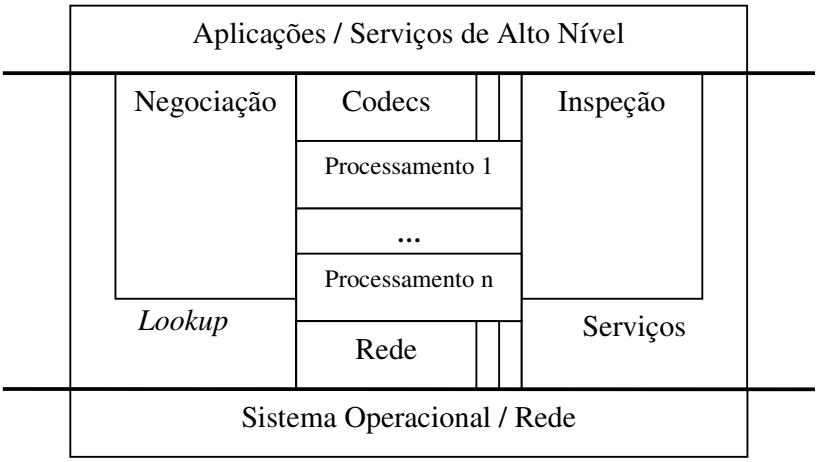


Figura 2.4: Arquitetura NFK [ROTH02]

Os módulos que compõem a arquitetura são do tipo *framework* e núcleo. Os módulos *framework* são partes fixas, oferecendo os serviços de *lookup* (procura outros nós na rede), Negociação (negocia parâmetros para uma conexão), Serviço (desempenha funções adicionais de configuração para conexões especiais, ex: conexão serial) e Inspeção (os serviços de nível

mais alto podem procurar propriedades para uma conexão específica). Os módulos núcleos são de três tipos: Rede (oferecem interface com os serviços de rede de nível mais baixo); Processamento (pré-processam pacotes de dados antes de enviar para a rede); e Codificador (codificam os dados para envio).

Esse formato possibilita que dois dispositivos diferentes se conectem de acordo com os seguintes passos: os dispositivos A e B se conectam (via TCP/IP, infravermelho ou cabo serial) e negociam os módulos que estão disponíveis nos dois lados da comunicação. Uma vez conectados, as aplicações podem enviar e receber dados independentemente da infra-estrutura de comunicação.

A arquitetura NFK deixa em aberto o protocolo de roteamento. Justificando que os desenvolvedores de aplicações podem criar algoritmos de roteamento de forma modular e independente de rede, um módulo simulador pode ser utilizado para projetar e testar um novo protocolo. Esse novo protocolo, depois de testado e compilado, pode ser utilizado no ambiente real sem modificações do protocolo utilizado no simulador.

2.2.2.2 Protocolo RKF

Este protocolo, descrito em [YAU03], tem por objetivo facilitar a computação de objetos distribuídos em ambientes heterogêneos de rede, incluindo redes móveis e fixas. Como mostra a Figura 2.5, o protocolo RKF pode ser utilizado dentro de um subsistema de comunicação de um *middleware* de objetos distribuídos (DOC).

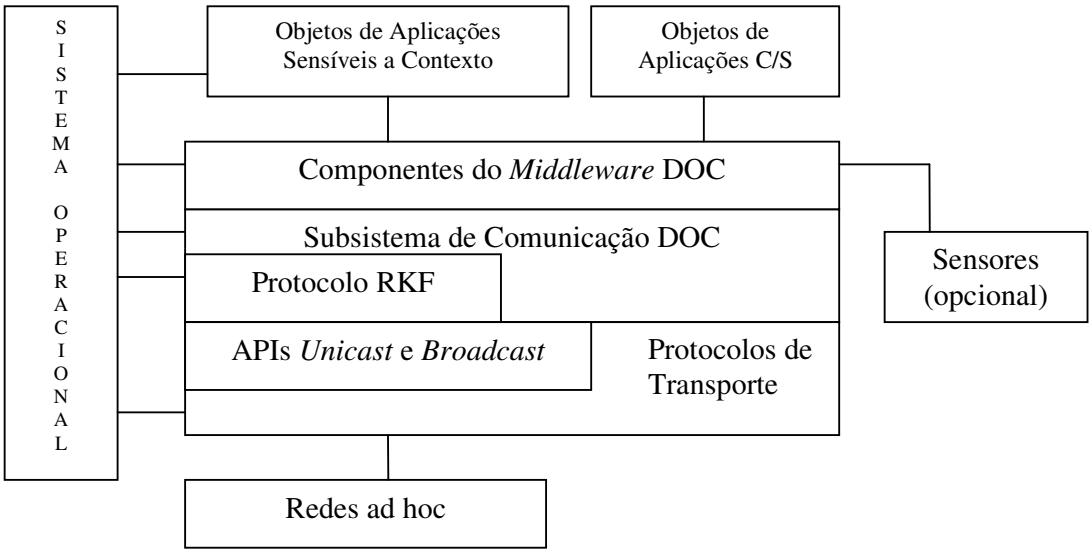


Figura 2.5: Arquitetura DOC com protocolo RKF [YAU03]

O protocolo RKF coopera com outros componentes, tais como ORB (*Object Request Broker*) e adaptadores de objetos, para facilitar tanto comunicações cliente-servidor quanto sensíveis a contexto.

Em uma comunicação cliente-servidor, o RKF cria um canal de comunicação cliente-servidor para um par de objetos remotos. Um objeto do cliente fornece a interface do objeto e o método de interesse, dados reais, e endereço do dispositivo destino. Baseado nessa informação, o RKF abre um canal de comunicação com o dispositivo destino. O RKF no dispositivo destino notifica então o adaptador do objeto responsável por ativar o objeto do servidor, que deverá ativar e invocar o método apropriado.

O objeto do servidor não precisa emitir uma resposta. Em vez disso, o servidor pode enviar algumas mensagens mais tarde usando RKF para abrir um canal de comunicação cliente-servidor separado.

O RKF cria um canal de comunicação sensível a contexto entre um par de objetos remotos. Antes de criar tal canal, o RKF deve primeiramente identificar os objetos locais que estão "prontos-a-serem-ativados" no contexto atual. Baseando-se nessa informação, deve-se descobrir os objetos compatíveis remotos que devem estar prontos-a-serem-ativados no contexto atual.

O RKF segue três fases antes de configurar um canal de comunicação sensível a contexto entre um par dos objetos. Uma propriedade do RKF é que não precisa confiar em nenhum protocolo de descoberta de vizinhos ou em nenhuma outra funcionalidade da rede (à exceção do *unicast* e *broadcast*) para realizar essas fases.

- Fase 1 – Broadcast de Informações Sensíveis a Contexto do Objeto.

O RKF descobre objetos novos escutando um tipo particular de sinais. Além disso, o RKF não inicia uma descoberta de objetos quando dispositivos novos se juntam à rede. Ao invés disso, o RKF somente transmite em *broadcast* a informação do objeto baseada na lista prontos-a-serem-ativados dos objetos.

Outros dispositivos, ao receber essas mensagens, iniciam um processo de Combinação de Pares para determinar a adequação de uma comunicação com o dispositivo de onde veio o *beacon*. O conteúdo dessas mensagens transmitidas muda quando a lista de prontos-a-serem-ativados é modificada. Por outro lado, se a lista pronto-a-ser-ativado se tornar vazia, então o RKF finaliza o processo de *broadcast* até que a lista não fique mais vazia.

- Fase 2 - Combinação de Pares.

Nessa fase, o RKF descobre os finais compatíveis de uma comunicação nos dispositivos remotos. Dado que cada canal de comunicação envolve um IM_PAIR(Par de

Interfaces e Método: o RKF vê uma interface de objeto e cada método deste objeto como um ponto final original de uma comunicação. Esses pontos finais são chamados de IM_PAIR). Dessa forma, um objeto pode ter muitos pontos finais de comunicação de entrada e saída. Quando um IM_PAIR que parte inicia a troca de dados do objeto, a combinação do par ocorre no lado do IM_PAIR que parte. Após uma combinação bem sucedida do par, o RKF notifica o adaptador do objeto para ativá-lo e invocar o método.

- Fase 3 – Troca de dados de objetos.

Nessa fase, o RFK transmite dados do objeto atual para outro, após as ativações necessárias serem executadas e os dados dos objetos coletados. O processo de ativação de um objeto esconde o destino real ou a fonte da informação.

Essa criação de canais de comunicação entre os pares é prejudicada pela característica *ad hoc* dos nós participantes. Segundo a própria avaliação de [YAU03], a execução em ambientes de computação ubíqua reduz significativamente o desempenho fim-a-fim da arquitetura RFK.

2.2.2.3 Arquitetura M2MI

O alvo dessa arquitetura são os sistemas colaborativos, onde múltiplos usuários com dispositivos computacionais entram e saem aleatoriamente e não são configurados para saber sobre outros dispositivos. Considerando particularmente redes *ad hoc* com proximidade física entre os dispositivos, a arquitetura baseia-se nas seguintes premissas [KAMI03]:

- Não existe endereço de dispositivos, permitindo que eles entrem e saiam do ambiente sem realizar configurações de rede;
- Mensagens são enviadas em *broadcast* para todos os dispositivos: como as transmissões são realizadas em áreas fisicamente próximas, todos os nós recebem os dados transmitidos;
- A relevância da mensagem é determinada pelo seu conteúdo, examinando o pacote recebido pelos bytes iniciais da mensagem; e
- O envio de mensagens é confiável: supondo que o *broadcast* atinge todos os nós, a perda de mensagens é rara.

Considerando os protocolos de roteamento *ad hoc* "complicados e utilizando espaço de memória substancial" [KAMI03], o uso de *broadcast* foi utilizado por ser simples, facilitando para os dispositivos o processo de envio de mensagens uns para os outros, diretamente ou utilizando um ponto de acesso.

2.2.2.4 iMAC

Em [CHEN02] foi proposto um serviço de acessibilidade a dados distribuídos para acessar dados multimídia dentro de um grupo cooperativo. Um grupo cooperativo é um conjunto de nós *ad hoc* que compartilham informações uns com os outros. Foi projetado nesse sistema um serviço de procura e oferecimento de dados, integrados com um serviço de replicação de dados e auxiliado por um protocolo de roteamento preditivo baseado em QoS. O design *cross-layer* fornece um *framework* integrado para a troca de informações entre os serviços de replicação, procura, propaganda e protocolo de roteamento.

A arquitetura é mostrada na Figura 2.6 e inclui:

- uma camada de aplicações que produz e compartilha dados multimídia com outros usuários da rede;
- um *middleware* que executa os serviços de acessibilidade a dados e apóia as aplicações na localização, acesso e replicação de dados; e
- uma camada de roteamento que computa rotas e repassa pacotes para outros nós da rede.

As camadas de roteamento e *middleware* compartilham informações para alcançar maior qualidade no acesso a dados.

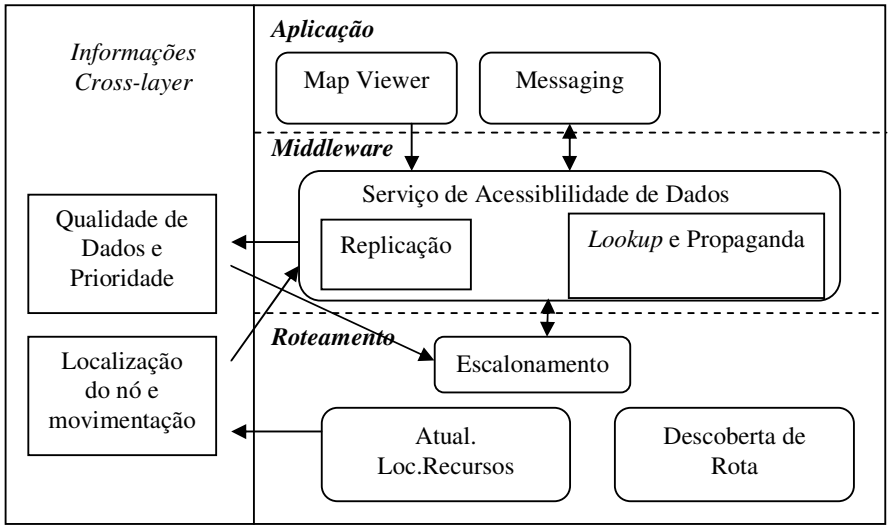


Figura 2.6: Arquitetura iMAC [CHEN02]

A camada de roteamento implementa um protocolo preditivo baseado em localização. O protocolo usa informações de localização geográfica de cada nó e padrões de movimentação com fins de descoberta e manutenção de rotas. Como parte do protocolo, um módulo de atualização de recursos-localização envia mensagens, periodicamente, em *broadcast* contendo a localização do nó e informações de recursos para os outros nós da rede. Através dessas mensagens, cada nó sabe a localização e o padrão de movimentação de outros

nós, o que permite que a camada de roteamento compute e mantenha rotas ativas com um conjunto específico de parâmetros de QoS.

Na camada *middleware* foi implementado um serviço de acessibilidade a dados que ajuda as aplicações a oferecer e compartilhar dados com outros usuários da rede. A tarefa de acessibilidade pode ser executada em dois passos:

- Passo 1: as informações de disponibilidade de dados são obtidas e apresentadas ao nível de aplicação. Clientes podem olhar as descrições dos dados disponíveis e acessar os dados até quando o fornecedor dos dados não estiver alcançável. O parâmetro de QoS, nesse passo, é a taxa de sucesso em conseguir alcançar os dados.
- Passo 2: a camada *middleware* recolhe dados de um *host* remoto com certos requisitos do nível de aplicação, como prazo de acesso e qualidade dos dados. Com essa finalidade, a camada *middleware* tem que traduzir os requisitos do nível de aplicação a um conjunto de parâmetros de QoS do nível de rede, como largura de banda e atraso, e configurar uma rota com esses parâmetros.

Para redes de baixa dinâmica, o protocolo é capaz de ajustar a rota atual ou recalcular uma nova rota. Para violações de QoS constantes, o *middleware* será notificado para aplicar adaptações funcionais, como trocar para um formato de dados mais comprimido.

2.2.3 Comparação entre as Arquiteturas

Para o problema do trabalho de campo, as características de maior interesse, como vistas na Seção 1.1, seriam o uso do contexto de localização física, a disseminação de dados pela área de trabalho e a adaptação do conteúdo para envio aos diversos tipos de dispositivos. O contexto utilizado direciona a atuação do uso das técnicas de adaptação de conteúdo. As arquiteturas vistas na Seção anterior podem ser comparadas levando em consideração esses itens.

A Tabela 2-1 resume algumas das características de cada arquitetura.

Tabela 2-1: Comparação entre as Arquiteturas

Item	NFK	RFK	M2MI	iMac
Atualização via *Broadcast*	Não	Não	Sim	Sim
Contextos considerados	Comunicação	Comunicação	Não	Dispositivo Comunicação Loc. Física QoS
Adaptação de Conteúdo	Não	Não	Não	Não

Primeiramente, o parâmetro Atualização via *Broadcast*, utilizado pelas arquiteturas M2Mi e iMAC, é fator negativo para o uso em ambientes de trabalho de campo. Os trabalhadores envolvidos em uma parte do trabalho receberão dados de participantes de outras áreas de trabalhos, gerando custos de comunicação e consumo de CPU e bateria não necessários. É necessário separar áreas por contexto para a melhor utilização dos recursos de comunicação. O *broadcast* dentro de uma área geográfica é um passo possível para a arquitetura iMAC, já que utiliza os dados de localização física como parte da arquitetura.

O uso do *broadcast* também limita a escalabilidade da arquitetura. O *broadcast* é adequado para sistemas pequenos enquanto apresenta limitações de desempenho no caso de sistemas e espaços grandes. Uma solução seria a adoção de *broadcast* com tempo de vida de pacotes (campo TTL) limitado para garantir que o escopo do *broadcast* seja limitado a uma área geográfica pequena de vizinhos.

No iMAC, todos os nós do ambiente sabem da posição física de todos os outros nós presentes. Essa replicação de todos os dados de localização também gera custos em termos de recursos de rede e de armazenamento nos dispositivos. A limitação do uso dessas informações apenas entre os vizinhos distantes a um salto reduz a carga de dados que trafega no ambiente e permite o roteamento de dados baseado em localização física. O uso de informações de localização física pelos protocolos de roteamento permite que rotas sejam geradas sabendo-se apenas as informações dos nós mais próximos. Detalhes sobre roteamento baseado em localização física dos nós serão vistos na Seção 3.4.2.

As arquiteturas NFK e RKF só propõem a negociação das conexões entre os nós. Mais precisamente, a RKF cria conexões permanentes entre fonte e destino, porém, em ambientes instáveis como as redes *ad hoc*, estas conexões são prejudicadas pela mobilidade e problemas de desconexões dos nós, gerando muitos erros nas conexões criadas.

O contexto dos dispositivos presentes no ambiente também deve ser considerada, pois um abiente de computação pervasiva possui dispositivos heterogêneos. Como no trabalho de campo os dispositivos de menor capacidade computacional utilizados como coletores de dados são encontrados em grande número, deixar de considerá-los ao gerar rotas e adaptar conteúdo pode impossibilitar a utilização da arquitetura neste ambiente. Nenhuma das arquiteturas vistas nesta Seção propõe métodos para melhorar a utilização dos recursos dos trabalhadores por adaptação de conteúdo.

Outro contexto que deve ser considerado é a heterogeneidade dos trabalhadores que serão clientes das aplicações fornecidas. Para a utilização de aplicações com contexto de localização física, como as usadas no trabalho de campo, a atualização de informações de

localização física de cada trabalhador é crucial para o melhor aproveitamento da aplicação e dos dados referentes ao trabalho de campo. Estas atualizações dependem da mobilidade dos trabalhadores, pois um trabalhador com pequena movimentação pela área física não necessita atualizar tantas vezes como outro trabalhador com alta movimentação.

2.3 Adaptação de Conteúdo

A presença de diversos tipos de dispositivos no trabalho de campo torna necessário o uso das técnicas de adaptação de conteúdo. O envio de dados com o mesmo formato para qualquer tipo de dispositivo pode gerar custos desnecessários de consumo de rede e dos recursos dos dispositivos [JOHN04].

Outro aspecto do trabalho de campo que a adaptação de conteúdo pode resolver é a utilização de contextos. Diversos desafios dos ambientes de computação pervasiva são referentes a contexto [HENR02]. Contexto pode ser definido como qualquer informação que possa ser utilizada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, lugar ou objeto que é considerada relevante para a interação entre um usuário e uma aplicação, incluindo os próprios usuários e as aplicações [DEY99].

Num ambiente que oferece serviços, vários tipos de dispositivos estarão presentes fisicamente. Com a identificação do contexto do dispositivo em relação a seus recursos computacionais, é possível adaptar o tráfego e a qualidade de apresentação de uma aplicação para sua melhor visualização e uso. O formato de envio de dados deverá ser adaptado de acordo com as possibilidades de cada dispositivo. Por exemplo, em um ambiente que esteja sendo utilizado por computadores portáteis, *desktops* e computadores de mão, a QoS oferecida para cada tipo de dispositivo deverá ser diferente.

A Figura 2.7 mostra uma aplicação servindo a vários dispositivos, sendo que cada categoria de dispositivos (em relação a sua capacidade computacional) recebe parâmetros de QoS diferentes.

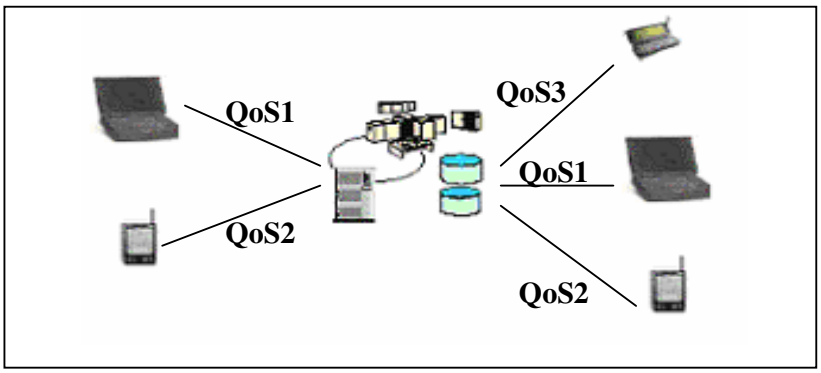


Figura 2.7: Aplicação servindo vários dispositivos heterogêneos

Os seguintes parâmetros de QoS, no nível de Aplicação, podem ser adaptados: taxa de *frames*, tamanho do *frame* e resolução, tempo de resposta, vazão, segurança e preço [BHAR02]. Esses parâmetros serão discutidos com mais detalhes na Seção 5.7.1.

Técnicas de adaptação incluem o uso de *proxies*. Estes geram diferentes fluxos já adaptados de acordo com as capacidades dos dispositivos de recepção. As técnicas incluem também a implementação de algoritmos de adaptação nos próprios dispositivos.

A primeira estratégia tem a vantagem de aliviar o processamento nos terminais e dispositivos e a segunda estratégia alivia a rede, diminuindo a carga de dados que trafega. Para maiores informações ver a Seção 2.3.2 sobre estratégias de adaptação.

2.3.1 Contexto

Os tipos de contexto podem ser divididos em [CHEN00]:

1. Computacional

As tecnologias de redes sem fio limitam o uso dos dispositivos por problemas relacionados à largura de banda limitada, alta latência e desconexões freqüentes. Os próprios dispositivos móveis possuem recursos limitados. CPU, memória, tamanho da tela, resolução, limitações de força, compressão e criptografia são alguns dos recursos que podem variar muito de um dispositivo para outro.

A mobilidade dos dispositivos impõe diversas restrições:

- espaços físicos: conexões ruins e desconexões freqüentes (planejadas ou não).
- espaços de conexões: grande número de enlace*s*; seleção de rotas entre enlace*s*.
- espaços de informações: grande número de aplicações e formato de dados selecionados para adaptar o ambiente dos usuários.

A Figura 2.8 resume esses principais desafios.

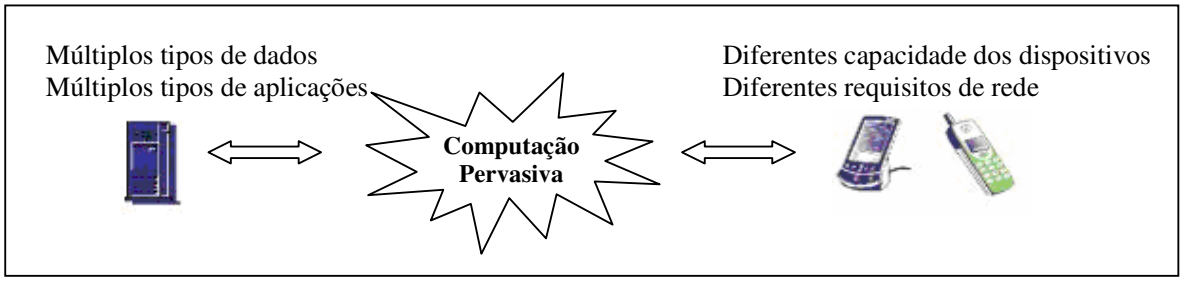


Figura 2.8: Desafios de Contexto

Exemplos de contexto computacional são: conectividade de rede, custos de comunicação, largura de banda e recursos próximos, como impressoras, monitores e estações de trabalho.

2. Usuário: perfil de usuário (suas preferências, tipo de assinatura, etc.), localização, mobilidade, pessoas próximas, até a situação social atual.
3. Físico: local onde se encontra no momento (sala, quarto, avenida) e suas características (luzes, nível de ruído, condições de tráfego).
4. Tempo: hora do dia, dia da semana, mês, e estação do ano.

Como o contexto é uma variável que influencia na qualidade percebida de um serviço oferecido, existe a necessidade de espaços de adaptação que habilitam as aplicações a funcionar em ambientes pervasivos em diferentes níveis de ambientes de operação, isto é, com diferentes configurações, através da adaptação de conteúdo.

2.3.2 Estratégias de Adaptação

Como se espera que nos ambientes de computação pervasiva as aplicações possam descobrir e tirar vantagem de informações contextuais, de acordo com o tipo de contexto ao qual a aplicação é sensível, devem-se adaptar os parâmetros da aplicação que está sendo servida.

Podem ser utilizadas informações contextuais de duas formas: as aplicações se adaptam automaticamente de acordo com o contexto descoberto (contexto ativo), ou apresentar o contexto ao usuário e/ou armazenar para uso posterior (contexto passivo). A computação consciente de contexto ativo é mais interessante porque leva à novas aplicações para dispositivos móveis e requer mais suporte de infra-estrutura. Isso pode eliminar cooperações desnecessárias com usuário e permitir que a tecnologia trabalhe de forma mais eficiente quanto possível [CHEN00].

Existem várias estratégias e técnicas de adaptação de conteúdo. As estratégias se referem à como realizar a adaptação e o local onde realizar adaptação propriamente dita.

2.3.2.1 Tipos de Estratégias de Adaptação

Como visto anteriormente, a adaptação de conteúdo é necessária quando existe uma incompatibilidade entre o fornecedor e a demanda por um recurso. [SATY01] define recurso como largura de banda de rede, energia, ciclos de computação e memória. São listadas três tipos de adaptação:

1. O cliente guia as aplicações para modificarem seu comportamento, com a intenção de que elas usem menos recursos escassos. A mudança causa redução na qualidade percebida pelo usuário;

2. O cliente pede ao ambiente (o *software* que gerencia a área de trabalho) que garanta um certo nível de recursos. Isso aumenta efetivamente o fornecimento de um recuso escasso para se compatibilizar com a demanda do usuário; e
3. O ambiente sugere uma ação corretiva ao usuário. Se o usuário aceitar a sugestão, é possível que o fornecimento se torne mais adequado à demanda.

2.3.2.2 Técnicas de Adaptação de Conteúdo

Existem diversos tipos de técnicas de adaptação de conteúdo. Como cada autor tem ma visão diferente sobre como tratar deste problema, serão mostradas técnicas propostas por três autores.

Em [LEI01] as técnicas de adaptação de conteúdo são classificadas em quatro grupos:

1. De acordo com o contexto alvo
    a. Adaptação à infra-estrutura técnica

Considerando como infra-estrutura as conexões de rede e a capacidade dos dispositivos, a tecnologia de adaptação deve poder fornecer uma apresentação apropriada a dispositivos diferentes.

    b. Adaptação à preferência do usuário

O usuário pode ter diferentes requisitos em termos de detalhes do conteúdo ou outros parâmetros, previamente conhecidos pelo ambiente.

Portanto, o sistema gerenciador do ambiente deve adaptar o conteúdo de acordo com a infra-estrutura física (rede e dispositivos) ou em relação ao usuário final, ou uma combinação dos dois métodos, o que no caso de ambientes de trabalho de campo é o mais indicado. Além de ser necessário se adaptar aos aspectos técnicos, os usuários podem ter vários níveis de preferências e hierarquia, com acesso a diferentes níveis de conteúdo transmitido.

2. De acordo com o momento da criação das mídias
    a. Adaptação Estática

Ocorre quando as informações são pré-processadas e armazenadas em múltiplas versões, com diferentes requisitos de qualidade e processamento. Quando a informação é solicitada, o sistema seleciona a versão apropriada dependendo do contexto do usuário.

    b. Adaptação Dinâmica

A informação é processada e entregue em tempo de execução. As alternativas disponíveis são determinadas para o contexto específico no tempo da apresentação. Por exemplo, quando a largura de banda da rede não é suficiente, pode-se eliminar dados menos importantes e enviar um resumo da apresentação multimídia. A diversidade dos dispositivos e

das conexões de rede torna difícil e caro criar diversos conteúdos multimídia para cada tipo de dispositivo no momento de adaptação. Além disso, a natureza das conexões de rede não é conhecida no momento da transformação. Conseqüentemente, tecnologias para adaptar dinamicamente informação multimídia para os dispositivos clientes e as diversas conexões de vários tipos de dispositivos tornou-se algo crítico.

3. De acordo com os tipos de mídia
    a. Mídia simples

Várias técnicas atuais da adaptação de melhor-esforço são consideradas como chaveamento entre qualidades ou formatos diferentes de elementos de mídia. Por exemplo, a maioria das imagens pode ser comprimida, significativamente, sem uma diminuição notável na qualidade. GIF-para-JPEG ou transformações de cor para preto e branco podem reduzir o tamanho físico de uma imagem.

b. Mídia Cruzada (*Cross Media*)

Essa adaptação permite transformar um tipo de mídia em outro que possa ser processado por um dispositivo particular. Por exemplo, a maioria de computadores de mão não é capaz de manipular os dados de vídeo devido à suas limitações de *hardware* e de *software*. Transformar vídeo em conjunto de imagens e retirar áudio ou legendas permitirá aos dispositivos acessar as informações contidas no vídeo. Assim, os usuários poderão receber informações úteis, em qualquer formato, que seus dispositivos podem manipular.

Em [ABDE99], as técnicas de adaptação são classificadas de acordo com suas aplicações:

1. Abstração de informação: seu objetivo é reduzir os requisitos de largura de banda para envio do conteúdo, através da compressão de dados, enquanto preserva a informação que tem o maior valor para o usuário. Por exemplo, sumário de informações e geração de imagens reduzidas.
2. Priorização de dados: deve distinguir a parte mais importante da menos importante, para que os níveis de qualidade de serviços possam ser fornecidos quando enviar dados através da rede. Pode ser alcançada através de um único tipo de dado através de esquemas de codificação, como codificação em camadas e compressão em multi-resolução.
3. Transformação de modalidade: é o processo de transformar conteúdo de um modo para outro, para que o conteúdo se torne útil para um dispositivo cliente específico. Por exemplo, transformar um vídeo em imagens, que possam ser vistas em dispositivos de baixa capacidade computacional.

4. Transcodificação de Dados: é o processo de transformar o conteúdo de páginas *web* para permitir acesso universal utilizando dispositivos ubíquos. Pode ser usado para adaptar vídeo, imagens, áudio e texto, estatisticamente ou dinamicamente, para se adaptar às preferências dos usuários ou especificações dos dispositivos. Por exemplo, conversão de vídeo (MPEG para MOV) e áudio (WAV para MP3).
5. Classificação de Propósitos: classifica o conteúdo de cada objeto de mídia em uma página *web*, para aumentar a eficiência do envio de informações, ao remover objetos redundantes ou priorizando-os de acordo com sua importância.

Outras técnicas, descritas em [THAM03], mostram como modificar vídeos e imagens para melhorar o desempenho de uma transmissão.

1. Transmissão progressiva: usada em imagens ou vídeo, esse tipo de transmissão é útil em muitas aplicações práticas tais como visualização de imagens (*image browsing*). Em tais situações, um usuário vê primeiramente uma versão grosseira de uma imagem reconstruída dos bits recebidos anteriormente, e enquanto mais partes de dados são recebidas, a qualidade da imagem é refinada até o final do fluxo de dados. A transmissão progressiva permite a recuperação rápida de uma imagem e dá ao usuário a opção de terminar a transmissão a qualquer momento.
2. Codificação em camadas ou hierárquica: é uma técnica desenvolvida para entrega de vídeo de forma escalável, em que o sinal é separado em componentes da importância visual diferentes. Originalmente, a idéia é que cada componente ou camada de vídeo possa ser codificado e transferido na rede com uma qualidade que combine sua importância, isto é, a qualidade melhora para cada componente que seja de nível superior que o anterior. As camadas vitais podem assim ser transferidas em uma classe com qualidade garantida, enquanto que uma camada que realça a qualidade poderia ser enviada como melhor-esforço [THAM03]. Como exemplo dessa técnica, o padrão MPEG definiu 4 classes de ferramentas de escalabilidade: particionamento de dados e escalabilidade SNR, espacial e temporal [ISO04]:
    a. Particionamento de dados: Ajuda a mascarar erros quando ocorrem erros de transmissão. Por exemplo, o fluxo de dados é transmitido em duas camadas com diferentes probabilidades de erro: uma camada contém os dados mais importantes, como cabeçalho e vetores de movimento; e a outra camada, dados de menor importância.
    b. Escalabilidade SNR: gera camadas de vídeo de mesma resolução espacial mas com qualidades diferentes de uma mesma fonte de vídeo. A camada mais baixa é

codificada para fornecer uma qualidade básica e a camada de realçamento é codificada para melhorar a camada inferior. A camada de realçamento, quando adicionada à camada inferior, gera uma reprodução de maior qualidade do vídeo original.

c. Escalabilidade Espacial: também gera camadas de resoluções espaciais diferentes de uma uma fonte de vídeo. A camada mais baixa fornece uma resolução espacial básica, enquanto que a camada de realçamento codifica o vídeo interpolado da camada base. Permite a mostrar o vídeo com diferentes resoluções espaciais no receptor.

d. Escalabilidade Temporal: gera camadas de vídeo onde a mais baixa é codificada por si mesma para fornecer uma taxa temporal básica e a camada de realçamento é codificada com previsão temporal em relação à camada inferior. Essas camadas, quando decodificadas e multiplexadas, geram um vídeo com a resolução temporal total do vídeo original.

O conceito de codificação escalável de vídeo evita a transcodificação, que requer recursos adicionais do sistema, causando atrasos. Permite decodificação de fluxos de dados à níveis de qualidade, resoluções espaciais e taxas de *frames* menores do que os parâmetros da seqüência de vídeo originais [MMV04].

2.3.2.3 Local de Adaptação

É o aspecto de arquitetura que se refere ao local onde a adaptação ocorre. Pode ser classificado em [LEI01]:

- Baseada no Servidor

Nesse tipo de adaptação, o servidor de conteúdo deve analisar o perfil do contexto e descobrir quanta largura de banda está disponível entre o cliente e o provedor do conteúdo. Com os dados, seleciona a estratégia de adaptação apropriada como na seguinte sequência.

1. CLIENTE solicita conteúdo de um SERVIDOR;
2. CLIENTE envia dados de PERFIL ao SERVIDOR;
3. SERVIDOR solicita informações de Largura de Banda da ROTA;
4. SERVIDOR seleciona estratégia;
5. SERVIDOR adapta e envia dados p/ CLIENTE.

Considerando o ambiente de trabalho de campo descrito neste trabalho, onde os provedores de serviços e conteúdos são os próprios Clientes, e que na geração de rotas o perfil dos dispositivos são descobertos, os Servidores possuem todas as informações necessárias para gerar o conteúdo adaptado de acordo com o Cliente alvo.

- Baseada em *Proxies*

Nesse tipo de arquitetura, o cliente se conecta através de um *proxy*, que decide e desempenha a adaptação pelo servidor de resposta, e então envia o conteúdo transformado de volta para o cliente. Essa adaptação permite que não haja nenhuma modificação no cliente ou no servidor, e também reduz a carga computacional no servidor de origem. Seqüência de funcionamento:

1. CLIENTE solicita conteúdo para um PROXY;
2. PROXY solicita dados para SERVIDOR;
3. PROXY solicita informações de PERFIL do CLIENTE;
4. PROXY recebe dados do SERVIDOR e adapta;
5. PROXY envia dados para CLIENTE.

O *proxy* pode ser um serviço do *middleware* e não é necessário que o cliente seja ciente da traduação no *proxy* (isto é geralmente transparente ao cliente e este não precisa solicitar).

Para o ambiente de trabalho de campo, este tipo de adaptação não é adequado devido à necessidade de infra-estrutura extra no campo.

- Baseada no Cliente

Dependendo principalmente da capacidade dos dispositivos, nesse tipo de adaptação a escolha da melhor representação é desempenhada pelo dispositivo após receber uma resposta inicial do servidor de origem. A seleção é realizada em uma lista de representações disponíveis enviadas junto com a resposta. Essa adaptação é vantajosa quando o servidor de origem não consegue determinar a capacidade do dispositivo ao examinar uma requisição, e quando *cache*s públicos são usados para distribuir a carga do servidor e reduzir a utilização da rede. A seguinte seqüência mostra um exemplo de uma adaptação baseada no cliente:

1. CLIENTE solicita conteúdo de um SERVIDOR;
2. CLIENTE recebe lista de tipos de conteúdo disponíveis;
3. CLIENTE avisa SERVIDOR que tipo de conteúdo deseja;
4. SERVIDOR adapta e envia dados p/ CLIENTE.

No caso deste trabalho, como visto anteriormente no caso da adaptação baseada em Servidor, o Servidor já sabe o contexto do Cliente. Então, este modelo não é necessário.

- Baseada em *Middleware*

Com partes de *software* executados nos clientes e nos servidores, esta abordagem permite que os servidores estejam em comunicação com os clientes invisivelmente, escondidos em camadas inferiores às aplicações. O *middleware*, que pode ser configurado por

um ou vários servidores, é responsável pela indicação ou adaptação de conteúdo de acordo com o dispositivo que o usuário está utilizando. Seqüência de funcionamento:

1. CLIENTE solicita conteúdo p/ *MIDDLEWARE*;
2. *MIDDLEWARE* pesquisa qual SERVIDOR/PROXY mais adequado e solicita conteúdo;
3. *MIDDLEWARE* envia dados de perfil do CLIENTE e dados de QoS da rota p/ SERVIDOR;
4. SERVIDOR envia conteúdo p/ CLIENTE.

Neste trabalho, como a camada de serviços realiza funções de busca de perfis, então este é um método adequado para a adaptação de conteúdo. Portanto, o método de adaptação de conteúdo desta arquitetura é um híbrido entre o método Baseado em Servidor e Baseado em *Middleware*.

2.3.3 Trabalhos Relacionados

Várias arquiteturas e *middlewares* foram projetadas para focalizar no problema específico de adaptação de conteúdo. Esta Seção mostra algumas dessas arquiteturas, detalhando o processo de adaptação de conteúdo utilizado por cada uma.

2.3.3.1 Context-Aware Decision Engine

Para resolver o problema de adaptação de conteúdo, causado principalmente pelas diferenças entre contextos multimídia variados e capacidade limitada dos dispositivos, Lum [LUM02] propõe um sistema de adaptação que decide a versão otimizada de conteúdo para apresentação e a melhor estratégia de derivar e gerar essa versão. O componente principal desse sistema é o *Decision Engine* (DE) que negocia as estratégias. O contexto computacional utilizado para a adaptação são as preferências dos usuários.

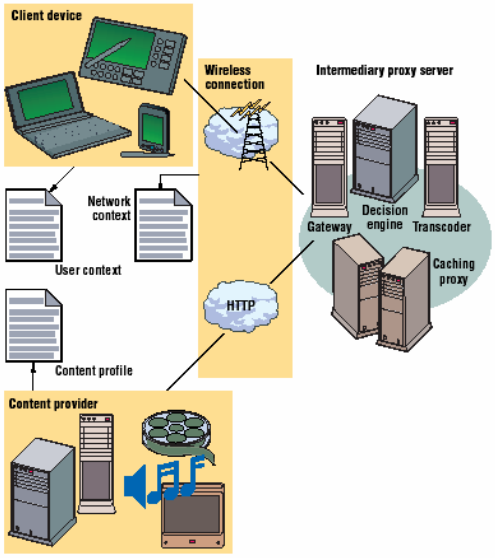


Figura 2.9: Arquitetura com *Decision Engine* [LUM02]

A Figura 2.9 mostra a arquitetura utilizada. Quando um usuário envia uma requisição ao sistema, isso ocorre através de um *proxy* intermediário para o provedor de conteúdo. Quando o sistema identifica o usuário, manda informações de contexto ao DE que reside no *proxy*. Através de um sistema de pontuação aplicado às diferentes versões de conteúdo, o DE executa um algoritmo que calcula a versão mais apropriada e compatível com o dispositivo cliente atual e com as características da rede. O resultado é enviado a um transcodificador, que gera a versão de conteúdo desejada, e o *proxy* envia o conteúdo adaptado ao dispositivo do usuário.

O DE negocia automaticamente as decisões de adaptação de conteúdo apropriadas que o transcodificador utilizará ao gerar a versão otimizada de conteúdo. Também tenta chegar à melhor relação para adaptação do conteúdo enquanto minimiza a degradação deste, devido a transcodificação com perdas. O DE se baseia na indicação das preferências do usuário, de acordo com sua percepção em diferentes domínios de qualidade e pode formular um método para expressar quantitativamente qualquer qualidade de conteúdo ao longo de vários eixos de qualidade e algoritmos para negociar por uma versão otimizada do conteúdo.

Uma desvantagem desta arquitetura para uso nos ambientes de trabalho de campo é a necessidade da utilização de dispositivos intermediários (*proxies, transcodificadores e gateways*). Não é desejável levar a campo equipamentos extras aos já necessários, devido às condições precárias de estrutura do ambiente.

Além disso, apesar de poder considerar o uso dos *proxies* nas Bases locais, até como um serviço adicional do dispositivo servidor da Base, o envio do conteúdo adaptado

indiretamente aos dispositivos fins através dos *proxies*, considerando a natureza *ad hoc* da comunicação e imprevisibilidade de localização dos usuários pelo campo, pode afetar o desempenho das aplicações, gerando grandes atrasos nas comunicações que necessitam de caminhos indiretos para serem entregues.

Caso os *proxies* pertencessem ao local de trabalho, por exemplo, fixos em certas áreas, eles poderiam ser usados também como centros de armazenamento intermediários, quando o usuário/terminal não está conectado. Isso ajudaria na entrega das informações para os dispositivos que saíssem do alcance de seus vizinhos de comunicação, permitindo-os receber informações importantes ao seu trabalho.

2.3.3.2 Arquitetura $2k^Q$

A arquitetura $2k^Q$ [XU00] (Figura 2.10) inclui: (1) componentes de serviços, (2) *QoSProxies* e (3) gerentes/monitores de recursos *(RMs)*. Os componentes de serviços são basicamente blocos de construção para a configuração dinâmica dos serviços multimídia. Um *QoSProxy* é executado em cada *host*, e tem o papel principal na configuração dos serviços. Um ou mais RMs são executados em cada *host* que está gerenciando ou monitorando a disponibilidade de diferentes recursos do sistema.

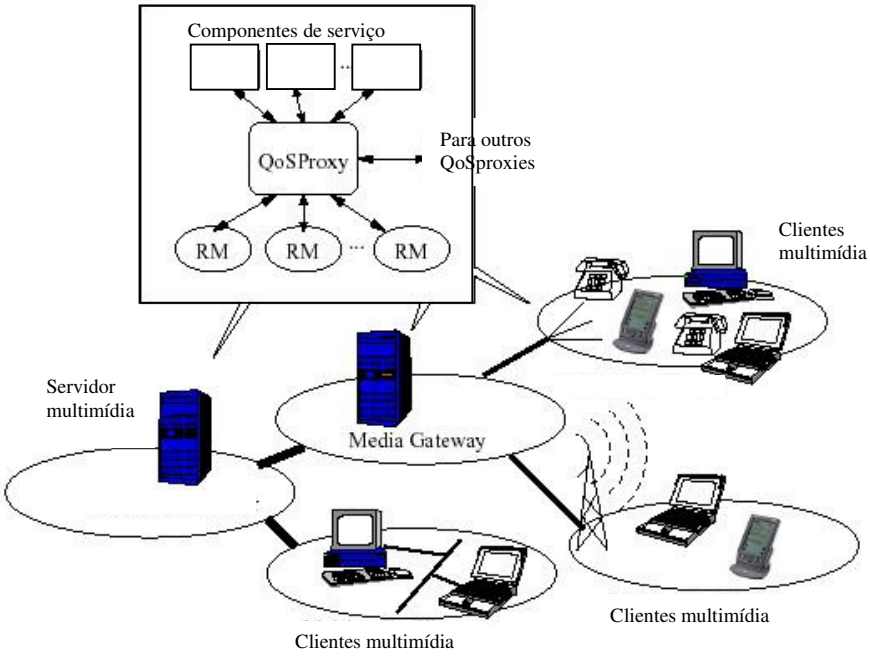


Figura 2.10: Visão geral da arquitetura $2K^Q$ [XU00]

Os principais serviços que compõe a arquitetura são:

- **Configuração Dinâmica de Serviço**

O serviço multimídia é composto por componentes de serviços, que são as unidades funcionais. Múltiplos componentes trabalham juntos para a entrega de um serviço. Para serviços multimídia ubíquos, onde há uma grande variedade de capacidade nos clientes, a configuração dinâmica do serviço é a seleção dos componentes mais adequados no servidor, cliente e entidades intermediárias, com o objetivo de alcançar qualidade fim-a-fim na entrega dos serviços.

Os *QoSProxies* cooperam para selecionar e iniciar um conjunto apropriado de *componentes de serviços* para cada requisição de serviço, de acordo com a disponibilidade reportada pelos RMs.

Nos servidores, clientes e *gateways*, cada *QoSProxy* mantém uma tabela de componentes de serviço com informações sobre os componentes registrados localmente. O provedor de serviços multimídia fornece informações, no lado do servidor, sobre as possíveis configurações de serviços.

Os *QoSProxies* também executam o protocolo REACT (*REsource Aware Configuration Protocol*), que determina a configuração adequada para cada cliente requisitante.

- **Gerenciamento de Recursos**

Os Gerentes/monitores de Recursos (RMs) tem as responsabilidades de monitorar a disponibilidade, reservar e escalonar os recursos do sistema. Presentes nos servidores, clientes e *gateways*, reportam a disponibilidade de recursos locais no lado do cliente (exemplo: capacidade de CPU e vida útil da bateria), recursos locais no lado do servidor (exemplo: similares aos dos clientes) e largura de banda da rede entre servidor e cliente, o que possibilita estimar gargalos na rede e ajudar no processo de configuração do serviço.

Os *QoSproxies* executam o protocolo REACT para determinar a configuração de serviço apropriada para cada cliente requisitante, isto é, os QoS*proxies* cooperam para selecionar e iniciar um conjunto apropriado de componentes de serviço para cada requisição de serviço, de acordo com a disponibilidade atual de recursos registrado nos Gerentes de Recursos (RMs).

O protocolo REACT funciona de acordo com a seguinte sequência:

1. Quando um cliente faz um pedido de serviço, a requisição é interceptada pelo *QoSproxy* lado-cliente;
2. O *QoSproxy* lado-cliente coleta dados sobre a disponibilidade de recursos locais e largura de banda fim-a-fim no RM local;

3. O *QoSproxy* lado-cliente então contacta o *QoSproxy* lado-servidor com a requisição s e as informações de recursos disponíveis;
4. O *QoSproxy* lado-servidor também coleta as informações atuais de recursos locais disponíveis do RM local;
5. Cria um vetor de disponibilidade R_{avail}, que será utilizado no calculo da configuração mais apropriada do serviço para cada *QoSproxy* envolvido; e
6. Finalmente, os *QoSproxies* envolvidos carregarão e iniciarão seus respectivos componentes de serviços.

Como visto na seqüência de passos, tal qual a arquitetura anterior (*Decision Engine*), esta arquitetura também necessita da utilização de equipamentos extras aos já existentes no trabalho de campo. Além disto, a criação de um vetor de disponibilidade em ambientes de redes *ad hoc* deve gerar inconsistências com muita rapidez, devido à natureza imprevisível de mobilidade dos usuários no ambiente.

2.3.3.3 *Middleware* iMash

A arquitetura iMASH (*Interactive Mobile Applications Support for Heterogeneous Clients*) [PHAN01]é produto da investigação da importância de *middleware* para aplicações móveis na Internet pelo Departamento de Computação da UCLA.

O principal problema identificado por este projeto é que a utilização de *middleware,* que realizam funções de adaptação de contexto e QoS se tornará mais crucial devido à proliferação de dispositivos móveis sem fio. O objetivo desse trabalho é mostrar uma arquitetura de *hardware* e *software* utilizando um nível *middleware* que dê suporte a clientes fixos e móveis heterogêneos, os quais interagem com um servidor de aplicações que fornece um repositório de dados.

A Figura 2.11 mostra o Servidor *Middleware* (MWS) posicionado entre os clientes e os Servidores de Aplicação (AS), que são repositórios de dados para os clientes. Os dados transmitidos pelos ASs são de natureza discreta, sem orientação à conexão.

Diferentemente das abordagens onde ASs devem prover serviços para um grande número de clientes simultâneos, o objetivo desta camada é permitir que o MWS trabalhe como o único cliente para um AS, permitindo, portanto, que o AS cumpra seu papel de provedor de dados, ao invés de gerente para clientes heterogêneos.

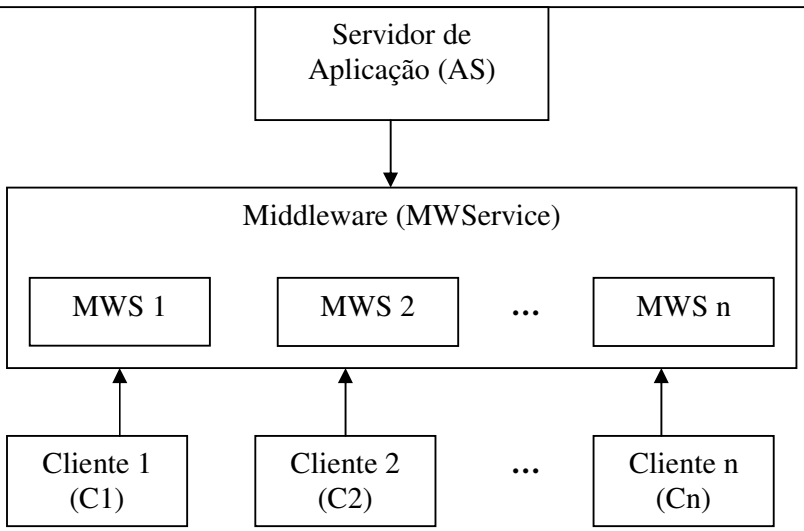


Figura 2.11: Visão geral da Arquitetura iMash [PHAN01]

O *middleware* iMASH foi projetado para oferecer:

- **Autenticação e Levantamento de Perfis de Usuários**

Os MWSs registram os clientes locais e seus perfis (exemplo: tipos de conexão de redes) para uso destas informações pelo serviço de conversão de apresentação.

- **Rastreamento de Localização**

Pode-se localizar à qual MWS um dispositivo está conectado, ou um MWS mais apropriado para servir um dado cliente.

- **Conversão de Apresentação**

A técnica de *Handoff* de Sessão de Aplicação (HSA) permite que usuários movam o estado atual de uma sessão de uma aplicação (discreta, sem orientação à conexão) executada em um dispositivo para outro com pouca ou nenhuma interrupção.

- **Adaptação de QoS na Rede**

Através das mesmas técnicas de *Handoff* de Sessão de Aplicação, utilizando também o *handoff* entre MWSs, uma aplicação terá o QoS mapeado para a nova rede onde se localiza o novo MWS.

Através do processo de *Handoff* de Sessão de Aplicação é possível transferir a aplicação que está sendo servida em um dispositivo para outro dispositivo com parâmetros diferentes. Quando um usuário deseja mudar de dispositivo, ele ativa o HSA para transferir todo o estado da sessão, como imagens renderizadas, texto ou preferências de usuários. Durante a transferência, esses dados podem ser filtrados para se adaptar à largura de banda disponível, ou limitações do novo dispositivo. O *runtime* do iMash executa todas as ações

necessárias para este *handoff*, liberando, portanto, o usuário de ter que salvar dados em um meio intermediário e reinstanciá-los depois no novo dispositivo [PHANT01].

A Figura 2.12 exemplifica o processo de *Handoff* de Sessão de Aplicação. Resumidamente, quando um cliente C_1 requisita um objeto *o* do AS (na Figura, passo 1), o AS retorna o objeto para o MWS (2), que o filtra para adaptá-lo ao dispositivo cliente, aplicando uma função $f_1(o)$ (3). O usuário em C1 modifica os dados com suas preferências, através de uma função g_1, resultando em g_1f_1 (o). Quando o usuário sai de C_1 para C_2, a modificação do usuário, g_1, e o objeto *o* são enviados ao próximo MWS, que aplica uma nova função de filtro f_2, adequada ao novo dispositivo.

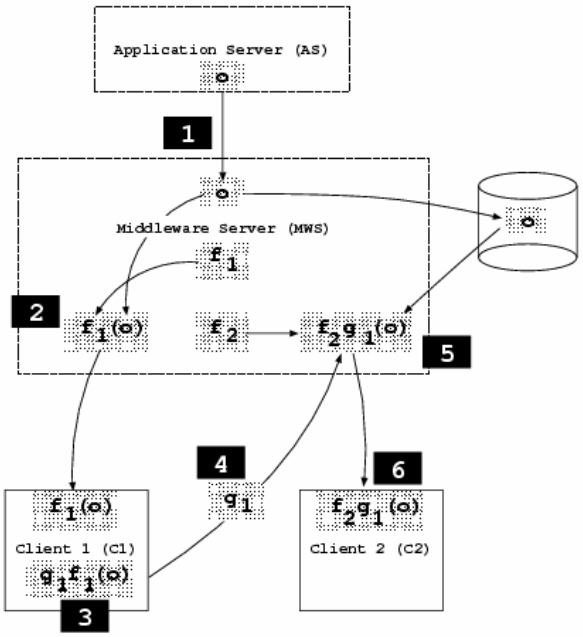


Figura 2.12: Handoff de Sessão de Aplicação [PHANT01]

Tal qual as outras arquiteturas já vistas, esta também depende de um novo equipamento participante, os MWSs. Além disto, o contexto dos usuários é restrito ao dispositivo utilizado, ignorando localização física, o que é muito interessante para o trabalho de campo.

2.3.3.4 Arquitetura Ninja

A arquitetura Ninja [GRIB00], desenvolvida na Universidade de Califórnia em Berkeley, busca habilitar serviços Internet robustos, escaláveis e distribuídos, e permitir que as classes emergentes de dispositivos heterogêneos acessem estes serviços.

Esse projeto identifica os principais problemas nos serviços de informação do futuro: a necessidade de adaptação à diversidade de formas nas quais os serviços podem ser oferecidos para serem iguais às capacidades particulares de diversos dispositivos e a utilização destes serviços por milhões de usuários. Isto torna necessário que serviços sejam escaláveis com o aumento da carga em períodos curtos de tempo e continuem disponíveis [GRIB00].

Os quatro elementos básicos da arquitetura, mostrados na Figura 2.13, são:

- *Units* (Unidades), os dispositivos heterogêneos usados;
- *Proxies* Ativos, os elementos de transformação usados para adaptação de dispositivo ou serviço específico;
- Bases, um *cluster* de estações de trabalho de alto desempenho, que o núcleo do serviço utiliza como ambiente de execução;
- *Paths* (Caminhos), uma abstração utilizada para compor outros elementos.

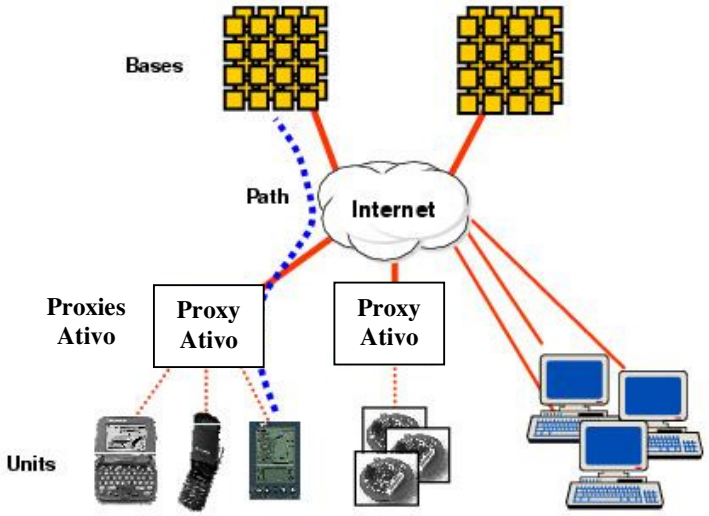


Figura 2.13: Visão Geral da Arquitetura Ninja [GRIB00]

Os principais serviços oferecidos pela arquitetura são:

- **Adaptação**

Ao invés dos serviços serem forçados a adaptar seu conteúdo e protocolos de acesso para permitir seu uso pelos diferentes tipos de dispositivos atuais e futuros, a arquitetura propõe o uso de intermediários de transformação. Os *proxies* ativos são posicionados entre clientes e serviços, isolando-os, e adaptam dados e protocolos de acesso às necessidades de dispositivos e serviços.

Os *proxies* ativos podem executar operações de otimização consciente de contexto e transformações de acordo com os dispositivos. São responsáveis pela adaptação dinâmica de

serviços, pois diminuem a heterogeneidade presente no ambiente através da tradução de protocolos de rede e formatos de dados entre clientes e serviços. Os *proxies* também podem condensar conteúdo dos serviços em formatos mais apropriados para dispositivos pequenos, como adaptar para telas pequenas com pouca resolução de imagens. Os *proxies* podem realizar estas operações filtrando no nível de aplicação, por exemplo, pulando seletivamente quadros MPEG em um vídeo *stream*; ou de protocolo, atrasando ou comprimindo dados para aumentar a vazão, de acordo com conhecimento das condições atuais da rede [GRIB00].

No nível de protocolo de rede, os *proxies* podem se comunicar com clientes através de protocolos especiais para dispositivos com conexões de baixa qualidade, de baixa capacidade computacional ou consumo de energia restrito. Podem ser utilizados para estabelecer conexões entre clientes e serviços que necessitem de tarefas complexas, como compressão de dados. Também reformulam o conteúdo de serviço em formatos mais apropriados aos dispositivos pequenos, filtrando na camada de aplicação.

- **Composição de Serviços**

Outras arquiteturas, em geral, podem manipular um conjunto fixo de dispositivos. De modo diferente, essa arquitetura dá suporte à composição dos serviços em *Paths*, e adaptação ao longo do caminho. O *Path* é um fluxo de dados através de múltiplos serviços em uma grande área, incluindo as operações de adaptação de dados para os dispositivos ou serviços através do caminho.

O estabelecimento dinâmico, que examina muitos caminhos em potencial antes de decidir por um caminho em particular, permite a otimização dos fluxos de dados. Como essa otimização é contínua até o fim da vida de um *Path*, a infra-estrutura pode se adaptar às características mutantes do ambiente de execução.

- **Serviço de Descoberta de Serviços (SDS)**

Para a criação de um caminho, devem-se localizar os serviços daquele caminho. O SDS permite que usuários e programas localizem os serviços apropriados através de grandes áreas baseado em pesquisas de atributos.

Os serviços se autodescrevem às instâncias do SDS. Essas descrições são publicadas e agregadas ao longo de uma hierarquia e clientes podem pesquisar essa hierarquia nas instâncias de SDS para localizar os serviços.

Para manipular as preocupações com a segurança, os SDSs devem controlar um conjunto de agentes que têm a habilidade de descobrir serviços, permitindo controle de acesso baseado em capacidade, i.e., esconder a existência de serviços ao invés de desabilitar acesso aos serviços locais.

Considerando o uso desta arquitetura no ambiente de trabalho de campo, é possível dizer que é impraticável o uso de Bases (os *clusters* do ambiente de execução). Só isto impossibilita seu uso no campo. Além disto, ainda são necessários os *proxies* intermediários, outro fator de difícil inserção no ambiente de trabalho.

2.3.3.5 *Middleware DaCaPo++*

A *DaCaPo*++ [STIL99] é um *middleware* que se auto-reconfigura dinamicamente para fornecer um ambiente estável às aplicações multimídia. O principal problema endereçado para as aplicações de multimídia e os serviços de comunicação necessários para estes ambientes, é a necessidade de suporte entre os sistemas fins da comunicação de maneira integrada e eficiente. Este suporte vem combinando especificações de requisitos de QoS, interfaces de aplicações, suporte a *multicast* e segurança.

Os objetivos do projeto DaCaPo++ são:

- fornecer um *middleware* capaz de manipular vários tipos de aplicações de forma modular;
- prover suporte entre sistemas fins que são adaptáveis às necessidades das aplicações; e
- permitir a especificação das necessidades das aplicações e da demanda de comunicação em termos de atributos de QoS e propriedades funcionais.

O *middleware* é localizado entre o acesso à rede e API (*Application Programing Interface*), como é apresentado na Figura 2.14.

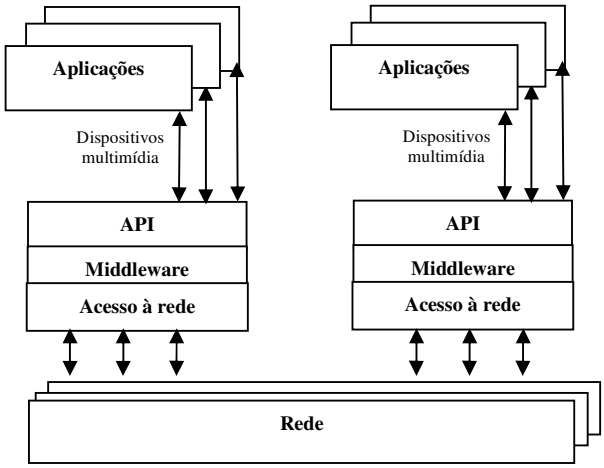


Figura 2.14: Visão Geral da Arquitetura DaCaPo++ [STIL99]

O *middleware* é composto de três módulos: A, C e T.O módulo A (de Aplicação) permite às aplicações acessarem qualquer tipo de serviço de um protocolo de comunicação configurado pela camada da API. O módulo C (de Comunicação) inclui funções relevantes dos protocolos de comunicação entre sistemas fins, como a verificação de soma ou o controle

de fluxo, que dão suporte em tempo de execução aos fluxos das aplicações. O módulo T (de Transporte) realiza as abstrações dos mecanismos, propriedades e semântica de redes de comunicação. Este módulo oferece diferentes níveis de garantias de desempenho de rede, como largura de banda ou nenhuma garantia.

O processo de configuração para realização de adaptação direcionada à aplicação é realizado por estes módulos e pelo protocolo requisitado, de acordo com os requisitos das aplicações, disponibilidade de recursos locais e pré-requisitos de redes.

O núcleo da arquitetura determina que instâncias da *middleware* residam no *end-system* e em cada estação de trabalho, enquanto múltiplas aplicações podem utilizar o mesmo núcleo do *middleware* ao mesmo tempo.

Os principais serviços oferecidos pela arquitetura são:

- **Especificação de Sessão**

Serve para oferecer às aplicações a flexibilidade de especificar suas necessidades de comunicação quando desejado. Elas podem especificar os fluxos necessários e mencionar quais requisitos de QoS específicos precisa para esses fluxos.

- **Protocolo *Lift***

É um mecanismo de transporte ativo que transfere dados entre rede e usuários, dividindo fluxos em dois caminhos de dados: *forward* (dados e controle) e *backward* (*feedback*).

- **Gerenciamento de Recursos**

Um dos principais objetivos da arquitetura é prover eficiência ao reduzir cópia de dados, através de pacotes (transportados pelo protocolo *Lift*) e *buffers*, que mantém os dados a serem transmitidos entre os módulos.

O Gerente de Recursos fornece uma camada de abstração para memória, recursos de CPU e temporizadores, simplificando a portabilidade, redução de *overhead* de gerenciamento de memória e minimizando a operação de cópia de memória. O Gerente de *Buffers* gerencia pacotes e *buffers*.

- **Segurança**

As funções de autenticação e criptografia são parte do *middleware* e os graus de segurança também são tratados como parâmetro de QoS. O nível de segurança fornecido é variável, dependendo da demanda do usuário, pois os recursos computacionais necessários para fornecer o nível mais alto de segurança geralmente excedem processamento de CPU disponível.

Para uso no trabalho de campo, esta arquitetura, diferentemente das anteriores, não especifica a utilização de novos equipamentos para fornecer seu serviço. Porém, como deixa de adaptar conteúdo de acordo com os variados tipos de dispositivos existentes, serão gerados conteúdos idênticos para dispositivos com capacidade computacional alta ou baixa. Além disso, a arquitetura não oferece suporte aos dados de localização física do ambiente.

2.3.3.6 Arquitetura da Carnegie-Mellon

Esta arquitetura foi desenvolvida com o intuito de criar uma infra-estrutura de suporte a aplicações ubíquas. Ela é composta de um *middleware* e serviços utilizados para que os alunos da Universidade Carnegie-Mellon possam desenvolver estas aplicações [WEBE00]. A Figura 2.15 mostra uma visão geral da arquitetura.

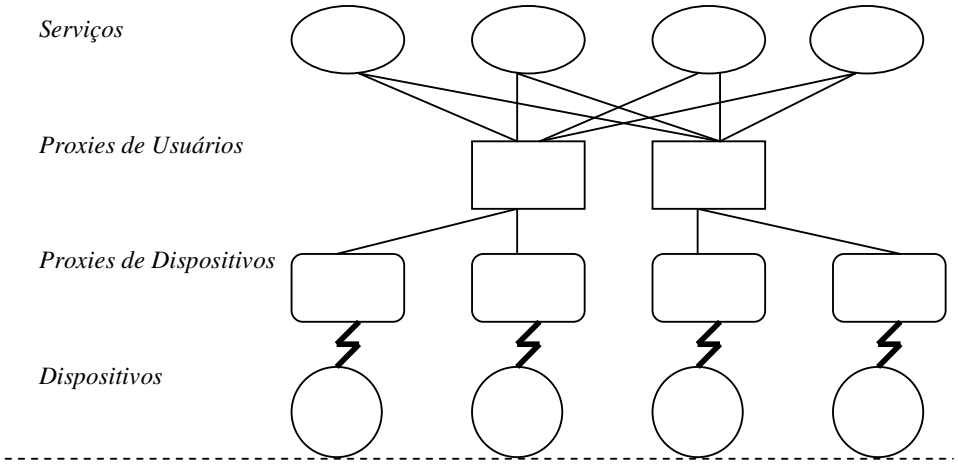


Figura 2.15: Diagrama da arquitetura [WEB00]

Em cada dispositivo existe um pequeno código que gerencia a comunicação das aplicações com a rede. No próximo nível, existem os *proxies* de dispositivos que encapsulam todas as informações específicas de um dispositivo e, visto pela rede, agem realmente como *proxies* para esses dispositivos. O *proxy* de usuário age como um *proxy* para usuário final. Nele estão os agentes e políticas definidas para cada usuário. No último nível estão os serviços, que interagem com os *proxies* de usuários, pois toda autenticação e autorização são baseadas em usuário, não em dispositivos.

No modelo de comunicação, uma aplicação consiste de um ou mais componentes, sendo que cada componente é associado com uma ou mais partes básicas do sistema: um dispositivo, um *proxy* de dispositivo, *proxy* de usuário ou serviço.

Essa arquitetura permite customizações, como permitir modificações da interface de usuário (baseado em características de dispositivos e preferências pessoais) e filtragem de entrada e saída baseada em critérios pessoais.

Da mesma forma que a arquitetura anterior, esta arquitetura também não insere novos equipamentos intermediários para a adaptação de conteúdo. Os *proxies* utilizados são apenas códigos inseridos nos equipamentos participantes da comunicação. Porém, para o trabalho de campo, ainda é necessária a utilização de contexto de localização física pelas aplicações, o que não é fornecido pela arquitetura.

2.3.4 Comparação entre Trabalhos Relacionados

Estes trabalhos apresentados foram comparadas em relação a alguns parâmetros:

1. Adaptação de conteúdo: se suporta ou não esse tipo de adaptação;
2. Adaptação de QoS: se suporta ou não adaptação de QoS de rede e comunicação;
3. Tipo de arquitetura em que é baseada: *Proxy* (P) ou *Middleware* (M);
4. Parâmetros principais: qual o principal parâmetro de adaptação da arquitetura. Pode ser a preferência do usuário, dispositivo, QoS ou recursos;
5. Necessidade de Infra-estrutura: se a arquitetura depende de infra-estrutura (de rede, *middleware* ou *proxies*) para seu funcionamento.

A Tabela 2-2 mostra cada arquitetura em relação a cada parâmetro.

Tabela 2-2: Comparação das Arquiteturas

Arquitetura	Adaptação de Conteúdo	Adaptação de QoS	Tipo de arquitetura	Parâmetros principais	Necessidade de Infra-estrutura
Decision Engine	Sim	Não	*Proxy*	Preferências dos usuários	Sim
$2k^Q$	Não	Sim	*Proxy*	Recursos fim-a-fim	Sim
IMASH	Sim	Sim	*Proxy*	Dispositivo	Sim
Ninja	Sim	Não	*Proxy*	Dispositivo	Sim
DaCaPo++	Não	Sim	*Middleware*	QoS e recursos	Não
Carnegie-Mellon	Sim	Não	*Proxy* virtual	Dispositivo e preferências	Não

Os parâmetros Tipo de Arquitetura e Necessidade de Infra-estrutura são relacionados. As arquiteturas do tipo *Proxy* necessitam da inserção de um equipamento no ambiente de trabalho para sua utilização. Mas como visto desde a Introdução (Capítulo 1), os ambientes de

trabalho de campo possuem apenas o equipamento da Base. Este equipamento, geralmente um computador portátil, não tem capacidade computacional para funcionar como um *proxy* adaptador, pois essas operações geram muito consumo de CPU. Este tipo de arquitetura é mais apropriado para uso em ambientes de trabalho com grande infra-estrutura de comunicação e de dispositivos.

Caso houvesse a possibilidade da utilização de um *proxy* adicional ao computador portátil na Base, ainda assim o envio de dados dos provedores de conteúdo ao *proxy* para reenvio para os dispositivos destinos acarretaria um aumento de custos de rede. Geraria grande tráfego na parte da rede onde se localiza a Base e deixaria os nós próximos sobrecarregados com o repasse de conteúdo, prejudicando os nós que necessitam dos serviços da Base e os serviços da própria Base.

No ambiente de trabalho de campo, o fornecedor de conteúdo é o responsável pela adaptação do conteúdo a ser enviado, pois não é possível contar com a ajuda dos outros nós, em caso da presença de diversos nós de baixa capacidade, o que é bastante comum no ambiente de trabalho de campo.

No caso das duas arquiteturas que não dependem de outros equipamentos (DaCaPo++ e Carnegie-Mellon), seus pontos fracos, para o trabalho de campo podem ser resumidos em:

- DaCaPo++ não adapta conteúdo, mas apenas QoS de tráfego. Deixando de considerar os diversos tipos de dispositivos presentes no ambiente, pode gerar alto custo computacional para os dispositivos de baixa capacidade, como será mostrado adiante na Avaliação de Desempenho (Capítulo 4).
- A arquitetura da Carnegie-Mellon, apesar de adaptar conteúdo de acordo com o tipo de dispositivo e usuário, realiza as operações de adaptação nos próprios dispositivos, através da combinação dos diversos *proxies* implementados em código nas diversas camadas. Esta adaptação local pode ser inviável para os dispositivos de baixa capacidade computacional existentes no trabalho de campo.

A adaptação deveria ser realizada somente pelos dispositivos de maior capacidade. No caso de dispositivos de baixa capacidade que também possam fornecer serviços, o conteúdo gerado só pode ser oferecido a um certo nível de qualidade, compatível com o próprio dispositivo. Os clientes podem ter a opção de aceitar os serviços do provedor com menor qualidade ou apenas recusá-los.

Com esta análise, pode-se concluir que a arquitetura de adaptação não pode depender de dispositivos além dos já existentes no ambiente de trabalho. Também deve-se considerar a existência de dispositivos de capacidade computacional heterogêneas, que podem ser

prejudicados pela falta do processo de adaptação de conteúdo. Como esses dispositivos estão presentes, o método de adaptação deve ser de fácil processamento, para não sobrecarregar as CPUs de baixa capacidade.

2.4 Conclusões

Neste Capítulo foram mostrados diversos aspectos relacionados à Computação Pervasiva, como os serviços de Adaptação de Conteúdo.

Na Seção de arquiteturas de computação pervasiva viu-se que alguns serviços são necessários para tornar as arquiteturas atuais mais adequadas ao trabalho de campo, como localização física e descoberta de serviços.

As arquiteturas analisadas possuem algumas desvantagens para o trabalho de campo. O uso do *broadcast*, limitando a escalabilidade da arquitetura: a separação das áreas por contexto já limitaria o escopo das mensagens e melhoraria a utilização dos recursos de comunicação. Outro problema verificado em uma das arquitetura é que todos os nós do ambiente mantêm informações sobre a posição física de todos os outros nós presentes também gera custos em termos de recursos de rede e de armazenamento nos dispositivos. A limitação do uso dessas informações apenas entre os vizinhos distantes um salto reduz a carga de dados que trafega no ambiente e permite o roteamento de dados baseando em localização física. Outros detalhes devem ser considerados além dos tipos heterogêneos de dispositivos, como os diversos tipos de trabalhadores presentes (com diferentes padrões de mobilidade e prioridade).

Da mesma forma, as arquiteturas de adaptação de conteúdo também necessitam de adaptações para o uso no trabalho de campo. O fornecedor de conteúdo deve ser responsável pela adaptação, pois não é possível contar com a ajuda dos outros nós, devido à presença de diversos nós de baixa capacidade. As arquiteturas não podem deixar de considerar os diversos tipos de dispositivos presentes no ambiente, o que pode gerar alto custo computacional para os dispositivos de baixa capacidade. A adaptação deveria ser realizada somente pelos dispositivos de maior capacidade. No caso de dispositivos de baixa capacidade que também podem fornecer conteúdo, o conteúdo gerado deve ser oferecido a um certo nível de qualidade, compatível com o próprio dispositivo.

O próximo Capítulo mostra outras tecnologias relacionadas, como Descoberta de Serviços, localização física e roteamento de dados em redes ad hoc.

Capítulo 3 - Tecnologias

3.1 Introdução

Diversas tecnologias são utilizadas nos ambientes de computação pervasiva. A descoberta de serviços é utilizada para melhorar a busca de informações de serviços disponíveis na rede. As tecnologias de localização física são usadas para fornecer o contexto de localização nestes ambientes. E os protocolos de roteamento *ad hoc* são utilizados em locais que as conexões de rede são realizadas de forma imprevisível e não fixas.

Este capítulo detalha cada uma destas tecnologias, que são utilizadas no trabalho de campo.

3.2 Descoberta de Serviços

Os protocolos de descoberta de serviços fornecem mecanismos para descobrir dinamicamente serviços disponíveis na rede. Fornecem informações necessárias para buscar, escolher o serviço correto e utilizá-lo [BETT00].

Os principais conceitos relacionados à descoberta de serviço são dispositivo, serviço, *lookup* e descoberta [MCGR00].

Dispositivos e serviços são as entidades que participam na "descoberta". Os serviços incluem qualquer tipo de serviço de rede que possa estar disponível. Um dispositivo pode estar representado na rede por um ou mais serviços. Por exemplo, uma impressora que fornece serviço de fax e impressão.

O termo "*lookup*" se refere ao processo de localizar um objeto ou recurso específico. A busca pode ser realizada pelo nome exato, endereço ou algum outro critério. O *lookup* é passivo, isto é, deve ser iniciado por quem procura e necessita de um serviço ou algum agente que responda à requisição. Como exemplos de *lookup* tem-se DNS [MOCK87], DHCP [DROM97] e diversos serviços de diretório e nomes existentes.

Descoberta se refere a um processo mais espontâneo, no qual várias entidades descobrem outras entidades na rede e se apresentam para outras entidades. Os protocolos de descoberta têm por objetivo deixar as redes mais fáceis de usar. O serviço de descoberta pode

ser usado para *lookup*, enquanto que o *lookup* não realiza a descoberta. As principais características dos protocolos de descoberta são:

- Descoberta e configuração de dispositivos e serviços espontâneos;
- Seleção de um tipo específico de serviço;
- Baixo requisito de administração humana (preferencialmente nenhuma);
- Adaptação automática para disponibilidade móvel e esporádica;
- Interoperabilidade entre fabricantes e plataformas.

3.2.1 Protocolos de Descoberta

Exemplos de protocolos de descoberta existentes são o *Service Location Protocol* [VEIZ97], *Jini* [SUN99], *Universal Plug and Play* [UPNP00] e Salutation [CHAK00]. Esses protocolos serão vistos a seguir.

3.2.1.1 *Service Location Protocol*

O protocolo *Service Location Protocol* (SLP) [VEIZ97] é um protocolo de descoberta automática de recursos para redes baseadas em IP. Baseia seu mecanismo de descoberta em atributos de serviços. Pode prover serviços de *hardware* e *software*.

A infra-estrutura do protocolo consiste de três tipos de agentes: de usuário, de serviço e de diretório. Os agentes de usuários recebem as interfaces dos serviços para as aplicações dos usuários que solicitam os serviços. Os agentes de serviços são responsáveis por publicar as interfaces dos serviços no Diretório de Agentes, tornando o serviço disponível para os agentes de usuários. O diretório de agentes mantém uma lista de serviços publicados na rede.

As principais funcionalidades do SLP são:

1. Obter as interfaces de serviços para os agentes de usuários;
2. Manter o diretório de publicação de serviços;
3. Descobrir os atributos dos serviços disponíveis;
4. Descobrir os diretórios de agentes disponíveis; e
5. Descobrir os tipos de agentes de serviços.

Um serviço é descrito por valores de configuração dos atributos possíveis. O protocolo provê suporte também para um mecanismo simples que mantém as interfaces nos casos onde o *hardware* do serviço está quebrado, mas os serviços continuam a serem anunciados.

3.2.1.2 Jini

Jini [SUN99] é uma arquitetura distribuída orientada a serviço. O objetivo de *Jini* é tornar a rede uma ferramenta flexível, facilmente administrada na qual clientes humanos e computacionais podem encontrar serviços de forma flexível e robusta. *Jini* foi projetada para tornar a rede uma entidade mais dinâmica que reflita melhor a natureza dinâmica dos grupos de trabalho, permitindo a habilidade de adicionar e suprimir serviços de forma flexível [BETT00].

Os serviços *Jini* podem representar dispositivos de *hardware*, *software* ou uma combinação dos dois. Uma coleção de serviços *Jini* forma uma federação, onde os serviços se coordenam com cada um.

Um dos componentes da arquitetura *Jini* é o serviço de *lookup* (JLS – *Jini Lookup Service*), que mantém informação dinâmica sobre os serviços disponíveis em uma federação. Cada serviço deve descobrir um ou mais JLSs antes de entrar na federação. A localização do JLS pode ser conhecida anteriormente ou ser descoberta usando *multicast*. Um JLS pode estar disponível à rede local ou às outras redes remotas (isto é, a Internet). O JLS pode também ter nomes de grupos de modo que um serviço possa descobrir um grupo específico em sua vizinhança.

Quando um serviço *Jini* quer se juntar a uma federação, primeiramente descobre um ou vários JLS das redes locais ou remotas. O serviço então envia seu *proxy* de serviço (isto é, um conjunto de classes Java) ao JLS. Os clientes do serviço podem usar este *proxy* para contactar o serviço original e invocar os métodos do serviço. Dado que os clientes do serviço interagem somente com os *proxies*, isto permite que os vários tipos de serviços, de *hardware* e *software*, sejam acessados de maneira uniforme.

Um usuário que procura por um serviço na rede primeiramente envia em *multicast* uma pergunta para encontrar o JLS. Se existir um JLS, o objeto remoto correspondente é carregado na máquina do usuário. O usuário então usa este objeto para encontrar o serviço requisitado. No *Jini*, a descoberta de serviço é feita pela combinação de interface ou atributo Java. Se o JLS contiver um serviço válido que implementa a interface especificada pelo usuário, um *proxy* para esse serviço é carregado na máquina do usuário. O *proxy* é usado então para chamar funções diferentes oferecidas pelo serviço.

3.2.1.3 *Universal Plug and Play*

O *Universal Plug and Play* (UPnP) [UPNP00], é uma arquitetura projetada para estender o *plug&play* original da Microsoft, que permite que dispositivos sejam adicionados

ao computador autometicamente, porém, neste caso, para os serviços em rede. O UPnP trabalha nas camadas de rede, implementando padrões neste nível. Isto envolve determinados protocolos opcionais da arquitetura, que podem ser executados de forma nativa pelos dispositivos.

O UPnP tenta certificar-se de que todos os fabricantes do dispositivo podem aderir rapidamente ao padrão proposto. Fornecendo um conjunto definido de protocolos de rede, o UPnP permite que os dispositivos construam suas próprias APIs que implementam estes protocolos - em qualquer linguagem ou plataforma que escolherem. O UPnP usa o SSDP (*Simple Service Discovery Protocol*) para a descoberta dos serviços em redes baseadas em IP. O protocolo pode ser operado com ou sem um serviço de *lookup* ou de diretório na rede. O SSDP opera sobre os protocolos padrões abertos existentes, usando o HTTP sobre UDP *unicast* ou *multicast*. O processo do registro envia e recebe dados no formato de hipertexto, mas tem uma semântica especial.

Quando um serviço quer se juntar à rede, ele primeiramente emite uma mensagem anúncio, notificando os participantes da rede sobre sua presença. No caso do anúncio *multicast*, o serviço emite sua propaganda para um endereço *multicast* reservado. Se um serviço do *lookup* (ou diretório) estiver presente, pode gravar essas propagandas emitidas. Outros serviços na rede também podem ver estas propagandas diretamente. A mensagem anúncio contém um URL que identifica o serviço anunciado e um URL para um arquivo que fornece uma descrição do serviço anunciando.

Quando um cliente quer descobrir um serviço, pode contactar o serviço diretamente através do URL que é fornecido na propaganda do serviço ou pode emitir um pedido de pesquisa pelo *multicast*. No caso de descobrir um serviço através do pedido de pesquisa pelo *multicast*, o pedido do cliente pode ser respondido pelo serviço diretamente ou por um serviço do *lookup* (ou o diretório). A descrição do serviço não desempenha um papel no processo da descoberta do serviço.

3.2.1.4 Salutation

O *Salutation* é um protocolo de gerência de sessão e descoberta de serviços. *Salutation* é um padrão aberto, independente de sistemas operacionais, protocolos de comunicação e plataformas de *hardware* [CHAK00]. O *Salutation* foi criado para resolver os problemas de descoberta e de utilização de serviços por um conjunto grande de dispositivos e equipamentos em um ambiente de alta conectividade e mobilidade. A arquitetura fornece aplicações e serviços e formula um método padrão para descrever e anunciar suas potencialidades. A

arquitetura permite também que aplicações, serviços e dispositivos procurem por uma potencialidade particular e que possam requisitar e estabelecer sessões interoperáveis com elas.

A arquitetura do *Salutation* define uma entidade chamada de Gerente do *Salutation* (SLM) que funciona como um corretor (*broker*) de serviços para serviços na rede. As diferentes funções de um serviço são representadas por unidades funcionais. As unidades funcionais representam as características essenciais de um serviço (por exemplo, fax, imprimir, scanner, etc). Além disso, os atributos de cada unidade funcional são capturados no registro da descrição da unidade funcional. O *Salutation* define a sintaxe e a semântica do registro da descrição da unidade funcional (por exemplo, nome e valor).

O gerente do *Salutation* pode ser descoberto por serviços de diversas maneiras:

1. Usando uma tabela estática que armazena o endereço de transporte do SLM remoto;
2. Emitindo, em *broadcast*, uma pesquisa de descoberta usando o protocolo definido pela arquitetura do *Salutation*;
3. Requisitando o endereço de transporte de um SLM remoto através de um diretório central. Este protocolo não é definido pela arquitetura do *Salutation*, entretanto, a especificação atual sugere o uso do protocolo SLP (visto na Seção 3.2.1.1);
4. O serviço especifica diretamente o endereço de transporte de um SLM remoto.

O processo da descoberta do serviço pode ser executado através de múltiplos SLMs. Um SLM pode descobrir outro SLM e determinar quais os serviços que estão registrados lá. A descoberta de serviço é executada comparando um tipo de serviço requisitado, como especificado pelo SLM local, com o(s) tipos(s) do serviço disponível(is) em um SLM remoto. Chamadas de procedimento remoto (RPC) são usadas para transmitir o(s) tipo(s) requisitado(s) do serviço do SLM local ao SLM remoto e transmitir a resposta do SLM remoto ao SLM local. O SLM determina as características de todos os serviços registrados em um SLM remoto manipulando a especificação do(s) tipo(s) requerido(s) do serviço. Pode também determinar as características de um serviço específico registrado em um SLM remoto ou na presença de um serviço específico em um SLM remoto combinando um conjunto específico das características.

3.2.2 Arquiteturas para Descoberta

A descoberta pode ser realizada baseando-se ou não no uso de um servidor [MACB01]. No caso de não utilizar um servidor, o cliente envia em *broadcast* sua pesquisa por serviços e os serviços respondem diretamente ao cliente solicitante.

No caso da descoberta baseada em servidor, os serviços são registrados no servidor, onde os clientes devem buscar as informações de serviços disponíveis. Pode ser dividida em descoberta *early-binding* (associação antecipada) e *late-biding* (associação tardia).

Na descoberta *Late-biding* (Figura 3.1), o cliente se comunica com o servidor (1), que retorna ao cliente (2) o endereço do local onde o serviço solicitado está disponível. Ao receber o endereço, o cliente então se comunica com o provedor do serviço (3).

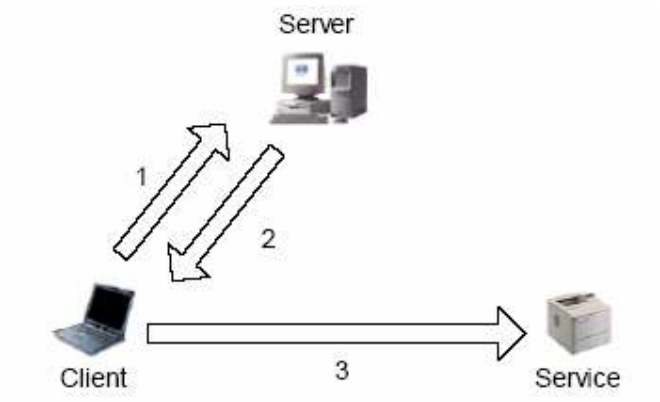


Figura 3.1: Descoberta *Late-biding* [MACB01]

Na descoberta *Early-biding* (Figura 3.2) o cliente se comunica com o servidor (1), que repassa o pedido do cliente diretamente ao provedor do serviço solicitado (2).

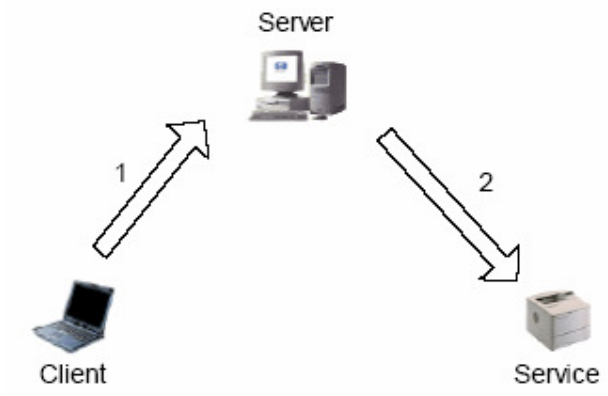


Figura 3.2: Descoberta Early-biding [MACB01]

3.2.3 Escopo da Descoberta (Escopos Geográfico e Espacial)

Todos os protocolos de descoberta discutidos anteriormente possuem escopo administrativo: encontram serviços e dispositivos dentro de um domínio administrativo definido da rede. Estes domínios são lógicos e não necessitam corresponder ao ambiente físico. "Próximo" significa próximo dentro da topologia da rede, não espaço físico.

Entretanto, freqüentemente, é necessário encontrar recursos fisicamente (não logicamente) próximos.

O fornecimento de posição física pode ser feito em duas maneiras [MCGR00]:

1. Os dispositivos e os serviços da rede conhecem sua posição e podem detectar o que e quem está perto deles.
2. Os dispositivos e usuários detectam sua própria posição e relatam-na aos serviços.

No primeiro caso, alguns componentes do sistema são encarregados de rastrear a localização de tudo dentro de seu escopo. No segundo caso, os dispositivos são encarregados de determinar sua própria posição e o relatá-la ao sistema. Um sistema real pode utilizar ambos os métodos.

Uma das formas de fornecer a ciência da localização física é oferecer um serviço padrão para detectar a presença dos usuários e dispositivos. Este serviço requer mecanismos que detectam a chegada e partida (detectores de infravermelho, rádio e sonar) e também políticas sobre quem e o que deve ser seguido, e sob que circunstâncias uma entidade deve ser admitida.

Este serviço poderia ser o serviço padrão de descoberta, ou um serviço separado que interage com o protocolo de descoberta. Uma alternativa é ter dispositivos que detectam sua própria posição. Isto pode ser feito com os *beacons* ou GPS. Enquanto cada dispositivo conhecer sua própria posição, é necessária haver comunicação para estabelecer a posição de outras entidades com o objetivo de se estabelecer o contexto geral.

3.2.4 Resumo - Descoberta de Serviços

Nos ambientes de computação pervasiva, é fundamental que o ambiente interaja com o usuário oferecendo os serviços disponíveis. O oferecimento e descoberta de serviços presentes no ambiente ocorre através da descoberta de serviços. Esta Seção mostrou alguns dos protocolos de descoberta existentes.

O SLP baseia o mecanismo da descoberta nos atributos dos serviços. Através de agentes, o protocolo recebe as interfaces dos serviços que serão utilizadas no Diretório de Agentes para escolha pelas aplicações.

O *Jini* utiliza o conceito de Federação, onde o serviço de *lookup* (JLS) mantém as informações de forma dinâmica sobre os serviços disponíveis. O JLS é é então contactado pelo usuário quando este quer solicitar um serviço. Um objeto remoto correspondente é carregado na máquina do usuário, que então usa esse objeto para encontrar o serviço requisitado.

O UP&P utiliza outros protocolos existentes para desempenhar suas funções. Os clientes que procuram um serviço podem contactar o serviço diretamente através do URL que é fornecido na propaganda do serviço (gerada no momento em que um serviço quer se juntar à rede), ou pode emitir um pedido de pesquisa por *multicast.*

O Salutation utiliza vários meios para descoberta, como tabelas estáticas, pesquisas em *broadcast*, uso de um diretório central ou diretamente, através de endereços conhecidos.

As arquiteturas de descoberta, como o *Late-binding* e *early-binding* possuem características diferentes em relação ao acesso a um provedor de conteúdo. Enquanto no primeiro caso o servidor de descoberta repassa para o cliente a informação do serviço requisitado, no segundo caso, o pedido de serviço é encaminhado diretamente ao provedor, economizando comunicação para o cliente.

Para o trabalho de campo, onde a infra-estrutura se resume aos dispositivos dos trabalhadores e a Base, é necessário utilizar estes protocolos com os equipamentos existentes. Será visto adiante, na Seção 5.8, uma forma de utilizar a descoberta de serviços de modo simples, utilizando somente os equipamentos dos usuários e da Base.

3.3 Tecnologias de Localização Física

Existem muitos métodos e tecnologias de determinação de posição física que podem ser empregados, porém nenhum método operará de forma eficaz em todos os ambientes pretendidos (por exemplo, ambientes fechados e terreno aberto) com exatidão suficiente para sua utilização. Estas tecnologias empregam vários meios tais como óptico, acústico ou rádio-freqüência (RF), cada um tendo vantagens em determinadas situações.

Os métodos de localização do RF, tais como a força recebida do sinal, o ângulo de chegada ou a diferença do tempo da chegada são potencialmente desejáveis porque podem ser integrados com os dispositivos de comunicação sem fio, especialmente em ambientes internos. O sistema global de posicionamento baseado em satélite (GPS) é o sistema RF mais utilizado, fornecendo a cobertura global ao ar livre. Rádios de alta frequência, como 802.11a, fornecem dados exatos de posição devido à sua exatidão inerente do sincronismo [AGRE02].

Sensores, em combinação com RF, IR (infravermelho) ou ultra-som são ideais para ambientes internos. Não somente fornecem informações espaciais de grão fino em taxas altas de atualização, como são não intrusivos, mas são baratos e robustos. Técnicas de localização baseadas em rádio, que são bem sucedidas em ambientes abertos, são prejudicadas por efeitos

multi-caminho (quando um sinal chega à antena do usuário por mais de um caminho) severos dentro de edifícios [HART99]

De modo geral, métodos múltiplos serão necessários para assegurar a determinação da posição que se opera em ambientes variados com a exatidão desejada.

3.3.1 Localização Física em Redes locais

Em redes 802.11b, a localização de um móvel pode ser calculada através de um algoritmo [KISH01]. Primeiramente interceptam-se cada par de círculos com o objetivo de obter um ponto situado exatamente no centro da região de sobreposição dos dois círculos (Figura 3.3). Se os círculos não intercedem, uma linha é construída unindo seus centros e, conseqüentemente, um ponto é escolhido na linha em proporção ao raio dos dois círculos (exemplo, se o círculo 1 tem raio 2, circulo 2 raio 3, e o centro dos dois estão separados por uma distância de 6, então o ponto central será escolhido (2/5)*6 de distância do centro do círculo 1). Dos pontos resultantes, deduz-se onde o usuário está localizado. Para completar, submete-se o grupo de pontos a um algoritmo de agrupamento recursivo.

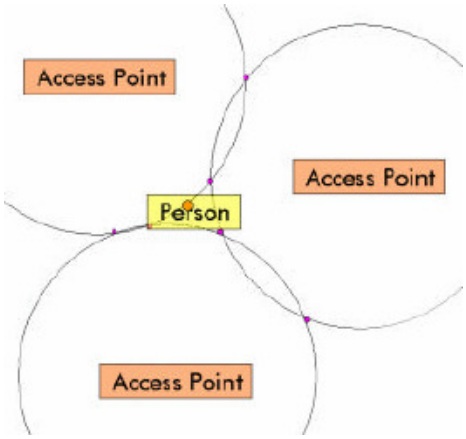


Figura 3.3: Localização em 802.11 [KISH01]

A operação do algoritmo é relativamente direta. Se o número dos pontos no grupo for 3, 2, ou 1, o caso básico emite um centróide do triângulo, um ponto médio, ou um único cálculo do ponto (respectivamente). De outra forma, um algoritmo de agrupamento é designado a quebrar os pontos em um, dois, ou três novos grupos. Se o resultado da aglomeração render um único conjunto, a recursão dividirá o grupo dos pontos na metade e recursará na outra metade arbitrariamente. A perda de exatidão neste caso é insignificante.

O centro do conjunto e dos dois conjuntos de pontos separados é aproximadamente equivalente; os dois conjuntos de pontos separados no caso de divisão forçada serão acoplados firmemente (como uma otimização, as tentativas futuras de se aglomerar são

evitadas, uma vez que o algoritmo acredita que todos os pontos pertencem a um único conjunto; cada metade dos pontos restantes é subdividida melhor que aglomerada). O algoritmo pára quando a recursão alcança um dos casos previamente mencionados. Idealmente, o ponto retornado é uma aproximação razoável da posição verdadeira do usuário.

3.3.2 Localização em Redes Celulares baseadas em RF

Existem várias técnicas diferentes que podem ser utilizadas para localizar um dispositivo celular. Soluções baseadas em infra-estrutura funcionarão com os *handsets* existentes por localizá-los baseando-se em suas características de transmissão [KAIN02]. A posição de localização é gerada através de qualquer um ou combinação dos seguintes métodos.

3.3.2.1 Método do Ângulo de Chegada

Uma das mais conhecidas abordagens ao problema da localização é através da utilização do Ângulo de Chegada (AOA), ou seja, o cálculo do ângulo com que determinado sinal chega às antenas. Também conhecido como Diferença de Chegada (DOA), o AOA é determinado pela variação de fase dos sinais recebidos ao longo de um grupo de antenas [UEVO03].

A localização de um alvo desejado, em duas dimensões, pode ser encontrada pela interseção de duas linhas de direção, cada uma formada por uma radial da estação base ao alvo móvel (Figura 3.4).

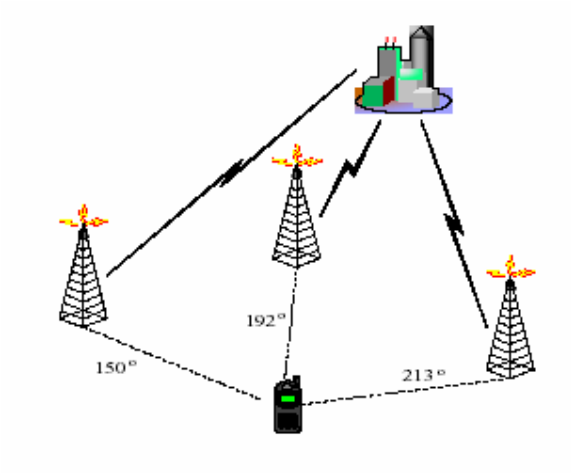


Figura 3.4: Métodos AOA [KHOK01]

Antenas altamente direcionais são necessárias, tornando o AOA difícil no aparelho de comunicação móvel. Esse método deve usar ao menos três estações base e dois ângulos medidos para determinar a localização do móvel na interSeção dos dois círculos. Este método, conhecido como triangulação, pode ser resolvido através de trigonometria, geometria analítica ou procura em tabelas prontas [KHOK01]. Ao adicionar mais locais à leitura do AOA, aumenta-se cada vez mais a precisão da estimação da fonte do sinal.

3.3.2.2 Métodos Tempo de Chegada e Diferença de Tempo de Chegada

No método Tempo de Chegada (TOA), a distância do transmissor até à antena base pode ser calculada através da interpretação do tempo de chegada da propagação de um sinal em cada receptor (Figura 3.5). Alternativamente, pode ser calculada a distância entre transmissores e um receptor de sinal. A TOA de cada sinal é comparada com o tempo (conhecido) de transmissão de cada transmissor [UEVO03].

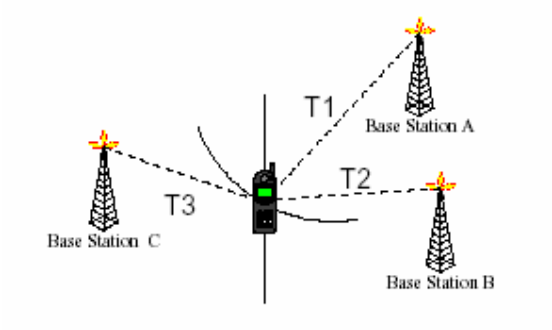


Figura 3.5: Método Tempo de Chegada [KAIN02]

A medida direta de TOA resulta em dois problemas:

- O TOA necessário em todos os transmissores e receptores devem ter relógios sincronizados precisamente (por exemplo, a imprecisão de apenas 1ms pode causar um erro de localização de 300m).
- O sinal transmitido deve ser marcado por um rótulo de tempo para que o receptor calcule a distância percorrida pelo sinal.

No método de Diferença de Tempo de Chegada (TDOA), os sistemas usam um Transmissor de Radio Freqüência (TFR) no dispositivo móvel e receptores em várias antenas utilizadas como estações base (Figura 3.6). O TFR emite sinais que são modulados com largura de banda suficiente de modo a fornecer uma precisão temporal alta (inversamente proporcional à largura de banda), obtendo assim, uma boa precisão na estimativa da localização desejada.

Nesta abordagem, a diferença de tempos de chegada do sinal a diferentes locais pode estimar diferenças de alcance. Quando três ou mais estimativas de diferentes locais estão disponíveis, a localização pode então ser calculada. Esta tecnologia requer a construção de uma infra-estrutura separada de antenas emissoras ou receptoras e/ou a instalação de transmissores em cada dispositivo móvel. A precisão deste tipo de tecnologia é determinada pelas limitações de largura de banda do sinal, bem como pela precisão do sistema e pelo ambiente em que o sinal se propaga.

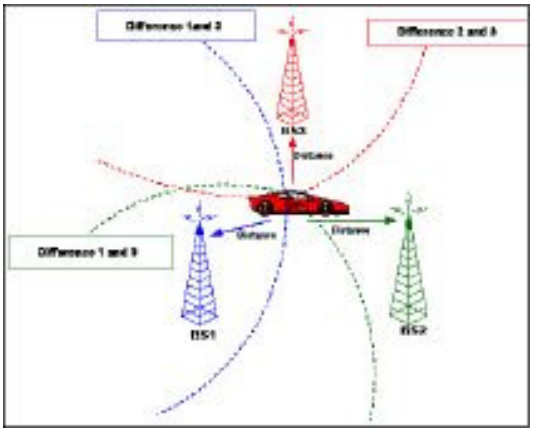


Figura 3.6: Método Diferença do Tempo de Chegada [KHOK01]

Outras desvantagens incluem a extrema precisão temporal necessária em cada receptor, à necessidade de recalibrar estes mesmos receptores devido a diferentes influências no sinal e à necessidade de enviar a informação do TOA (bem como o padrão do sinal) para um processador central. A principal vantagem da TDOA é o fato de não ser estritamente necessário modificar os dispositivos móveis e a sua precisão ser de cerca de 125 metros [UEVO03].

Muitas vezes o TDOA é preferível ao TOA porque, em diversas implementações, menos dados devem ser trocados na interface aérea (carga de comunicação) e a computação do TDOA é realizada em um processador central [KAIN02].

3.3.2.3 Diferença de Tempo Observada

A diferença de tempo observada (OTD) é um método baseado no móvel, desde que um *host* móvel execute as medidas dos sinais de rádio da rede. O OTD medido na estação móvel é definido como o intervalo do tempo entre a recepção dos sinais de duas estações bases diferentes (Figura 3.7) [EBLI01].

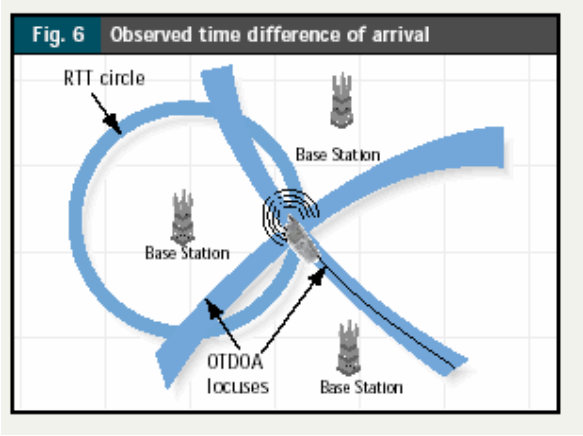


Figura 3.7: Método Diferença de Tempo Observada [EBLI01]

3.3.3 GPS

O Sistema de Posicionamento Global (*Global Positioning System*, GPS) tem como principal propósito prover sinais que permitam que qualquer usuário no planeta, equipado com um receptor GPS, possa calcular corretamente e em tempo real sua posição geográfica (longitude, latitude e altitude).

O padrão de uso para os usuários comuns (não militares), é o Sistema de Posicionamento Padrão (*Standard Positioning System*, SPS). Está disponível para qualquer usuário no mundo, em todas condições climáticas e de forma contínua.

O GPS é formado por dois segmentos. O segmento espacial inclui 24 satélites e o segmento terrestre pode ser subdividido em dois sub-segmentos, de Controle e de Usuário [ALOU96].

- **Segmento Espacial**

Formado pelos satélites que estão em órbita, consiste de 24 satélites, em 6 órbitas diferentes com 4 satélites em cada órbita. Os satélites percorrem a órbita em torno da Terra a cada 12 horas, a uma altitude de aproximadamente 11.000 milhas náuticas. Cada satélite têm 28 graus de visualização sobre a Terra e estão inclinados 55 graus em relação à linha do Equador. Vários pontos da Terra por alguns momentos são visualizados por 6 a 10 satélites focalizando a mesma área. Isto fornece redundância, desde que apenas 4 satélites são requeridos para uma determinação tri-dimensional de posição. Na Figura 3.8, temos basicamente a distribuição de cada órbita em relação à Terra [GPS03].

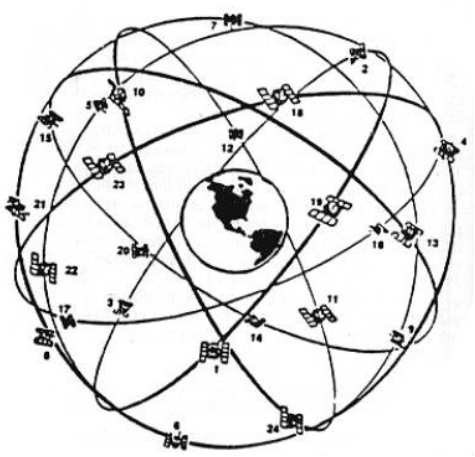


Figura 3.8: Distribuição de órbita [PUC03]

- **Sub-segmento de Controle**

Todos os 24 satélites são controlados pelo Segmento de Controle em terra. Este controle é feito por uma estação de controle *Master* localizado no Colorado, nos Estados Unidos. Ela é responsável por monitorar o rastro dos satélites com o auxílio de cinco estações de monitoramento espalhados pela Terra, processando todos os dados e então enviando a correção e sinais de controle para os satélites. O segmento de controle monitora o desempenho total do sistema, corrige posições do satélite e reprograma o sistema com o padrão necessário.

A localização de cada estação de monitoramento oferece um monitoramento constante de cada satélite. Todas as estações de monitoramento trilham os satélites, determinam seu desempenho operacional, checam os parâmetros e passam estas informações para a Estação de Controle Master. A estação Master pode então determinar os parâmetros de órbita de um satélite e transferir dados de correção para o mesmo satélite.

A determinação desse número de satélites circulando o globo, mais os planos de órbita dos satélites, junto com a estrutura de comando e controle, faz com que o GPS assegure que um número mínimo de quatro satélites sempre estará disponível para oferecer, seja de dia ou de noite, em qualquer lugar da superfície da Terra, uma posição precisa de determinado objeto (através de um receptor - um GPS *Receiver*). O mesmo aspecto de precisão espacial com o mínimo de recurso necessário, oferecendo uma informação segura, é assegurado na localização de cada estação de monitoramento. Também é assegurando a posição exata de cada satélite, supervisionando-o a todo o momento. Estes dois fatores são necessários para assegurar uma precisão tridimensional na determinação da posição - localização geográfica.

- **Sub-segmento de Usuário**

O Segmento do Usuário do GPS consiste dos GPS *Receivers* e a comunidade de usuários. Os GPS *Receivers* convertem os sinais dos satélites em posição, velocidade, e tempo estimado. Quatro satélites, no mínimo, são requeridos para computar as quatro dimensões: x, y, z (posição) e t (tempo) (Figura 3.9) [GPS03] .

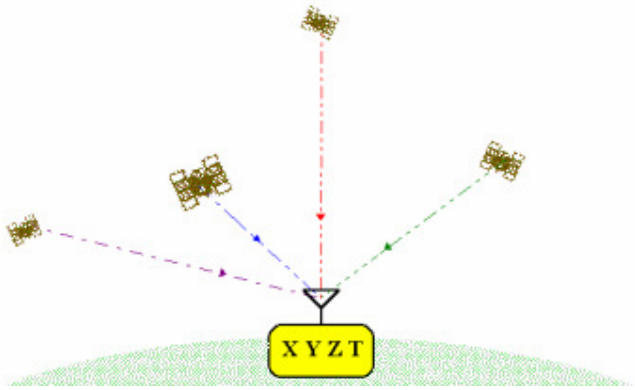


Figura 3.9: Recepção do sinal [DANA00]

Essa posição (x, y, z) pode ser determinada com os valores da distância de três posições diferentes conhecidas pelo método de triangulação (Figura 3.10).

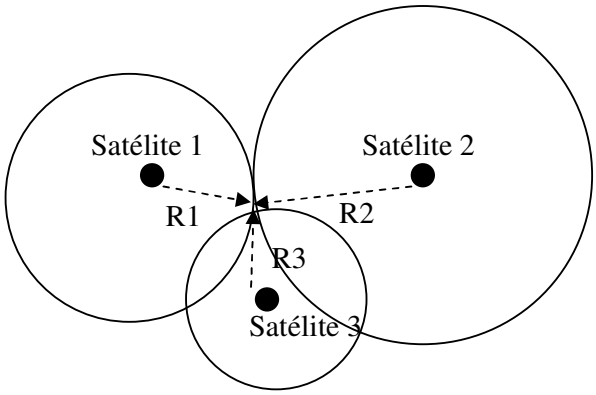


Figura 3.10: Método de triangulação

Conhecendo a distância em relação a três ou mais satélites, o receptor pode calcular a sua posição com base numa série de equações. Em teoria, a distância pode ser calculada multiplicando o tempo que o sinal demora a chegar pela velocidade a que este viaja (a velocidade da luz), no entanto na prática são necessários cálculos mais sofisticados, uma vez que podem existir inúmeras interferências, como as condições atmosféricas ou as

perturbações do Sol. A informação de três satélites é necessária para calcular a longitude e a latitude, no entanto são necessários quatro para calcular também a altitude [UEVO03].

Sob condições ideais o sistema permite, apenas com um receptor, uma precisão de cerca de 25 metros. No entanto o sinal tem sido degradado pelo exercito dos Estados Unidos, que não permitia, até agora, uma precisão horizontal inferior a 100 metros e 150 metros verticalmente. Estes limites de precisão foram estabelecidos por questões de segurança, para evitar a recepção de dados muito precisos por adversários potenciais. Os GPS Receivers são usados para orientação à navegação, posicionamento, disseminação do tempo, e outros recursos.

A navegação é uma das funções do GPS, e é usada por usuários de aeronaves, navios, veículos, e por indivíduos que usam o receptor portátil. Também é usado por observatórios astronômicos, por empresas de telecomunicações, e laboratórios científicos; podendo configurar sinais controlados por freqüências específicas para um determinado propósito, estudo ou experiência [PUC03].

3.3.4 GPS Assistido

A aproximação da posição baseada no telefone móvel integrado com um receptor do GPS é atrativa desde que uma boa exatidão pode ser conseguida, ao menos em áreas abertas [NARE99]. A desvantagem é que os sinais de satélite são atenuados por edifícios e, conseqüentemente, a recepção do número adequado dos satélites em áreas urbanas densas ou dentro dos edifícios pode ser difícil. O alto consumo de energia e o tamanho do telefone móvel também são problemas potenciais. Em todo caso, um esforço considerável está sendo realizado para combinar as tecnologias de celular e GPS e telefones integrados de GSM/GPS já têm sido introduzidos no mercado.

Especialmente os métodos rede-assistidos do GPS têm recebido atenção considerável e levados à padronização do GSM. Enquanto sistemas de GPS comuns falham em locais fechados, este sistema rede-assistido permite que os aparelhos sejam localizados dentro de casa, em veículos em movimento e sob folhagem fechada dentro de um *canyon* [HAYE03].

No conceito proposto pela empresa *SnapTrack Inc.*, os sinais do GPS são recebidos em estações fixas e a mensagem decodificada da navegação é transmitida ao servidor de localização a ser usado na computação da posição (Figura 3.11). Desta maneira, é suficiente para o módulo do GPS do *handset* realizar somente as medidas de pseudo-alcance (alcance até o servidor de cálculo de posicionaento) aos satélites, sem a necessidade decodificar a mensagem da navegação.

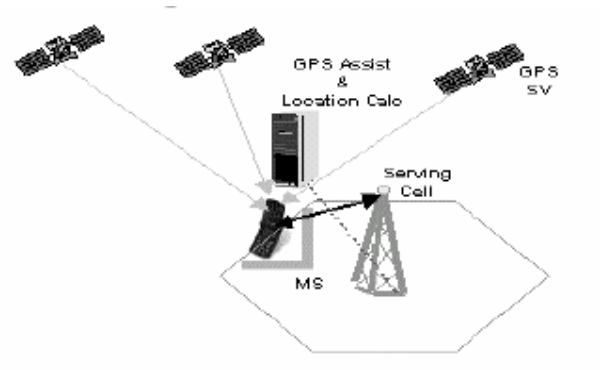


Figura 3.11: Abordagem GPS rede-assitido com cálculo de localização em uma rede [NARE99]

Conseqüentemente, as medidas do pseudo-alcance podem ser realizadas rapidamente e com uma quantidade menor de força recebida. A vantagem adicional do tempo de sincronização é ganha transmitindo a posição bruta do usuário e as freqüências *doppler* (o efeito doppler é a alteração da freqüência notada pelo observador em virtude do movimento relativo de aproximação ou afastamento entre uma fonte de ondas e o observador) ao usuário através da rede celular. O receptor emite as pseudo-escalas (escala de localização até o servidor de localização) observadas ao servidor de localização, que calcula então a posição do usuário usando também as pseudo-escalas medidas de um receptor GPS fixo para obter uma solução exata do GPS.

O tempo necessário para medidas de pseudo-alcance pelo móvel é de somente alguns segundos. Economias consideráveis no consumo de potência podem ser conseguidas, pois o receptor GPS necessita ser ligado por somente um período curto de tempo. Por causa dos requisitos de força recebida, o sistema é solicitado a fornecer posições até no interior de edifícios. Os resultados de testes com a tecnologia mostram que a exatidão (no nível da probabilidade de 68%) varia de quatro metros ao ar livre a oitenta e quatro metros dentro de um edifício (quatro metros da parede externa).

3.3.5 Estratégias de Rastreamento de Localização

Existem diversos métodos de determinação de localização e diversas classificações [WAN99]. Em duas classificações diferentes, podem ser baseadas ou não em infra-estrutura, ou divididas em métodos dinâmicos e estáticos.

3.3.5.1 Baseadas ou não em Infra-estrutura

Na classificação proposta em [AGRE02], estas estratégias podem ser ou não baseadas na utilização da infra-estrutura.

Os Métodos Baseados em Infra-estrutura são divididos em Esquemas Ativos e Passivos (Figura 3.12). Nos Esquemas Ativos, existem servidores de sinalização (*beacons*) que fornecem medidas e sinais de tempo que são recebidos pelos clientes (usuários ou dispositivos). A computação de localização é desempenhada pelos clientes (como em GPS). *Beacons* podem ser requisitados ou constantemente gerados. Nos Esquemas Passivos, os *beacons* são gerados pelos clientes e a localização é computada por um ou mais servidores.

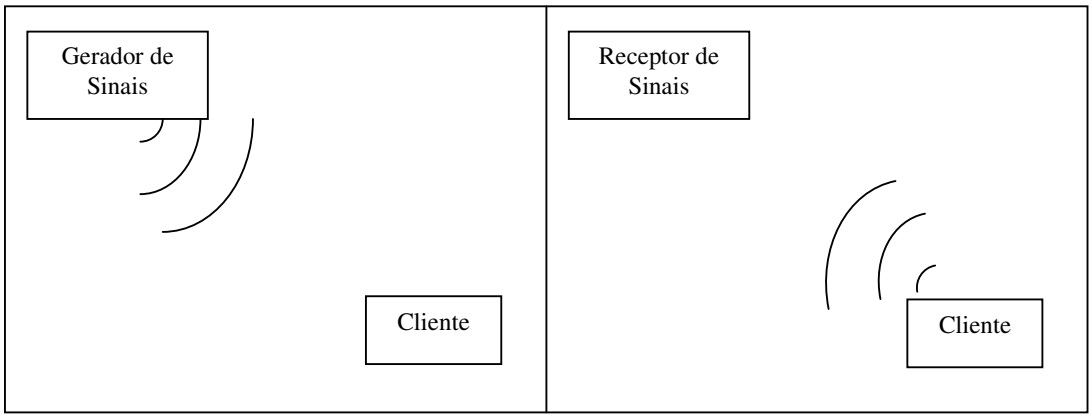


Figura 3.12: Esquema Ativo e Esquema Passivo

Nos métodos Não Baseados em Infra-estrutura, a localização é executada somente pelos clientes. Podem ser executados de maneira independente ou cooperativa.

O modo Independente envolve uma única operação de um cliente de uma posição conhecida, com capacidade de auto-rastreamento (como navegação imercial). O modo Cooperativo envolve medidas de múltiplos clientes para determinação de posição. Estes esquemas não podem determinar localização individualmente e devem se auto-organizar para executar a comunicação necessária para compartilhar as informações de medições.

3.3.5.2 Estratégias Dinâmicas e Estáticas

As estratégias também podem ser classificadas em termos de seus mecanismos de disparo. As estratégias de rastreamento classificadas em [BAR96] são divididas em Dinâmicas e Estáticas.

Nas estratégias Dinâmicas, os clientes decidem quando e onde atualizar. Elas podem ser baseadas em tempo, movimento e em distância.

Na Baseada em Tempo (periódica), cada cliente atualiza o sistema a cada T unidades de tempo (Figura 3.13). A procura por um cliente é conduzida nas áreas de localização em que o cliente pode alcançar em T unidades de tempo da última área de localização na qual ele

informou sua posição. As áreas de busca podem ser calculadas através da velocidade de movimentação do cliente.

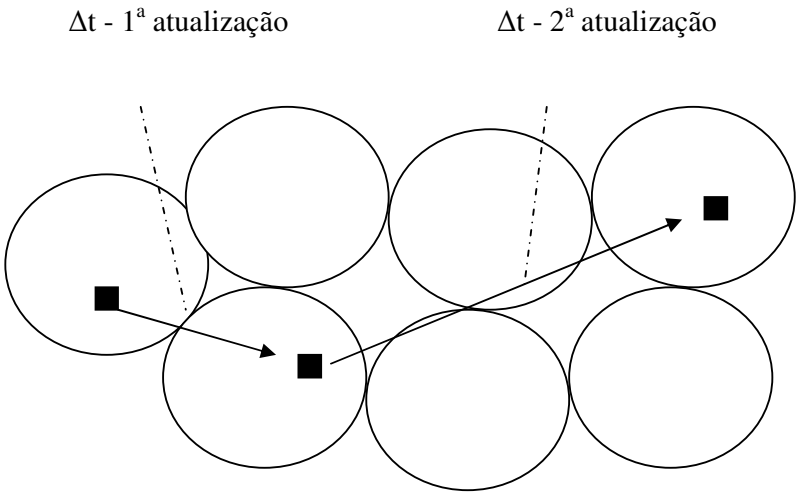


Figura 3.13: Estratégia baseada em tempo

Na **Baseada em Movimento**, cada cliente atualiza o sistema todo momento em que completou M movimentos entre áreas de localização (Figura 3.14). Um movimento pode ser definido como realizar movimentação e realizar uma parada, antes de recomeçar a movimentação. A busca pelo cliente ocorre em todas as áreas que o cliente pode alcançar em M movimentos da última área na qual ele informou sua posição.

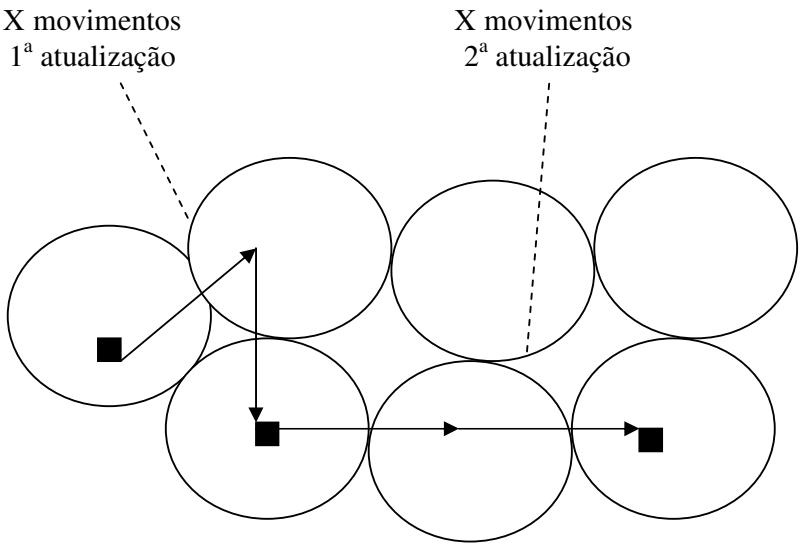


Figura 3.14: Estratégia baseada em movimentos

Na **Baseada em Distância**, cada cliente atualiza o sistema toda hora que está a uma distância D da última área de localização que ele informou sua posição. A busca pelo cliente

ocorre em todas as áreas localizadas a uma distância menor que D que o cliente pode alcançar da última área na qual ele informou sua posição (Figura 3.15).

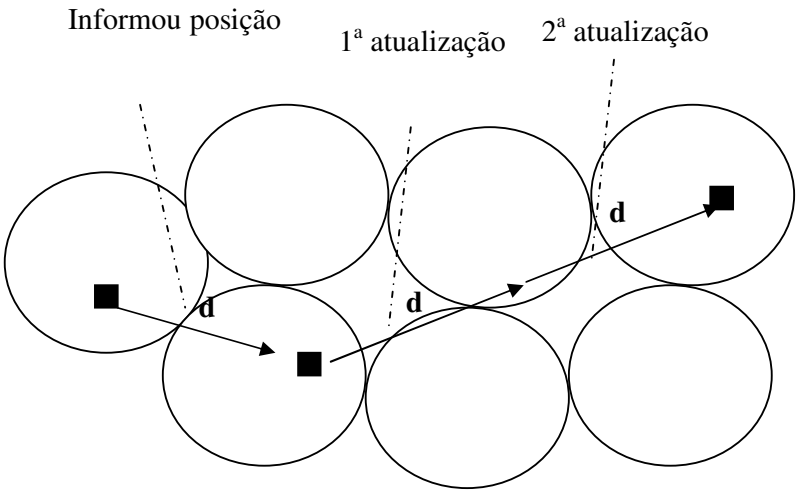


Figura 3.15: Estratégia baseada em Distância

Nas estratégias Estáticas, o sistema decide quando e onde os clientes devem atualizar suas informações. Podem ser subdivididas em Áreas de *Paging*, Áreas de *Paging* Sobrepostas, Centros de Reportagem e Centros de Reportagem Generalizados.

Nas Áreas de *Paging*, as áreas de localização são divididas em grupos separados que são referidos como áreas de *paging*. Os dispositivos informam sua posição somente quando chegam em uma nova área de *paging*. A busca pelo dispositivo é conduzida somente em uma área.

Nas Áreas de *Paging* Sobrepostas: similar à estratégia anterior, exceto que as áreas são agrupadas em grupos separados.

Na estratégia de Centros de Reportagem (CR), um subconjunto das áreas de localização é selecionado e designado como CR. Um usuário deve se 'apresentar' somente quando entrar em um CR. A busca por qualquer usuário é restrita à vizinhança de um CR, que é um conjunto de todas as células para as quais o usuário pode se mover sem cruzar outro CR.

De uma forma mais geral, a estratégia Centros de Reportagem Generalizados (CRG), adiciona uma regra que o usuário nunca atualiza duas visitas consecutivas ao mesmo CR. Isto evita que um usuário que está rodeando a área, entrando e saindo várias vezes, gera atualizações desnecessárias.

Sem se enquadrar nestas categorias anteriores, como estratégias intermediárias, a atualização pode também ser dos tipos Nunca Atualiza (NA) e Atualiza Sempre (AS). Na estratégia NA, o ambiente não tenta manter o rastreamento dos usuários enquanto se

movimentam, e só realiza busca por este usuário quando ele é chamado. Na AS, a rede atualiza toda vez que o usuário se move entre áreas. Como resultado, não existem mensagens de busca e usuários rastreados são localizados em curtos períodos de tempo, porém, o custo deste rastreamento intensivo é muito alto, em termos de largura de banda e bateria dos terminais.

3.3.6 Resumo - Localização Física

Esta Seção mostrou diversas tecnologias de localização física existentes para espaços abertos, como as tecnologias de RF (celular e 802.11) e GPS. Para os ambientes de computação pervasiva, a tecnologia de redes 802.11 é bastante comum, sendo que para o trabalho de campo, seu uso de modo infra-estruturado (uso de antenas como ponto de acesso) é restritivo devido à necessidade de antenas. Com o uso desta tecnologia no modo *ad hoc*, onde cada nó se comunica diretamente com seus vizinhos físicos (a ser visto na próxima Seção), as redes 802.11 têm sido bem aceitas e utilizadas.

Também foram mostradas algumas estratégias de atualização de informações de localização. Estas estratégias são utilizadas principalmente em sistemas de telefonia celulares.

As estratégias de Rastreamento de Localização se diferenciam pelo apoio (ou não) na infra-estrutura física existente, para cálculo da posição física de um nó. Como a não existência de infra-estrutura é uma característica marcante do ambiente de trabalho de campo, a tecnologia GPS permite que o próprio nó sabe e atualiza sua posição física, não necessitando de apoio de outras entidades. O uso de estratégia baseada em infra-estrutura é mais adequada para ambientes fechados, com grande número de dispositivos sensores de localização (como sensores de Infravermelho e Rádio Freqüência).

Nas propostas vistas na Seção 2.2.2, as arquiteturas para redes *ad hoc* não propõem o uso de estratégias de rastreamento diversas. Como será visto no Capítulo de avaliação de desempenho deste trabalho (Capítulo 4), o uso de uma estratégia para todos os usuários é dispendiosa, em termos de custos de rede (aumento da carga e atrasos). Na avaliação de desempenho, também será comentado o uso das tecnologias vistas neste Capítulo nos ambientes de trabalho de campo.

3.4 Roteamento em Redes Ad Hoc

Uma rede *ad hoc* é uma coleção de *hosts* móveis sem fio que formam uma rede temporária sem a ajuda de uma administração central ou serviços padrões de suporte

regularmente disponíveis nas redes de longo-alcance às quais os *hosts* normalmente se conectam [JOHN94]. Não existem roteadores disponíveis e, portanto, cada nó se torna um roteador em potencial (Figura 3.16), onde cada nó deve reenviar tráfego em nome dos outros nós.

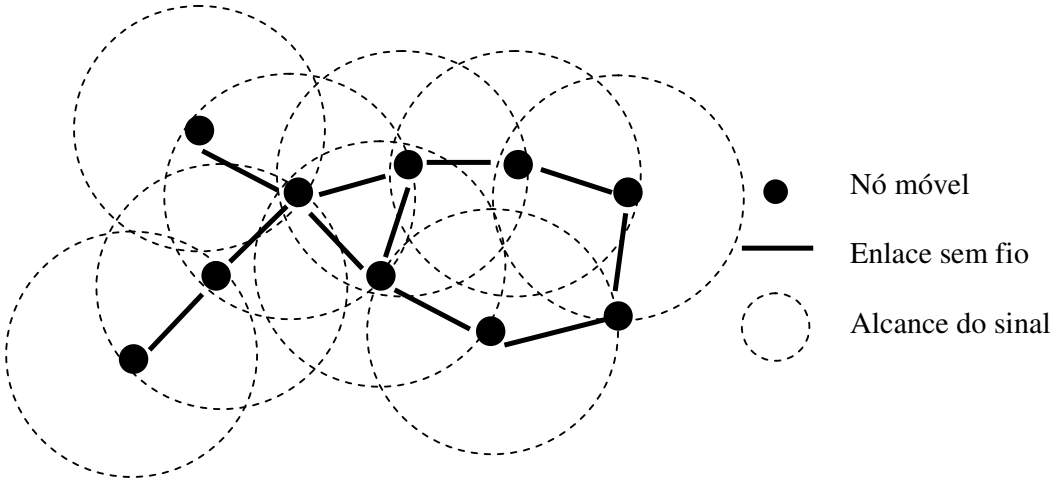


Figura 3.16: Rede Ad Hoc [MOHA03]

Protocolos de roteamento em redes *ad hoc* definem um esquema para transferir mensagens de um nó a outro, decidindo o caminho a percorrer entre fonte e destino.

As próximas seções detalham alguns dos protocolos de roteamento existentes.

3.4.1 Protocolos de Roteamento Convencionais

Os protocolos convencionais possuem como principal parâmetro na escolha da rota apenas o menor caminho percorrido entre o nó fonte e destino. Não consideram os aspectos de qualidade de comunicação, de capacidade dos dispositivos e localização física. São os protocolos DSDV [PERK94], DSR [JOHN03], AODV [PERK03] e TORA [PARK97].

3.4.1.1 DSDV

O protocolo DSDV (Destination-Sequenced Distance-Vector Routing) [PERK94] é um protocolo pró-ativo, isto é, seu algoritmo tenta avaliar continuamente as rotas de modo que quando um pacote necessitar de encaminhamento, a rota já seja conhecida e possa ser utilizada imediatamente.

Em cada nó da rede existe uma tabela com todos os possíveis destinos dentro da rede, e o número de saltos até cada um deles. Cada entrada nesta tabela é marcada com um número de sequência determinado pelo nó destino. Este número distingue rotas velhas de rotas novas, evitando a formação de loops.

Para realizar a atualização de rotas, o protocolo dispõe de dois tipos de pacotes *full-dump*, que transmitem as informações de roteamento. Pacotes do tipo *incremental* completam a informação enviada no último *full-dump* realizado.

Cada *broadcast* de novas rotas inclue o endereço do nó destino, o número de saltos até o destino, o número de sequência da informação original sobre o destino e um número de sequência para o *broadcast*. O DSDV utiliza sempre a rota com o número de sequência mais recente e, caso hajam duas atualizações com o mesmo número de sequência, utiliza-se a rota com menor métrica.

3.4.1.2 DSR

O protocolo DSR (*Dynamic Source Routing*) [JOHN03] realiza roteamento baseado na fonte, no qual o nó emissor determina a seqüência completa de nós pelos quais retransmitir os pacotes. Quando um nó necessita de uma rota para outro nó, ela é determinada dinamicamente. O DSR consiste de dois mecanismos: Descoberta de Rota e Manutenção de Rota.

Quando um nó móvel quer enviar um pacote para um destino, primeiramente ele verifica seu *cache* de rotas para determinar se já existe uma rota para tal destino. Se existir uma rota não expirada, o nó usará esta rota para enviar o pacote. Caso contrário, será iniciada uma Descoberta de Rota através de um *broadcast* de um pacote de Requisição de Rota. Este pacote contém os endereços dos nós fonte e destino e um número de seqüência único "request id", gerado pelo nó fonte. Cada nó na rede mantém uma lista de pares (endereço fonte, *request* id) que recebeu recentemente de qualquer *host*, para evitar duplicação de requisição de rotas.

Ao receber um pacote de requisição, se o nó já recebeu o par ou encontra seu próprio endereço na requisição, descarta a cópia. Caso contrário, acrescenta seu próprio endereço ao registro da rota no pacote e o reenvia em *broadcast* o pedido de rota aos seus vizinhos.

Quando o pacote chega ao destino, o nó destino envia um pacote de *Route Reply* para a fonte com uma cópia da rota. Se o nó pode completar o pedido de rota pelo seu próprio *cache*, ele pode enviar um pacote resposta para a fonte sem ter que repassar adiante. Além disso, qualquer nó que participa da descoberta de rota pode "aprender" rotas ao repassar pacotes de dados e guardar estas informações em seu *cache* de rotas.

Quando um nó envia ou repassa pacotes para um destino, a Manutenção de Rota é usada para detectar se a topologia da rede mudou, de forma que a rota usada por um pacote possa estar quebrada. Cada nó da rota, quando retransmitir um pacote para o próximo nó é responsável por detectar se o enlace para o próximo salto está quebrado.

Muitos protocolos MAC de redes sem fio, como o IEEE 802.11, retransmitem cada pacote até que um reconhecimento na camada de enlace seja recebido ou até que um número máximo de tentativas de retransmissão seja realizado.

Quando os mecanismos de retransmissão e reconhecimento detectam que um enlace está quebrado, o nó que detectou retorna um pacote de erro de rota. Quando um pacote de erro de rota é recebido ou ouvido, o salto com erro é removido de qualquer *cache* de rota e todas as rotas que contém este salto devem ser truncadas naquele ponto.

A fonte pode tentar usar outra rota existente em *cache* para o destino, ou chamar Descoberta de Rota novamente, para achar uma nova rota.

3.4.1.3 AODV

O protocolo AODV (*Ad-hoc On-Demand Distance Vector Routing*) [PERK03] é uma combinação dos protocolos DSR e DSDV. Utiliza o conceito de números de seqüência do DSDV e utiliza os mecanismos de descoberta e manutenção de rota do DSR. O AODV é chamado de um sistema de aquisição de rota sob-demanda puro: os nós não se baseiam em caminhos ativos, não mantém nenhuma informação de roteamento nem participam de nenhuma troca periódica de tabelas de roteamento.

Quando um nó fonte necessita enviar um pacote a um nó destino para o qual não tem nenhuma informação de roteamento em sua tabela, o processo da descoberta de rota é iniciado. O nó fonte transmite em *broadcast* um pedido da rota (RREQ) a seus vizinhos. Cada nó que repassa o pacote de RREQ E cria uma rota reversa para si de volta ao nó fonte. Cada nó mantém dois contadores separados: um número de seqüência do nó e um *broadcast*_id. Broadcast_id é incrementado quando a fonte emite um RREQ novo. Junto com o endereço da fonte, identifica unicamente um RREQ. Além do endereço IP do nó fonte, número de seqüência atual e *broadcast* ID, o RREQ contém também o número de seqüência mais recente para o destino do qual o nó fonte está ciente.

Um nó que recebe o RREQ pode enviar uma resposta da rota (RREP) se for o destino ou se tiver uma rota ao destino com o número de seqüência correspondente maior ou igual aquele contido no RREQ. Se este for o caso, ele envia em *unicast* um RREP de volta à fonte. Se não, ele envia em *broadcast o* RREQ. Cada nó que participa no repasse de retorno do pacote RREP à fonte do RREQ cria uma rota para o nó fonte. Cada nó somente registra o salto seguinte, e não a rota inteira, como no roteamento de fonte.

Os nós registram o endereço IP da fonte do RREQ e seu *Broadcast*_id. Se receberem um pacote de RREQ que já processaram, rejeitam o RREQ e não o repassam. Como o RREP se propaga de volta à fonte, os nós configuram ponteiros para o destino.

Uma vez que o nó fonte recebe o RREP, pode começar a enviar os pacotes de dados ao destino. Se a fonte, posteriormente, receber um RREP que contenha um número de seqüência maior ou contiver o mesmo número de seqüência com uma contagem menor do salto, pode atualizar sua informação de roteamento para esse destino e começar a usar a melhor rota. As rotas são mantidas da seguinte forma: se o nó fonte se mover durante uma sessão ativa, pode reiniciar o procedimento da descoberta de rota para gerar uma nova rota ao destino.

Quando o destino ou algum nó intermediário se move, um RREP especial é emitido aos nós fonte afetados. Uma vez que o salto seguinte se torna inalcançável, o nó acima da ruptura propaga um RREP não solicitado com novos números de seqüência e contagem do salto infinita a todos os vizinhos ativos fluxo acima. Aqueles nós repassam subseqüentemente essa mensagem a seus vizinhos ativos. Este processo continua até que todos os nós fonte ativos sejam notificados.

Após receber a notificação de um enlace quebrado, os nós fonte podem reiniciar o processo da descoberta se ainda necessitarem do destino. A falha de um enlace pode ser detectada usando mensagens “hello” ou usando reconhecimentos da camada de enlace (LLACKS). Também pode ser indicada se houverem novas tentativas de enviar um pacote à falha seguinte do salto. O benefício principal do AODV sobre DSR é que a rota da fonte não necessita ser incluída com cada pacote. Esta diferença resulta em uma redução de *overhead* do protocolo.

3.4.1.4 TORA

TORA (*Temporally-Ordered Routing Algorithm*) [PARKY97] é um protocolo distribuído baseado em um algoritmo de enlace reverso. Garante que todas as rotas são livres de *loop* e provê múltiplas rotas para qualquer par fonte-destino que necessite de rotas para aliviar congestionamentos. É iniciado pela fonte e cria um conjunto de rotas para um dado destino somente quando desejado.

O protocolo possui três funções básicas: criar, manter e apagar rotas. Três pacotes de controle distintos são usados para realizar três funções: pergunta (QRY), para criar as rotas, *update* (UPD), para criar e manter rotas, e *clear* (CLR), para apagar rotas. Durante a fase da criação e manutenção de uma rota, os nós usam uma altura métrica para estabelecer um grafo

acíclico dirigido (DAG) fixado no destino. A altura associada com cada nó é uma quíntupla ordenada hi = (ti, oidi, ri, di, i), onde [CORS01]:

- *ti* é um marcador de tempo associado ao momento da falha de um enlace;
- *oidi* é o ID único do nó que definiu o novo nível de referência;
- *ri* é um bit indicador de reflexão;
- *di* é um parâmetro de ordenação de propagação;
- *i* é o ID original do próprio nó.

Os primeiros três elementos da quíntupla representam o nível de referência e os valores dos últimos dois representam um delta. A altura (*di*) é ordenada em ordem alfabética, assim aos enlace*s* são atribuídos um sentido (fluxo acima ou abaixo) baseado na altura relativa aos nós vizinhos. Durante o roteamento, um nó só pode rotear informações a um nó de nível mais baixo.

Em cada nó da rede, uma cópia separada do algoritmo TORA é executada para cada destino. Quando um nó necessita de uma rota a um certo destino, ele transmite um pacote de QRY que contém o endereço do destino para o qual requer uma rota. Este pacote se propaga na rede até que alcance o destino ou um nó intermediário que tem uma rota ao destino.

O receptor do QRY então transmite um pacote de UPD que lista sua altura em relação ao destino (se o receptor for o destino, a altura é 0). Enquanto este pacote se propaga na rede, cada nó que recebe o UPD ajusta sua altura a um valor maior do que a altura do vizinho do qual o pacote UPD foi recebido. Isto cria uma série de enlace*s* dirigidos do remetente original do QRY ao nó que gerou inicialmente o pacote de UPD.

Um protocolo subjacente, IMEP (*Internet MANET Encapsulation Protocol* [CORS99]) provê entrega confiável e em ordem de todas as mensagens de controle do roteamento de um nó a cada um de seus vizinhos, mais a notificação ao protocolo de roteamento sempre que um enlace a um de seus vizinhos é criado ou quebrado. Para detectar o status de um enlace e manter uma lista dos vizinhos de um nó, cada nó IMEP transmite periodicamente um pacote de *Beacon*, que é respondido por cada nó que o escuta com um pacote *Hello*.

Quando um nó perde seu último enlace fluxo abaixo em conseqüência da falha de um enlace, o nó seleciona uma nova altura tal que ele se transforma em um máximo global, definindo um novo nível de referência, que é mais elevado do que todos os níveis de referência previamente definidos. Os enlace*s* são invertidos para refletir a mudança de se adaptar ao novo nível de referência.

Quando um nó detecta uma divisão na rede, onde uma parte da rede é separada fisicamente do destino, o nó gera um pacote CLR que restaura o estado do roteamento e remove as rotas inválidas da rede. No TORA, cada nó deve manter uma estrutura que descreve a altura dos nós com o status de todas os enlace*s* conectados por uma conexão mantida pela rede. Nos termos da exigência de largura de banda, cada nó deve estar em coordenação constante com nós vizinhos, a fim de detectar mudanças na topologia e cobertura.

3.4.2 Roteamento Baseado em Localização Física

Esta Seção mostra dois protocolos de roteamento que se baseiam em localização física e utilizam a técnica de *geocasting*.

Geocasting é um variante de *multicasting*. Diferencia-se por nomear membros de um grupo os *hosts* que estão localizados dentro de uma região geográfica específica, isto é, a região do *geocast* [JIAN02]. Os nós elegíveis para receber pacotes são especificados implicitamente por uma região física; a associação em um grupo *geocast* muda sempre que um nó se move para dentro ou fora da região do *geocast*. Supõe-se que cada nó tem disponível sua própria posição e sempre que cada nó na região do *geocast* recebe um pacote *geocast*, fará uma "inundação" do pacote *geocast* a todos seus vizinhos. Ou seja, a inundação de pacotes *geocast* ocorre dentro da região do *geocast*. Um efeito desta suposição é que um protocolo *geocast* necessita somente de um nó na região *geocast* para receber pacotes *geocast* da fonte.

Portanto, os protocolos que serão visto a seguir não realizam comunicação *unicast*, entre dois nós, apenas entre um nó e o grupo *geocast*.

3.4.2.1 GeoGRID

O protocolo GeoGRID [LIAO01] evita o inundamento de todos os nós, pois os autores consideraram o este inundamento uma operação cara. Com um inundamento, é provável que um pacote do *geocast* será recebido diversas vezes de diversos vizinhos e que as colisões ocorrem freqüentemente. O artigo considera também *multicast* baseado em árvore instável devido à incerteza elevada da mobilidade dos *hosts* em uma rede *ad hoc*.

A idéia básica do GRID, predecessor *unicast* do GeoGRID, é dividir a rede em grades lógicas. Em cada grade, o *host* perto do centro da grade é eleito para ser o nó responsável para propagar pacotes do *geocast* às grades vizinhas. Estes *hosts* são chamados *gateways*. Os pacotes de *geocast* são emitidos de forma grade-a-grade através dos *gateways*. Assim, os

gateways são responsáveis por enviar pacotes *geocast* às grades vizinhas, o que diminui o *overhead* de mensagens aliviando os outros *hosts* da inundação de pacotes. A suposição é que os tamanhos da grade são construídos tais que um *gateway* pode se comunicar com ao menos um ou mais *gateways* sem intermediários.

Antes de enviar um pacote *geocast*, nenhuma rota *geocast*-específica é estabelecida. Quando repassa um pacote *geocast*, uma região retangular é usada, para definir um sentido de envio e diminuir *overhead* de transmissão. Um pacote recebido fora da região do envio é rejeitado. Se não, se um *gateway* dentro da região de envio receber um pacote, ele realiza um *broadcast* do pacote a seus *gateways* vizinhos, contanto que o pacote não seja uma duplicata de um pacote já retransmitido. Dentro da região do destino um nó entrega um pacote recebido a suas camadas de aplicação superiores.

Além deste esquema, os autores propuseram o GeoGRID baseados em tíquetes. Neste segundo esquema, um *gateway* dentro da região de repasse (região de encaminhamento) envia pacotes *geocast*, mas somente um número limitado de *gateways* fará isso. Para limitar o número de *gateways*, um *gateway* que repassar um pacote envia para pelo menos três vizinhos, em vez de para todos os vizinhos e o remetente inicial limita o espalhamento da inundação especificando um número de tíquetes.

A idéia é que cada tíquete é responsável por carregar uma cópia do pacote *geocast* à região destino. Assim, selecionando um determinado número dos tíquetes, o remetente inicial determina não somente o *overhead* de entrega do *geocast*, mas também a probabilidade do sucesso da entrega. Se um *gateway* não estiver dentro da região do destino, selecionará até três *gateways* vizinhos cujas grades estão mais perto da região do destino e dentro da região de encaminhamento. O pacote do *geocast* é enviado então aos *gateways* selecionados e os tíquetes são compartilhados uniformemente entre eles. Se somente um tíquete for deixado, um pacote é enviado sempre a exatamente um vizinho. Mesmo se um *gateway* receber uma mensagem duplicada, isto é, tíquetes de dois *gateways* vizinhos diferentes, não rejeitará a mensagem *geocast* duplicada. Ao invés disso, o *gateway* enviará a mensagem duplicada como descrito anteriormente, pois cada tíquete é responsável por carregar uma cópia da mensagem *geocast* à região do destino. Se um *gateway* que recebe um pacote *geocast* estiver dentro da região destino, realiza *broadcast* do pacote dentro dessa região para conseguir uma alta taxa de chegada.

3.4.2.2 GeoTORA

GeoTORA [KO00] é um outro protocolo do tipo *geocast* para redes *ad hoc*. É baseado no protocolo TORA (Seção 3.4.1.4). No TORA, um gráfico acíclico dirigido (DAG) é mantido para cada destino. O DAG mostra para cada nó o sentido ao nó destino e é usado para o encaminhamento de um pacote a um destino começando em qualquer nó. O algoritmo do GeoTORA é baseado em uma modificação do *anycast* de TORA. Um DAG é mantido primeiramente para cada grupo do *anycast*. Entre membros do grupo do *anycast* não há nenhum sentido no DAG, isto é, são tudo destinos possíveis. Os sentidos dentro do DAG são definidos atribuindo uma altura a cada nó. Um pacote é enviado sempre a um vizinho com altura mais baixa. Basicamente, a altura é uma distância aos membros da região de destino. É atribuída a altura 0 a membros do grupo *geocast*.

O DAG inicial é criado da seguinte forma: quando um nó requer primeiramente uma rota a um grupo do *geocast*, transmite uma pergunta a todos os vizinhos. A pergunta é repassada em *broadcast* até que um membro do DAG seja encontrado (os nós vizinhos da região do destino já são membros). Um sinalizador em cada nó ajuda identificar as duplicatas, que são rejeitadas. Ao receber uma pesquisa, um membro do DAG responde transmitindo sua altura a seus vizinhos. Um nó que espera uma conexão ao DAG ajusta sua própria altura a altura mínima de todos os vizinhos aumentados por um e transmite sua altura. Como os nós se estão movendo, o DAG não é estável. Porém manter o DAG é conseguido sem inundação.

GeoTORA reage às mudanças no DAG se um nó não tiver mais enlace*s* de saída. Então, o sentido de um ou mais enlace*s* é mudado, que é chamado de reversão de enlace*s*. Os nós vizinhos são afetados por esta medida somente se seu último enlace de saída mudar para um enlace de entrada, o que significa que tem que repetir o processo da reversão de enlace. Se os enlace*s* dirigidos do DAG forem seguidos para enviar um pacote *anycast*, está entregue finalmente a um nó aleatório do grupo do *anycast*. *Geocasting* com este algoritmo trabalha como segue. Começa com um *anycast* a um membro aleatório do grupo *geocast* usando o processo descrito anteriormente. Após receber o primeiro pacote *geocast*, o membro do grupo *geocast* gera uma inundação do pacote dentro da região do *geocast*. [MAIH04]

3.4.3 Discussão dos Protocolos de Roteamento

Os protocolos vistos nesta Seção são comparados com os seguintes parâmetros:

1. Localização Física: se utilizam informações de localização na geração de rotas;
2. Coleta de Informações: se trazem informações sobre os dispositivos pertencentes às rotas geradas;

3. *Unicast* e *Geoacast*: se oferece comunicação *unicast e geocast,* respectivamente.

A Tabela 3-1 mostra a comparação dos protocolos. Todos os parâmetros da tabela são importantes para o trabalho de campo.

Tabela 3-1: Comparação dos Protocolos

Protocolo	**Localização Física**	**Coleta de Informações**	***Unicast***	***Geocast***
DSR	Não	Não	Sim	Não
AODV	Não	Não	Sim	Não
TORA	Não	Sim	Sim	Não
GeoGRID	Sim	Não	Não	Sim
GeoTORA	Sim	Sim	Não	Sim

O parâmetro de localização física é importante para a geração de novas aplicações para o trabalho de campo, pois diversos tipos de trabalhadores em campo dependem destas informações para melhor execução do seu próprio trabalho.

A coleta de informações é necessária para ajudar outros serviços da arquitetura de trabalho de campo. Como será visto adiante, o uso das informações dos dispositivos de uma rota ajudam a adaptação do conteúdo que trafega no ambiente. É também importante poder gerar rotas entre dois pontos, pois um trabalhador pode necessitar uma comunicação com outro trabalhador ou com a Base de trabalho. Também necessário o uso de comunicações do tipo *geocast*, onde todos os trabalhadores de uma certa área de trabalho receberão informações importantes ao seu trabalho.

De acordo com estes requisitos, verifica-se que o protocolo GeoTORA, possui quase todos os requisitos, menos a comunicação *unicast*. Porém, os dados de localização física são referentes apenas a áreas, não a nós pontuais.

Portanto, para gerar um protocolo adequado aos requisitos do trabalho de campo, o protocolo AODV foi estendido para gerar comunicações *unicast* baseadas em localização física e a coleta de dados dos dispositivos da rota. Este protocolo foi escolhido por ter comportamento de coleta de informações na rota e possuir uma implementação razoável no software de simulação NS, usado na avaliação de desempenho.

O protocolo X-AODV (conteXt-AODV) será mostrado no Capítulo 4.

3.5 Conclusões

Neste Capítulo foram mostrados diversos aspectos relacionados à Computação Pervasiva, como Descoberta de Serviços, Localização Física e Roteamento de dados.

Os protocolos de roteamento de dados, para prover os novos serviços necessários às arquiteturas, necessitam de adaptações para oferecer um ambiente mais adequado ao trabalho de campo, como a separação das áreas por contexto e geração de rotas *unicast*.

O próximo Capítulo mostra o protocolo de roteamento X-AODV, adaptado para os requisitos do ambiente de trabalho de campo.

Capítulo 4 - Protocolo X-AODV

4.1 Introdução

No ambiente de trabalho de campo, a cooperação entre os participantes é obtida através de troca de mensagens pelos meios de comunicação entre eles. Como visto neste trabalho, o contexto do ambiente (usuários e localização física) influi nos dados transmitidos.

Usando como base o protocolo de roteamento AODV (Seção 3.4.1.3), o protocolo X–AODV (conteXt-AODV) foi projetado para facilitar a comunicação entre os participantes através da utilização de contexto no roteamento de dados. A geração e a escolha de rotas considera a Classe de Dispositivos (Seção 5.5) e a localização física dos nós. A manutenção e exclusão de rotas seguem o mesmo funcionamento do protocolo AODV original.

Diferentemente dos protocolos de roteamento por localização física vistos na Seção 3.4.2, o X-AODV possibilita a comunicação *unicast*. As próximas seções mostram definições básicas e detalham o funcionamento do protocolo.

4.1.1 Definições Básicas

Antes de apresentar o protocolo, algumas definições são necessárias para melhor entendimento. Um cenário de trabalho de campo, exemplificando os conceitos a serem apresentados, é mostrado na Figura 4.1.

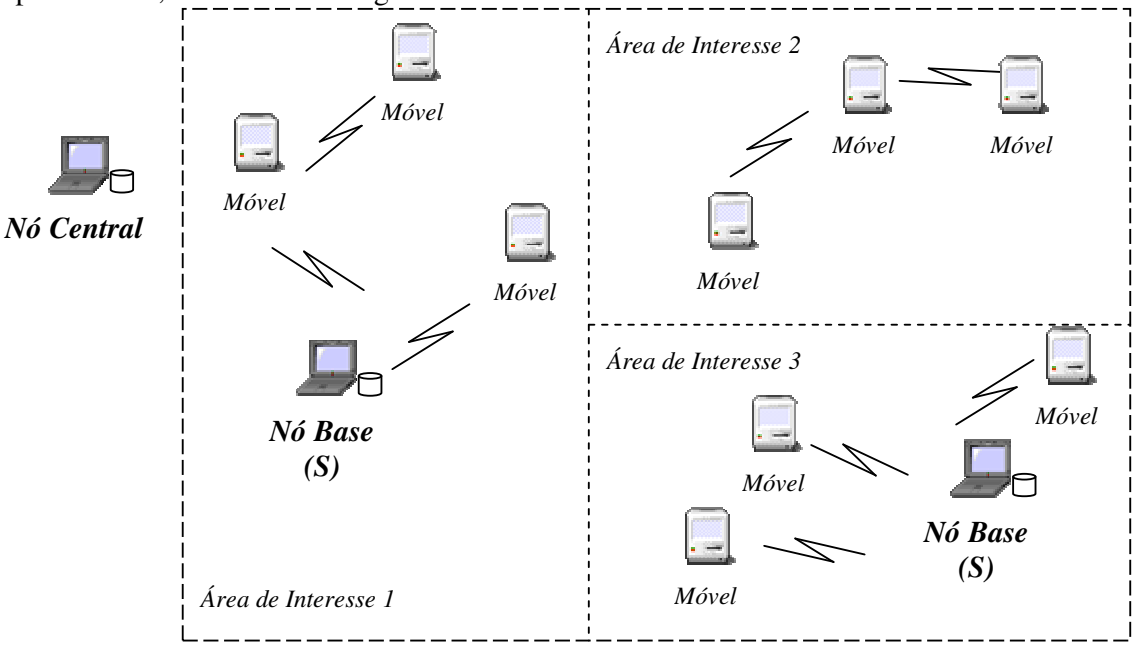


Figura 4.1: Cenário de trabalho de campo

- **Áreas de Interesse**

A área geográfica total do ambiente de trabalho de campo é previamente conhecida e mapeada por GPS. É sub-dividida em áreas menores, chamadas de Áreas de Interesse (AI), ou seja, que possuem interesses específicos em comum. Por exemplo, uma mina que possui três áreas, sendo uma de exploração de minério, outra para futura investigação e área de preservação ambiental. A sub-divisão por contexto facilita a disseminação de informações, pois pode ser utilizada para localizar usuários que sempre trabalham nessas áreas ou transmitir dados somente para as áreas que possuem interesse nelas.

A divisão física da AI, baseada por assunto de trabalho em comum, é realizada sem permitir sobreposição de áreas, para facilitar a utilização do protocolo de roteamento.

- **Servidor/Nó Base (S)**

Base é o dispositivo computacional de alta capacidade que fica localizado em uma base operacional, para apoio dos trabalhadores de campo, como visto na Seção 1.1. O nó Base é responsável por manter os dados relativos à AI, os dados de localização física dos nós pertencentes à área e o conteúdo disponibilizado para os usuários. Cada AI tem seu próprio nó Base. Nesse dispositivo ficam os Repositórios de Perfis (RP), de Mobilidade (RM) e o Diretório de Aplicações (DAP). Todos os dispositivos utilizados pelos trabalhadores se comunicam com o servidor Base.

- **Servidor/Nó CENTRAL**

Localizado fora a área de trabalho de campo, é um servidor de dados (banco de dados, base GIS, etc) principal, onde os dados do trabalho realizado são processados. Por exemplo, todos os dados coletados no campo após o final de uma etapa poderiam ser então incorporados à base de dados principal para estudos, projeções, comparações com dados antigos, etc.

- **Agrupamentos**

O conjunto de nós agrupados em uma Área de Interesse é unicamente identificado por AI. A Figura 4.2 exemplifica uma área de trabalho de campo formada por múltiplas AI's: AI_1, AI_2, AI_3 e AI_4.

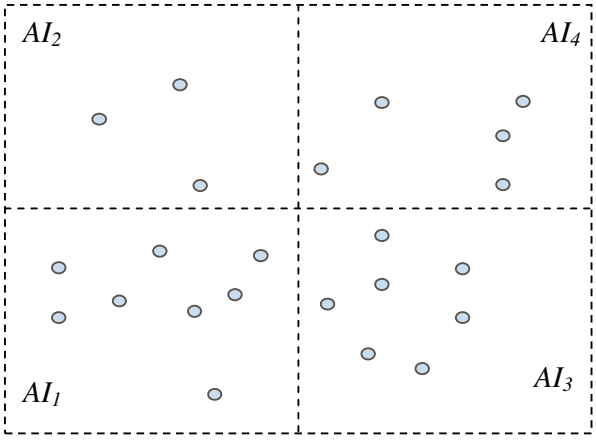


Figura 4.2: Exemplo de conjunto de *AI's*

- **Nós Bordas e Comuns**

Os outros nós, além dos servidores, podem ser do tipo Borda (B) e Comum (C). Os nós Borda, B_i, são os nós localizados na parte mais externa de cada AI e funcionam como *gateways* entre cada AI. Os detalhes sobre nós Borda serão mostrados na Seção 4.1.2. O restante dos nós é do tipo Comum, apenas participantes do grupo, localizados dentro da área geográfica da AI.

4.1.2 Informações de Localização Física

As informações de localização física são utilizadas de diversas maneiras. Os Clientes e os dados sobre sua Base correspondente são carregados na memória. As informações sobre a área de trabalho (AI) de cada cliente também são armazenadas na memória e correspondem aos dados das bordas. Essas informações são utilizadas para a troca de dados entre diferentes áreas de interesse. Tais informações são dividas em:

1) Informação de cada nó

   Cada Cliente possui o recurso de GPS, podendo autolocalizar-se.

2) Localização de Vizinhos

Vizinhos são todos os nós ao alcance da comunicação sem fio de cada nó. Cada nó armazena as informações de localização física do seu vizinho, com o objetivo de gerar roteamento através de localização física (Seção 3.4.2).

Como exemplifica a Figura 4.3, o conjunto de vizinhos de cada nó é dado por: VZ_A={B}; VZ_B={A,C}; VZ_C={B,D}; VZ_D={C}; VZ_E={ }.

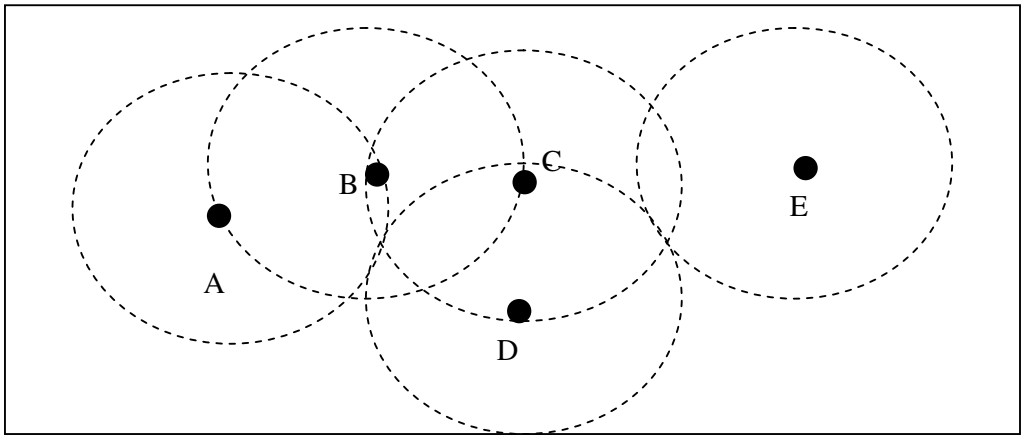


Figura 4.3: Identificando Vizinhos

A distância máxima para ser considerado vizinho depende do alcance do nó. Os nós, quando estão lado a lado, devem possuir contato de alcance (Equação 4-1), caso contrário, não são considerados vizinhos (no exemplo, nós C e E).

Com o conhecimento da energia (e) gasta na transmissão de sinais, é possível saber a distância (d) máxima do alcance dessa comunicação.A Equação 4-1 mostra a relação entre energia e distância.

$$e = kd^c \qquad \text{(Equação 4-1)}$$

onde *k* e *c* são constantes para um sistema sem fio específico [LI01].

3) Localização da Base

Como a Base é montada em um local fixo, para apoio dos trabalhadores de campo (clientes), esses devem sincronizar seus equipamentos com a Base antes de sair a campo, a localização física da Base é armazenada em cada Cliente. Caso a Base seja movida pela área, sua nova localização física será atualizada para todos os nós (via *geocasting*, Seção 5.9.2).

4) Informações enviadas para registro no RM

Será enviado ao Repositório de Mobilidade da Base apenas a diferença entre a posição do Nó em relação à Base, gerando menor custo de dados transmitidos.

5) Informações de Borda

Juntamente com a informação da localização da Base, os Clientes são informados sobre os limites da AI à qual pertencem. Esses limities ficam armazenados na forma das coordenadas da área. Essas informações são necessárias para o uso dos trabalhadores que devem se limitar a sua área de atuação, além de serem úteis no roteamento. Como visto na Seção 4.1.1, os nós Borda realizam a comunicação entre duas AI's. O processo do roteamento entre duas AI's será visto na Seção 4.1.7.

O nó é eleito Borda quando um nó Borda de outra AI está a seu alcance e não existe nenhum vizinho Borda da sua AI em seu alcance, como mostra a Figura 4.4. Isso permite que a comunicação entre duas AI's seja realizada.

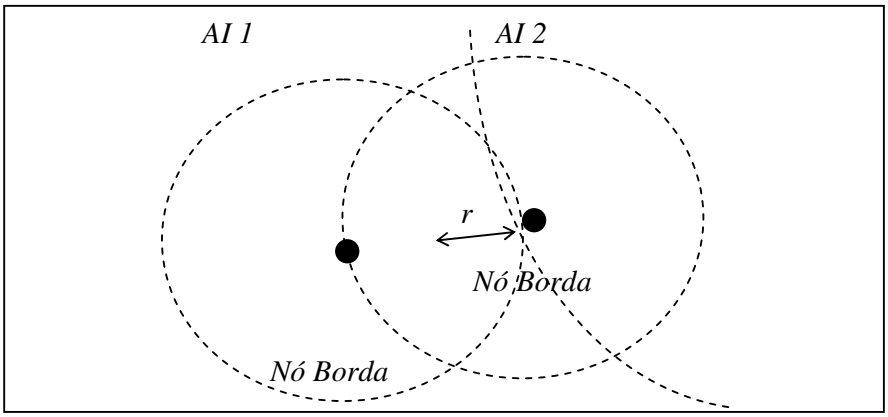


Figura 4.4: Troca de status para Borda por aproximação de outro Borda

No momento da troca de status (de nó Comum para Borda) é realizada a troca de mensagens Atualiza Vizinho (Seção 4.1.5), que informa aos vizinhos físicos seu novo status.

6) Informações de AI's

Os nós Borda monitoram as AI's com as quais mantém contato. Em suas tabelas de vizinhos (Seção 5.8.2), os outros nós Borda são armazenados, juntamente com a informação de qual AI pertencem.

7) Controle de potência de transmissão

Como cada nó conhece seus vizinhos e suas posições físicas e, de acordo com a Equação 3.1, conhece a sua capacidade de transmissão, pode calcular a distância física entre eles (Figura 4.5). Com isso, pode ser realizado o controle de potência de transmissão de dados. Além de gastar menos energia para realizar as comunicações, esse controle também diminui a interferência com os outros nós ouvintes da transmissão.

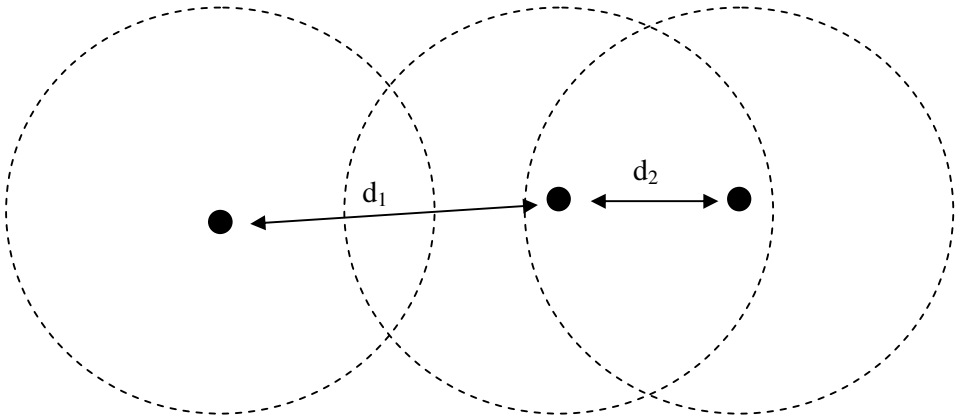


Figura 4.5: Potência de transmissão variável de acordo com a distância física

O nó, sabendo para qual outro nó irá transmitir, verifica a distância física (*d*) entre eles. Dependendo dessa distância, aumenta ou diminui a potência da transmissão para chegar ao nó em questão. Caso o objetivo do envio seja uma mensagem *geocasting* (envio para os nós pertencentes a uma certa área geográfica – Seção 3.4.2), a potência deve incluir o vizinho mais distante.

Esta é uma sugestão do uso das informações de localização física para melhor aproveitamento do uso de força dos nós.

4.1.3 Premissas

Para projetar um protocolo que use informações de contexto ao gerar e escolher rotas, as informações de contexto devem estar disponíveis. Portanto, cada nó:

1. Sabe sua posição física atual, pelo uso do seu módulo GPS;
2. Conhece as bordas da Área de Interesse à qual pertence, carregada em sua memória na sincronização inicial na Base;
3. Reconhece os vizinhos a seu alcance (ver Seção 4.1.2);
4. Reconhece os serviços oferecidos por seus vizinhos (ver Seção 5.8.2).

Quando existir tráfego na rede, todos os nós lêem os pacotes X-AODV e verificam se tem interesse na mensagem transmitida.

4.1.4 Formato dos Pacotes X-AODV

Os pacotes X-AODV são uma extensão dos pacotes do protocolo AODV original. Porém, para a implementação do protocolo X-AODV, foi adicionado o campo LOCFIS

(abreviação de localização física), que utilizou o campo RESERVED existente dos pacote originais para sua criação.

A Tabela 4-1 mostra o tamanho do campo RESERVED do protocolo AODV nas mensagens originais e depois da inclusão do campo LOCFIS, que utiliza 15 bytes de cada mensagem. Os pacotes originais do AODV e suas descrições são detalhados em [PERK03].

Tabela 4-1: Campo Reserved no AODV e X-AODV

Mensagem AODV	**RESERVED Original (AODV)**	**Diferenca LOCFIS e RESERVED**	**Tamanho da Mensagem AODV**	**Tamanho da Mensagem X-AODV**
RREQ	11	4	186	190
RREP	9	6	155	161
RERR	15	0	155	161
RREP-ACK	8	7	15	22

O campo LOCFIS é composto de 3 sub-campos:

a. Tipo de mensagem: 1 byte. Este campo é usado para indicar qual a operação que deverá ser executada. Pelo tipo da mensagem, os nós que a recebem podem tomar decisões sobre o próximo passo da entrega de dados. A Tabela 4-2 mostra os valores possíveis do campo Tipo de Mensagem.

Tabela 4-2: Valores do campo Tipo de Mensagem

Valor	**Descrição**
0	Pedido de Rota (PRT)
1	Volta de Pedido de Rota (VPRT)
2	Envio de dados (EDS)
3	Pedido de Listagem de Serviços (PLS)
4	Pedido de Serviço Específico (PSE)
5	Atualiza Vizinhos (AVZ)
6	Busca Vizinhos (BVZ)
7	Entre Vizinhos (EVZ)
8	Entre Bordas (EBD)
9	*Geocasting* (GEO)
10	Atualiza Posição Física (APS)

b. Número de seqüência do pacote, SEQ_V: 2 bytes (diferente do número de seqüência original do pacote, ainda existente e incluído na parte do cabeçalho original, para controle de falhas e recepção);

c. Uso Múltiplo: 12 bytes. Este campo poderá ser usado de diversas formas, como inserir o endereço físico de um nó. O uso deste campo será detalhado em cada mensagem que o utiliza.

As mensagens serão detalhadas na próxima Seção.

4.1.5 Tipos de Mensagens da Arquitetura

Esses tipos de mensagens possibilitam que todos os nós que recebem e repassam pacotes tenham conhecimento dos objetivos do tráfego atual e possam tomar decisões sobre como reencaminhar os pacotes ou até mesmo como armazenar dados importantes referentes aos outros nós participantes.

a. Atualiza Vizinhos (AVZ)

Com a intenção de gerar roteamento baseado em localização física (Seção3.4.2) e melhorar a busca por serviços em sua área, essa mensagem possibilita a troca de informações entre vizinhos que estão ao alcance de sua comunicação sem fio (Figura 4.6).

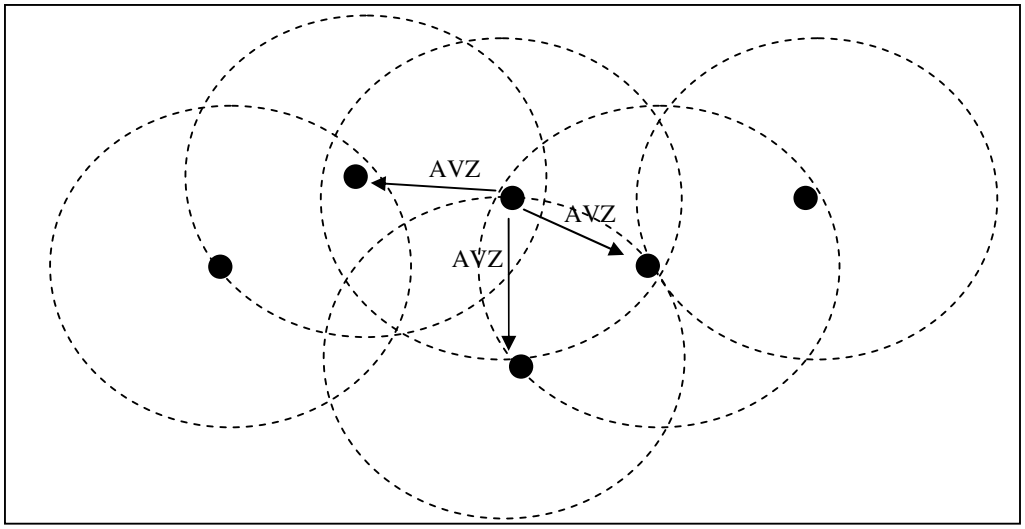


Figura 4.6: Alcance das mensagens Atualiza Vizinhos

Basicamente, o procedimento de atualizar vizinhos segue esta sequência:

- Quando um nó chega ao alcance de outro nó, envia uma mensagem AVZ contendo seus serviços disponíveis. Os detalhes sobre Descoberta de Serviços serão mostrado na Seção 5.8.
- Recebe de seus vizinhos uma mensagem com seus dados de serviços.

- Os dados recebidos são armazenados na tabela Vizinhos (Tabela 5-7).
- Se após um certo ΔT (*time-out*) deixar de receber transmissões de algum vizinho, apaga os dados referentes a ele de sua tabela.

Como em um roteador que utiliza tabelas estáticas, um nó deve atualizar sua tabela de Vizinhos para apagar rotas não existentes. O valor do ΔT depende da mobilidade do próprio nó: um nó de classe de mobilidade alta deve ter esse *time-out* menor do que um nó de mobilidade baixa.

Essa tabela também é utilizada pelo SLF. Como o roteamento baseado em localização física possui a vantagem de que um nó só precisa saber da localização de seus vizinhos (Seção 3.4.2), o nó também armazena os dados de localização atual do vizinho.

Quando um vizinho gerar tráfego diferente de AVZ, o nó também analisa o pacote enviado e verifica se o vizinho está atualizando sua posição física para a Base (mensagem Atualiza Posição Física), com o objetivo de atualizar o RM. Após atualização, o nó armazena os novos valores de localização em sua tabela.

Para garantir que as mensagens AVZ sejam processadas apenas pelos nós ao alcance de 1 salto do nó emissor, o número de seqüência da mensagem AVZ deve ser igual a 00. Quando o vizinho recebe, verifica se é 00. Se for, armazena os dados na tabela Vizinhos. Caso contrário, descarta a informação.

b. Pedido de Serviço Específico (PSE)

Como visto na Seção 5.8, quando um nó quer pedir um determinado serviço, deve procurar o Diretório do grupo, localizado na Base. No pedido de serviço que é enviado em direção à Base, é inserido no campo Tipo da mensagem a indicação de que o nó emissor vai solicitar um tipo específico de serviço. Os nós intermediários que recebem o pacote de pedido de serviço verificam em suas tabelas se possuem dados de algum vizinho que seja provedor de um serviço. Caso positivo, abre a mensagem (os dados do serviço solicitado são inseridos em uma camada superior do pacote) para verificar qual o serviço que está sendo solicitado.

Se esse nó possuir em sua tabela Vizinhos o endereço do provedor do serviço solicitado, roteia o pedido para o provedor do serviço. Caso não possua, roteia em direção à Base. Por fim, a Base verifica em seu diretório o endereço do provedor e repassa para o provedor, que, por sua vez, se comunica com o Cliente.

c. Pedido de Listagem de Serviços (PLS)

Caso a mensagem indique que o nó deseja conhecer todos os serviços disponíveis em sua AI, a mensagem é roteada diretamente para a Base (Seção 5.8.3), que envia uma lista dos serviços disponíveis (cada um com as respectivas classes de serviços disponíveis).

d. Pedido de Rota (PRT)

Sinaliza que o pacote está apenas solicitando rota a um certo destino.

O nó emissor pede uma rota e armazena no campo Uso Múltiplo sua posição física atual. O tamanho do campo, de 12 bytes é suficiente para armazenar o endereço físico do nó. Como a latitude/longitude de um ponto é dada por 10 algarismos, mas será transmitida apenas a difernça de latitude e longitude entre o nó e a Base (Seção 4.1.2), os valores transmitidos são de apenas 6 bytes cada. A geração da rota é realizada de acordo com o processo descrito na Seção 4.1.6.

Os nós que recebem o pacote verificam se é de um vizinho registrado. Se SEQ_V=00, armazenam o valor da posição física atual do emissor em sua tabela Vizinhos. Ao repassar o pacote, incrementam SEQ_V (SEQ_V=SEQ_V+1). Se receberem outro pacote (SEQ_V') e SEQ_V'<SEQ_V, descartam-no.

e. Volta de Pedido de Rota (VPRT)

Sinaliza que o pacote está retornando uma rota solicitada ao emissor do pedido de rota. Os valores das Classes de Dispositivos pertencentes à rota gerada são armazenados no campo Múltiplo Uso do pacote a cada salto.

f. Envio de dados (EDS)

Sinaliza que é uma transmissão entre fonte e destino específicos, com rota gerada anteriormente Como no funcionamento do protocolo AODV, a rota fica armazenada em cada nó e, assim, os nós que receberem esse pacote verificam se pertencem à rota. Caso contrário, descartam o pacote.

g. Atualiza Posição Física (APS)

É gerada pelo nó de acordo com a sua Classe de Mobilidade e Prioridade (Seção 5.5). A mensagem é roteada até a Base de sua AI. No campo Uso Múltiplo do pacote são inseridos os valores da posição física atual do nó, de acordo com a diferença entre os valores da posição do nó e da Base (mostrado na Seção 4.1.2). Se SEQ_V=00, os nós que recebem o pacote verificam se é de um vizinho registrado e armazenam o valor da posição física atual do nó emissor em sua tabela Vizinhos. Em seguida, repassam o pacote com SEQ_V=SEQ_V+1.

h. Busca Vizinhos (BVZ)

O nó é definido SOZINHO quando está sem nenhum outro nó a seu alcance, isto é, VZ = { }. Tal situação ocorre quando o nó não recebe qualquer pacote dos seus vizinhos durante um certo Δt. Então, sua tabela Vizinhos é esvaziada, e o nó aguardará receber novas de mensagens. Se até um certo Δt receber alguma mensagem de vizinhos, armazena em sua

tabela o nó vizinho emissor. Se após Δt continuar com VZ = { }, o nó envia uma mensagem Busca Vizinhos. Esta mensagem é importante pois o nó pode estar rodeado de vizinhos de baixa prioridade, que só enviam mensagens quando realizada uma Busca sob-demanda (ver Seção 5.6.4). A BVZ é recebida como a AVZ, com a diferença que os nós vizinhos devem responder com uma AVZ.

i. Entre Vizinhos (EVZ)

São mensagens enviadas somente entre vizinhos do alcance de um certo nó. O nó receptor não repassa para outros nós. Funciona como um *broadcast* restrito aos vizinhos. Esta mensagem é util para que os nós atualizem seus vizinhos em relação à seus serviços disponíveis ou para troca de mensagens com limitação de saltos.

j. Entre Bordas (EBD)

As mensagens são trocadas entre os nós Borda, tanto entre nós de mesma e de diferentes AI's, para uso na geração de rotas. Os outros nós que recebem as mensagens e não são do tipo Borda, descartam-nas.

k. *Geocasting* (GEO)

As mensagens são enviadas para aviso de todos os nós dentro de uma AI. Os nós que recebem as mensagens verificam no campo Uso Múltiplo qual a AI alvo do pacote. Se forem da AI indicada, recebem o pacote. Caso contrário, descartam-no.

l. Gera Busca na mesma AI (GBM)

Para uso do SLF na busca de um determinado nó, os nós que recebem o pacote o roteiam até a Base, visando a verificação no RM. Caso algum nó no caminho possua a localização do nó procurado na tabela Vizinhos, retorna a localização do nó alvo para o nó solicitante.

m. Gera Busca em outra AI (GBO)

A mensagem é utilizada para procurar um nó localizado em outra AI. Inserindo a identificação do nó e sua AI no campo Uso Múltiplo, a mensagem é enviada para os nós Borda, que se comunicam com as outras AI's, e o pacote é enviado até o RM da AI alvo. O procedimento para a comunicação entre o nó inicial e um nó de outra AI é o mesmo descrito na Seção 4.1.8.1, com a diferença de não gerar a rota, mas somente a transferência da informação de localização física.

Para facilitar a visualização do uso dessas mensagens pelo protocolo, um modelo básico de estados e os algotrimos de uso das mensagens serão mostrados na Seção 4.1.8.

4.1.6 Geração de Rotas dentro de uma AI

O funcionamento da geração de rotas no X-AODV é basicamente o mesmo do protocolo AODV, porém, diferentemente do protocolo original, utiliza informações de localização física. Além disso, no momento da geração de rota, o momento do restorno dos dados da rota gerada também inclui a aquisição dos valores de Classe dos Dispositivos de cada nó.

No momento da criação de enlace*s*, pelo algoritmo do AODV, a busca pelo nó destino gera tráfego por todas as direções, já que não é possível saber em qual direção o nó alvo se encontra. Utilizando as informações de contexto de localização física presentes em todos os nós, através da política de atualização dos vizinhos, é possível localizar os nós alvos com menor custo de roteamento. A Figura 4.7 mostra um cenário de funcionamento do AODV, onde a busca é realizada em todos os nós, pois o alvo tem localização desconhecida.

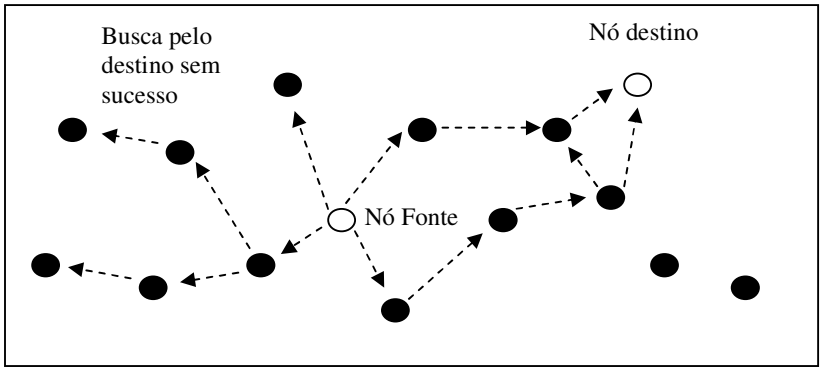


Figura 4.7: Geração inicial de enlace*s* no AODV

No X-AODV, no momento em que um nó gera a solicitação de rota para um nó alvo, com o conhecimento da posição física de seus vizinhos e do alvo, o nó realiza a geração de rotas enviando no seu pacote o endereço físico do nó alvo, e os nós vizinhos continuam o processo de gerar a rota caso estejam na mesma direção do nó alvo, possuam o endereço do alvo em sua tabelaVizinhos ou uma rota para o alvo computada previamente (Figura 4.8).

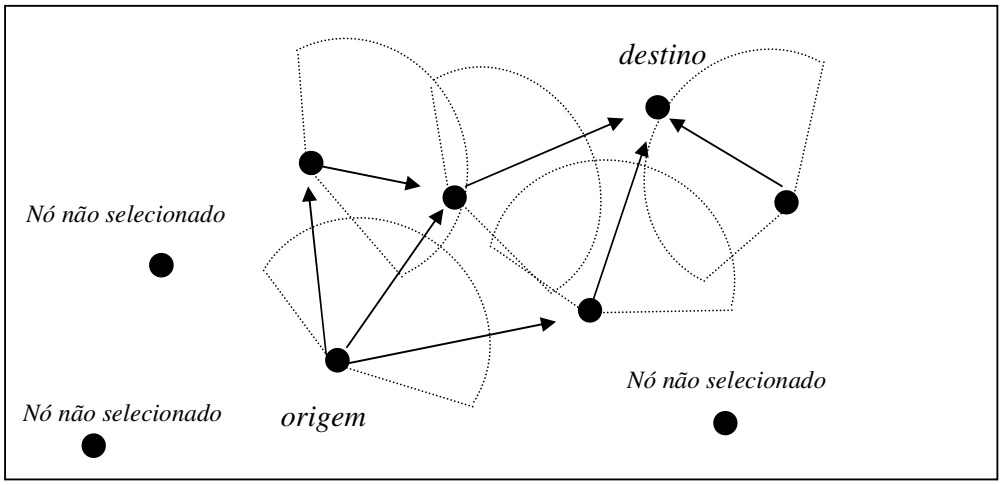


Figura 4.8: Seleção dos Vizinhos para geração de rota

A Figura mostra o nó de origem só enviando pacotes RREQ para os vizinhos que estão na direção do nó alvo, verificando os Vizinhos com relação à sua posição física relativa ao nó destino. Cada nó que recebe os pacotes realiza a mesma seleção. Os vizinhos que estão fora do ângulo pré-definido de busca não são selecionados para a geração da rota.

O diagrama de sequência (Figura 4.9) mostra a a troca de mensagens para a geração da rota no X-AODV.

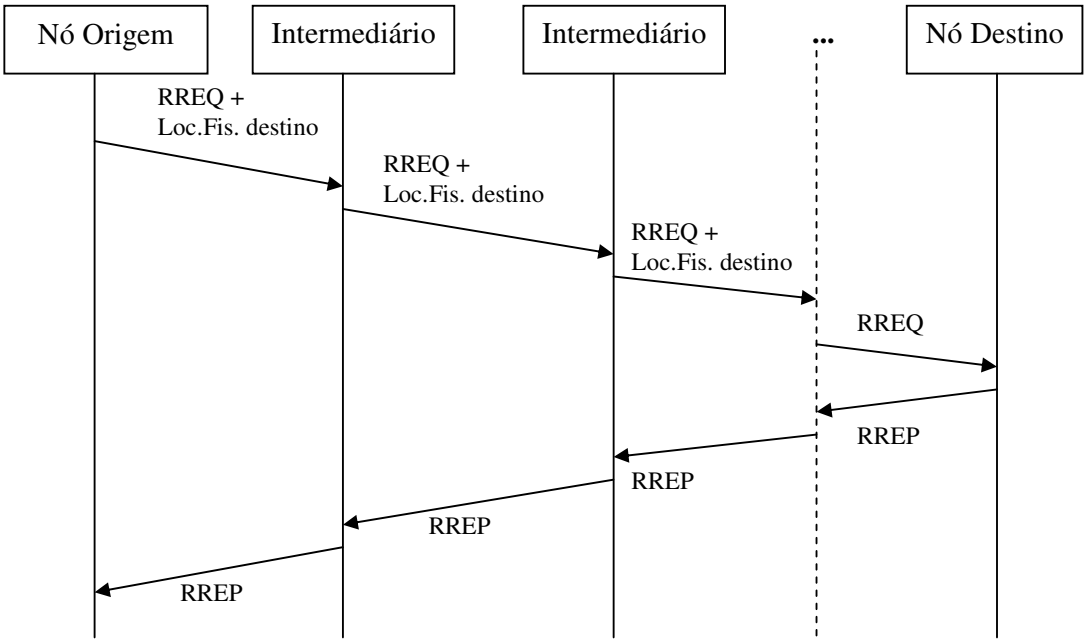


Figura 4.9: Sequência de envio de mensagens

Após o processo de geração de uma possível rota, o nó inicial recebe as rotas possíveis entre ele e o destino. A escolha da rota mais adequada considera os parâmetros de Classe de Dispositivos (CD) dos nós pertencentes às rotas possíveis.

4.1.7 Geração de Rotas entre diferentes AI's

A comunicação entre duas AI's diferentes pode ser necessária quando o trabalhador de uma área de contexto necessita trocar dados com outro trabalhador, ou até mesmo com a Central (Seção 4.1.1). Considera-se a Central como uma AI. Como os participantes de uma AI não sabem quais participantes de outra AI estão ativos ou não, ou se eles mudaram de AI, o SLF deve ser acionado para localizar os Clientes alvo. Quando um nó *i* da AI_n deseja realizar uma comunicação com um nó *j* de AI_m desconhecida, AI_n deve acionar as outras AI's do ambiente de trabalho de campo para localizar o Cliente *j*.

O conhecimento das AI's presentes é responsabilidade dos nós Borda. Cada Borda mantém o registro de quais outros nós Borda estão a seu alcance (Seção 4.1.2) e armazena os dados dos outros Bordas: suas respectivas AI's, quais outras AI's estão a seu alcance direto e quais AI's outros nós borda de sua AI estão em alcance.

Um exemplo é mostrado na Figura 4.10. O nó Borda B_1 registra dados dos nós B_2 (pertencente a AI_2) e B_3 (pertencente a AI_3, contato direto com AI_2 e AI_4, através de B_4). Portanto, B_1 sabe que pode alcançar AI_2, AI_3 e AI_4 e pode escolher qual caminho deve seguir até chegar ao alvo.

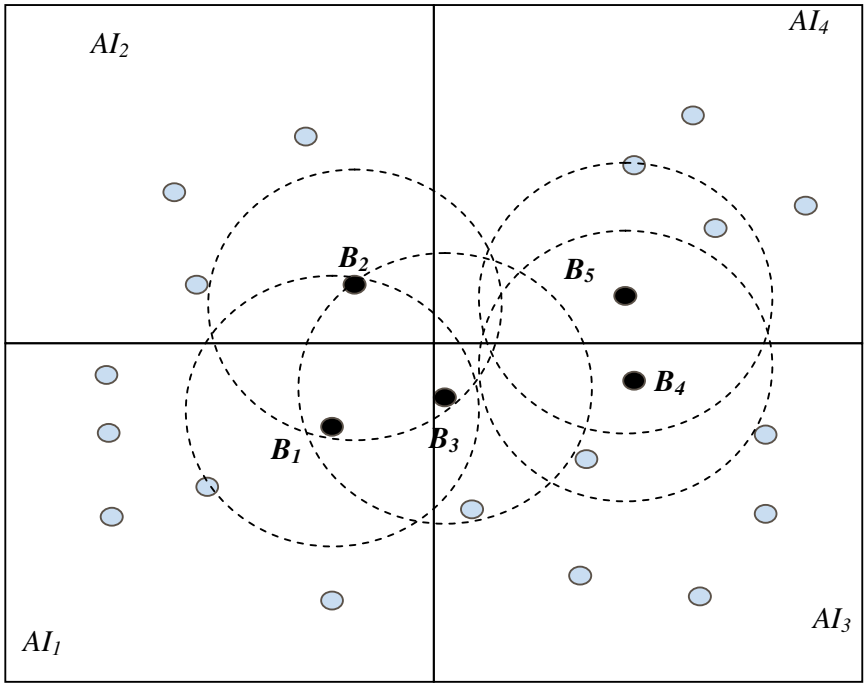


Figura 4.10: Exemplo de conhecimento dos nós Borda

O X-AODV roteia os dados dentro de uma AI utilizando a geração padrão de rotas por contexto, por exemplo, entre um nó qualquer e um nó Borda. A troca de mensagens entre dois nós Borda utiliza mensagens do tipo EBD (Entre Bordas) para permitir a escolha da Borda correta pelo nó Borda da AI original. Entre o novo nó Borda e o nó alvo, a geração da rota é novamente realizada da forma padrão.

Os nós Bordas mantêm então as tabela de rotas entre as AI's. Quando um Borda perde contato com outro nó Borda, deve atualizar outros nós Borda de sua AI, que por sua vez, atualizam os Bordas em contato. Como visto na Seção 4.1.5, para manter as informações entre as AI's, o X-AODV aproveita os pacotes que são gerados nos momentos da geração de rotas e atualização para divulgar os dados de vizinhos. Portanto, os nós Borda também utilizam as tabelas Vizinhos para manter os dados das AI's vizinhas, através do conhecimento dos outros nós Borda.

Este procedimento é recomendável para ambientes de trabalho de campo que não possuam mais do que 2 saltos de AI's, isto é, onde um Borda consegue alcançar uma AI de forma indireta passando por até uma AI intermediária. Para manter a escalabilidade para 3 ou mais saltos, deve-se utilizar uma hierarquia de roteamento entre as AI's.

Portanto, para providenciar a comunicação entre os nós *i* e *j*, já sabendo à qual AI *j* pertence, deve-se seguir este procedimento:

a. Repassar o pedido de Localização e Rota para os nós Borda (B) da AI_n atual;
b. B_n verifica em sua tabela de Bordas qual outro nó Borda pertence a AI_k desejada ou possui contato;
c. Envia para o Borda B_k indicado;
d. O Borda B_k repassa os pacotes para o RM da Base da AI_k;
e. No caminho entre B_k e a Base, os nós que repassam os pacotes verificam o tipo da mensagem e podem conferir se possuem em sua tabela Vizinhos comunicação com o nó *j*. Caso um nó *l* seja vizinho de *j*, repassa o pacote para *j*;
f. *j* sinaliza de volta para B_k sua rota.
g. B_k repassa rota para B_n, que por sua vez repassa rota para o nó *i*.

Em outra possibilidade, para providenciar a comunicação entre os nós *i* e *j*, sabendo que *j* não pertence a AI_n atual (de acordo com o algoritmo de Busca sob-demanda) e não sabendo à qual AI *j* pertence, deve-se seguir este procedimento:

a. Repassar o pedido de Localização e Rota para os nós Borda (B_n) da AI_n atual;
b. Cada B_n deve repassar o pedido para outros nós Borda B_k das AI_k com quem tenha contato de alcance;

c. Cada Borda B_k repassa os pacotes para o RM da Base da AI_k;
d. No caminho entre B_k e a Base, os nós que repassam os pacotes verificam o tipo da mensagem e podem conferir se possuem em sua tabela Vizinhos comunicação com o nó *j*. Caso um nó *l* seja vizinho de *j*, repassa o pacote para *j*;
e. *j* sinaliza de volta para B_k sua rota.
f. B_k repassa rota para B_n, que por sua vez repassa rota para o nó *i*.

Para os dois casos, se o nó procurado não for encontrado, o protocolo retorna uma resposta de erro, como no protocolo original.

4.1.8 Algoritmos X-AODV

O funcionamento do protocolo segue alguns algoritmos: Recebimento de mensagens, Roteamento na mesma AI, Entre AI's e Localiza. Cada algoritmo é acionado dependendo da funcionalidade requerida pelo Cliente. A aplicação indica o procedimento que quer realizar à camada de roteamento dependendo do Tipo de Mensagem (Seção 4.1.5) indicado. Um diagrama de estados é apresentado para melhor visualização do algoritmo de troca de mensagens.

4.1.8.1 Algoritmos de Roteamento

Os algoritmos apresentados nesta Seção são divididos em três partes. O algoritmo ROTEAMENTO_MESMA_AI mostra como funciona a geração de rota quando o nó fonte e destino pertencem à mesma área. Este algoritmo chama uma função LOCALIZA, que é responsável pela busca de um nó dentro da AI.

Quando o nó destino pertence à outra AI, é ativado o algoritmo ROTEAMENTO_OUTRA_AI, que faz a comunicação entre os nós Borda. Este algoritmo também utiliza a função LOCALIZA, pois a geração de rotas entre dois Bordas, ou entre um Borda e o nó alvo, precisa também localizar um certo nó dentro da mesma AI. No retorno da solicitação de rota é utilizado o esquema descrito em 4.1.6.

Em cada algoritmo estão comentrários sobre seu funcionamento.

```
ROTEAMENTO_MESMA_AI (nó) {
// Chama função Localiza(nó), que seleciona os nós para repassar o pedido de rota.
        LOCALIZA(nó j);
    }
```

```
LOCALIZA (nó){
        Enquanto (nó_atual != j) faça {
// No caminho entre um nó i e um nó j, os nós que repassam o pacotes verificam o tipo da
// mensagem e podem conferir se possuem em sua tabela Vizinhos comunicação com o nó j.
// Caso um nó l seja vizinho de j, repassa o pacote para j.
                Se SEQ_V já recebido
                        Descarta pacote;
//O Número de sequência evita realizar o mesmo procedimento mais de uma vez
                Se j ∈ VIZINHOS {
                        Repassa pacote para j;
                }
                Senão {
                        SEQ_V = SEQ_V+1
                        Repassa pacote para todos seus vizinhos;
                }
        }
}

ROTEAMENTO_ENTRE_AIs (nó j) {
        Enquanto (nó atual != j) faça {
// Repassa o pedido de Localização para os nós Borda (B_n) da AI_n atual;
                LOCALIZA(B_n);
                ROTEAMENTO_MESMA_AI (nó B_n);
// Cada B_n deve repassar o pedido para outros nós Borda B_k das AI_k com quem tenha contato
de alcance;
                Envia mensagem EBD;
// EBD = Entre Bordas
// Cada Borda B_k repassa o pacotes para o RM da Base_k;
                ROTEAMENTO_MESMA_AI (entre B_k e Base_k);
                B_k recebe rota entre B_k e j;
                B_k repassa rota para B_n, que por sua vez repassa rota para o nó i.
//Rota gerada:  i - B_n, B_n – B_k e B_k – j;
```

Como exemplo do uso dos algoritmos, o caso da geração de rotas entre o nó 1 da AI_1 e nó 10 da AI_4, mostrado na Figura 4.11.

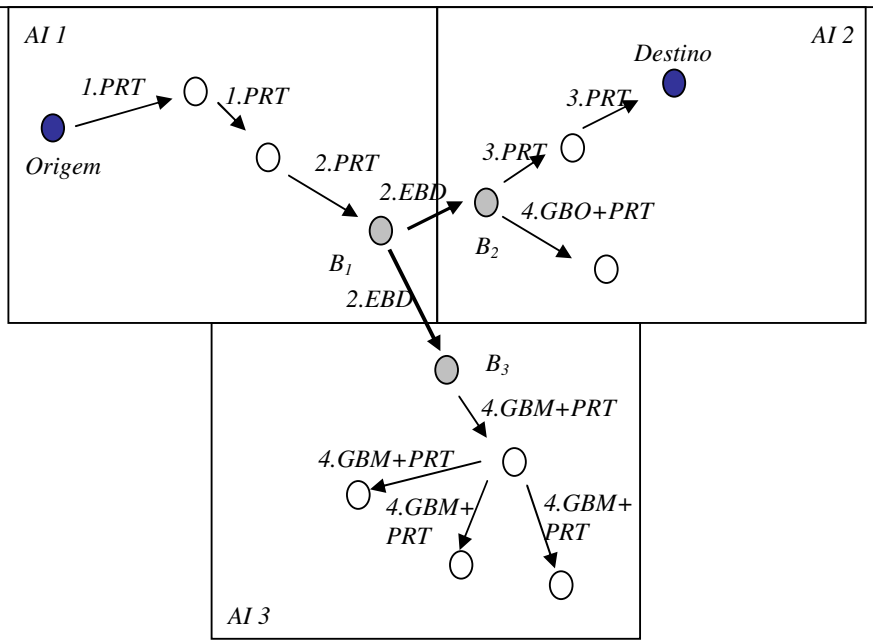


Figura 4.11: Exemplo de roteamento do X-AODV

Na Figura, cada fase é representada por um número antes de cada nome de mensagem. Quando o nó Origem solicitou uma rota a um nó Destino (sabendo à qual AI ele pertencia), ele enviou uma mensagem PRT para o nó Borda B_1 (fase 1), contendo a AI à qual Destino pertencia e a indetificação do nó, no campo Uso Múltiplo. Quando o nó borda B_1 recebeu a mensagem, verificou que deveria repassar para o no borda da AI_1, enviando então uma mensagem EBD para B_2 (fase 2). O nó B_2 solicitou então uma rota para o nó Destino, enviando uma mensagem PRT de pedido de rota (fase 3). Quando o Destino recebeu a mensagem, realizou a volta do pedido de rota para o nó B_2. B_2 então retornou para B_1, que por sua vez retornou para o nó Origem.

Caso o nó Origem não soubesse à qual AI destino pertencia, no momento da fase 2, de realizar a troca de mensagens EBD, todos os nó borda vizinhos de B_1 receberiam a mensagem contendo a identificação do nó Destino. Cada Borda deveria então realizar a busca do nó, usando a mensagem GBM (Gera Busca na Mesma AI, fase 4), repassando com um pedido de rota para o no Destino. Quando o nó fosse encontrado, retornaria a volta do pedido de rota, como descrito no caso anterior.

4.1.8.2 Mensagens Recebidas

Quando um nó recebe uma mensagem, ele verifica cada pacote para ver o campo Tipo da mensagem. De acordo com o tipo, o nó realiza diferentes procedimentos. Resumindo a

Seção 4.1.5, onde foram mostradas as mensagens do X-AODV, o algoritmo abaixo mostra o que acontece de acordo com cada tipo de mensagem.

```
Caso (Tipo de Mensagem)
        PEDIDO DE ROTA:
                Repassa para próximo nó da Rota;
        VOLTA DE PEDIDO DE ROTA:
                Armazena seu dado de Ci no campo Uso Múltiplo;
                Repassa para próximo nó da Rota;
        ENVIO DE DADOS:
                SE SEQ_V = 00
                        Armazena dados de ILF do vizinho;
                SENÃO Descarta;
                Repassa para próximo nó da Rota;
        ATUALIZA VIZINHO:
                SE SEQ_V = 00
                        armazena dado do vizinho;
                SENÃO descarta;
                Repassa para próximo nó da Rota;
        PEDIDO DE LISTAGEM DE SERVIÇO
                Repassa para próximo nó da Rota;
        PEDIDO DE SERVIÇO ESPECÍFICO
                Verifica serviços na tabela Vizinhos;
                SE achar serviço solicitado, repassa para o vizinho o pedido de
serviço;
                SENÃO
                        Repassa para próximo nó da Rota;
        ATUALIZA POSIÇÃO FÍSICA
                SE SEQ_V = 00
                        armazena dados de ILF do vizinho;
                SENÃO descarta;
                Repassa para próximo nó da Rota;
        GEOCASTING {
                Verifica AI do emissor inicial do pacote;
                SE AI do emissor = AI atual e SEQ_V ainda não recebido,
                        Abre mensagem;
                SENÃO
                        Descarta;
```

Repassa para Vizinhos; }

SEQ_V = SEQ_V+1.

O diagrama de estados (ANEXO A) da Figura 4.12 representa os estados do protocolo em relação ao uso das novas mensagens criadas. O tipo das mensagens recebidas são representadas por suas siglas, que foram apresentadas na Seção 4.1.5.

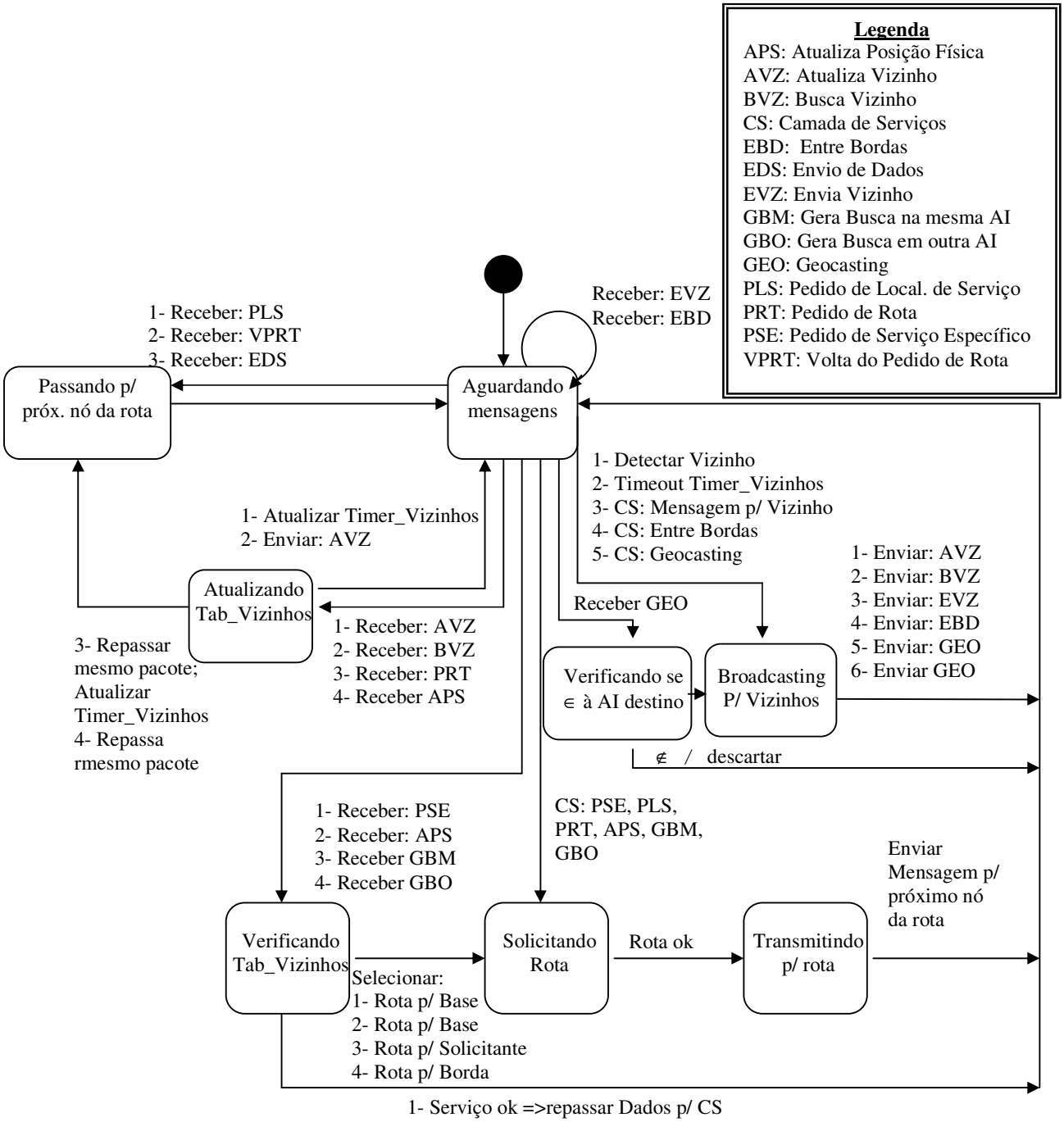


Figura 4.12: Diagrama de Estados simples

O diagrama mostra a sequência de estados que um nó passar ao receber cada tipo de mensagem. Inicialmente, fica no estado AguardandoMensagens. No momento que uma mensagem chega, o nó analisa quem tipo de mensagem foi recebido, para tomar decisões sobre o que fazer. Na figura, como cada estado pode receber diversos tipos de mensagens que mudarão seu estado atual para o próximo estado, as transições de saída do estado estão numeradas com o mesmo número das condições de entrada.

Ao receber uma mensagem, o estado pode mudar para Passando para o próximo nó da rota (onde apenas repassa os pacotes), Atualizando Tab-Vizinhos (onde atualiza informações de localização física dos vizinhos para depois ir ao estado onde repassará os pacotes), Verificando Tab_Vizinhos (onde deve procurar por informações de serviços de vizinhos e localização) e Verificando se $\in$ à AI destino (onde pode gerar um *broadcast* para os vizinhos ou descartar à mensagem, se não pertencer à área de *geocasting*). As operações que cada mensagem solicita ao protocolo estão descritas em 4.1.5 e no algoritmo desta Seção.

4.2 Conclusões

Este Capítulo apresentou o protocolo de roteamento X-AODV. O protocolo usa informações de contexto de dispositivos (Classe de Dispositivo e localização física) para a criação de rotas entre fonte e destino das aplicações. A coleta de dados de perfis dos dispositivos permite a seleção de rotas com a presença de menos dispositivos de baixa capacidade computacional. Como será visto na Avaliação de Desempenho, o dispositivo de baixa capacidade é prejudicado pelo tráfego que ele necessita repassar para seus nós vizinhos. A escolha de outras rotas possíveis per mite tentar utilizar rotas com o maior número possível de dispositivos de maior capacidade computacional.

O aproveitamento das mensagens que trafegam na rede mantém cada nó consciente de seus vizinhos, o que facilita a busca por outros nós e serviços disponíveis no ambiente. O uso da localização física no roteamento permite que rotas mais otimizadas sejam criadas, já que o pré-conhecimento da localização dos nós da vizinhança permite o direcionamento da rota para o local alvo.

O próximo Capítulo detalha a Arquitetura de Computação Pervasiva (APTC), projetada para melhorar o trabalho de campo, ao oferecer novas formas de realizar os serviços tradicionais e gerar também novos paradigmas.

Capítulo 5 – Arquitetura Pervasiva para Trabalho de Campo

5.1 Introdução

Como visto na Seção 1, houve a necessidade de serem criados novos serviços para fornecer um ambiente adequado para trabalho de campo. Esses novos serviços foram agrupados em uma camada de serviços, para permitir que as aplicações e usuários, separados em uma camada superior, comuniquem-se abstraindo os aspectos de rede de comunicação, usufruindo as vantagens do uso dos novos serviços.

A **Arquitetura Pervasiva para Trabalho de Campo (APTC)** é dividida em três camadas, cada qual com suas funções, entidades e algoritmos. A camada superior de aplicação inclui usuários, dispositivos e aplicações. Nela, os aspectos de rede são transparentes. A camada intermediária é responsável por oferecer os serviços da arquitetura. A camada inferior, de rede, engloba todas as redes de comunicação do ambiente. Uma visão geral da arquitetura é mostrada na Figura 5.1.

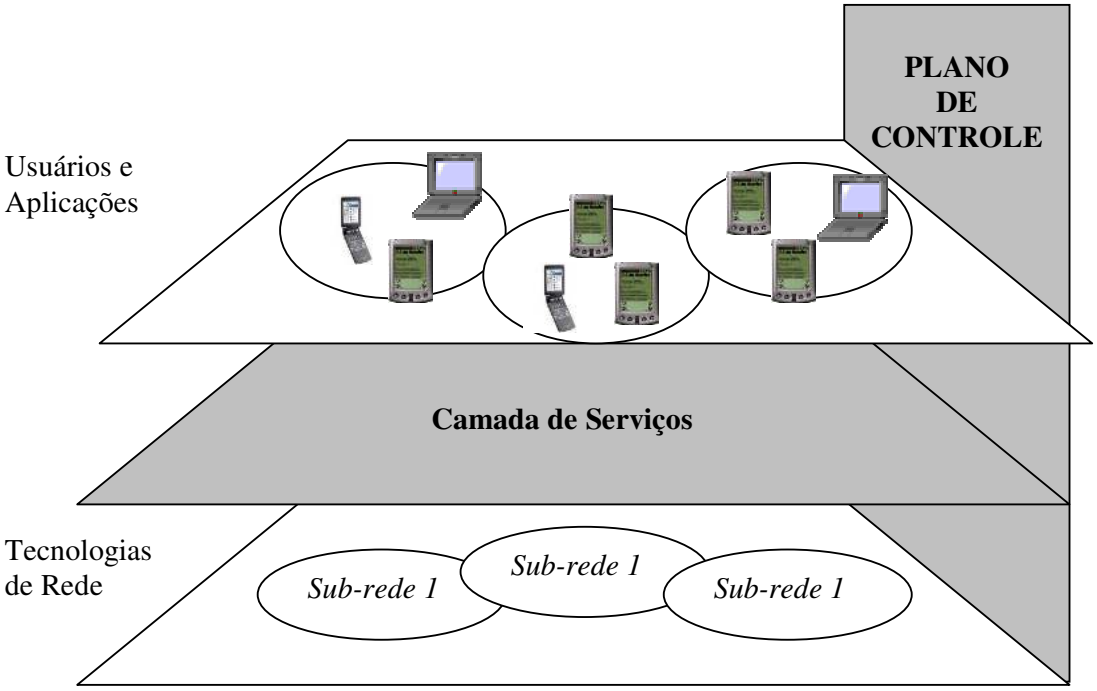


Figura 5.1: Visão geral da arquitetura

As entidades presentes do ambiente de computação pervasiva para trabalho de campo são:

- **Usuários**

São as pessoas físicas que trafegam pelo ambiente, portando ou não algum dispositivo computacional. Considera-se, neste trabalho, que cada usuário possui um módulo GPS que transmite seus dados de localização.

- **Dispositivos**

A classe de dispositivos engloba desde dispositivos computacionais, como computadores portáteis e de mão , como outros dispositivos de apoio, todos com acesso à rede de comunicação. Os dispositivos são classificados em:

Móveis: são os dispositivos que tem acesso à comunicação e informação, mesmo em movimento, como os computadores portáteis, computadores de mão e celulares. São os dispositivos mais comuns na área sem infra-estrutura do trabalho de campo.

Fixos: estão conectados à rede de comunicação, mas permanecem geralmente fixos na mesma posição física. São mais comuns na área com infra-estrutura, fora da área do trabalho de campo.

- **Aplicações**

As aplicações são oferecidas aos usuários e aos dispositivos do ambiente. Para esta arquitetura, são consideradas aplicações aqueles *softwares* que utilizam os serviços disponibilizados pela arquitetura para seu próprio funcionamento.

- **Provedores de Serviços**

Os nós que disponibilizam aplicações para os usuários são chamados de Provedores de Serviços (PS). Sua arquitetura é a mesma de um nó Comum, com a diferença que podem oferecer serviços, como por exemplo, fornecer uma imagem através de sua câmera acoplada. Na Seção 5.8.1 será mostrado se oferecem serviços.

As próximas seções detalham as camadas da arquitetura.

5.2 Camada de Clientes e Aplicação

As entidades que estão presentes no ambiente são as aplicações, os usuários e os dispositivos (sendo os dois últimos englobados em uma classe chamada de Clientes). A Figura 5.2 exemplifica como aplicação e clientes interagem com a camada de serviços.

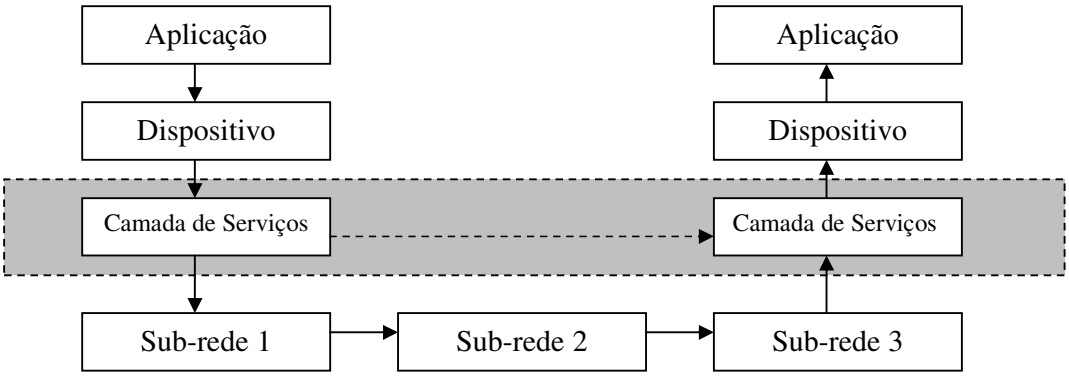


Figura 5.2: Interação da arquitetura

A interação entre as entidades é realizada através da camada de serviços. Aplicação e Clientes não estão cientes da infra-estrutura, apenas se comunicam com a camada intermediária, que decide qual serviço chamar ou apenas realiza o roteamento das informações entre fonte e destino. Os aspectos de redes de comunicação são transparentes, pois o serviço de roteamento é realizado pela camada intermediária.

5.2.1 Integração no Ambiente

As entidades integram-se no ambiente de diferentes formas. Os Clientes (usuários e dispositivos) devem se registrar no ambiente através de armazenamento de perfil e atualização de posição física, enquanto que as aplicações se integram no ambiente registrando-se nos Diretórios de Aplicações, localizados nos servidores Base.

- Integração de Clientes: A integração ao ambiente ocorre na seguinte seqüência:
    1. Detecção da chegada do Cliente no ambiente pelos nós vizinhos;
    2. Registro do Cliente - Armazenamento de Perfil; e
    3. Atualização de sua posição física.
- Integração de Aplicações: uma aplicação se torna disponível no ambiente depois que se registra no Diretório de Aplicações, localizado no nó Base de sua Área de Interesse.

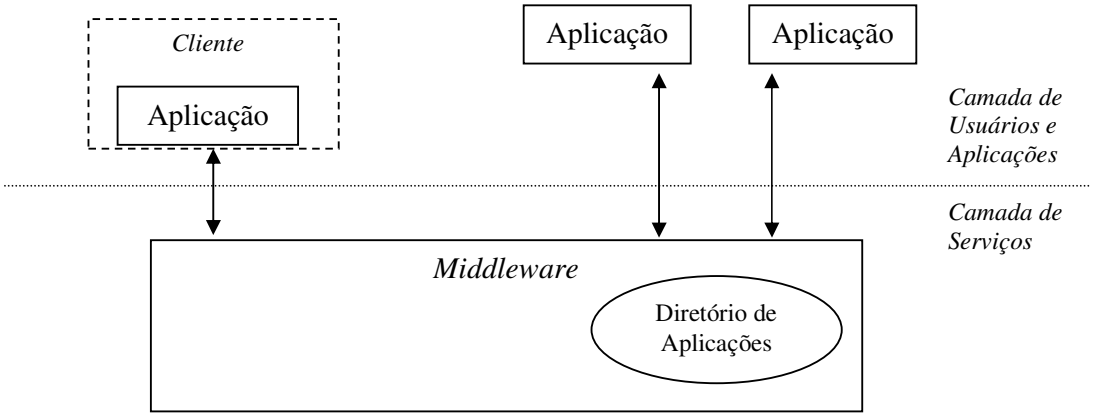


Figura 5.3: Integração no ambiente

Como mostra a Figura 5.3, o Diretório de Aplicações (DAP) é o responsável por aceitar e manter o registro das aplicações oferecidas no ambiente. Através da camada de serviços, as Aplicações se publicam no Diretório e oferecem seus serviços ao ambiente. Detalhes sobre o diretório e oferecimento de serviços serão mostrados na Seção 5.8.

5.3 Camada de Serviços

Esta camada abstrai o nível inferior de infra-estrutura e provê serviços de alto nível para a camada superior. Esta camada de interligação entre entidades (Clientes, aplicações, serviços e repositórios de dados) é composta de:

- Serviço de Localização Física (SLF)
- Serviço de Descoberta de Serviços (SDS)
- Serviço de Adaptação de Conteúdo (SAC)
- Repositórios de Perfis (RP) e Mobilidade (RM)
- Diretório de Aplicações (DAP)
- Protocolo de Roteamento por Contexto (X-AODV)

Os serviços serão mostrados em detalhes nas próximas seções.

A Figura 5.4 mostra a relação entre serviços e camadas.

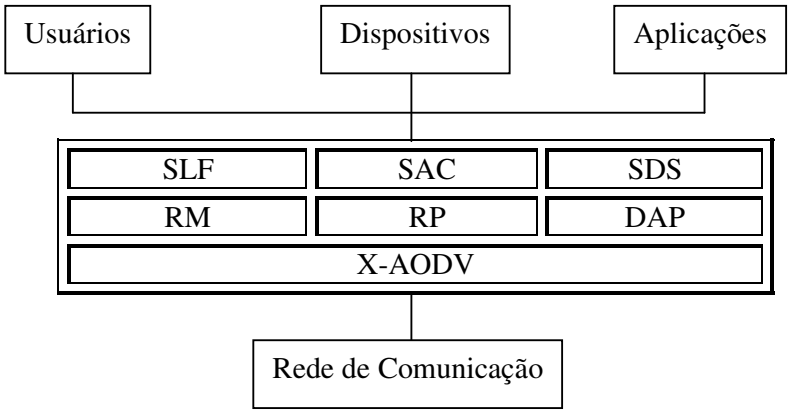


Figura 5.4: Serviços e camadas

Na arquitetura apresentada neste trabalho, a camada de serviços tem como funções:

1. Admitir Clientes no sistema;
2. Dar suporte à mobilidade de usuários e dispositivos;
3. Realizar publicação de aplicações;
4. Gerenciar roteamento de informações por contexto; e
5. Localizar e oferecer aplicações.

Os servidores Base e os Clientes possuem configurações diferentes da camada de serviços. Enquanto os servidores precisam manipular os repositórios de dados e atender os Clientes com os serviços, os Clientes possuem tarefas mais leves para se manterem no ambiente de trabalho.

A Figura 5.5 mostra a camada de serviços nos servidores. Além dos serviços e protocolo citados anteriormente, nos servidores existe um módulo CONTROLE do tipo *cross-layer*, responsável por verificar os pacotes manipulados pelo protocolo de roteamento e gerenciar a troca de dados entre o roteamento e os serviços. Como foi visto anteriormente, o roteamento utiliza diversos tipos de mensagens para otimizar a troca de dados entre os participantes do ambiente de trabalho de campo.

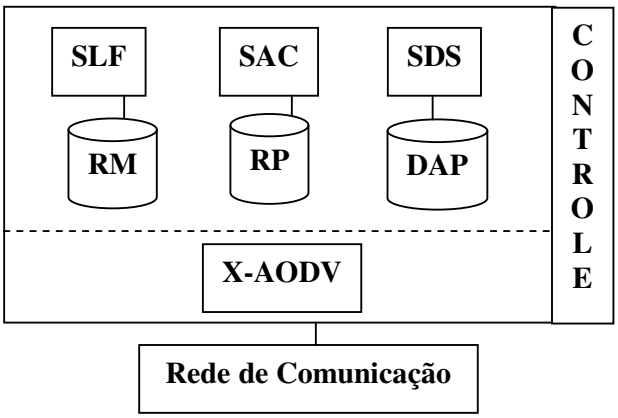


Figura 5.5: *Middleware* nos servidores Base

Nos servidores Base de cada AI estão localizados os repositórios de dados (RM e RP) daquela área. Os Clientes localizados na mesma AI utilizam os serviços oferecidos por seu servidor. Nos Clientes é mantida uma versão mais simples dos servicos (Figura 5.6).

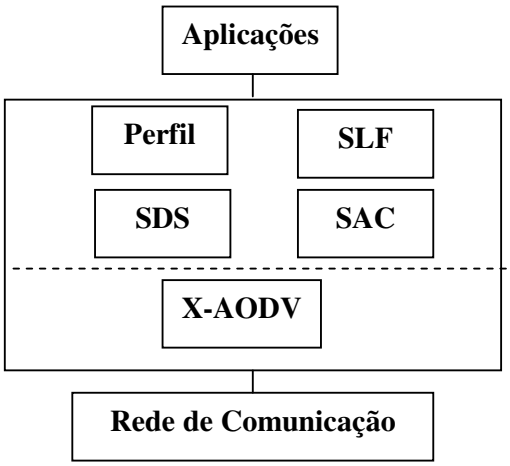


Figura 5.6: *Middleware* nos dispositivos dos usuários

Diferentemente da versao utilizada nas Bases, nos Clientes são armazenadas as definições de perfil de dispositivos (Seção 5.5), as regras de atualização de informações de localização física (Seção 5.6.3), o perfil dos servicos (seus e dos vizinhos), juntamente com o protocolo de roteamento da arquitetura.

5.3.1 Funcionamento da Arquitetura

Um cliense se registra no ambiente enviando primeiramente seus dados de localização fisica atual, seus dados de perfil e os serviços que quer oferecer (Figura 5.7).

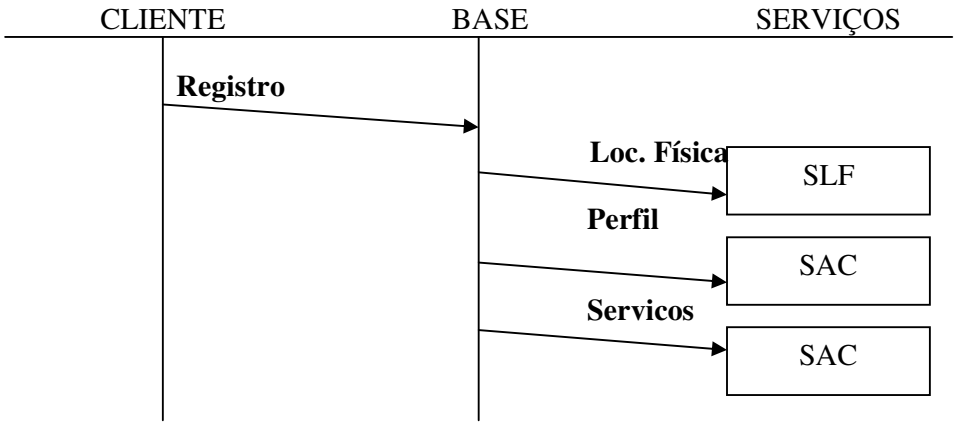


Figura 5.7: Registro do Cliente

A Base recebe as informações enviadas pelo Cliente e repassa para os serviços que mantém e que são responsáveis por cada tipo de informação.

5.4 A Camada de Tecnologias de Comunicação

Esta camada inclui as tecnologias de comunicação que podem estar presentes em um ambiente de computação pervasiva. Sistemas como redes Celulares, GPS, redes WLAN e *Bluetooth* podem integrar um ambiente de comunicação global quando os dispositivos possuem múltiplas interfaces de tecnologias, podendo comunicar-se em vários sistemas ao mesmo tempo e/ou tempos separados.

5.4.1 Tipos de Tecnologias

São consideradas entidades as diferentes tecnologias de comunicação e posicionamento que integram esta arquitetura.

- Comunicação: Bluetooth, Redes sem fio 802.11, Ethernet.
- Posicionamento: GPS.

Cada tecnologia de comunicação possui características próprias de transmissão, variando parâmetros de largura de banda e alcance.

As tecnologias têm interface com a camada de serviços, que se torna responsável pelo fluxo de mensagens e escolha da melhor rota.

Para o ambiente de trabalho dar suporte às diversas tecnologias, o nó Base, que mantém os respositórios (de Mobilidade e de Perfis) deve possuir interfaces aéreas de múltiplas tecnologias para poder trocar dados entre dispositivos heterogêneos. A rede IP permite troca de dados de forma transparente à camada de serviços.

O uso de dados de localização física no formato latitude/longitude (Seção 4.1.2) permite que o Repositório de Mobilidade mantenha informações sobre os dispositivos de todas as interfaces, supondo-se que eles possuem GPS acoplado. Isso permite o ambiente conheer a posição física de todos os nó, independentemente da tecnologia de comunicação.

5.5 Utilização de Perfis

Na camada de serviços apresentada, a utilização de Perfis permite abstrair componentes do ambiente para os desenvolvedores de aplicações. Cada Cliente possui uma classificação para Classe de Dispositivo, Classe de Prioridade e Classe de Mobilidade.

A Classe de Dispositivo parametriza o tipo de dispositivo e a qualidade de comunicação. A Classe 1 engloba dispositivos com alta capacidade computacional, como computadores portáteis e *desktops*. A Classe 2 inclui dispositivos com menor capacidade de processamento, como computadores de mão , e a Classe 3 inclui dispositivos de baixo poder computacional (ex: telefone celular e computadores de mão de menor capacidade).

A Classe de Prioridade é configurada pelos administradores dos ambientes de trabalho e permite realizar transmissões de dados com diferentes níveis de prioridades: pode ser alta (A), média (M) ou baixa (B).

A Classe de Mobilidade depende do comportamento de movimentação dos dispositivos pela área de trabalho: pode ser alta (A), média (M) ou baixa (B). A descrição com mais detalhes está na Seção 5.5 deste trabalho.

Tais informações sobre os clientes são armazenadas no Repositório de Perfis (RP). O repositório pode também ser utilizado para Autenticação de Clientes. A autenticação pode ser realizada ao só permitir que Clientes previamente cadastrados no sistema tivessem acesso aos serviços da Arquitetura, aumentando a segurança dos dados.

5.5.1 Formato dos dados armazenados no RP

Os dados são diferentes para usuário, dispositivo e mapeamento de informações de usuários para vários dispositivos. A Tabela 5-1 mostra osdados armazenados no RP.

Tabela 5-1: Formato de dados no RP

Cliente	Classe de Dispositivo	Classe de Prioridade	Classe de Mobilidade
Usuário	---	A, M ou B	A, M ou B
Dispositivo	1, 2 ou 3	A, M ou B	A, M ou B

Na primeira utilização do sistema, os dados dos Clientes são inseridos no RP. Para usuários, são configuradas suas classes de Prioridade e geradas suas identificações. Para dispositivos, é gerada a identificação e os dados de *hardware* detectados e armazenados.

5.6 Serviço de Localização Física (SLF)

Cada servidor Base de uma AI é responsável por manter o SLF. O servidor recebe dados de localização física dos nós pertencentes à área conforme uma Estratégia Híbrida de Atualização de Informações, detalhada na Seção 5.6.3.

Esta Seção mostra a estrutura do SLF e os algoritmos desse serviço.

5.6.1 Estrutura do SLF

Nos servidores localizados nas Bases, o SLF é composto de alguns módulos internos, mostrados na Figura 5.8, onde cada um possui funções específicas.

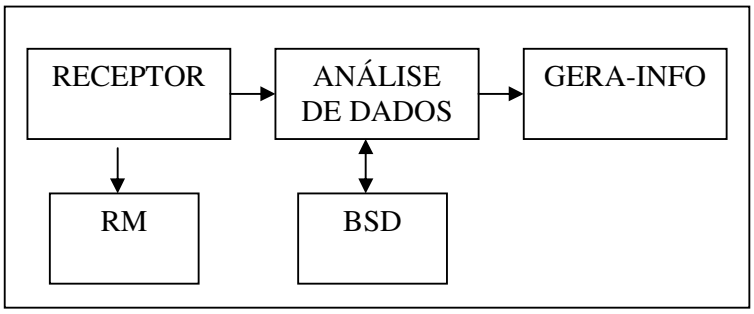


Figura 5.8: Módulos internos do SLF

O módulo RECEPTOR é responsável por receber as informações que chegam e interpretá-las. Quando recebe dados de um nó com sua atualização de localização, armazena-a no Repositório de Mobilidade (RM), conforme formato descrito na próxima sub-Seção.

Caso seja um pedido de informação de localização de alguma entidade no ambiente, passa o pedido para ANÁLISE DE DADOS, que busca no RM os registros atuais das entidades. Se a entidade não está registrada, é ativado o BSD (Busca Sob-demanda, Seção 5.6.4). A análise gerada é enviada ao GERA-INFO, que formata as informações geradas. Por fim, a informação formatada é enviada para quem fez a solicitação inicial.

5.6.2 Repositório de Mobilidade

Em cada área de interesse, o nó Base se torna responsável por manter os dados de localização física dos outros nós, para posterior atualização da movimentação dos nós na área. Os nós dos grupos (Normais e Borda) geram seus dados de posição física e transmitem para o

nó Base de acordo com a freqüência definida no algoritmo de atualização de informações de localização física (Seção 5.6.3), como mostrado na Figura 5.9.

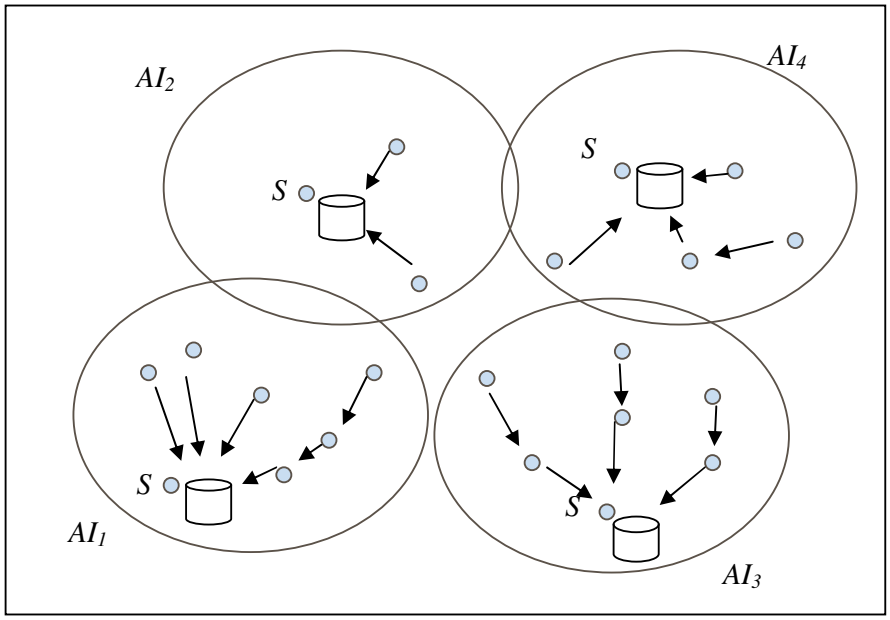


Figura 5.9: Atualização de dados de localização física nos RM

Os dados recebidos pelas Bases são inseridos no Repositório de Mobilidade (RM). Os dados armazenados no RM possuem o seguinte formato, mostrado na Tabela 5-2.

Tabela 5-2: Formato do RM

Cliente	Data	Hora	Localização
ID	dd/mm/aaaa	hh:mm:ss:msms	Latitude, longitude

Quando o RM é consultado pelo SLF, retorna a última posição conhecida do Cliente. O histórico de movimentação pode ser utilizado para previsão de movimentação dos nós pela área de trabalho.

5.6.3 Estratégia Híbrida de Atualização de Informações de Localização

A periodicidade da busca de dispositivos e usuários depende do tipo de tecnologia utilizada. Atualizações Periódicas são necessárias para usuários e dispositivos móveis com tendência de grande movimentação, pois existe a necessidade de manter a consistência dos registros de localização. Para dispositivos que não possuem muita mobilidade, a busca pode ser realizada quando houver demanda de informação de localização mais atual.

O uso de dados de localização e movimentação pode gerar informações de qual classe de mobilidade os Clientes pertencem. Os Clientes podem ser classificados quanto a sua mobilidade:

- Mobilidade Alta: são os Clientes que se movimentam com grande freqüência, por mais de um certo limiar ao dia. Por exemplo, trabalhadores equipados com computadores de mão, celular; caminhão interligado ao sistema.
- Mobilidade Média: são os Clientes que se movimentam com relativa freqüência, mas até um certo limiar ao dia. Por exemplo, trabalhador de escritório trabalhando com computador portátil.
- Mobilidade Baixa: são os Clientes que praticamente ficam imóveis. Por exemplo, computadores *desktop*.

Este trabalho propõe um método híbrido de atualização de informações de localização física. Como será mostrado no Capítulo 6, a variação das estratégias de atualização de dados pode diminuir os custos com o tráfego de dados. Esta estratégia classifica as entidades rastreadas como: Usuários simples (sem dispositivos), Dispositivos e UxD (cruzamento de dados do usuário e o(s) dispositivo(s) que está utilizando).

Um método de atualização é utilizado dependendo do tipo da entidade.

a. Usuários simples

Em áreas externas, com uso de Módulo GPS, utilizam-se Métodos Não Baseados em Infra-estrutura Independente (Seção 3.3.5.1), com necessidade de atualização periódica, com intervalo entre atualizações de Δt (valor que deve ser definido pelo gerente do sistema).

b. UxD

Quando se quer enviar dados para o dispositivo mais próximo de um determinado usuário, o serviço compara as informações de localização mais atuais do Cliente e localiza os dispositivos com maior proximidade.

c. Dispositivos

Através de experimentos de simulação (Seção 6.3), verificou-se que cada estratégia de atualização (Seção 3.3.5) tem suas vantagens e desvantagens em relação à comunicação no ambiente. A estratégia baseada em tempo tem maior granularidade, porém maior carga de dados, enquanto que a estratégia de Busca sob-demanda tem menor custo de rede, porém menor granularidade.

Esta variação do uso das estratégias de atualização podem gerar economias de tráfego na rede, ao mesmo tempo que mantém a consistência dos dados de localização. Dependendo da prioridade do Cliente pesquisado, deve-se usar uma estratégia que tem maior granularidade

para um Cliente mais importante, mas que tem maior custo (maior geração de tráfego). No caso de um Cliente de menor prioridade, pode-se utilizar uma estratégia que gere menor custo.

A escolha da estratégia de atualização de informações de localização física é realizada através do cruzamento dos dados de Mobilidade e Prioridade de cada nó, de acordo com a Tabela 5.3, gerada de acordo com os resultados obtidos nas simulações.

Tabela 5-3: Escolha de Taxa de Atualização de Localização Física

Classe de Mobilidade	Classe de Prioridade	Estratégia de Atualização
Alta	Alta	Baseada em Tempo
	Média	Baseada em Tempo
	Baixa	Baseada em Movimento
Média	Alta	Baseada em Tempo
	Média	Baseada em Movimento
	Baixa	Baseada em Movimento
Baixa	Alta	Baseada em Movimento
	Média	Baseada em Movimento
	Baixa	Nunca Atualiza

Para dispositivos com mobilidade alta (modo de prioridade alta ou média) e dispositivos de mobilidade média (modo de prioridade alta) será utilizada a **Estratégia Dinâmica Baseada em Tempo** (Seção 3.3.5.2): a posição de localização do dispositivo deve ser atualizada a cada *t* unidades de tempo, sendo *t* diferente para cada tipo (Atualização Periódica). Dependendo do nível de mobilidade, o intervalo de tempo (Δt) entre atualizações varia. Clientes de mobilidade alta (MA) necessitam de atualizações mais freqüentes do que Clientes de mobilidade média (MM); portanto, $\Delta t_{MA} < \Delta t_{MM}$. Tal estratégia é indicada para manter a consistência e a precisão dos dados de localização física dos nós que podem variar de posição em curtos espaços de tempo. A **Estratégia Dinâmica Baseada em Movimento** será utilizada nos casos de menor prioridade, quando o ambiente não necessitar de grande exatidão de informações.

No caso dos dispositivos de baixa mobilidade, essa estratégia é utilizada quando a prioridade do Cliente é alta, devido a sua pequena movimentação pela área de trabalho. A atualização em intervalos maiores não gerará prejuízos à manutenção da base do RM. Porém, em casos de baixa prioridade, será utilizada a Estratégia Nunca Atualiza (Seção 3.3.5.2), isto

é, a rede não tenta manter rastro do dispositivo após seu registro no sistema, e começa uma busca somente quando informações sobre a localização desse dispositivo são solicitadas (Busca sob Demanda - BSD).

A estratégia híbrida proposta, que utiliza diversas classes de atualização, gera menor custo de manutenção das informações de localização física no RM. Enviar dados de localização física com diferentes taxas de envio dessas atualizações para cada classe reduz a carga de sinalização da arquitetura, como será mostrado na avaliação de desempenho (Capítulo 4).

5.6.4 Busca Sob-demanda

O próximo procedimento mostra o funcionamento da Busca sob-Demanda, quando um Cliente/Aplicação solicita à Base informação de um dado nó.

1. Chegada na BASE, no Módulo RECEPTOR, de pedido de localização de CLIENTE;
2. Receptor se CLIENTE tem Nível de Prioridade Baixo ou Nível de Mobilidade Baixo (condições para BSD);
    a. Verifica REPOSITÓRIO DE MOBILIDADE;
    b. Se o CLIENTE não registrado no REPOSITÓRIO DE MOBILIDADE ou não atualiza a mais de um Δt, ativa BUSCA SOB-DEMANDA;
        i. GERA-BUSCA na AI (envio da mensagem GBM - Gera Busca na Mesma AI, descrita na Seção 4.1.5);
        ii. Se não localizar, ativa módulo GERA-BUSCA nos REPOSITÓRIO DE MOBILIDADE das outras AI's e armazena resposta (envio da mensagem GBO - Gera Busca em Outra AI, descrita na Seção 4.1.5);
    c. Nova informação de localização física é armazenada no REPOSITÓRIO DE MOBILIDADE local;
    d. Repassa informação de localização física para módulo GERA-INFO;
    e. Envia informação de localização física para nó Solicitante.

O diagrama de atividade (ANEXO B) da Figura 5.10 simplifica a visão do procedimento.

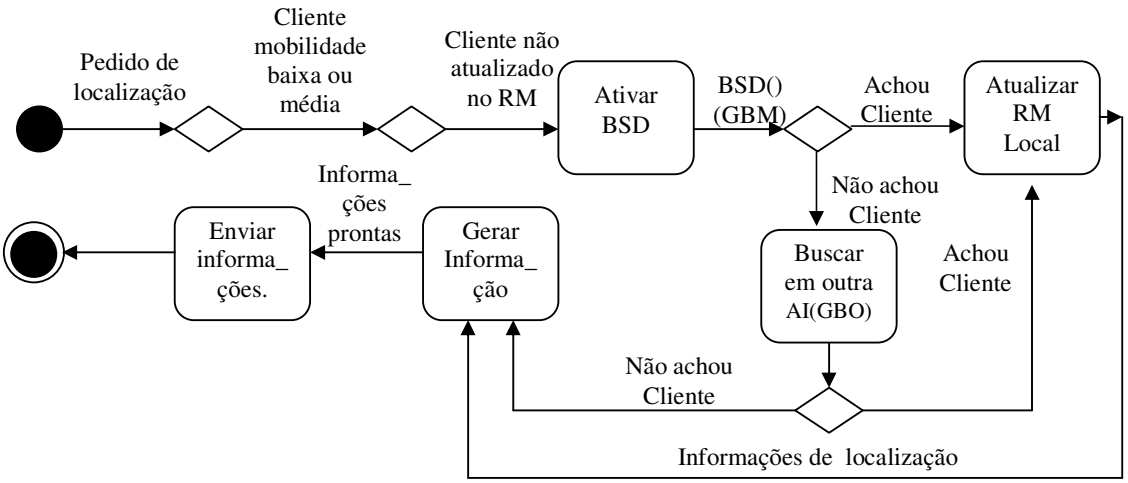


Figura 5.10: Diagrama de Funcionamento do SLF

O pedido de localização de um Cliente é recebido, e se tiver as duas condições para BSD (Mobilidade baixa ou média e o registro do Cliente estar desatualizado no RM), então a Busca é ativada dentro da AI, utilizando as mensagens GBM (Gera Busca na mesma AI, Seção 4.1.5). Se o Cliente não for encontrado, então a busca deve ser realizada nas outras Ais, através do uso das mensagens GBO (Gera Busca em Outra AI). Caso o Cliente tenha sido encontrado, a informação é atualizada no RM local e repassada para o nó solicitante. Caso não tenha sido encontrado, deve retornar uma mensagem de aviso de que o nó não foi achado.

5.7 Serviço de Adaptação de Conteúdo (SAC)

Para adaptação de conteúdo, as entidades do ambiente de execução da camada de serviços estão envolvidas em vários níveis. As aplicações devem ter seu conteúdo adaptado para melhor utilização dos recursos do ambiente e para serem utilizadas com melhor desempenho pelos Clientes. Os serviços cooperam na camada de serviços para a realização desta adaptação. Na arquitetura, a camada de serviços é responsável por intermediar Aplicações e Clientes ou entre Clientes, que desejam enviar conteúdo um para o outro.

Nos ambientes onde a adaptação de conteúdo é realizada, a negociação do conteúdo a ser enviado requer os seguintes elementos:

1. Descrição do Contexto no qual o conteúdo será usado: contexto do cliente, capacidades do servidor, tipo do conteúdo, etc.
2. Perfil do Conteúdo: cada tipo e formato de conteúdo têm várias características e parâmetros que podem ser usados para descrever a mídia. Estas informações incluem as

característica do conteúdo, como tipo de mídia (vídeo, áudio,etc) e formato (vídeo H.261, MPEG, etc).

3. Métodos de Adaptação: usados para adaptar ou substituir o conteúdo com a variante (tipo, formato, qualidade) apropriada;
4. Estratégia de Combinação: um algoritmo que combina os diferentes perfis (cliente, documento, servidor, etc) para determinar o melhor método de adaptação.

Cabe à camada de serviços verificar os níveis de contexto (representados pelos parâmetros de Classes definidos na Seção 5.5), através de dados coletados pelo protocolo de roteamento (ver Seção 5.8) e dos perfis dos Clientes, para realizar a adaptação com a técnica mais adequada a cada caso. A Figura 5.11 mostra o funcionamento geral do serviço, da solicitação do Cliente ao momento o provedor envia o conteúdo.

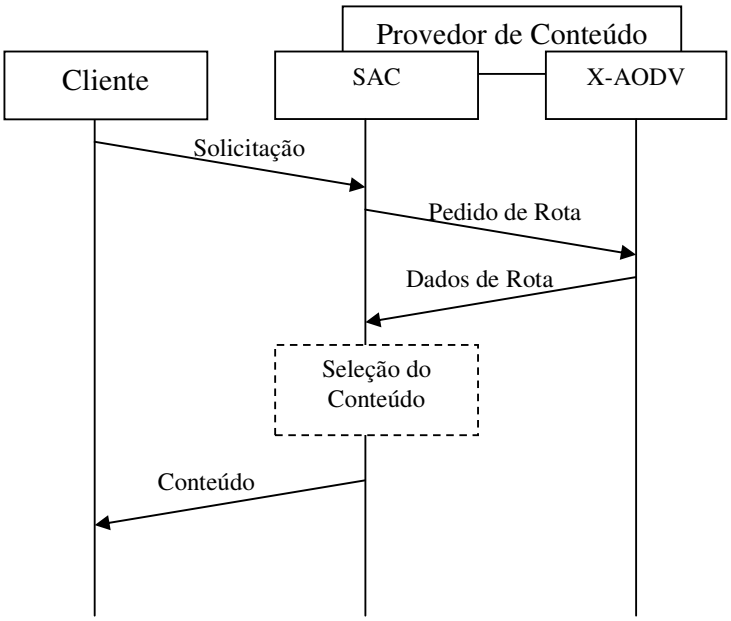


Figura 5.11: Funcionamento geral da adaptação

A Seleção do Conteúdo é controlada por uma estratégia de negociação, através do Protocolo de Negociação de Adaptação (PNA), que permite conciliar as limitações dos clientes às capacidades de adaptação dos provedores de conteúdo.

5.7.1 Protocolo de Negociação de Adaptação (PNA)

O protocolo PNA decide onde a adaptação deve ocorrer e indica qual o melhor método de adaptação, de acordo com o contexto dos dispositivos geradores do conteúdo, adaptadores

e receptores. Para implementar essas requisições, o processo de adaptação de conteúdo é executado em 4 fases:

1. Coleta de dados dos dispositivos presentes na rota,

Na primeira fase, quando uma aplicação ou usuário pretende realizar uma transferência de conteúdo para outro usuário, as informações sobre Classe de Dispositivos de todos os usuários entre Fonte e Destino são coletadas pelo protocolo de roteamento e repassadas para a camada de serviços. O nó emissor sabe então o perfil (classe) dos nós que estão presentes no caminho entre o provedor da aplicação e o receptor. Esta informação é necessária para o protocolo PNA indicar, na próxima fase, qual a classe máxima de conteúdo (CAmáx, Seção 5.7.1.1) pode ser enviado na rede e também para a camada de serviços escolher a rota mais apropriada em relação aos dispositivos. Detalhes sobre a geração e escolha de rotas são mostrados no Capítulo 4.

2. Negociação de parâmetros,

Na segunda fase, com a informação da classe máxima de conteúdo que pode ser enviado já selecionada, o PNA verifica quais os parâmetros detalhados do dispositivo receptor, que foram enviados de volta ao provedor de conteúdo no momento em que recebeu a rota para o destino. A escolha da classe de conteúdo a ser enviado depende dos parâmetros recebidos da rede e do dispositivo fim.

3. Escolha do local de adaptação

A escolha do local de adaptação, seleciona o dispositivo que realizará a adaptação do conteúdo, se necessário. A escolha depende da possibilidade do próprio provedor realizar a adaptação, e em caso negativo, o PNA deve escolher qual o nó da rede mais apropriado para realizar esta tarefa, como um dispositivo vizinho, presente na rota calculada ou que possua maior poder computacional (maior Classe de Dispositivo).

4. Geração do conteúdo personalizado.

Após a determinação do local, a geração do conteúdo (quarta fase) é realizada. O objetivo é customizar apresentações para os diferentes tipos de clientes do ambiente.

A próxima seqüência resume o processo de adaptação.

1. Pedido de rota ao destino pelo nó provedor de conteúdo (fonte);
2. Geração de informações de rota;
3. A fonte recebe a classe de dispositivo (CD) dos nós pertencentes à rota e os parâmetros de dispositivo (PD) do nó destino;
4. O PNA verifica a Classe de Adaptação (CA) máxima de acordo com a rota;

5. O PNA verifica os parâmetros do dispositivo fim e do conteúdo, determinando o método de adaptação a ser realizado;
6. Fonte gera conteúdo;
7. A Camada de serviços adapta e envia os dados ao destino especificado.

As próximas sub-seções detalham todo processo de adaptação.

5.7.1.1 Escolha da Classes de Adaptação (CA)

Os tipos de conteúdo que trafegam na rede, e, portanto, devem ser adaptados, são Texto, Imagem, Vídeo e Áudio. Cada tipo de dado é classificado em três categorias de qualidade, para se adaptar aos tipos de Clientes, de baixa, média ou alta capacidade computacional.

Com a indicação das classes de dispositivos selecionadas na geração da rota, o nó emissor verifica a Classe de Adaptação máxima (CAmáx) que pode gerar o conteúdo, antes da verificação dos Parâmetros de Dispositivo, que determina o método final para a geração do conteúdo. Estes dois procedimentos pertencem à Fase 2 do processo de Adaptação. A Tabela 5-4 resume as Classes de Adaptação.

Tabela 5-4: Classes de Adaptação de Conteúdo [SMIT99]

Conteúdo	Classe de Adaptação	Parâmetros
Texto	**1**	Completo
	2	Sumário
	3	Título
Imagem	**1**	8 bits de cores
	2	4 bits de cores
	3	2 bits preto e branco
Áudio	**1**	50 kbps
	2	25 kbps
	3	8 kbps
Vídeo	**1**	1 Mbps
	2	128 kbps
	3	28 kbps

O próximo algoritmo mostra como funciona a escolha da Classe de Adaptação máxima.

Definições:

CD_{fonte} = Classe do dispositivo fonte: provedor de conteúdo/serviços;

$CD_{destino}$ = Classe do dispositivo destino: cliente;

CD_{interm} = Classe do dispositivo do nó intermediário de menor valor, isto é, a classe do nó de menor capacidade presente na rota escolhida;

$CA_{máx}$ = Classe de Adaptação máxima selecionada na primeira parte;

SE (CD_{fonte} <= $CD_{destino}$) e (CD_{fonte} <= CD_{interm})
$CA_{máx} = CD_{fonte}$;
SE (CD_{Interm} <= CD_{fonte}) e (CD_{Interm} <= $CD_{destino}$)
$CA_{máx} = CD_{Interm}$;
SE ($CD_{destino} \geq CD_{fonte}$) e ($CD_{destino} < CD_{intermed}$)
$CA_{máx} = CD_{destino}$.

Este algoritmo seleciona o nó de menor Classe de Dispositivo (isto é, capacidade computacional) entre fonte e destino.

O nó escolhido será utilizado para escolher a classe máxima de adaptação, isto é, o conteúdo gerado deve ser adequado ao dispositivo de menor capacidade na rota que levará os dados entre o provedor e o receptor do conteúdo. Como será visto na Seção 6.4.2, os dispositivos de menor capacidade são prejudicados pelo tráfego que repassam para outros nós. A Seção 6.4.3.3 mostra um exemplo de cenário onde o uso de diferentes valores de $CA_{máx}$ afetam o desempenho do ambiente.

Portanto, ao limitar o conteúdo que será repassado na rede, os dispositivos de baixa capacidade não são prejudicados e, ao mesmo tempo, outros dispositivos conseguem receber o conteúdo desejado. No trabalho de campo, o recebimento das informações, mesmo com qualidade inferior à capacidade do dispositivo, permite que as aplicações continuem funcionando e o trabalho sendo realizado.

5.7.1.2 Seleção dos Parâmetros de Dispositivos (PD)

Ainda na Fase 2 do processo de adaptação, após a determinação da $CA_{máx}$ de adaptação, serão verificados os Parâmetros de Dispositivos.

Diferentemente da Classe de Dispositivos, que engloba os diferentes tipos de dispositivos em três classe gerais de acordo com sua capacidade computacional, os Parâmetros de Dispositivos detalham, para cada dispositivo considerado, os seus parâmetros computacionais exatos. Esses parâmetros, quando recebido pelo provedor de conteúdo, são utilizados para a geração dos dados a serem enviados. Esses valores são limitados pelo parâmetro da classe de adaptação calculado anteriormente.

Para garantir que o conteúdo requisitado seja mostrado de forma apropriada, é essencial incluir as características e capacidades do dispositivo no processo de personalização do conteúdo. Informações sobre o dispositivo incluem as características de *hardware*, como tipo de dispositivo, velocidade do processador, resolução de tela, número de cores, memória disponível, tamanho da tela, e capacidade de entrada e saída.

As características de *software*, como sistema operacional, codificadores de áudio e vídeo também podem ser incluídos no perfil do dispositivo. O Perfil de Agente de Cliente (*User Agent Profile*[2], UAProf) criado pelo WAP Fórum e o padrão MPEG 21[3] incluem ferramentas de descrição para capacidades de dispositivos.

O perfil descreve o recurso de acordo com a seguinte sequência: Item, Endereço, Domínio do Recurso e Comentários (Figura 5.12). Item é o nome do recurso; endereço representa o endereço de rede, no formato de URL; domínio identifica se é um recurso de hardware ou de software; e comentários são usados para detalhar o recurso.

```
<rdf:Description ID="ITEM">
    <rdf:type rdf:resource="ENDEREÇO/>
    <rdfs:domain rdf:resource="#DOMÍNIO DO RECURSO" />
    <rdfs:comment>COMENTÁRIOS</rdfs:comment>
  </rdf:Description>
```

Figura 5.12: Template do perfil do UaProf

Para melhor entendimento, a Figura 5.13 mostra um exemplo[4] de perfil do UAProf para um dispositivo.

[2] http://www.openmobilealliance.org/tech/profiles/ www.openmobilealliance.org/tech/profiles/
[3] http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm
[4] http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405

```
<!--  ***** Component: HardwarePlatform *****   -->
- <rdf:Description ID="BitsPerPixel">
    <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property" />
    <rdfs:domain rdf:resource="#HardwarePlatform" />
    <rdfs:comment>Description: The number of bits of color or grayscale information per pixel,
        related to the number of colors or shades of gray the device can display. Type: Number
        Resolution: Override Examples: "2", "8"</rdfs:comment>
  </rdf:Description>
- <rdf:Description ID="ColorCapable">
    <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property" />
    <rdfs:domain rdf:resource="#HardwarePlatform" />
    <rdfs:comment>Description: Indicates whether the device's display supports color. "Yes"
        means color is supported. "No" means the display supports only grayscale or black and
        white. Type: Boolean Resolution: Override Examples: "Yes", "No"</rdfs:comment>
  </rdf:Description>
- <rdf:Description ID="CPU">
    <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property" />
    <rdfs:domain rdf:resource="#HardwarePlatform" />
    <rdfs:comment>Description: Name and model number of the device CPU. Type: Literal
        Resolution: Locked Examples: "Pentium III", "PowerPC 750"</rdfs:comment>
  </rdf:Description>
- <rdf:Description ID="ImageCapable">
    <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property" />
    <rdfs:domain rdf:resource="#HardwarePlatform" />
    <rdfs:comment>Description: Indicates whether the device supports the display of images. If
        the value is "Yes", the property CcppAccept may list the types of images supported.
        Type: Boolean Resolution: Locked Examples: "Yes", "No"</rdfs:comment>
  </rdf:Description>
- <rdf:Description ID="InputCharSet">
    <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Property" />
    <rdf:type rdf:resource="http://www.w3.org/TR/PR-rdf-schema#Bag" />
    <rdfs:domain rdf:resource="#HardwarePlatform" />
    <rdfs:comment>Description: List of character sets supported by the device for text entry.
        Property's value is a list of character sets, where each item in the list is a character set
        name, as registered with IANA. List items are separated by white space. Type: Literal
        Resolution: Append Examples: "US-ASCII", "ISO-8859-1", "Shift_JIS"</rdfs:comment>
  </rdf:Description>
```

Figura 5.13: Exemplo de Perfil de Dispositivo

Este perfil descreve os parâmetros de BitsPerPixel (bits por pixel), ColorCapable (capacidade de suporte a cores), CPU (tipo de CPU) , ImageCapable (se dá suporte para imagens) e InputCharSet (tipos de caracteres com suporte), e cada recurso é descrito dentro do próprio perfil nos campos de comentários.

5.7.1.3 Negociação dos Parâmetros de Adaptação

Nessa fase é realizada a seleção do Método de Adaptação mais apropriado ao conteúdo gerado e uma Estratégia de Combinação. Essa estratégia é um algoritmo que combina os diferentes perfis (cliente, documento e servidor) para determinar qual o melhor método de adaptação para ser utilizado em cada caso. A combinação utiliza os dados de Perfil de Dispositivo do destino e Classe de Adaptação máxima selecionados anteriormente,

juntamente com os dados do nó fonte e seus serviços de capacidade adaptativa (codificadores e transformadores de dados).

O provedor do conteúdo deve manter o perfil do conteúdo (características e funcionalidades) que está fornecendo. Isso inclui também o perfil do recurso (descreve o recurso de mídia usada e o método de adaptação que existe no servidor).

Durante a negociação, o perfil do dispositivo (PD) e o perfil do conteúdo requisitado são comparados e o conjunto de restrições é armazenado na memória de acordo com seus tipos. De acordo com o conteúdo, o servidor pode pegar o parâmetro do dispositivo que corresponde ao recurso usado pelo conteúdo requisitado. Por exemplo, no caso do envio de um vídeo (exemplo: vídeo colorido, codificador MPEG), o provedor verifica os parâmetros de tela e codificadores do cliente.

Se a descrição do recurso for compatível com os requisitos do método de adaptação, o servidor verifica se a saída gerada pelo método de adaptação combina com os requisitos do cliente. Se combinar, a camada de serviços aplica o métodos de adaptação no recurso original e envia o recurso criado ao cliente. Em caso negativo, i.e., nenhum método de adaptação pode ser aplicado, outro local de adaptação deve ser selecionado.

De forma resumida, a negociação realiza o seguinte procedimento:

1. Verifica o tipo do conteúdo a ser enviado;
2. Compara o tipo do conteúdo com o Perfil do Dispositivo destino;
3. Compara a capacidade (serviços adaptativos) do dispositivo fonte;
4. Escolhe o método de adaptação; e
5. Escolhe o local de adaptação (opcional).

A seguir, são descritas as estratégias de combinação entre Perfil do Dispositivo e Fonte, a escolha do método de adaptação e a escolha do local de adaptação.

a) Estratégia de Combinação

Dentro da Combinação, comparam-se o tipo do conteúdo com o Perfil do Dispositivo destino e a capacidade (serviços adaptativos) do dispositivo fonte.

A Tabela 5-5 mostra os parâmetros que são comparados na primeira parte da combinação, representados por um “X”. A tabela foi gerada de acordo com os tipos de conteúdo e as operações possíveis de se realizar com cada um. De acordo com o tipo de conteúdo a ser enviado, compara-se o conteúdo com a capacidade do cliente, se possui suporte ao tipo do dado a ser enviado. Considera-se que, no ambiente de trabalho, os diversos

dispositivos presentes também podem possui diversas capacidades de *software*, de codificadores, etc.

Tabela 5-5 : Parâmetros de Comparação

Conteúdo	Codec	Formato	Cores	Resolução
Áudio	X	---	---	---
Imagem	---	X	X	X
Vídeo	X	---	X	X
Texto	---	X	---	---

Após a determinação destes parâmetros, o PNA verifica qual o método de adaptação mais apropriado, de acordo com as capacidades do servidor e os formatos disponíveis para entrega.

O próximo passo é decidir o método de adaptação mais adequado, caso seja necessário realizar algum tipo de adaptação. Caso adaptação não seja necessária, o conteúdo já pode ser enviado.

b) Escolha do Método de Adaptação

A adaptação pode ser realizada em duas formas: uso do mesmo formato com parâmetros modificados (exemplo: resolução) ou transformação de modalidade (ver Seção 2.3.2.2), quando um dispositivo não possuir capacidade para nenhum formato disponível ou o provedor de conteúdo não conseguir adaptar no formato necessário.

Para mudanças dos parâmetros de envio, quando Cliente e Servidor possuem os mesmo formatos/codificadores disponíveis, o provedor, ao receber os Parâmetros de Dispositivo do nó destino, realiza mudanças de acordo com a Tabela 5-6. Essa tabela mostra quais operações devem ser realizadas para cada tipo de conteúdo.

Tabela 5-6: Mudança dos Parâmetros do Conteúdo

Conteúdo	Cores	Resolução	Taxa
Áudio	---	---	Diminuição para se adequar à taxa máxima que a classe de adaptação permite
Imagem	Diminuição do número de cores	Diminuição para tamanho da tela do dispositivo	---
Vídeo	Diminuição do número de cores	Diminuição para tamanho da tela do dispositivo	Diminuição para se adequar à taxa máxima que a classe de adaptação permite

Esta tabela foi gerada considerando o algoritmo da escolha da Classe de Adaptação máxima. A adaptação é direcionada ao dispositivo de menor capacidade entre fonte e destino.

As mudanças a serem realizadas incluem somente diminuição dos parâmetros de qualidade. Como será visto na Seção 5.8, o provedor publica a classe máxima de conteúdo que ele pode fornecer, limitando a qualidade do conteúdo. Para mudanças no caso de adaptação, portanto, só estão disponíveis operações de redução de qualidade.

Quando não existir acordo com o tipo de formato de envio, é possível mudar de formato ou executar a transformação da modalidade. A hierarquia de adaptação segue a seguinte seqüência:

- Áudio: Mesmo Codec → Redução de Qualidade → Outro Codec;
- Imagem: Mesmo Formato → Redução de Qualidade → Outro Formato.
- Vídeo: Mesmo Codec → Redução de Qualidade → Outro Codec → Transformação em Imagens;
- Texto: Mesmo Formato → Outro Formato → Transformação em ASCII;

A escolha da sequência de operações foi definida de acordo com cada tipo de dado. A prioridade é a geração de dados dentro do próprio provedor para um codec/formato apropriado, sem perda da qualidade. Se isso não for possível, utiliza-se o mesmo formato para a diminuição da qualidade. Como última opção, caso o codec/formato não esteja disponível, muda-se o formato.

A escolha entre a troca de formato e a transformação da modalidade depende do local de adaptação. Como será visto na Análise de Desempenho (Seção 6.4.3.2), a transformação ou adaptação realizada mais próximo da Fonte é mais vantajosa em termos de tempo de resposta, isto é, o tempo entre a transmissão pela Fonte e o recebimento no Destino é menor quando o conteúdo é adaptado na fonte, ou em nós mais próximos dela. Quanto mais distante da Fonte o conteúdo for adaptado, maior o tempo de resposta total.

Portanto, antes de escolher o método de adaptação, deve-se decidir o local da realização da adaptação. Após a escolha do local é possível decidir se é melhor adaptar localmente para outro formato, transformar a modalidade ou enviar o conteúdo sem adaptação para outro nó realizar o processo.

c) Escolha do Local de Adaptação

Nesta etapa é escolhido o dispositivo que irá modificar o conteúdo. A camada de serviços participa neste processo verificando se a adaptação será realizada no próprio

provedor do conteúdo ou se será realizado por algum nó vizinho presente na rota entre fonte e destino.

Existem duas possibilidades para escolher um outro nó para modificar o conteúdo gerado. Ou o nó provedor não possui os codificadores necessários para combinar com os codificadores do nó destino, ou a capacidade do nó provedor é inferior às capacidades da vizinhança.

Quando o conteúdo é enviado pela rede, geralmente passa por diversos dispositivos intermediários (Figura 5.14). Eles podem ser usados para serviços de adaptação, o que pode aliviar o problema de clientes com recursos limitados e servidores sobrecarregados.

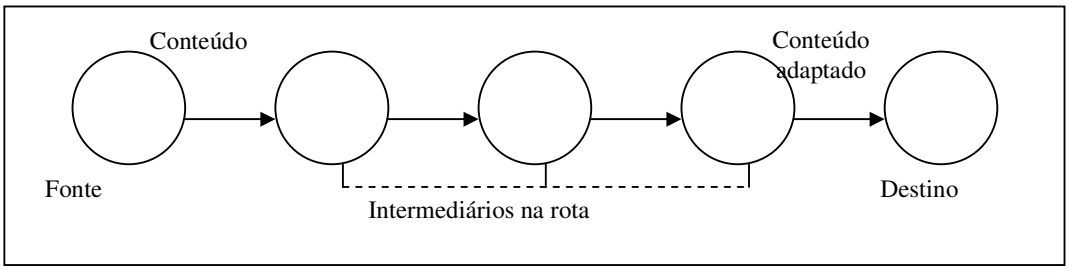


Figura 5.14: Intermediários entre Fonte e Destino

O perfil de um nó vizinho (ver Seção 5.8.2) deve incluir a descrição dos serviços de adaptação que ele pode prover. Estes serviços podem ser descritos pelo protocolo de Descoberta de Serviços utilizado na arquitetura. A descrição deve incluir, por exemplo, os formatos de entrada e saída. Caso o nó fonte seja de capacidade inferior ao vizinho que oferece o serviço necessário para a adaptação, a camada de serviços indica ao nó fonte para enviar o conteúdo e o vizinho, ao receber os dados, gera o conteúdo adaptado e o envia para o destino.

Com rota entre Fonte e Destino estabelecida anteriormente, a camada de serviços realizará uma pesquisa de serviços de acordo com uma hierarquia de busca:

1. Vizinhos pertencentes à rota entre Fonte e Destino;
2. Outros nós da Rota;
3. Outros Vizinhos;
4. Base.

O nó mais próximo da Fonte que seja capaz de atender os requisitos de adaptação ou transformação do conteúdo é escolhido. Como será visto na Seção 6.4.3.2, quanto menor a distância (saltos) entre Fonte e nó adaptador, menor o tempo de resposta do sistema.

Após a escolha do local e a determinação do método mais apropriado, o conteúdo pode ser gerado e enviado.

5.7.2 Estrutura do SAC

Como visto na Seção 5.3, nos Clientes são armazenados seus registros de perfis, indicando sua classificação de acordo com sua Classe de Dispositivo e os Parâmetros de Dispositivos (PD), com descrições detalhadas de suas capacidades.

Nas entidades geradoras de conteúdo (servidor Base ou um nó que oferece serviços no Diretório de Aplicações), o SAC é composto de dois'módulos, como mostra a Figura 5.15.

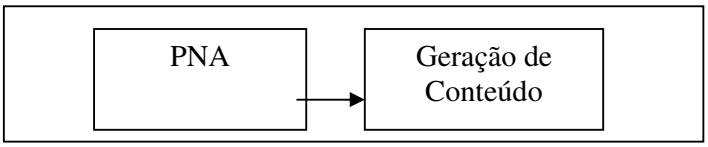


Figura 5.15: Módulos do SAC

O PNA, descrito na Seção anterior, é responsável por ler as informações recebidas e analisá-las. Determina que tipo de ação deve ser realizada. A ação de adaptação é desempenhada pelo Módulo de Geração de Conteúdo (MGC), que realiza as ações indicadas pelo PNA. O conteúdo adaptado é então enviado para o destino.

5.7.2.1 Funcionamento Geral do SAC

O próxima sequência de passos mostra o funcionamento do serviço, desde a solicitação do Cliente, do início do serviço da Aplicação até o momento em que a camada de serviços ativa o SAC para adaptar o conteúdo transmitido, resumindo todo o procedimento descrito na sub-Seção 5.8.

- A APLICAÇÃO ou CLIENTE (FONTE, provedor de conteúdo) solicita rota para DESTINO (receptor do conteúdo);
- A camada de serviços solicita rota FONTE-DETINO ao X-AODV;
- O X-AODV gera informações para criação da rota e coleta dados de dispositivos na rota;
- Quando a rota for calculada, a FONTE verifica se a APLICAÇÃO é dependente de contexto de localização. Se for, a FONTE solicita informações de Localização Física do CLIENTE à Base;
- O PNA escolhe a Classe de Adaptação máxima de acordo com os dados coletados pelo X-AODV;
- O PNA compara os Parâmetro de Dispositivos do cliente com os parâmetros do conteúdo e da Fonte;
- O PNA escolhe local de adaptação;

- O Provedor disponibiliza conteúdo para ser enviado.
- O nó gerador de conteúdo recebe o conteúdo a ser adaptado (caso não seja a própria Fonte);
- O Módulo GERAÇÃO DE CONTEÚDO recebe as Classes de Adaptação e Parâmetros de Dispositivos e gera o novo conteúdo;
- O Conteúdo é adaptado e enviado.

O SAC, portanto, realiza a coleta das informações (de rota, de classes de dispositivos, de parâmetros de dispositivos) para a geração do conteúdo a ser enviado. Também escolhe onde será realizada a adaptação, permitido pelo uso das classes de dispositivos e conhecimento dos serviços dos vizinhos. Sem o conhecimento dos serviços dos vizinhos, no momento da geração do conteúdo, a camada de serviços deveria verificar no DAP (Diretório de Aplicações) quais os nós que poderiam gerar o conteúdo de acordo com as regras selecionadas. Isso geraria mais custos (envio de mensagens de controle) e atrasos no envio do conteúdo (por gerar rota até a Base e depois ir buscar o conteúdo). Detalhes sobre a descoberta de serviços são mostrados na próxima Seção.

5.8 Serviço de Descoberta de Serviços (SDS)

Cada Cliente participante do ambiente de trabalho de campo pode ser um Provedor de Serviços (PS). Estes provedores possuem recursos ou informações que podem compartilhar com outros participantes. Por exemplo, um dispositivo que possui o recurso de câmera de fotografia acoplada ao dispositivo que pode gerar imagens de certo local de trabalho ou outro dispositivo que armazena informações sobre certa área e pode compartilhá-las sob demanda.

A possibilidade de prover serviços dentro do ambiente de trabalho é muito útil, dado que as informações contextuais de localização física podem gerar dados mais precisos sobre aspectos relacionados diretamente ao objeto de estudo dos trabalhadores.

Supondo que um Cliente Cl_i possui uma câmera de fotografia acoplada e ele oferece os serviços ao ambiente, outros Clientes podem solicitar uma imagem atual da área de trabalho onde Cl_i está localizado para efeitos de comparação com sua posição atual.

Como num sistema real um servidor de descoberta de serviços deve ficar localizado em um endereço conhecido por todos os clientes do sistema, o servidor da Base é responsável por manter o Diretório de Aplicações.

Este trabalho de Tese não propõe uma nova arquitetura de Descoberta de Serviços, como as mostradas na Seção 3.1, mas oferece a abordagem de utilizar os métodos existentes

e estendê-los para incluir os fatores de localização física e capacidade heterogênea dos dispositivos presentes no ambiente.

5.8.1 Provedores e Usos de Classes

Como visto anteriormente neste trabalho, os dispositivos presentes no ambiente são heterogêneos e com a presença de nós de baixa capacidade. Dando continuidade ao conceito de Classes de Dispositivos (Seção 5.5) e Adaptação de Conteúdo, a Descoberta de Serviço também utiliza as classes como parâmetros para oferecimento de Serviços.

O nó solicitante é verificado através dos mesmos métodos de verificação da adaptação de conteúdo. O nó provedor de serviços, além de publicar as interfaces dos serviços, também publica sua Classe de Dispositivo, permitindo que os outros nós saibam antecipadamente a classe máxima de conteúdo (Seção 5.7.1) que o provedor está oferecendo.

Por exemplo, um certo serviço do provedor PS*j* é solicitado por um nó *i*. O nó *i*, ao receber as ofertas dos serviços, recebe também o valor de classe de adaptação máxima que o provedor PS*j* pode atender: neste caso, classe de adaptação 2. Sendo o nó *i* de Classe de Dispositivo (CD) inferior à Classe de Adaptação máxima que PS*j* pode oferecer, *i* solicita o recebimento do serviço.

5.8.2 Serviços de Vizinhos

Como mostrado anteriormente (Seção 4.1.2), cada nó conhece os nós que estão em seu alcance e sua localização física atual. Em cada nó é mantida uma tabela (Tabela 5-7) com as informações dos vizinhos, incluindo os serviços que cada um oferece. O processo de troca de informações entre vizinhos está detalhado na Seção 4.1.5.

Tabela 5-7: Vizinhos

Nó	Classe de Dispositivo	Serviços	Posição Física	Status	AI
ID	1, 2 ou 3		Lat, long	(B, S ou C)	

Dentre os serviços que vizinhos podem oferecer e ajudar um ao outro neste tipo de ambiente estão os serviços adaptativos, onde um nó provedor de serviços (de vídeo, de imagem, etc) solicita ao seu vizinho para receber os dados e adaptá-los antes de enviar para o nó destino. Se o nó desejar um serviço oferecido pelo vizinho, realiza o pedido diretamente, sem o uso do Diretório de Aplicações.

5.8.3 Diretório de Aplicações

Localizado na Base, este diretório armazena os dados de serviços disponíveis no ambiente. Cada nó que desejar se tornar provedor publica seus serviços no Diretório, de acordo com o procedimento padrão dos protocolos de Descoberta de Serviços (Seção 3.2.1).

Nesta arquitetura, para otimizar a busca por serviços solicitados no Diretório, serão utilizadas as informações contidas nas tabelas Vizinho de cada nó.

O nó que deseja um certo serviço gera um pacote de Pedido de Serviço Específico (PSE) encapsulando no pacote de roteamento (Seção 4.1.5) para a Base a informação sobre o serviço que está procurando. Cada nó que recebe o pacote de roteamento verifica o tipo da mensagem. Sendo PSE, verifica qual serviço o nó deseja, incluído no campo Múltiplo Uso (4.1.4) do pacote. Caso o serviço solicitado esteja presente na tabela Vizinhos, o nó que está analisando o pacote envia o pedido de serviço para o provedor diretamente, que por sua vez fornece o serviço solicitado (Seção 3.2.2), como exemplifica a Figura 5.16.

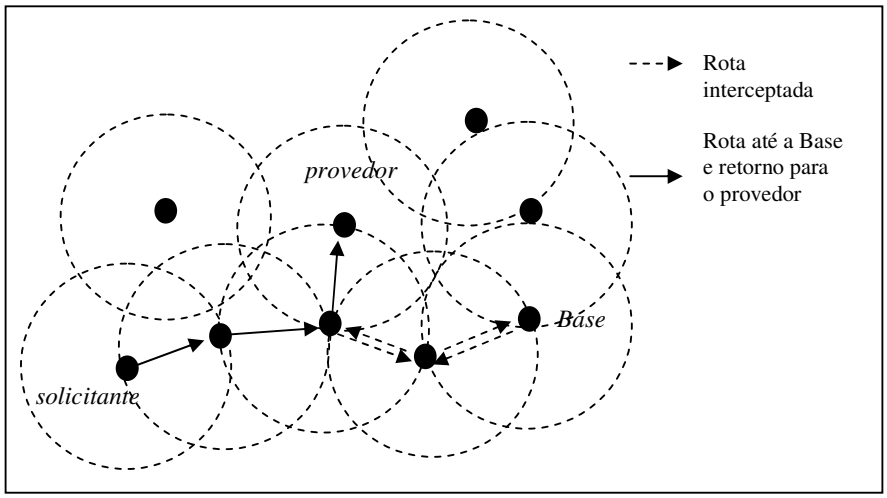


Figura 5.16: Interceptação de Pacote de Pedido de Serviço Específico

Caso nenhum nó entre o solicitante e a Base conheça o serviço específico, o pedido chega até a Base, que consulta o Diretório e repassa o pedido para o provedor, como também mostra a Figura anterior.

A comunicação direta com a Base é realizada para os Clientes que desejam conhecer todos os serviços disponíveis em sua AI. A Base consulta os serviços atuais e retorna ao Cliente a lista de serviços e os provedores.

Quando o provedor recebe a solicitação de serviço, deve solicitar uma rota até o destino. Na geração de rotas para o cliente destino, o provedor de serviços recebe os parâmetros de Classes de Dispositivos e realiza Adaptação de Conteúdo, caso seja necessário.

5.8.4 Uso de *Geocasting* para Oferecer Serviços

Pode ocorrer de que um novo serviço fundamental para os trabalhadores ser adicionado ao Diretório. Para avisar todos os nós da AI, a Base gera um aviso via *geocasting* (Seção 3.4.2), avisando em todos os nós dentro de uma AI que chegaram novos dados na base de dados (Figura 5.17).

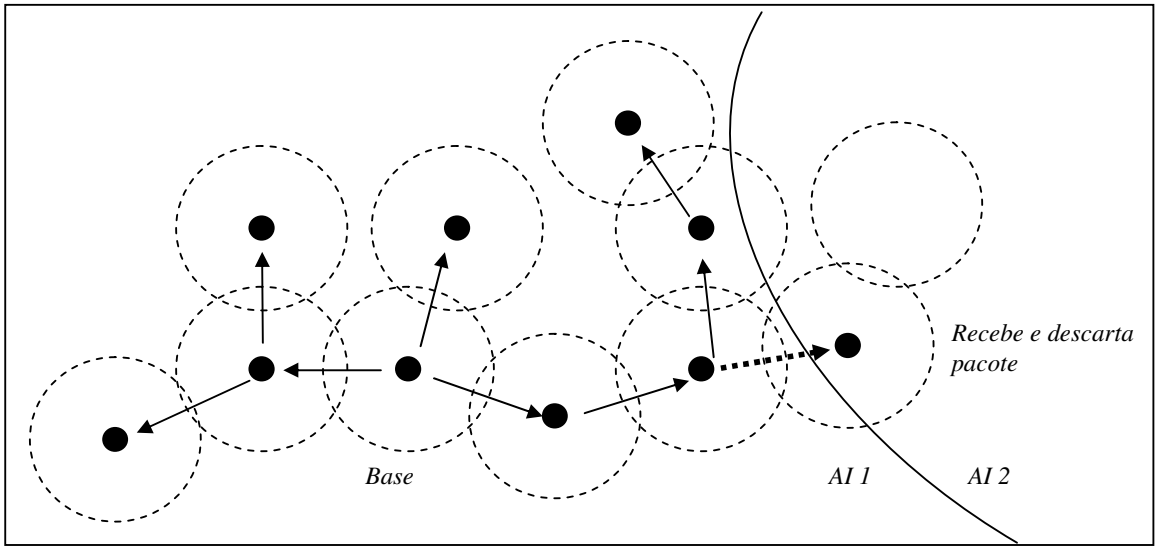


Figura 5.17: *Geocasting* para oferecer serviço indispensável

Quando um nó receber o aviso *geocasting*, verifica se pertence à AI indicada (a indicação da AI do *geocast* é inserida no pacote – ver Seção 4.1.5). Caso contrário, descarta o pacote. O g*eocasting* pode ser limitado a uma AI, ou usado para outra AI. Nesse caso, os dados devem ser roteados até um nó Borda da AI destino, que repassa os pacotes para todos os nós de sua AI.

5.9 Outras situações

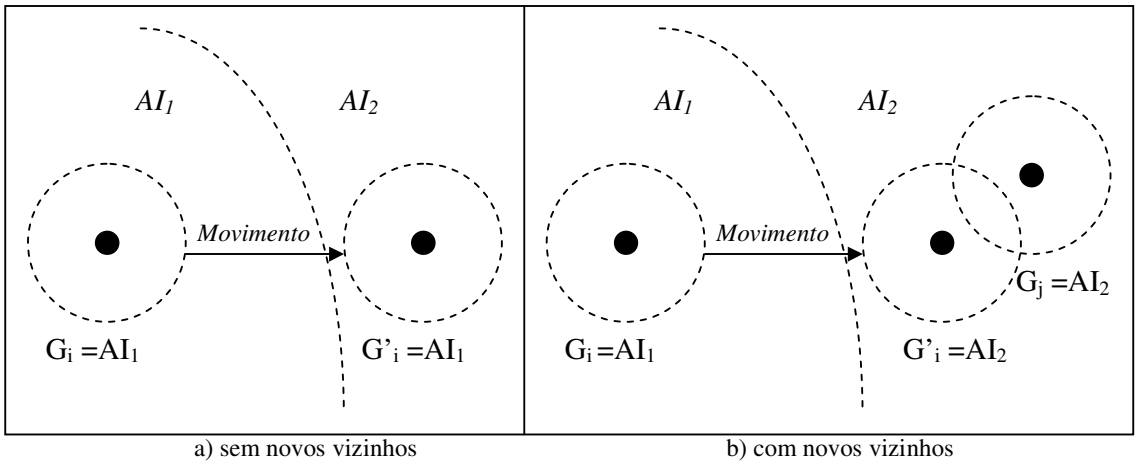
Esta Seção mostra como manipular situações específicas que podem ocorrer no decorrer do tempo, durante o uso da arquitetura.

5.9.1 Troca de AI

Quando um nó chega à fronteira de sua AI, se torna Borda (Seção 4.1.2). Se o nó continuar em direção para fora da AI atual e entrar no limite de outra AI, o nó muda de associação para a nova AI.

Ao sair da AI verifica se há algum nó ao seu alcance. Enquanto não se comunicar com algum nó da nova AI, mantém sua associação com a AI anterior (apesar de estar fora da área definida). Quando entrar em contato com um nó da nova área, troca de associação (5.18). O

nó deve, portanto, fazer registro no RM da AI atual e atualizar seus novos vizinhos através das mensagens Atualiza Vizinhos.



a) sem novos vizinhos b) com novos vizinhos

Figura 5.18: Troca de AI

O nó sem novos vizinhos deve manter associação com a AI anterior senão estaria fora de qualquer agrupamento, pois como ainda não teve contato com outros nós, não sabe se está fora ou dentro da área de atuação de outro agrupamento. É possível que tenha saído da área de trabalho de campo e não encontrará outra AI para se associar. No momento em que entrar em contato com outros nós da nova AI, pode realizar o registro na nova área de trabalho.

5.9.2 Base fora do ar ou de alcance de todos os nós

Se a Base ficar fora do ar com conhecimento prévio, gera mensagens *geocasting* avisando os nós do grupo para não enviarem suas atualizações e pedidos até um Δt determinado ou até a Base reenviar outro *geocast*, avisando para os nós de sua volta.

No caso da Base demorar mais do que um certo Δt configurado no sistema para retornar, o nó de maior capacidade, de baixa prioridade e mobilidade (CD = 1; CM=3 e CP=3) pode assumir o papel de Base para receber as atualizações dos nós e a publicação dos serviços.

O nó assumido como Base avisa o grupo através de *geocasting*. Esse aviso é necessário para informar sua localização física para todos os nós, permitindo que a nova Base receba a atualização de todos os nós do agrupamento.

5.10 Conclusões

Este Capítulo apresentou a arquitetura APTC, uma arquitetura de computação pervasiva, para uso em trabalho de campo composta de diversos serviços. Os serviços são divididos em Localização Física (SLF), Adaptação de Conteúdo (SAC) e Descoberta de Serviços (SDS).

As informações coletadas nas gerações de rotas pelo protocolo de roteamento (que será visto adiante) também são utilizadas pelos serviços e aplicações, que geram o conteúdo mais apropriado a cada dispositivo. Verificando a Classe de Conteúdo máxima para a geração dos dados a serem enviados, e com o conhecimento dos Parâmetros de Dispositivo do nó alvo, ocorre a negociação de parâmetros pela aplicação, que gera o conteúdo mais adequado para tráfego na rede e para o dispositivo alvo.

As informações sobre os nós vizinhos permite que outros nós utilizem seus serviços como local de adaptação, permitindo que dispositivos de baixa capacidade, ou que não tenham condições (*software*) para realizar a adaptação, ainda tenha possibilidade de oferecer serviços para outros nós do sistema.

O uso de classes de Mobilidade e Prioridade também otimiza a geração de atualização de localização física. Como os trabalhadores possuem comportamento diferente no local de pesquisa, com mobilidade diversa pelo ambiente, a atualização de suas informações também deve ocorrer de formas diferentes.

O próximo Capítulo mostra a análise de desempenho de diversos aspectos relacionados a esta arquitetura.

Capítulo 6 - Avaliação de Desempenho

6.1 Introdução

A técnica de avaliação de desempenho utilizada neste trabalho é a Simulação. Cada execução de uma simulação gera uma amostra dos valores pesquisados. Como uma amostra não é suficiente para garantir a precisão do resultado obtido, diversas simulações devem ser realizadas para ser possível calcular uma média do parâmetro estudado e gerar um intervalo de confiança daquele valor [JAIN92].

Este capítulo mostra os detalhes das simulações realizadas, seus fatores e parâmetros, e os resultados obtidos em cada experimento, onde os valores estão apresentados em média e seu intervalo de confiança.

Também são mostradas as fórmulas de tempo de resposta, geradas com a técnica de modelagem, e que são utilizadas para mostrar resultados dos protocolos e do serviço de adaptação de conteúdo.

6.2 Simulação

Uma das técnicas de avaliação de desempenho é a Simulação. Esta técnica envolve contruir um modelo para o comportamento do sistema e uma abstração apropriada da carga de trabalho [KANT92]. Uma das vantagens dos modelos de simulação é a flexibilidade, que permite que quase qualquer comportamento seja facilmente simulado.

De forma geral, a simulação pode ser feita com diversos níveis de detalhes, dependendo dos objetivos do estudo. Podem existir diversos modelos para um mesmo sistema a vários níveis de detalhes, de forma que os resultados de um estudo de um modelo mais detalhado possam calibrar os modelos de mais alto nível.

6.2.1 Tipos de Simuladores

Existem diversos tipos de simuladores, como *Emulation*, Monte Carlo, simuladores *trace-driven* e simuladores de estados discretos [JAIN92].

No tipo *emulation*, um terminal emulador simula um tipo de terminal em outro terminal. Um processador emulador emula um conjunto de instruções de processador em outro. A simulação do tipo Monte Carlo é uma simulação estática ou sem um eixo de tempo. Essas simulações são utilizadas para modelar fenômenos probabilísticos que não tenham suas

características mudadas com o tempo. Uma simulação que utiliza um *trace* como entrada é chamada de simulação *Trace-Driven.*

Uma simulação que usa modelos de estados discretos é chamada de simulação de estados discretos. Nos sistemas computacionais, modelos de estados discretos são usados quando os estados do sistema são descritos por um número de *jobs* em um dispositivo. O termo discreto não se aplica a valores de tempo. Um simulador de estados discretos pode ter valores de tempo discreto ou contínuo.

O simulador utilizado neste trabalho é do tipo simulador de estados discretos.

6.2.2 Ambiente de Simulação - Network Simulator

Este trabalho utilizou o *Network Simulator* (NS-2) [NS00], que é um simulador desenvolvido pela Universidade de Berkeley. Ele tem como objetivo realizar simulações de tecnologias, protocolos e arquiteturas de comunicação para redes de computadores. Este simulador apresenta ainda a capacidade de estudar o comportamento de aplicações de geração de tráfego, e de diferentes mecanismos de gestão de filas de pacotes.

O NS-2 é um simulador discreto de eventos, onde as atividades físicas são transformadas em eventos. Estes eventos são enfileirados e processados de acordo com as suas ocorrências de escalonamento.

O simulador tem um modelo de programação em dois níveis: o núcleo do simulador é implementado em C++ que interage com módulos programados em OTcl, uma extensão orientada a objeto do Tcl. Enquanto o núcleo implementa a lógica e as primitivas da máquina de simulação (*simulation engine*), o OTcl é usado para desenvolver os scripts de simulação e descrever as funcionalidades dos elementos de rede simulados. Ambas as linguagens podem ser utilizadas para implementar protocolos de rede a serem simulados. Devido este modelo de programação, o NS-2 é um simulador de protocolo de propósito geral, flexível, e extensível.

Uma extensão do simulador foi desenvolvido pelo projeto Monarch [MONA98], que acresentou ao simulador básico, uma extensão para rede ad hoc. Os protocolo desenvolvidos incluem quatro protocolos de roteamento para redes *ad hoc*: DSDV, DSR, AODV e TORA. Para a subcamada MAC existem duas opções: IEEE 802.11 ou TDMA, enquanto que para a camada de transporte estão disponíveis os protocolos TCP e UDP. É possível definir ainda qual é o tipo de antena utilizada, qual é o consumo de energia, e o alcance do sinal de rádio de cada nó móvel. Os nós móveis são dispostos, através de coordenadas cartesianas, em um espaço tridimensional delimitado e gradeado, sendo que a resolução da grade pode ser ajustada. Para movimentar um nó móvel neste espaço são informadas as coordenadas de seu destino e a sua velocidade até este destino [MELE03].

6.2.3 Intervalo de Confiança da Média

Intervalos de confiança geram uma faixa de valores estimados que provavelmente incluem parâmetros desconhecidos da população, sendo este intervalo calculado de um conjunto de valores de amostras de dados [JAIN92].

A largura do intervalo de confiança mostra quanta incerteza existe sobre o parâmetro desconhecido. Um intervalo muito grande pode indicar que mais dados devem ser coletados antes que algo em definitivo seja dito sobre o parâmetro. Intervalos de confiança são mais informativo do que resultados simples de testes de hipóteses (onde se decide *rejeitar* H_o ou *não rejeitar* H_o) desde que sejam fornecidos um intervalo de valores plausíveis para o parâmetro desconhecido.

Se amostras independentes são tiradas repetidamente de uma mesma população, e um intervalo de confiança calculado para cada amostra, então um certo percentual (nível de confiança) dos intervalos incluirá o parâmetro desconhecido da população. Intervalos de confiança são calculados de forma que esta percentagem seja de 95%, mas podem produzir intervalos de confiança de 90%, 99%, 99.9% para os parâmetros desconhecidos.

O nível de confiança é o valor probabilístico 1- α (chamado de coeficiente de confiança) associado com um intervalo de confiança. É geralmente expresso por uma porcentagem. Por exemplo, se $\alpha = 0.05 = 5\%$, então o nível de confiança = (1 - 0.05) = 0.95, que é um nível de confiança de 95%.

O intervalo de confiança da média especifica uma faixa de valores dentro do parâmetro desconhecido da população, neste caso a média, pode estar.

6.3 Resultados de Simulação

O objetivo de uma avaliação quantitativa é comparar, através de experimentos, vários aspectos referentes à arquitetura e conseguir dados comparativos entre variações de um mesmo tema. Foram realizadas simulações para analisar diversos aspectos da arquitetura, como adaptação, localização física e escalabilidade.

6.3.1 Visão Geral da Simulação

Os dados inseridos no simulador se dividem em Parâmetros e Fatores. Os parâmetros são dados fixos, padrão para todos os experimentos de cada cenário. Os fatores são os parâmetros que variam em cada experimento de um cenário.

Alguns dos parâmetros e fatores são:

1. Número de nós: número total de nós presentes na área total;

2. Parâmetros da área: medida do tamanho da área de trabalho: base x altura;
3. Alcance dos nós: raio de alcance de sua comunicação sem fio;
4. Número de conexões: nós que estão transmitindo dados;
5. Mobilidade dos nós: classe de mobilidade.

As métricas são os resultados obtidos nas simulações. Definição das métricas:

1. Fração de pacotes enviados: a razão dos pacotes recebidos pelos pacotes enviados. Expresso em %, mostra a porcentagem dos pacotes recebidos no destino especificado no momento do envio. FPE igual a 100% significa que todos os pacotes que foram enviados foram recebidos. Quanto menor o FPE, pior a qualidade da transmissão.

   FPE = pacotes recebidos/pacotes enviados (Equação 6-1)

2. Atraso fim-a-fim: inclui todos os atrasos gerados entre o momento da transmissão e a recepção dos pacotes, como atraso no *buffering* durante a latência de descoberta de rotas, enfileiramento na interface de rede, atraso de retransmissão do MAC, tempo de propagação e transferência.
3. Carga de Roteamento Normalizada – CRN (*Normalised Routing Load*): é o número de pacotes de roteamento transmitida por pacotes de dados enviados ao destino.

   CRN= número de pacotes de roteamento transmitida / (Equação 6-2)
   pacotes de dados enviados ao destino

Estas métricas não são totalmente independentes. Por exemplo, uma baixa fração de pacotes enviados significa que o atraso é avaliado com base em poucas amostras. Quanto maior o caminho a ser percorrido, maior a probabilidade de pacotes perdidos. Portanto, com uma baixa fração de pacotes enviados, as amostras são geralmente favorecidas em favor de caminhos menores e, portanto, geram menos atraso.

6.3.2 Metodologia de Simulação

Os cenários de simulação usam áreas de trabalho de 1000x1000m, com os nós tendo alcance de transmissão de sua comunicação sem fio de 200m. O uso de 50 e 100 nós nas simulações foi determinado pelo tamanho dos grupos de trabalhadores alvo deste trabalho. Grupos de trabalho de tamanho médio, como utilizados por grupos de pesquisadores de diversas áreas, tais como em minas a céu aberto e pesquisa científica de campo.

Cada experimento foi simulado 10 vezes, e foram calculados intervalos de confiança dos valores obtidos de 95%.

6.3.3 Experimento 1 – Divisão da área em AIs

Neste experimento, foi simulada a divisão da área de trabalho em sub-áreas de interesse (AI). Os cenários utilizados foram dois (representados na Figura 6.1):

- Área total: o servidor de localização física se localiza no centro geográfico da área de trabalho como um todo. Todos os nós devem atualizar suas informações neste servidor.
- Múltiplas AIs: vários servidores são espalhados pela área de trabalho, subdividindo-a em várias sub-áreas de interesse (Seção 4.1.1). Os nós se reportam para o servidor fisicamente mais próximo (Figura 6.1).

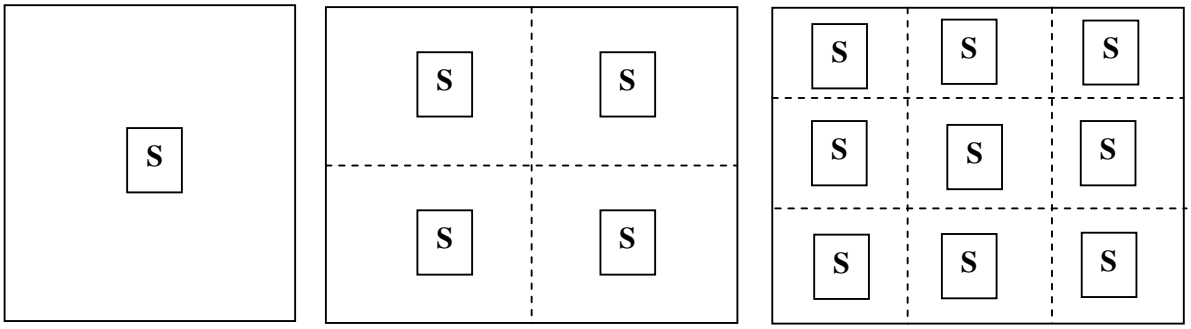


Figura 6.1: Área total e múltiplas AIs (4 e 9)

Os fatores analisados neste cenário tinham como objetivo verificar se a divisão da área de trabalho traria ganhos no desempenho de comunicação para o ambiente de trabalho. A Tabela 6-1 mostra os parâmetros utilizados nas simulações.

Tabela 6-1: Parâmetros de simulação do Cenário 1

Parâmetros / Fatores	Valor
Número de nós	100
Alcance dos nós	200m
Parâmetros da área	1000x1000m
Taxa de atualização de ILF (λ)	λ = 1/10 at/seg
Mobilidade	Velocidade = 5 m/s
Número de conexões (nós que enviam atualização)	10, 50 e 75
Divisão da Área	Área Total, Múltiplas AIs (4 e 9 Ais)

O estudo sobre a fração de pacotes enviados é mostrado no Gráfico 6-1. O melhor desempenho foi mostrado pelo cenário de 9 AIs, com 97% a 100% na relação pacotes enviados/recebidos. O que significa que quase a totalidade dos pacotes que foram enviados

foram recebidos com acerto. No caso de uma única área, o pior desempenho, com variação de 94% a 96% de pacotes recebidos, mostra que uma solução que concentra todos as atualizações para fora da área de trabalho é prejudicial ao funcionamento de um sistema que necessita de saber a localização física de seus nós com regularidade.

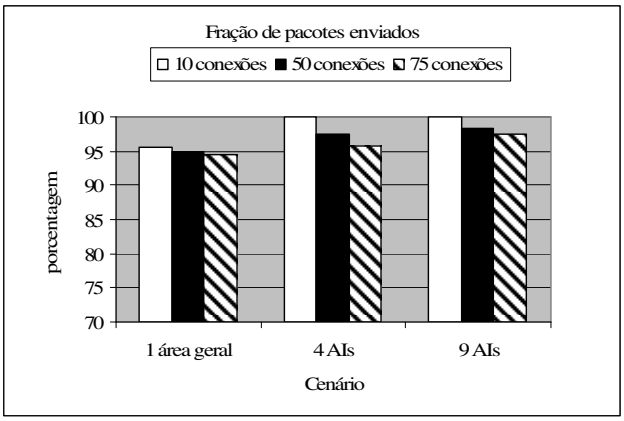


Gráfico 6-1: Divisão da área de trabalho – Fração de Pacotes enviados (intervalo de confiança = ± 1)

Como ponto alto dda divisão de áreas em áreas menores, também se considera que os grupos de trabalho podem repassar mensagens dentro de sua AI, evitando enviar dados para trabalhadores de outras AIs, reduzindo o escopo das mensagens que trafegam.

Em relação ao atraso fim-a-fim gerado nas transmissões (Gráfico 6-2), é possível notar maior diferença entre os cenários. As sub-áreas conseguem menor atraso em todos os casos de número de conexões.

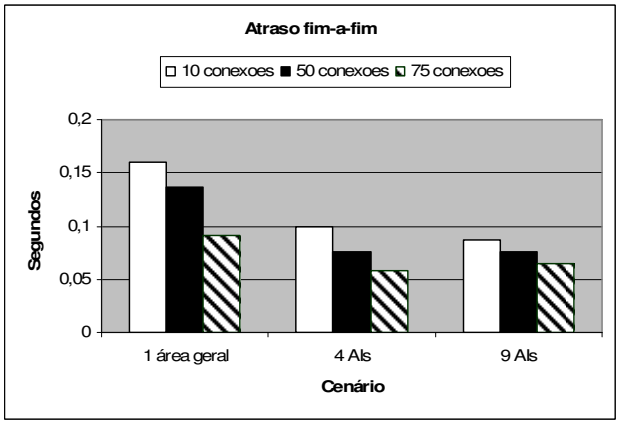


Gráfico 6-2: Divisão da área de trabalho - Atraso fim-a-fim (intervalo de confiança = ± 0,02)

Na área única, considerando que diversos tráfegos de pacotes pode ser repassado por um mesmo nó, certos nós se sobrecarregam e isso gera maiores tempos de transmissão.

Portanto, para os dois aspectos analisados, a divisão em subáreas pela área de trabalho de campo possui melhor desempenho.

6.3.3.1 Experimento 2 – Efeitos de Mobilidade

Neste experimento foi simulada a variação da mobilidade dos nós para verificar o efeito desta na carga de trabalho transmitida pela rede em casos de atualizações periódicas de dados de localização física. Considerando a existência de diversos tipos de usuários, que possuem mobilidade diversa pela área de trabalho, foi criada a classificação por tipos de mobilidade - Classe de Mobilidade (Seção 5.5). Este estudo mostra que para cada Classe de Mobilidade uma estratégia de atualização é necessária.

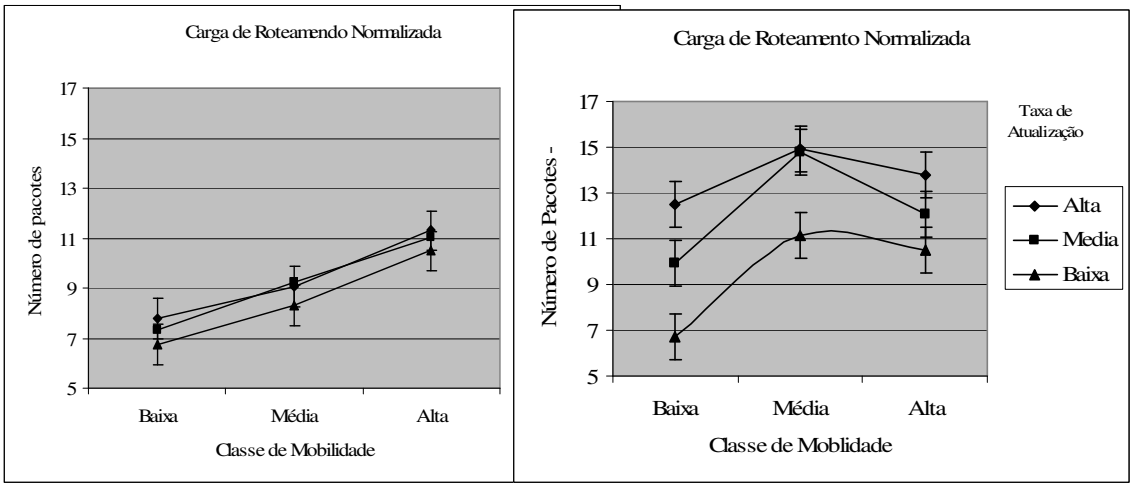
A Tabela 6-2 mostra os parâmetros utilizados nas simulações.

Tabela 6-2: Parâmetros de simulação do Experimento 2

Parâmetros/Fatores	Valor
Número de nós	50 e 100
Alcance dos nós	200m
Parâmetros da área	1000x1000m
Número de Ais	4
Número de conexões (nós que enviam atualização)	50 e 100
Mobilidade dos Nós	Baixa: Velocidade = 1m/s
	Média: Velocidade = 10 mm/s
	Alta: Velocidade = 20m/s
Taxas de Atualização (λ) (Freqüência de atualização do nó)	Baixa: λ_B=1/100 at/seg
	Média: λ_M= 1/50 at/seg
	Alta: λ_A= 1/10 at/seg

Uma operação de atualização demora o tempo necessário para gerar o pacote com a informação do nó e enviar pela rede. Considera-se neste cenário a geração e manutenção das rotas entre o nó gerador e a Base.

No estudo de comparação entre Classes de Mobilidades foram realizadas simulações para as classes Baixa, Média e Alta, variando a taxa de atualizações. O Gráfico 6-3 mostra a comparação entre as diferentes classes de mobilidade. O comportamento do sistema mostra que quanto maior a taxa de atualização, maior a Carga de Roteamento Normalizada.



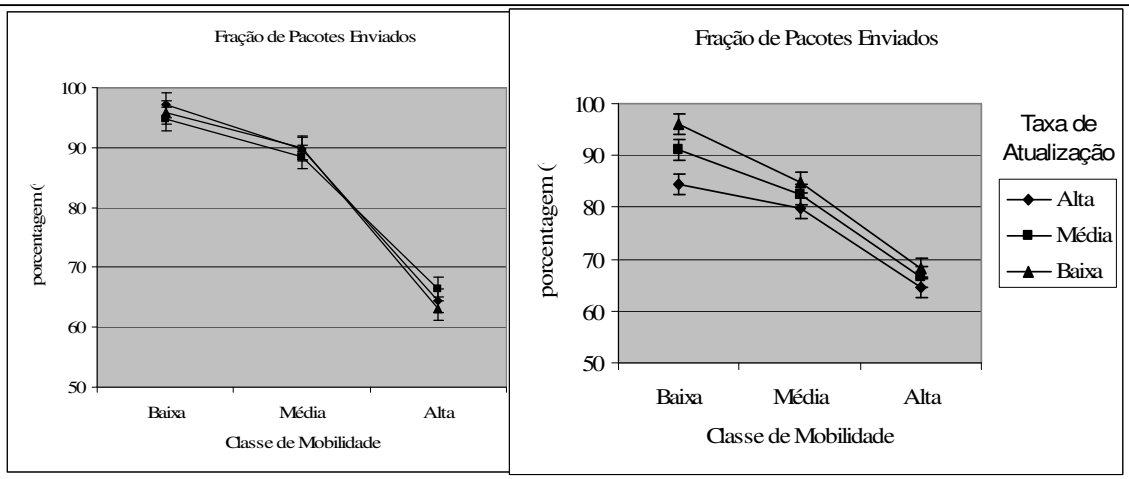
(a) (b)

Gráfico 6-3: Comparação de carga normalizada de roteamento a)100 e b)50 conexões

Pode-se analisar o aumento da CRN com o aumento da mobilidade dos nós. Isso ocorre devido ao aumento das mensagens de controle de roteamento, enquanto que a taxa de dados transmitida permanece constante. Quanto maior a mobilidade, maior a probabilidade de mudanças e atualizações de rotas, aumentando a carga de roteamento.

Para uma mesma categoria de mobilidade, verifica-se que quanto maior a taxa de atualizações, maior a carga normalizada. Porém, considerando a Equação 6.2, onde a carga normalizada de pacotes é o número de pacotes de roteamento em relação ao número de pacotes de dados, se consideramos que o roteamento permanece constante e aumenta-se a taxa de atualizações (e dados), teoricamente a CRN deveria diminuir com o aumento de geração de dados. Mas como mostram os próximos gráficos, para altas taxas de mobilidade e de atualizações, a perda de pacotes é muito alta, o que interfere no resultado na CRN. Para grandes perdas de pacotes, a CRN aumenta, pois é diretamente proporcional aos pacotes de dados enviados.

O Gráfico 6-4 mostra que a perda de pacotes de dados é muito alta para as maiores taxas de atualização.



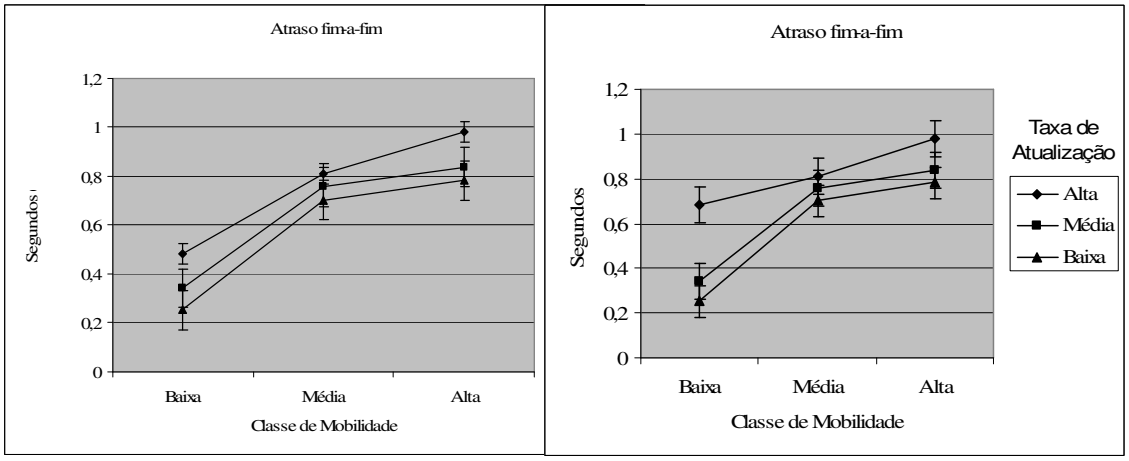
(a) (b)

Gráfico 6-4: Comparação de fração de pacotes enviados (a) 100 e (b) 50 conexões

Para uma mesma categoria de mobilidade, a perda de pacotes é aproximada, independente das taxas de atualização. Porém, quanto menor a mobilidade do nó, maior seu Fração de Pacotes Enviados: $FPE_{mob_baixa} > FPE_{mob_média} > FPE_{mob_alta}$. Isto ocorre porque altas taxas de mobilidade não permitem que as rotas sejam atualizadas a tempo, o que gera a perda de pacotes.

Para uma mesma categoria de taxas de atualização, verificam-se valores aproximados de FPE, com uma pequena variação de aproximadamente 3% . Portanto, quanto menor a taxa de atualização, melhor o FPE: $FPE_{atual_baixa} > FPE_{atual_média} > FPE_{atual_alta.}$. A perda de pacotes de dados é muito alta para as maiores taxas de atualização.

Comparando os três tipos de mobilidade também pode-se analisar o desempenho do sistema em relação ao atraso fim-a-fim. O Gráfico 6-5 mostra os valores gerados.



(a) (b)

Gráfico 6-5: Comparação de Atraso fim-a-fim (a) 100 e (b) 50 conexões

Para uma mesma categoria de mobilidade, nota-se o maior atraso fim-a-fim nos nós de alta mobilidade. A alta mobilidade gera mais mudanças de rotas, que consomem mais tempo para a transmissão. Nos nós de mobilidade menor, a mudança de rotas é realizada com menos freqüência, o que ajuda a manter o atraso baixo.

Conclui-se que:

- os nós de mobilidade alta possuem pior desempenho nos três parâmetros analisados;
- altas taxas de atualização degradam o desempenho do sistema;
- nós de mobilidade baixa e média consomem menos recursos e geram menos atraso.

Com estes dados verifica-se que é necessário ter diferentes tipos de estratégias de atualização de dados para diferentes tipos de mobilidade de nós. Esta é a justificativa da Estratégia Híbrida de Atualização de Informações, vista na Seção 5.6.3.

Para confirmar que o uso de estratégias variadas melhora o desempenho do sistema, foram simulados cenários incluindo taxas diferentes de atualização para cada tipo de mobilidade. Os próximos gráficos mostram a variação dos três parâmetros analisados quando é incluído o cenário Variado (nós de mobilidade baixa = 50%, média = 30% e alta = 20%, onde cada classe de mobilidade possui parâmetros próprios de atualização, de acordo com a Tabela 4-2).

Primeiramente, verifica-se que o atraso fim-a-fim diminui para as classes de atualização média e alta (Gráfico 6-6), perdendo apenas para um cenário onde todos os nós geram atualizações com taxas baixas.

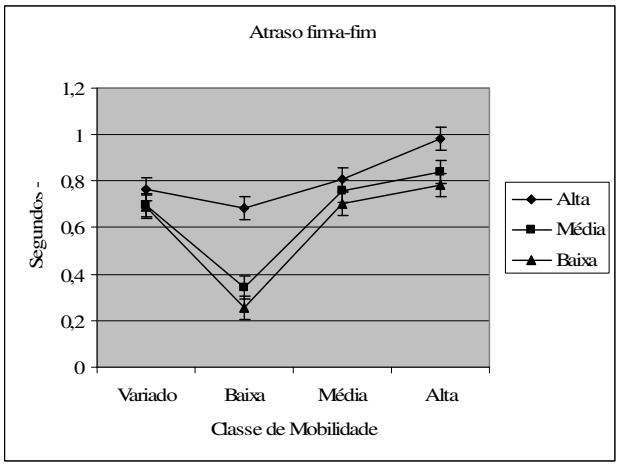


Gráfico 6-6:Atraso fim-a-fim comparando diversas estratégias

O Gráfico 6-7 mostra que a CRN é aproximada para todas as taxas de atualizações no cenário Variado, tendo melhor desempenho que as taxas de mobilidade alta e média, perdendo também apenas para o cenário onde todos os nós possuem mobilidade baixa.

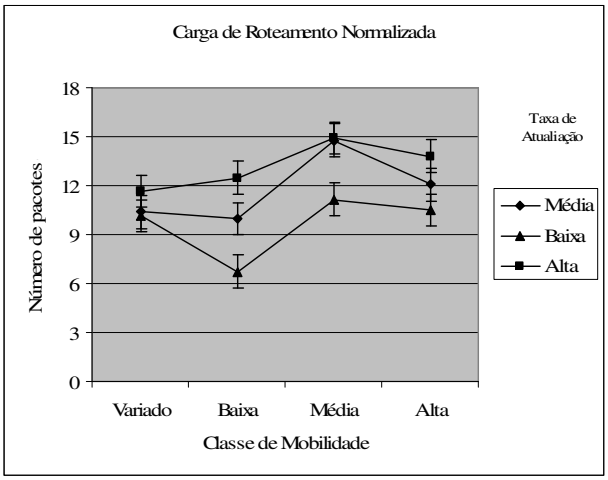


Gráfico 6-7: CRN comparando diversas estratégias

Finalmente, o FPE do cenário Variado é equivalente ao cenário de mobilidade média, perdendo em comparação ao de mobilidade baixa, mas com desempenho melhor do que o cenário de mobilidade alta (Gráfico 6-8).

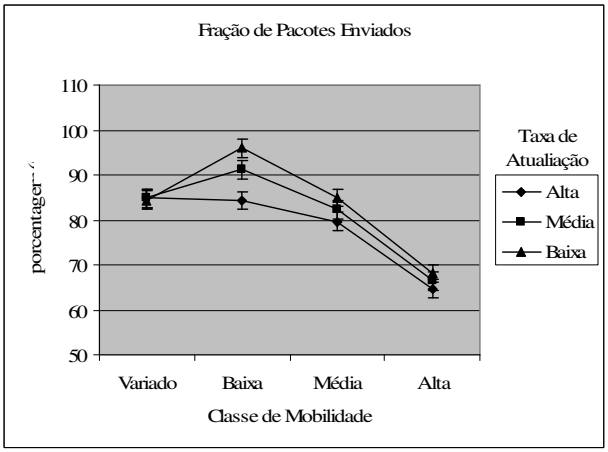


Gráfico 6-8: FPE comparando diversas estratégias

O desempenho do cenário Variado é prejudicado por existirem muitos nós de mobilidade alta, que já foi visto ter menor desempenho de todos os cenários.

Com estes experimentos confirma-se que a variação dos tipos de estratégias de atualizações melhora o desempenho de um sistema que necessita manter dados de localização física (ou até mesmo outros tipos de informações).

Para gerar ainda menos dados de atualizações, diminui-se o número de nós que geram atualizações periódicas. Esta é a justificativa de se utilizar, na Estratégia Híbrida de Atualização de Informações, a técnica Nunca Atualiza (NA) para nós de baixa prioridade, onde os nós só atualizam sua informação de localização física quando solicitados, gerando

uma Busca sob-Demanda. Portanto, os nós do trabalho de campo devem ter sua classificação realizada de tal forma a só permitir a classificação de nós de Alta Mobilidade para os nós com maior importância/prioridade ao sistema.

6.3.4 Experimento 3 – Estratégias de Localização

As estratégias de localização, descritas na Seção 3.3.5, foram comparadas em diversos cenários. Com o conhecimento de que variar as taxas de atualização melhora o desempenho do sistema (Seção anterior), o objetivo dessas simulações foi descobrir qual das estratégias é a melhor em termos de eficiência (% de atualizações transmitidas com sucesso) e menor custo (menor geração de tráfego na rede). As estratégias simuladas foram: Atualização Periódica baseada em Distância, em Tempo e Busca sob-demanda. A Tabela 6-3 mostra os parâmetros utilizados nas simulações.

Tabela 6-3: Parâmetros de simulação do Experimento 3

Parâmetros/Fatores	Valor
Número de nós	100
Alcance dos nós	200m
Parâmetros da área	1000x1000m
Número de AIs	4
Número de conexões (nós que enviam atualização)	50
Mobilidade dos Nós	Baixa: Velocidade = 1m/s
	Média: Velocidade = 5 m/s
	Alta: Velocidade = 20m/s
Estratégias de Atualização (Freqüência de atualização do nó)	BSD, Busca sob-demanda, λ_{BSD}=1/80 at/seg
	M, Baseada em Movimento, λ_M= 1/40 at/seg
	T, Baseada em Tempo, λ_T= 1/20 at/seg

Como visto no Gráfico 6-9, para cada classe de mobilidade a eficiência das estratégias são aproximadas. E de acordo com o cenário anterior, os nós de maior mobilidade possuem o menor desempenho em relação à eficiência de transmissão.

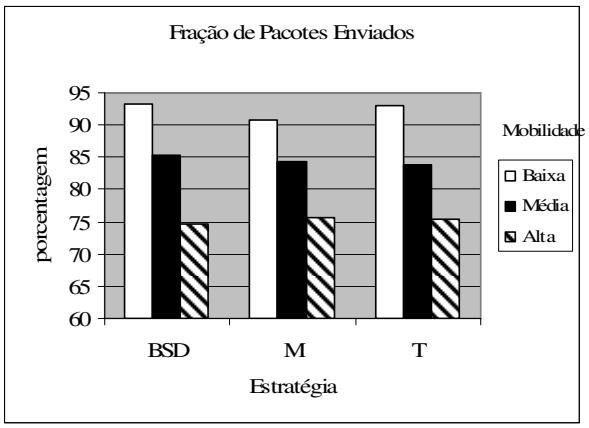


Gráfico 6-9: FPE para estratégias variadas (intervalo de confiança = ± 2)

Comparando o custo de transmissão, em termos de número de pacotes gerados (de dados e roteamento, Gráfico 6-10, a) e b), respectivamente), a estratégia de Busca sob-demanda gera o menor número de pacotes, e menor custo, enquanto que a estratégia baseada em tempo gera maiores custos.

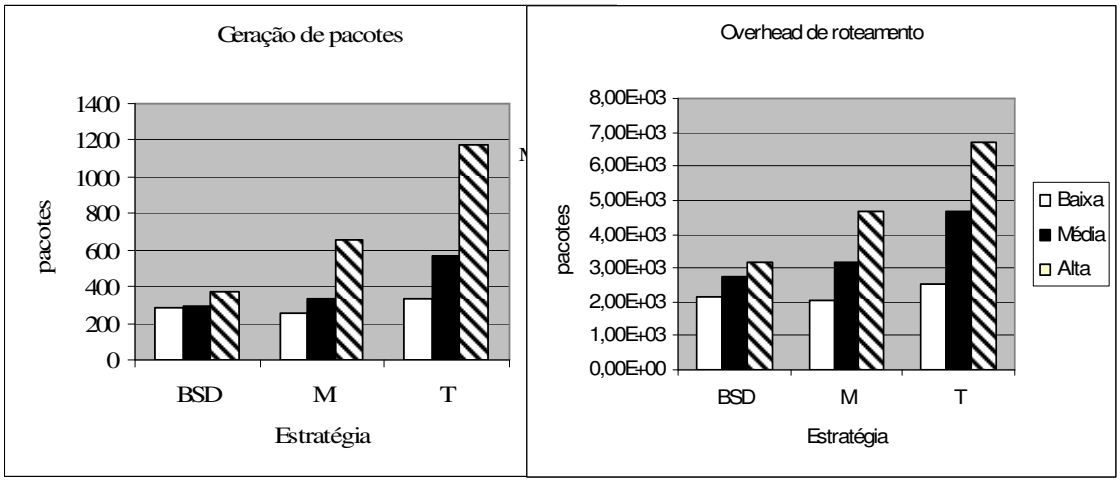


Gráfico 6-10: Relação entre pacotes gerados e estratégias (intervalo de confiança = ± 12)

Apesar do maior custo, a estratégia baseada em tempo gera menores atrasos nas transmissões para todas as classes de mobilidade, em relação às outras estratégias (Gráfico 6-11).

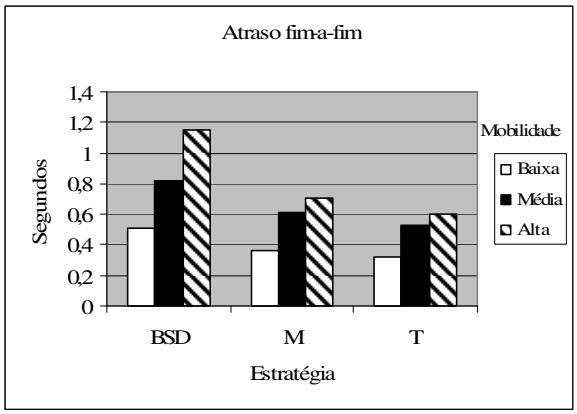


Gráfico 6-11: Atraso fim-a-fim para estratégias variadas (intervalo de confiança = ± 0,04)

Portanto, cada estratégia tem suas vantagens. A Figura 6.2 resume as vantagens de cada uma.

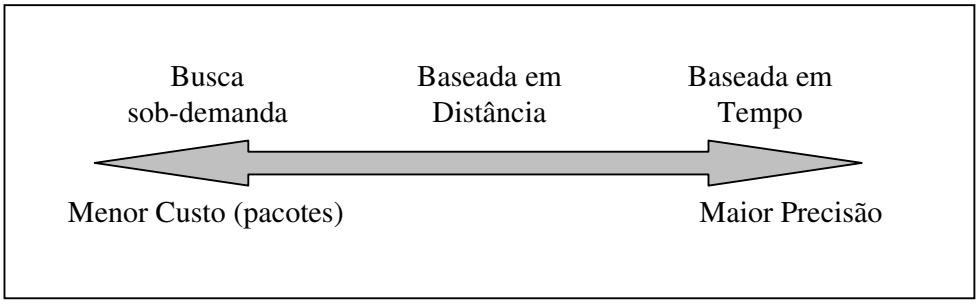


Figura 6.2: Relação entre as estratégias

O menor custo gera também menor precisão nas informações de localização física, enquanto que a maior precisão consome mais recursos de transmissão.

Um gerente de um ambiente de trabalho de campo deve considerar estes aspectos ao classificar os usuários e dispositivos de acordo com as classes de mobilidade e prioridade. Quanto maior precisão nas informações de localização necessitar, a troca de estratégias pode gerar maiores custos. Quando a categoria de prioridade de um Cliente for rebaixada ou sua classe de mobilidade for mudada, a troca de estratégia reduz a carga de pacotes na rede.

6.3.5 Experimento 4 - Roteamento

Este experimento tem por objetivo comparar o desempenho do protocolo AODV com o do protocolo X-AODV. Foram realizadas simulações dentro de uma única área de trabalho e entre duas AIs. A Tabela 6-4 mostra os parâmetros utilizados neste experimento.

Tabela 6-4: Parâmetros de simulação do Experimento 4

Parâmetros/Fatores	Valor
Número de nós	100
Alcance dos nós	200m
Parâmetros da área	1000x1000m
Número de Conexões	50
Mobilidade dos Nós	Velocidade = 1m/s
Taxa de Atualização	λ= 1/20 at/seg

Na primeira parte deste experimento, os protocolos foram comparados em sua utilização dentro de uma mesma área e trabalho. O Gráfico 6-12 mostra a comparação dos protocolos em relação a Carga de Roteamento Normalizada. O Protocolo X-AODV possui menor CRN devido a seu comportamento de selecionar o número de nós para os qual vai repassar os pedidos de rota, portanto, enviando menos pacotes de roteamento do que o AODV original.

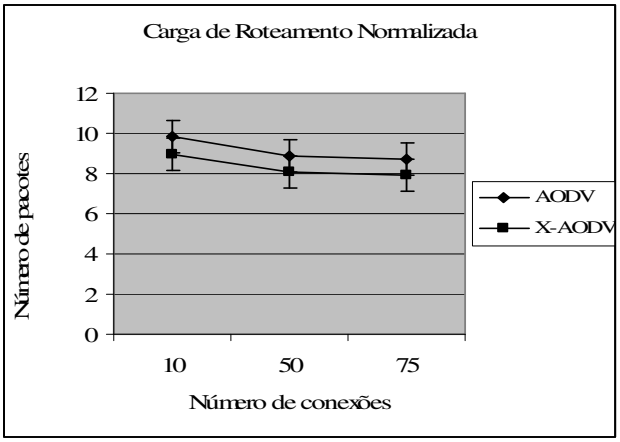


Gráfico 6-12: Comparação – Carga de Roteamento Normalizada

Por outro lado, como podem existir nós selcionados que não possuem todo o caminho final até o nó alvo, a perda de pacotes é maior no X-AODV, diminuindo seu desempenho em relação ao AODV. O Gráfico 6-13 mostra que o AODV possui melhor desempenho, com a quase totalidade dos pacotes enviados.

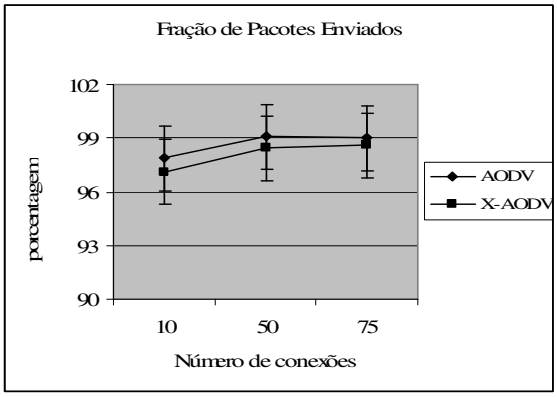


Gráfico 6-13: Comparação Fração de Pacotes Enviados

Em relação ao atraso fim-a-fim (Gráfico 6-14), o AODV possui menor atraso, pois possui pacotes de menor tamanho e a menor taxa de erros permite que as rotas estejam corretas na maior parte do tempo.

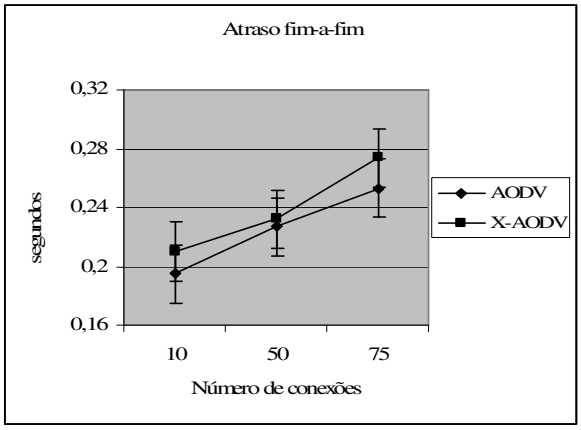


Gráfico 6-14: Comparação – Atraso fim-a-fim

A próxima Seção mostrará que a diferença entre os tamanhos dos pacotes do AODV e X-AODV gera diferença entre os atrasos. Em relação a comunicação entre duas AIs, o atraso é menor quando a comunicação não é quebrada em fases (Seção 4.1.7), pois a manutenção de várias sub-rotas acarreta na maior demora do envio dos dados (Gráfico 6-15).

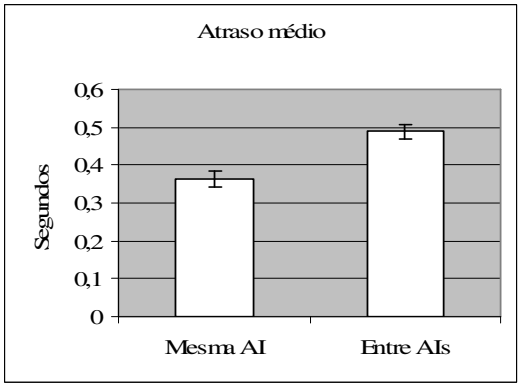


Gráfico 6-15: Roteamento – Atraso médio

A Fração de Pacotes Enviados diminui quando se está repassando dados entre duas áreas. O Gráfico 6-16 mostra que, enquanto transmitindos dentro da mesma AI, o desempenho do protocolo foi superior à transmissão entre duas áreas.

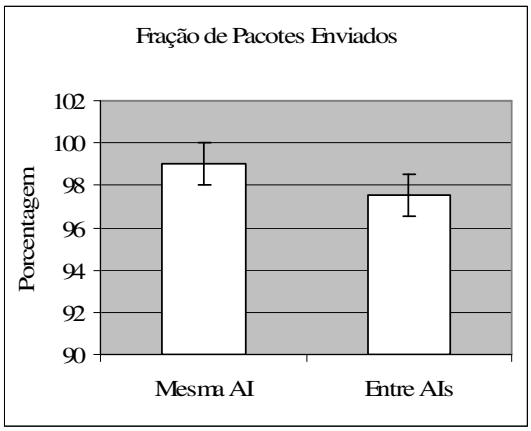


Gráfico 6-16: Roteamento – Fração de Pacotes enviado

Considerando que a geração do número de pacotes de manutenção de rotas é menor quando se está utilizando rotas menores, a Carga de Roteamento Normalizada é menor quando se transmite entre duas AIs (Gráfico 6-17).

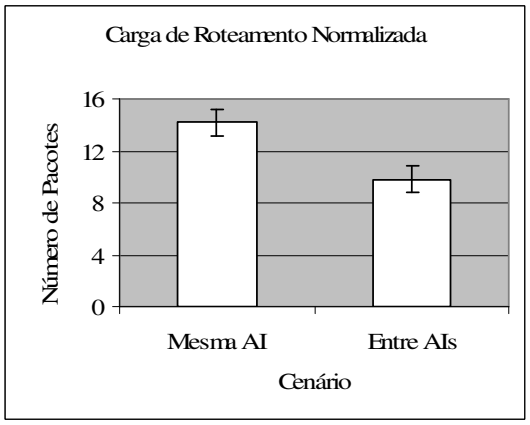


Gráfico 6-17: Roteamento - Carga de Roteamento Normalizada

De modo geral, o desempenho do protocolo na comunicação entre duas AIs é menor do que quando utilizado dentro de uma mesma AI. Porém, como visto no Experimento 1 (Seção 6.3.3), a diminuição do escopo de transmissão da mensagens para apenas dentro de uma mesma área gera um melhor desempenho para o ambiente de trabalho. Isto compensa o fato do protocolo ter menor desempenho quando da comunicação entre duas áreas, pois o maior tráfego de dados é entre os Clientes de AIs de mesmo interesse.

6.4 Modelagem Matemática

Esta Seção utiliza técnicas de modelagem matemática para avaliar dois aspectos da arquitetura. O primeiro aspecto a ser analisado é a utilização do protocolo X-AODV em relação ao protocolo AODV. O outro aspecto é mostrar que a utilização de diversos tipos de dispositivos no mesmo ambiente sem o uso de adaptação de conteúdo quando dispositivos de baixa capacidade estão no meio do roteamento de dados é prejudicial a estes dispositivos.

6.4.1 AODV versus X-AODV

Esta Seção compara os protocolos AODV e X-AODV com relação ao Tempo de Resposta de Rede. Quer-se mostrar que o X-AODV, apesar de ter um tamanho de pacotes maior do que o AODV possui desempenho semelhante ao AODV original principalmente se considerar que os pacotes de tamanho maior do X-AODV são utilizados com outros fins além do simples roteamento.

6.4.1.1 Tempo de Resposta

A métrica de desempenho Tempo de Resposta para uma transmissão de rede considera o tempo total entre o envio de dados e sua recepção. O tempo de resposta pode ser dividido em tempo de rede e tempo no *host* (Figura 6.3) [MENA02].

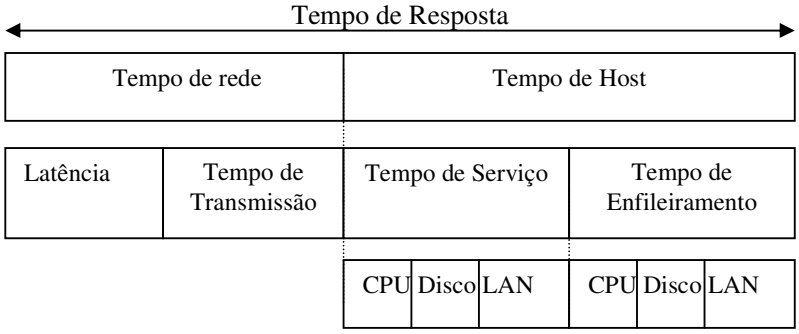


Figura 6.3: Divisão do Tempo de Resposta [MENA02]

O tempo de rede inclui todo o tempo gasto pelas mensagens enviadas entre fonte e destino e é dividida em Latência e Tempo de Transmissão. A latência indica o número de RTT (*Round Trip Times*) envolvidos na troca de mensagens entre fonte e destino. É dependente da natureza dos protocolos utilizados. O tempo de transmissão é o tempo necessário para transmitir todos os bytes entre fonte e destino e depende da largura de banda do enlace mais fraco.

Os pacotes de roteamento, como o AODV e X-AODV, são encapsulados em datagramas IP para envio na rede. No cálculo do tempo de transmissão, deve-se considerar o encapsulamento de pacotes.

As equações para o cálculo do tempo de transmissão entre dois pontos em redes diferentes, foram baseadas nas equações para cálculo do tempo de resposta de [MENA02].

Seja:

- N = número de redes entre os nós *i* (fonte) e *j (destino)*;
- $NHops_n$ = número de *hops* (saltos) entre nós *i* e *j* dentro da rede *n*;
- TamanhoMensagem = tamanho, em bytes, das mensagens trocadas entre os *hosts*;
- MTU_n = MTU (*Maximum Transfer Unit*, tamanho máximo dos dados que trafegam na rede), em bytes, da rede n;
- AODVOvhd = *overhead* do protocolo AODV;
- XAODVOvhd = *overhead* do protocolo X-AODV;
- IPOvhd = *overhead* do protocolo IP;
- MSS = tamanho máximo do segmento, em bytes. Considerando que não há fragmentação, $MSS \leq \min_{n=1}^{N} MTU_n - XAODVOvhd - IPOvhd$;
- FrameOvhd = *overhead*, em bytes, dos frames da rede n;
- $Overhead_n$= *overhead* total (XAODV + IP + FRAME), em bytes, para todos os frames necessários para carregar a mensagem na rede n. É dado por:

  $Overhead_n$=NDatagramas(XAODVOvhd + IPOvhd + FrameOvdh) (Equação 6-3)
- LarguraBanda = largura de banda, em Mbps, da rede n;
- NDatagramas = número de datagramas transmitidos na rede n para carregar uma mensagem. É dado por:

$$NDatagramas = \left\lceil \frac{Tamanho_mensagem}{MSS} \right\rceil \quad \text{(Equação 6-4)}$$

O tempo de serviço de transmissão dentro de uma rede *n* o tempo necessário para envio dos dados (considerando tudo o que é transmitido, inclusive o *overhead*), dividindo-se pela larura de banda da rede. O tempo de transmissão é então dados por:

$$TempoTramissão_n = \frac{(Tamanho_Mensagem + Overhead_n)x8}{10^6 xLarguraBanda_n} \quad \text{(Equação 6-5)}$$

Considerando para esta análise o tempo de resposta de rede (R_{ij}) para uma transmissão entre os nós *i* e *j*, este tempo dve calcular o tempo de transmissão entre todos os saltos, incluindo a latência da rede. Tem-se então:

$$R_{ij} = \sum_{k=1}^{n\ redes} NHops_k \ x(Latência_média_k + Tempo_Transmissão_k) \quad \text{(Equação 6-6)}$$

A Tabela 6-5 mostra os valores de PDU (*Protocol Data Unit*, que é o tamanho máximo de um pacote do protocolo), *Overhead*, e Área máxima de dados, que é a diferença entre PDU e *Overhead*.

Tabela 6-5: Tamanho de pacotes para os protocolos utilizados

Protocolo	Nome do PDU	Tamanho máx. do PDU (bytes)		Overhead (bytes)	Área máxima de dados (bytes)
AODV	Pacote	RREQ	186	186	0
		RREP	155	155	0
		RERR	155	155	0
		RREP-ACK	15	15	0
X-AODV	Pacote	RREQ	190	190	0
		RREP	161	161	0
		RERR	161	161	0
		RREP-ACK	22	22	0
IPv6	Datagrama	65535		40	65495
802.11b	MPDU	1534		34	1500

Como protocolos de roteamento não são projetados para transferência de dados, não existe área de dados. No caso do X-AODV, além do roteamento, um novo campo foi gerado, aproveitando um campo nao utilizado do protocolo (Seção 4.1.4) para a melhorar a troca de dados entre os Clientes.

6.4.1.2 Resultados

Foram comparados os dois protocolos em termos do Tempo de Resposta, para cada tipo de pacote dos protocolos. Estas comparações mostram que os dois protocolos geram dados aproximados, de forma geral a não modificar o comportamento do sistema.

A Tabela 6-6 mostra os principais parâmetros e fatores do sistema.

Tabela 6-6: Parâmetros do cálculo de Tempo de Resposta

Parâmetro	Valor
Número de Hops	5 e 10
Tamanho do MTU	2312
Largura de Banda da rede	2Mbps e 11 Mbps

O Gráfico 6-18 mostra a variação do número de datagramas gerados por cada tipo de pacote, utilizando a Equação 6-4.

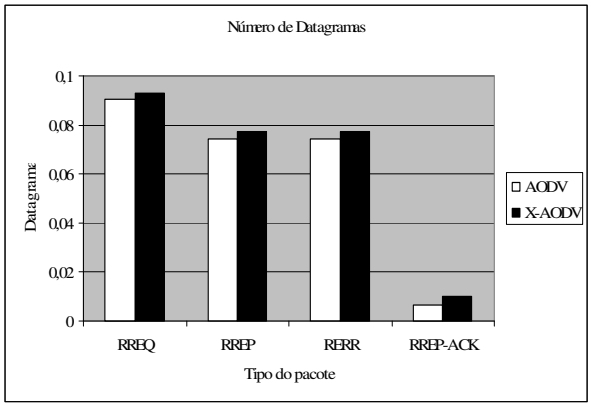


Gráfico 6-18: Número de Datagramas Gerados

A diferença entre o número de datagramas gerados chega a aproximadamente 10%. Isto influi no *overhead* que, pela Equação 6-6, é afetado pelos datagramas gerados. O Gráfico 6-19 mostra que o *overhead* gerado chega a variar 14% entre o AODV e X-AODV.

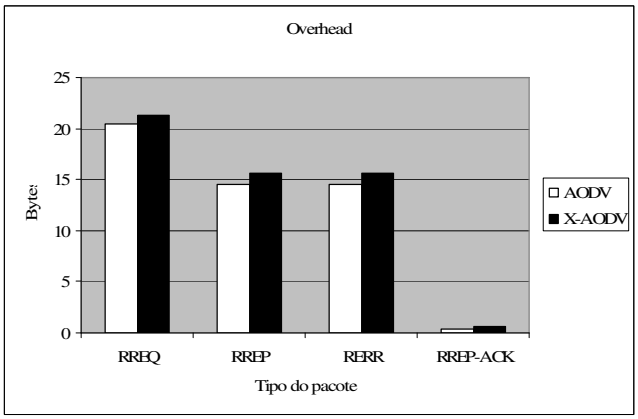


Gráfico 6-19: Overhead gerado por cada protocolo

Calculando o tempo de resposta pela Equação 6-6, verifica-se ainda uma diferença percentual de 15% em uma comunicação a 2Mbps e 12% em 11MBps (Gráfico 6-20). Porém, em valores absolutos, a diferença do tempo de transmissão dos dois protocolos chega, em média dos tipos de pacotes, a $1{,}29 \times 10^{-5}$ e $2{,}38 \times 10^{-5}$ segundos, para 2 e 11Mbps, respectivamente.

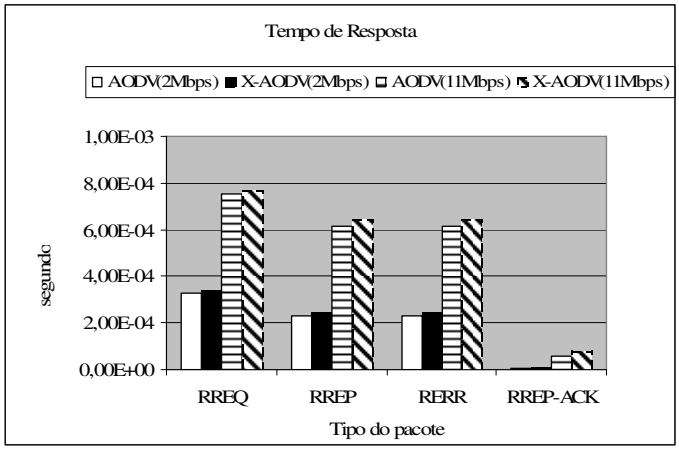


Gráfico 6-20: Tempo de resposta para largura de bandas diferentes (2 e 11 Mbps)

Comparando o tempo de resposta (Gráfico 6-21) com o aumento do número de salto*s* em uma transmissão, de 5 para 10 nós, a variação entre os protocolos chega a 14% e 10% respectivamente. Porém, em valores absolutos, a diferença do tempo de transmissão dos dois protocolos chega em média a $2{,}29 \times 10^{-5}$ e $4{,}77 \times 10^{-5}$ segundos, para 5 e 10 salto*s* na transmissão, respectivamente.

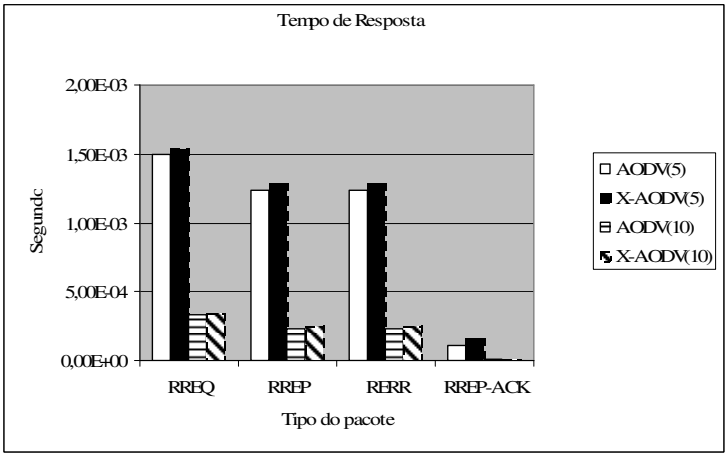


Gráfico 6-21: Tempo de resposta para número de salto*s* diferentes (5 e 10)

Estes valores mostram que, no cômputo geral do tempo de uma transmissão em relação ao aumento dos tamanhos dos pacotes, o protocolo X-AODV gera valores aceitáveis de tempo de resposta, com diferenças pequenas em relação ao protocolo original.

6.4.2 Nós de Capacidade Heterogênea

O objetivo desta Seção é mostrar que nós de menor capacidade computacional tem pior desempenho em suas tarefas quando utilizados como roteadores de tráfegos de alta carga. A métrica utilizada será a Utilização de um nó, dependente do tráfego que o nó repassa.

6.4.2.1 Cálculo da Utilização

Através da Lei da Utilização [MENA02], a utilização de um nó *i* é dado por:

$$U_i = \lambda_i . S_i \quad \text{(Equação 6-7)}$$

onde:

- λ_j = Taxa de chegadas de mensagens no nó *i*;
- Si = Tempo de Serviço do nó i.

Essas variáveis são utilizadas para o cálculo do tempo de serviço de roteamento.

6.4.2.2 Cálculo do Tempo de Serviço de Roteamento

Como visto na Seção 6.4.1.1, o tempo de resposta de uma transmissão depende do tempo de rede como do tempo do *host*. O tempo do *host* é dividido em tempo de serviço e tempo de enfileiramento.

O tempo de serviço (S) é o período no qual uma requisição está recebendo serviço da CPU, disco, segmento de rede ou linha de comunicação. O tempo que uma requisição espera para ter acesso a um recurso é chamado de tempo de espera (W). A soma do tempo total de espera em um recurso por uma dada requisição é o tempo de enfileiramento (Q). O tempo de residência (R_i') de uma requisição em um recurso *i* é a soma do tempo gasto no recurso, as demandas de serviço e os tempos de enfileiramento.

Como visto anteriormente (Seção 3.4), nas redes *ad hoc* todos os *hosts* atuam como roteadores, repassando tráfego para os outros nós. O uso dos nós como roteadores gera maior consumo dos recursos de cada nó ao processar as informações de roteamento. O roteador deve enfileirar as mensagens que chegam, inspecioná-las e repassá-las para os nós adequados.

O tempo do processo de roteamento é conhecido como Latência de Roteamento. O tempo de serviço total de roteamento de um nó é dado pela Equação:

TempoServiçoRoteamento = NPacotes x LatênciaRoteamento (Equação 6-8)

Considerando a Latência de Roteamento como o tempo de serviço (S) gasto pelo nó ao analisar e repassar o pacote, Q como o tempo de enfileiramento das mensagens e W o tempo de espera pelo recurso de envio pela rede, tem-se para um nó *j*:

$$LatênciaRoteamento_j = \sum_{k=1}^{i} \left(S_i + W_i + Q_i \right) \quad \text{(Equação 6-9)}$$

onde i é o número de pacotes de roteamento que o nó processa.

O tempo de serviço (S) de um nó *j* para o roteamento depende da CPU do dispositivo, sendo calculado por:

$$S_j = \lambda_j \text{ x } S_{cpu} \text{ x } S_{lan} \text{ x TamanhoMensagem} \quad \text{(Equação 6-10)}$$

onde:

- λ_j = Taxa de chegadas de mensagens de roteamento no nó *j*;
- $\mu_{cpu} = 1/S_{cpu}$: Taxa de serviço de CPU para análise do roteamento;
- $\mu_{lan} = 1/S_{lan}$: Taxa de serviço da interface de rede para transmissão.

As taxas de serviços de CPU dependem do dispositivo utilizado. A Tabela 6-7 resume as taxas de serviços de CPU para dispositivos das três Classes de Dispositivos em termos literais, para efeitos de comparação. Os valores de utilizados nos cálculosVelocidade de CPU, taxa de CPU e LAN, e geração de mensagens foram tirados de [MENA02].

Tabela 6-7: Taxas de serviços para dispositivos variados

Dispositivo	Classe de Dispositivo	Velocidade da CPU	μ_{cpu}	μ_{lan}
CPU_1	1	V_1	μ_c	μ_l
CPU_2	2	$V_1/2$	$\mu_c/2$	$\mu_l/2$
CPU_3	3	$V_1/3$	$\mu_c/3$	$\mu_l/3$

A próxima sub-Seção mostra os resultados obtidos com a utilização das fórmulas apresentadas.

6.4.2.3 Resultados

Foram comparadas as três categorias (classes 1, 2 e 3) de dispositivos em termos do Tempo de Resposta e Utilização, para diferentes tamanhos de dados transmitidos. A Tabela 6-8 mostra os principais parâmetros e fatores do sistema.

Tabela 6-8: Parâmetros do Cenário 5

Parâmetro	Valor
Taxa de chegada de mensagens	3 msg/s
Tamanho dos dados enviados (bytes)	20, 20.000, 100.000 e 1.000.000
Número de pacotes gerados	0,1, 100, 500 e 50.000

Considerando a latência de roteamento (Equação 6-9) como função apenas do tempo de serviço de roteamento, pode-se verificar no Gráfico 6-22 a variação do tempo em função do tamanho dos dados transmitidos para cada classe de dispositivo.

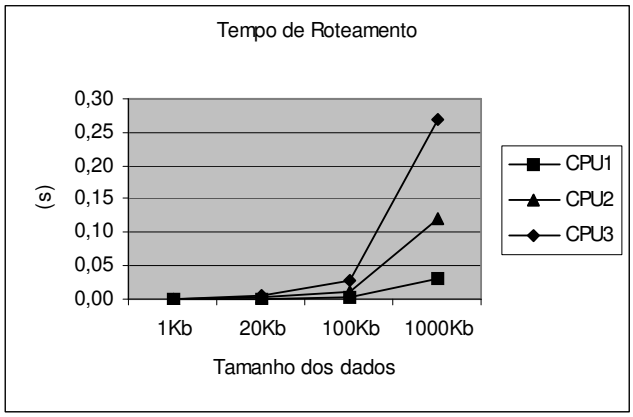


Gráfico 6-22: Tempo de Roteamento de cada tipo de CPU variando o tamanho dos dados roteados

O gráfico mostra diferenças aparentes apenas nos arquivos de 100Kb e 1000Kb, porém os valores obtidos mostram que para todos os tamanhos de arquivos, a CPU_3 possui tempo de roteamento nove vezes maior do que a CPU_1 e 4 vezes maior do que a CPU_2. A CPU_2 gera tempo de roteamento duas vezes e meia maior do que a CPU_1.

Com os valores obtidos do tempo de roteamento, utilizando a Lei da Utilização (Equação 6-7), pode-se observar como as diferentes CPU's se comportam de acordo como tamanho dos arquivos a serem roteados (Gráfico 6-23).

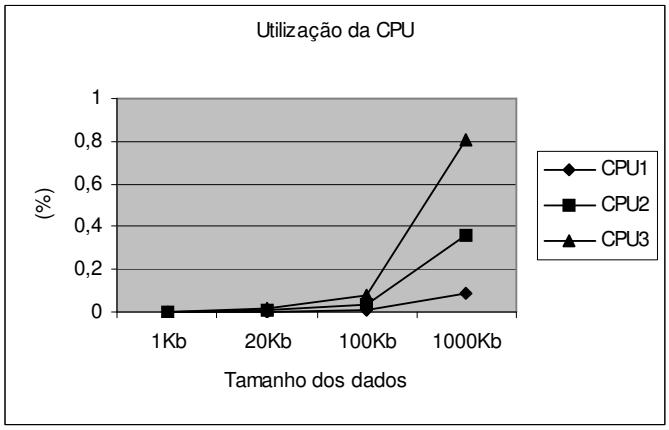


Gráfico 6-23: Utilização de cada tipo de CPU variando o tamanho dos dados roteados

Para arquivos até 100Kb o consumo das CPUs é baixo (até 15%), porém, para arquivos de 1000Kb, a utilização da CPU de menor capacidade chega a 80%, enquanto que as outras CPUs (2 e 1) gastam 38% e 8%, respectivamente.

Isso mostra que uma utilização alta inviabiliza os dispositivos de menor capacidade de processarem seus próprios dados, tendo que gastar mais energia em roteamento do que em seu próprio benefício. Desta forma, o conteúdo que vai ser roteado na rede deve considerar os

dispositivos de classe 3 e ser adaptado para uma classe de conteúdo que possa ainda ser roteado sem prejudicar os participantes do ambiente.

6.4.3 Tempo de Resposta na Adaptação de Conteúdo

Esta Seção tem por objetivo analisar certos aspectos referentes ao Serviço de Adaptação de Conteúdo, mostrado na Seção 5.7, através do uso do Tempo de Resposta do sistema, analisado nos dois sub-itens anteriores.

O tempo de resposta de uma transmissão é o tempo total de transmissão de dados entre sua Fonte e o Destino. No caso do SAC, quer-se analisar como alguns dos parâmetros do ambiente influenciam no tempo de resposta total de uma transmissão.

6.4.3.1 Tempo de Resposta *versus* Modalidade de Adaptação

Esta Sub-Seção tem por objetivo mostrar que o uso de conteúdo adaptado diminui o tempo de resposta do sistema. Utilizou-se um cenário com 7 nós posicionados em linha, onde o nó i se conecta aos nós *i+1* e *i-1*, sendo a Fonte o nó 0 e o último, nó 6, o destino.

A Tabela 6-9 mostra, para vídeo e áudio, os parâmetros do conteúdo.

Tabela 6-9: Parâmetros de Vídeo e Áudio

Conteúdo	Formato	Tempo (min)	Taxa	Resolução
Vídeo	Original	5	25 fps	300x300
	Adaptado	5	15 fps	200x200
Áudio	Original	5	25 kbps	---
	Adaptado	5	15kbps	---

No Gráfico 6-24 tem-se a comparação, para o vídeo, como o conteúdo original e adaptado influenciam no tempo de resposta.

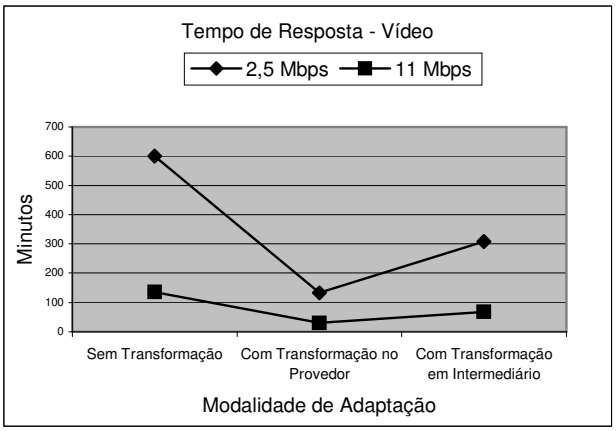


Gráfico 6-24: Tempo de Resposta de transmissão de vídeo para três modalidades diferentes

A transmissão sem adaptação gera tempo de resposta superiores às outras modalidades de adaptação, pois envia o vídeo com maior carga. Ao reduzir o conteúdo original no próprio provedor, adaptando a um certo cliente, tem-se uma redução muito grande no tempo de resposta total. Quando a transformação é realizada em um nó intermediário, no exemplo, nó 3, a redução é menor, mas ainda tem tempo de resposta menor do que no caso sem a transformação.

O Gráfico 6-25 mostra o mesmo experimento para um caso de áudio, que foi transmitido no formato original, sendo transformado no próprio provedor e no nó intermediário. O comportamento do tempo de resposta é o mesmo do caso anterior.

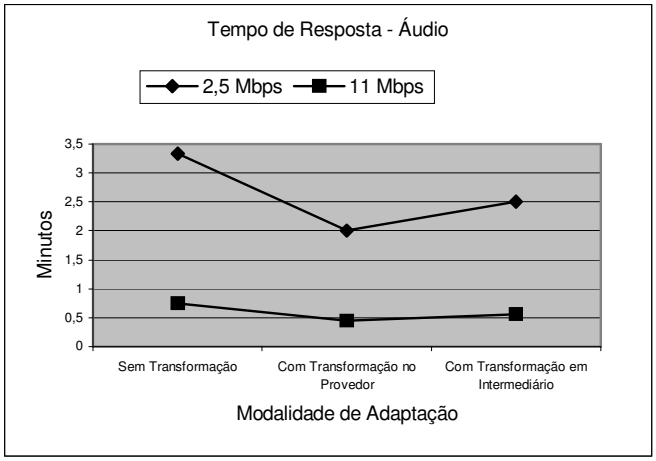


Gráfico 6-25: Tempo de Resposta de transmissão de áudio para três modalidades diferentes

Portanto, conclui-se que a adaptação melhora o tempo de resposta do sistema ao reduzir a carga de dados que trafega na rede. A adaptação no próprio nó provedor de conteúdo oferece menor tempo de resposta do que a adaptação em um nó intermediário, pois o tráfego que chega até o nó adaptador possui maior carga, sendo reduzida apenas depois desse nó.

O próximo cenário mostra como o sistema se comporta utilizando diversos locais de adaptação.

6.4.3.2 Tempo de Resposta *versus* Local de Adaptação

Este cenário mostra como a escolha do local onde o conteúdo será adaptado influencia no tempo de transmissão. O cenário é composto de 8 nós, sendo que cada nó i somente se comunica com os nós *i-1* e *i+1*, tal qual o cenário anterior.

A Tabela 6-10 mostra os parâmetros para áudio e vídeo antes e depois da adaptação.

Tabela 6-10: Parâmetros de Áudio e Vídeo

Conteúdo	Formato	Tempo (min)	Taxa	Resolução
Vídeo	Original	5	25 fps	300x300
	Adaptado	5	15 fps	200x200
Áudio	Original	5	25 kbps	---
	Adaptado	5	15kbps	---

O Gráfico 6-26 mostra como o sistema se comporta ao mudar o local de adaptação. A adaptação do conteúdo no próprio provedor é mais vantajosa do que a escolha de um nó mais afastado. A geração do vídeo com seus parâmetros originais, enviado pelo provedor, e adaptado para os novos parâmetros em outros nós, faz o tempo de resposta crescer quanto mais longe do nó fonte o conteúdo chega em tamanho original.

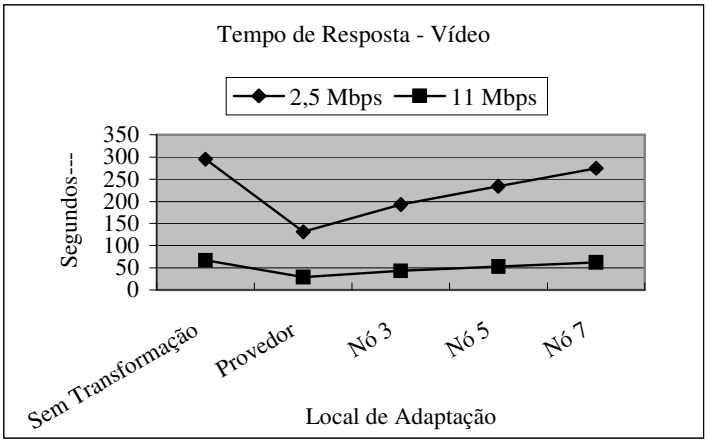


Gráfico 6-26: : Tempo de Resposta de transmissão de vídeo para locais de adaptação diversos

O mesmo comportamento ocorre na transmissão de áudio (Gráfico 6-27).

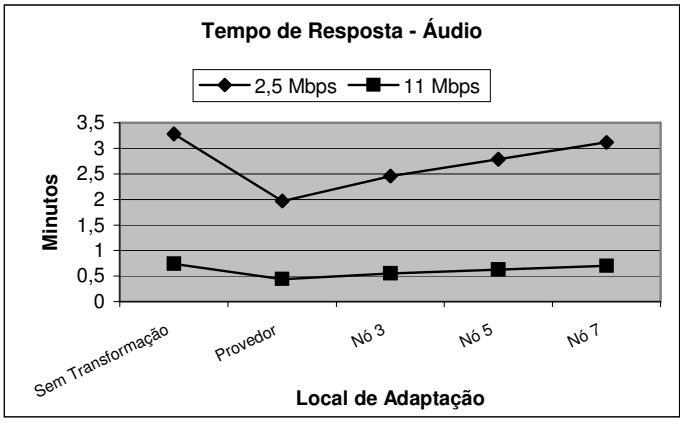


Gráfico 6-27: Tempo de Resposta de transmissão de áudio para locais de adaptação diversos

Portanto, é melhor adaptar o conteúdo (para outro formato ou transformação de modalidade) no próprio provedor do que enviar o conteúdo para outro nó apenas mudar os parâmetros de qualidade. Isto justifica a escolha do provedor como local de adaptação, mesmo tendo que mudar os formatos, do que enviar para outro nó, como mostrou a Seção 5.7.1.3. A escolha de outro nó deve ser realizada quando o próprio provedor não tiver capacidade de fornecer o conteúdo transformado. Quando outro nó for escolhido, deve ser o nó mais próximo do provedor de conteúdo.

6.4.3.3 Tempo de Resposta *versus* Uso de Classe de Adaptação máxima

Este cenário mostra como a escolha da Classe de Adaptação influencia o tempo de resposta (tempo de roteamento, visto na Seção 6.4.2) em um determinado nó.

O cenário utilizou vídeo e áudio, com três Classes de Adaptações máximas para cada conteúdo, como mostra a Tabela 6-11.

Tabela 6-11: Parâmetros de Áudio e Vídeo para cada Classe de Adaptação

Conteúdo	Classe de Adaptação Máxima	Tempo (min)	Taxa (kbps)
Vídeo	1	5	500
	2	5	128
	3	5	28
Áudio	1	5	50
	2	5	25
	3	5	8

O Gráfico 6-28 mostra o tempo de resposta (roteamento) de um vídeo, em cada CAmáx selecionada. Quanto maior a CA utiliza, maior a carga de dados do vídeo, e por conseqüência, maior tempo de transmissão.

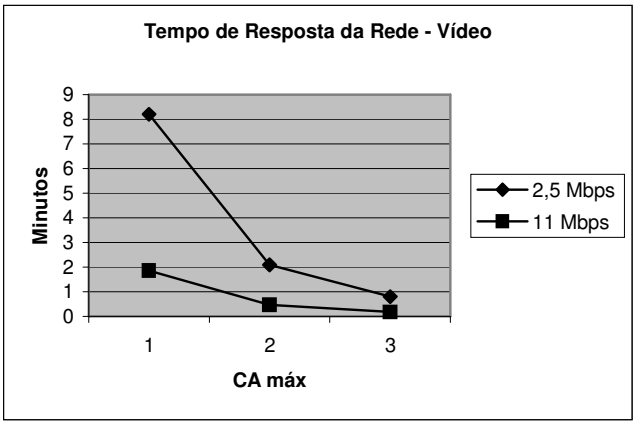


Gráfico 6-28: Tempo de Resposta de transmissão de vídeo de acordo com CAmáx

Da mesma forma, o tempo de resposta do áudio é afetado por diferentes CA's (Gráfico 6-29).

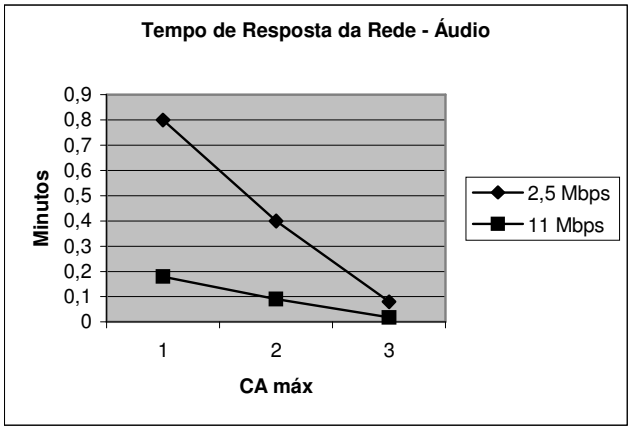


Gráfico 6-29: Tempo de Resposta de transmissão de áudio de acordo com CAmáx

Para melhor visualizar o efeito da CA escolhida em diferentes tipos de dispositivos, foram realizados cálculos do tempo de roteamento utilizando diferentes CPU's, uma de cada Classe de Dispositivo, significando que a CPU 1 > CPU 2 > CPU 3 em termos de processamento, como na Seção 6.4.2.3.

O Gráfico 6-30 mostra como a $CA_{máx}$ = 1 possui maior tempo de resposta em comparação com as outras CA's.

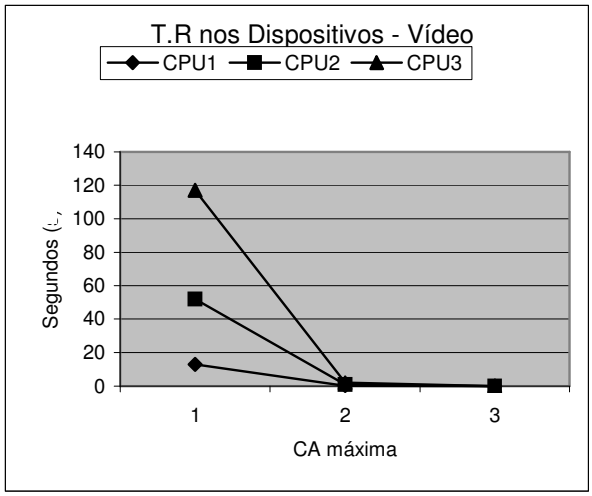


Gráfico 6-30: Tempo de Resposta de roteamento de vídeo de acordo com CAmáx

Para melhor visualizar o efeito da escolha da CA em cada tipo de dispositivo, o Gráfico 6-31 detalha, para cada CPU, o tempo de resposta de roteamento. Notadamente a CPU classe 3 é o dispositivo mais prejudicado nesta transmissão.

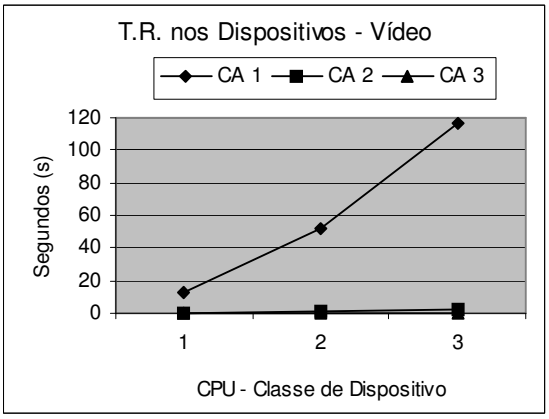


Gráfico 6-31: Tempo de Resposta de roteamento de vídeo de acordo com CPU

O mesmo comportamento ocorre na transmissão de um áudio. O Gráfico 6-32 e o Gráfico 6-33 mostram como a escolha da $CA_{máx}$ afeta o desempenho, prejudicando principalmente a CPU de menor poder computacional.

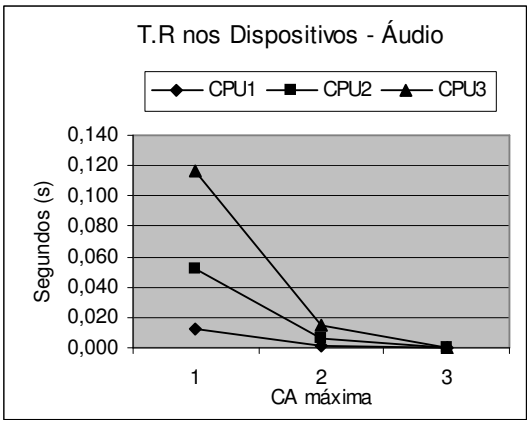


Gráfico 6-32: Tempo de Resposta de roteamento de áudio de acordo com CAmáx

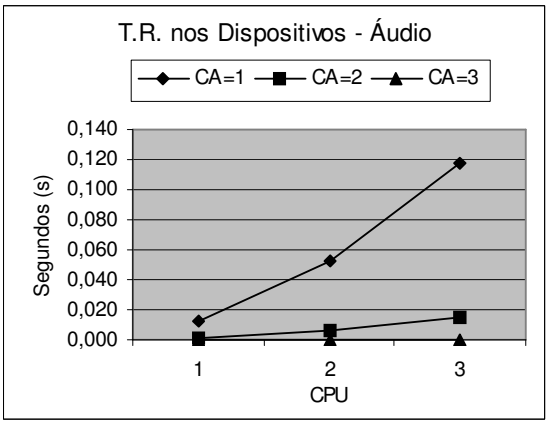


Gráfico 6-33: Tempo de Resposta de roteamento de áudio de acordo com CPU

Como na Seção 6.4.2, que mostrou que os dados transmitidos na rede e que devem ser repassados pelos nós intermediários afetam o desempenho dos nós de baixa capacidade, conclui-se que a transmissão de dados sem limites de qualidade pode prejudicar os nós intermediários. A limitação da qualidade, através de um parâmetro como a $CA_{máx}$ ajuda os nós de baixa capacidade, permitindo que eles não sejam prejudicados por altas cargas de tráfego.

6.5 Conclusões

Neste Capítulo foram mostrados diversos aspectos referentes ao ambiente de trabalho de campo: localização física, adaptação de conteúdo, mobilidade, estratégias de atualização, roteamento e uso de dispositivos.

O uso da divisão da área de trabalho em diversas sub-áreas contribui para a melhora do desempenho do sistema, gerando menores valores de atraso de transmissão de dados e maior taxa de pacotes recebidos.

Foi mostrado que a mobilidade dos nós do ambiente afeta a carga de dados que trafega no ambiente. A classificação dos nós de acordo com sua mobilidade gera menores custos em termos de carga de trabalho.

O uso de uma estratégia híbrida de atualização de localização física, onde se varia a forma de atualização de informações, também economiza recursos do ambiente quando utilizada de acordo com a classificação de mobilidade dos nós.

O experimento de roteamento mostrou que o X-AODV possui desempenho pior do que o AODV quando dentro de uma área de trabalho. Porém, a diminuição do escopo de transmissão da mensagens para apenas dentro de uma mesma área gera um melhor desempenho para o ambiente de trabalho. Isto compensa o fato do protocolo ter menor desempenho quando da comunicação entre duas áreas, pois o maior tráfego de dados é entre os Clientes de AIs de mesmo interesse.

Na modelagem, foi mostrado que o protocolo X-AODV possui desempenho semelhante ao do protocolo AODV original. Mesmo gerando uma pequena variação nos números de tempo de resposta do sistema, a vantagem do X-AODV é o uso de alguns bytes dos pacotes de roteamento para troca de informações entre os nós que compartilham o ambiente.

Viu-se que o uso de dispositivos de menor capacidade no meio de rotas pode afetar seu desempenho. O uso de sua capacidade de processamento para envio de dados de outros dispositivos gera altos custos de uso de seus recursos.

A escolha do local de adaptação também afeta o desempenho do sistema. Foi visto que é melhor gerar conteúdo adaptado, no provedor ou em nós intermediários, para diminuir a carga do sistema e melhorar o tempo de resposta de transmissão.

A escolha de um nó mais próximo da fonte para realizar a adaptação também trás vantagens ao ambiente de trabalho, pois o conteúdo original percorre menor caminho na rede, sendo depois reduzida sua carga ao ser adaptado.

O uso de uma Classe de Adaptação máxima também reduz o tempo que os nós intermediários são utilizados para roteamento. Viu-se que quanto maior a qualidade do conteúdo, maior o tempo gasto em roteamento. Também foi mostrado que quanto menor a capacidade da CPU utilizada como roteador, maior o tempo gasto pela CPU no processo de roteamento, e por conseqüência, maior utilização da CPU. Portanto, a adaptação de conteúdo visando à capacidade de dispositivos intermediários melhora o desempenho destes dispositivos.

Capítulo 7

Conclusões

7.1 Conclusões Gerais

Este trabalho contribuiu em diversos aspectos para a melhoria dos ambientes de trabalho de campo e gera novos desafios para os desenvolvedores de aplicações voltadas a esse tipo de ambiente e de outras áreas de estudos.

7.1.1 Contribuições

As principais contribuições deste trabalho podem ser resumidas em:

- Uso de classes de dispositivos

Essa classificação diferencia os vários tipos de usuários do sistema e facilita os serviços de adaptação de conteúdo e é usada como contexto para a geração de rotas e como parâmetro no oferecimento de serviços.

- Uso de classes de mobilidade

Essas classes permitem que diferentes padrões de atualização de informações de localização física sejam utilizados por dispositivos de mobilidade variada.

- Um esquema de atualização de informações de localização física baseada nos diferentes tipos de mobilidade e prioridades dos Clientes do ambiente

O uso de diferentes esquemas de atualização dependente da classe do usuário permite que o ambiente aumente a porcentagem de pacotes recebidos com sucesso e reduza a carga de sinalização gerada e os atrasos impostos ao tráfego.

- Negociação de adaptação

O uso da classe de dispositivo juntamente com a classe da adaptação facilita a indicação do conteúdo mais adequado para cada classe de Cliente do ambiente. Uma estratégia de Combinação, que verifica os parâmetros dos dispositivos, e os combina com o conteúdo a ser enviado e a capacidade adaptativa do servidor, juntamente com a escolha do melhor local onde a adaptação deve ocorrer, trás ganhos de desempenho ao sistema, diminuindo o tempo de resposta do envio dos dados e diminuindo a utilização de nós intermediários com roteamento.

- Uso de informações de localização física para geração da divisão da área de trabalho por contextos de localização.

A redução do escopo das mensagens enviadas na área de trabalho é reduzida para a Área de Interesse do nó. Essa redução de área permite que o ambiente contextualize os trabalhadores, enviando mensagens de seu contexto. Também reduz a carga total da rede, pois mensagens recebidas fora de sua AI não são repassadas para outros nós, pois serão descartadas pelos nós Borda de outras AI's.

- Um protocolo de roteamento baseado em contexto

O protocolo X-AODV, uma evolução de outro protocolo existente, o AODV, roteia as informações baseando-se em informações de localização física e contexto de dispositivos através de:

  - Conhecimento dos vizinhos;
  - Conhecimento da capacidade dos nós presentes em cada rota; e
  - Geração de enlace*s* entre nós mais próximo fisicamente dos nós alvo das rotas.

Este protocolo realiza roteamento evitando os dispositivos de baixa capacidade e indica ao serviço de adaptação que reduz a qualidade do conteúdo a ser enviado devido aos nós de baixa capacidade presentes na rota. Isto permite que esses nós mais fracos não sejam prejudicados ao repassar tráfego para os outros nós.

Estas contribuições ao trabalho de campo também podem ser utilizadas por qualquer ambiente de computação pervasiva para melhora dos serviços prestados aos usuários.

7.1.2 Trabalhos Futuros

Este trabalho poderá servir de base para o desenvolvimento de novos estudos, principalmente para os relacionados a seguir:

- O armazenamento das informações desatualizadas permite a utilização do repositório por um serviço de Gerenciamento de Mobilidade, que pode utilizar técnicas conhecidas de previsão de movimentação para
  - notar a mudança de classe de mobilidade para modificar a classe de mobilidade do nó;
  - realizar a disseminação de informações com antecedência;
  - oferecer serviços avançados de adaptação de conteúdo, isto é, prevendo-se a movimentação do Cliente por diferentes área de contexto, adaptar previamente o conteúdo, de acordo com o contexto previsto;

- Ampliação da arquitetura para o uso concorrente com diferentes tecnologias de localização e redes;
- Utilização das classes de prioridade para otimizar o tráfego de pacotes;
- Estudo das técnicas de adaptação (Abstração de informação, Priorização de dados, Transformação de modalidade, *Data transcoding e* Classificação de Propósitos) para descobrir qual a técnica mais adequada dependendo da capacidade do dispositivo provedor de serviços. Da mesma forma que o roteamento por nós de baixa capacidade precisa ser adaptado, quando um nó dessa classificação é um provedor de conteúdo, as diferentes técnicas de adaptação podem gerar diferentes usos de sua capacidade. Portanto, devem existir técnicas de transformação e geração de conteúdo mais adequadas ao perfil de cada classe de dispositivos;
- O desenvolvimento de um esquema de balanceamento de carga, que possa ajudar o ambiente escolher nós com menor utilização para uso no processo de adaptação de conteúdo em outros nós; e
- Refinamento do protocolo X-AODV, para tratar do controle de potência da transmissão dependendo do contexto dos dados a serem transferidos. Por exemplo, a potência de transmissão, para todos os vizinhos ou vizinho específico, em função da distância entre o nó fonte e o nó destino;

Referências Bibliográficas

[ABDE99] ABDELZAHER, Tarek; BHATTI, Nina. Web Content Adaptation to Improve Server Overload Behavior. **Computer Networks**, v. 31, n.16, p.1563-1577, 1999.

[AGRE02] AGRE, J. et al. A Layered Architecture for Location-based Services in Wireless Ad Hoc Networks. Networks. In: IEEE Aerospace Conference, 2002, Big Sky (EUA). **Proceedings...** 2002, v.3, p.1085-1097.

[ALOU96] ALOUINI, Mohamed-Slim. GPS: An Overview. **Tunisian Scientific Magazine**, 1996. Disponível em: <http://citeseer.nj.nec.com/31114.html>. Acesso em: 01/01/2002.

[BANA00] BANAVAR, Guruduth et al. Challenges: An Application Model for Pervasive Computing. In: International Conference on Mobile Computing and Networking, 2000, Boston (EUA). **Proceedings...** New York: ACM Press, 2000. p.266 – 274.

[BAR96] BAR-NOY, A. et al. Topology-based Tracking Strategies for Personal Communication Networks. **ACM-Baltzer Mobile Networks and Nomadic Applications**, v.1, n.1, p.49-56, 1996.

[BETT00] BETTSTETTER, Christian; RENNER, Christoph. A Comparison of Service Discovery Protocols And Implementation of the Service Location Protocol. In: EUNICE Open European Summer School, 2000, Twente (Holanda). **Proceedings...** Set. 2000.

[BHAR02] BHARGAVA, B. Guest Editorial: Quality of Service in Multimedia Networks. **Mutimedia Tools and Application**, v. 17, n. 2-3, jul. 2002.

[BHAT01] BHATTACHARYA, A. et al. Towards a Novel Architecture to Support Universal Location Awareness. In: Workshop on Pervasive Computing and Information Logistics, 2001, Viena (Áustria). **Proceedings…** Set. 2001, vol.1, p. 182-189.

[BROW96] BROWN, Martin G. Supporting User Mobility. In: World Conference on Mobile Communications, 1996, Canberra (Austrália). **Proceedings…** Set. 1996, p.69-77.

[BROWN96] BROWN, P.J. The Stick-e document: a framework for creating context-aware applications. Electronic Publishing: Origination, Dissemination and Design, 1996, Palo Alto (EUA). **Proceedings…** 1996, p.259-272.

[CHAK00] CHAKRABORTY, Dipanjan; CHEN, Harry. Service Discovery in the Future for Mobile Commerce. **Crossroads**, v.7, n.2, p.18-24, dez. 2000.

[CHAL99] CHALMERS, Dan; SLOMAN, Morris. A Survey Of Quality Of Service In Mobile Computing Environments. **IEEE Communications Surveys**. Segundo trimestre, 1999. Disponível em:
<http://www.comsoc.org/pubs/surveys>. Acesso em: 01/01/2003

[CHEN00] CHEN, Guanling; KOTZ, David. A Survey of Context-Aware Mobile Computing Research. **Relatório Técnico**. Department of Computer Science, Dartmouth College (EUA).

Novembro, 2000. Disponível em: <ftp://ftp.cs.dartmouth.edu/TR/TR2000-381.ps.Z>. Acesso em: 01/04/02

[CHEN02] CHEN, K; SHAH, S.H.; NAHRSTEDT, K. Cross Layer Design for Data Accessibility in Mobile Ad Hoc Networks. **Wireless Personal Communications: An International Journal,** Hingham (EUA), v.21, n.1, p-49-75, abr. 2002.

[CORS01] PARK, Corson. Temporally Ordered Routing Algorithm (TORA) Version 1: Functional Specification. **Internet Draft**, jul. 2001. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002.

[CORS99] M. S. Corson et al. An Internet MANET Encapsulation Protocol (IMEP) Specification. **Internet Draft**, ago.1999. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002.

[DANA00] DANA, Peter H. Global Positioning System Overview. **Relatório Técnico**. Department of Geography, University of Texas at Austin, Janeiro, 2000. Disponível em: <http://www.colorado.edu/geography/gcraft/notes/gps/gps.html>. Acesso em: 01/05/2003.

[DECI04] **Decimal Degrees Conversion Worksheet**. Disponível em: <http://www.geo.msu.edu/geo221/deconv.htm>. Acesso em 01/07/2004.

[DEY99] DEY, Anind K; ABOWD, Gregory D. Towards a Better Understanding of Context and Context-Awareness. In: International symposium on Handheld and Ubiquitous Computing, 1999, Karlsruhe (Alemanha). **Proceedings...** Londres: Springer-Verlag, 1999. P.304-307.

[DROM97] DROMS, R. Dynamic Host Configuration Protocol. **IETF Request For Comments**, mar.1997. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002

[EBLI01] EBLING, Maria R.; HUNT, Guerney D. H.; LEI, Hui. Issues for Context Services for Pervasive Computing. **Alcatel Telecommunications Review**, p.71-76, jan. 2001.

[FOX98] FOX, Armando et al. Adapting to Network and Client Variation Using Infrastructural Proxies: Lessons and Perspectives. **IEEE Personal Communications**. V.5, n.4, p.10-19, ago. 1998.

[GAED98] GAEDKE et al. Web Content Delivery to Heterogeneous Mobile Platforms. Lecture Notes In Computer Science. In: Workshops on Data Warehousing and Data Mining: Advances in Database Technologies, 1998. **Proceedings…** Londres: Springer-Verlag, 1998, v.1552, p.205 – 217.

[GPS03] GPS - Princípio Básico. Disponível em: http://www.redemegatron.com.br/protesat/gps.htm. Acessado em: 05/05/2003.

[GRIB00] GRIBBLE, S. et al. The Ninja Architecture for Robust Internet-scale Systems and Services. **IEEE Computer Networks**, Special Issue on Pervasive Computing, v.35, n. 4, p.473-497, 2001.

[GRIM00] GRIM, Robert et al. A System Architecture for Pervasive Computing**.** 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system, 2000, Kolding (Dinamarca). **Proceedings…** New York: ACM Press, 2000. p.177-182.

[HART99] HARTER, A. et al. The Anatomy of a Context-Aware Application. In: ACM/IEEE international conference on Mobile Computing and Networking, 1999, Seattle (EUA). **Proceedings...** New York: ACM Press, 1999, p.59-68.

[HAYE03] HAYES, Nicki. Locating Your Location Based Service Provider. **Wireless Developer Network**. Disponível em: <http://www.wirelessdevnet.com/channels/lbs/>. Acesso em 30/12/2003

[HENR02] HENRICKSEN, Karen; INDULSKA, Jadwiga; RAKOTONIRAINY, Andry. Modeling Context Information in Pervasive Computing Systems. Lecture Notes In Computer Science. In: First International Conference on Pervasive Computing, 2002, Berlim. **Proceedings…** Londres: Springer-Verlag, 2002. p. 167–180.

[HEUS03] HEUSSE, Martin et al. Performance Anomaly of 802.11b. In: INFOCOM, 2003, São Francisco (EUA). **Proceedings…** Mar. 2003.

[ISO04] International Organization for Standardisation. **Overview of the MPEG Standard**. Disponível em:
http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm. Acessado em: 01/12/2005.

[JAIN92] JAIN, Raj. **The Art of Computer Systems Performance Analisys**. EUA: Wiley & Sons, 1992.

[JIAN02] JIANG, X.; CAMP, T. A Review of Geocasting Protocols for a Mobile Ad Hoc Network. In Grace Hopper Celebration, 2002, Vancouver (Canadá). **Proceedings…** 2002.

[JOHN03] JOHNSON, David B.; MALTZ, David A.; Hu, Yih-Chun. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. **Internet Draft**, abr. 2003. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002

[JOHN04] Johnson, Thienne; Sadok, Djamel. A Lightweight Architecture To Provide Information Dissemination Throughout Non-Infrastructured Fieldwork Environments. In: Mobility Aware Technologies and Applications, 2004. **Proceedings…** Campinas, 2004.

[JOHN05] Johnson, Thienne; Sadok, Djamel. A Lightweight Pervasive Architecture for Fieldwork Environments. IN: IEEE Wireless. Communications & Networking Conference, Nova Orleans (EUA), 2005. **Proceedings**... Nova Orleans, 2005.

[KAIN02] KAIN, Carl W. Location-Based Wireless Services: Finding People Everywhere. **Σ-Mitretek Technology Summaries**. Primavera, 2002. Disponível em
<http://www.mitretek.org/pubs/Sigma_Pubs_spring02/>. Acesso em 01/05/2003.

[KAMI03] KAMINSKY, Alan; BISCHOF, Hans-Peter. New architectures, protocols, and middleware for ad hoc collaborative computing. In: Middleware 2003 Workshop on

Middleware for Pervasive and Ad Hoc Computing, 2003, Rio de Janeiro. **Proceedings…** 2003.

[KHOK01] KHOKHAR, Sarfraz. Which Location-Estimation Technologies Meet Market Needs? **Bussiness Geographics**, Março/Abril, 2001. Disponível em: <http://www.geoplace.com/bg/2001/0401/0401tt.asp>. Acesso em:05/05/2003.

[KISH01] KISHAN, Arun et al. Halibut: An Infrastructure for Wireless LAN Location-Based Services. **Relatório Técnico**, 2001. Disponível em: <http://fern2.stanford.edu/cs444n/>. Acesso em 01/03/2003.

[KO00] KO, Young-Bae; VAIDYA, Nitin H. GeoTORA: A Protocol for Geocasting in Mobile Ad Hoc Networks. **Relatório Técnico,** Department of Computer Science, Texas A&M University, mar. 2000. Disponível em: <citeseer.ist.psu.edu/article/ko00geotora.html>. Acesso em: 01/12/2003.

[KOKR00] KORKEA-AHO, Mari. Context-Aware Applications Survey, 25.04.2000. **Relatório Técnico**. Department of Computer Science, Helsinki University of Technology, abr. 2000. Disponível em: <htto:llwww.hut.fi/~mkorkeaa/doc/context-aware.html>. Acesso em: 01/03/2003.

[LEI01] LEI, Zhijun; GEORGANAS, Nicolas D. Context-based Media Adaptation in Pervasive Computing. In: Canadian Conference on Electrical and Computer Engineering, 2001, Toronto (Canadá). **Proceedings…** Maio, 2001.

[LEON96] LEONHARDT, U. et al. Location Service in Mobile Computing Environments. **Computers and Graphics**, v. 20, n.5, 1996.

[LI01] LI, Qun; ASLAM, Javed; RUS, Daniela. Online Power-aware Routing in Wireless Adhoc Networks. In: International Conference on Mobile Computing and Networking. 2001, Roma. **Proceedings …** 2001, p.97 – 107.

[LIAO01] LIAO, Wen-Hwa; TSENG, Yu-Chee; SHEU, Jang Ping. GRID: A Fully Location-Aware Routing Protocol for Ad Hoc Networks. **Telecommunication Systems**, v.18, n.1-3, p.37-60, 2001.

[LUM02] LUM, W.Y.; LAU, F.C.M. A Context-Aware Decision Engine for Content Adaptation. **IEEE Pervasive Computing**, v.1, n.3, p.41-49, jul. 2002.

[MACB01] MACBETH, Adam. An auto-configuring server-based discovery system. **Relatório Técnico**, maio 2001. Disponível em: <http://www.cs.nyu.edu/rgrimm/one.world/papers.html>. Acesso em: 01/05/04.

[MAIH04] MAIHÖFER, Christian. A Survey of Geocast Routing Protocols. **IEEE Communications Surveys and Tutorials**, v.6, n.2. 2º Bimestre, 2004. Disponível em: <http://www.comsoc.org/livepubs/surveys/public/2004/apr/maihofer.html>. Acesso em: 01/06/2004.

[MCGR00] MCGRATH, Robert E. Discovery and Its Discontents: Discovery Protocols for Ubiquitous Computing. **Relatório Técnico**, 2000, University of Illinois at Urbana-Champaign, Champaign (EUA). Disponível em: http://www.cs.uiuc.edu/Dienst/UI/2.0/Describe/ncstrl.uiuc_cs/UIUCDCS-R-2000-2154. Acesso em: 15/02/2002

[MELE03] Mele, Alexandre; Endler, Markus. Um Framework para a Simulação de Redes Móveis Ad hoc. **Monografias em Ciência da Computação**, Departamento de Informática, PUC-Rio, Setembro 2003.

[MENA02] MENASCÉ, Daniel; ALMEIDA, Virgílio. **Capacity Planning for Web Services: Metrics, Models and Methods.** Upper Saddle River: Prentice Hall. 2002.

[MMV04] MMV Lab - Multimídia and Vision Laboratory, University of London. **Scalable Video Coding**. Disponível em: <http://www.elec.qmul.ac.uk/mmv/svc.html>. Acesso em: 01/12/2004.

[MOCK87] MOCKAPETRIS, P. Domain Names – Concepts and Facilities. **IETF Request For Comments**, nov. 1987. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002

[MOHA03] MOHAPATRA, Prasant; GUI, Chao. Qos in Mobile Ad Hoc Networks. **IEEE Wireless Communications**, v.10, n.3, p. 44-52, jun. 2003.

[MONA98] Monarch. **The CMU Monarch Project's Wireless and Mobility Extensions to NS, Extensions to NS**. 1998. Disponível em: <http://www.monarch.cs.cmu.edu/>. Acesso em 10/08/2001.

[NARE99] NARENTHIRAN, K.; TAFAZOLLI, R.; EVANS, B.G. Simple Positioning Method For Location Tracking in Mobile Satellite Communication Systems. 1999. In: 18th AIAA International Communication Satellite Systems Conference, 2000, Oakland (USA). **Proceedings…** Abr. 2000.

[NIST00] NIST. **Smart Spaces and Their Related Areas**, 2000. Disponível em: <http://www.nist.gov/smartspace/smartSpaces/>. Acesso em: 01/02/2002.

[NS00] **The Network Simulator**. Disponível em: http://www.isi.edu/nsnam/ns/. Acesso em: 30/01/2001.

[PARK97] PARK, V. D.; CORSON, M. S. A highly adaptive distributed routing algorithm for mobile wireless networks. In**:** Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution. 1997, Washington. **Proceedings...** 1997. p. 1405-1413.

[PERK03] PERKINS, C.; BELDING-ROYER, E. e DAS, S. Ad hoc On-Demand Distance Vector Routing. **IETF Request For Comments**, jul. 2003. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002

[PERK94] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: ACM SIGCOMM Conference. **Proceedings…** Ago. 1994, p.234–244.

[PHAN01] PHAN, Thomas; GUY, Richard G.; BAGRODIA, Rajive. A Scalable, Distributed Middleware Service Architecture to Support Mobile Internet Applications. **Wireless Mobile Internet**, v.9, n.4, p.27-33, jul. 2001.

[PHANT01] PHAN, T. et al. Handoff of Application Sessions Across Time and Space. In: IEEE International Conference on Communications, 2001, Helsinki (Finlândia). **Proceedings…** 2001, v.5, p.1367-1372.

[PUC03] PUC-Rio. Curso de Física Introdutória 1999. **Relatório Técnico.** Disponível em: <http://www.fis.puc-rio.br/fis_intr/gps.htm>. Acesso em: 01/09/2003.

[ROTH02] ROTH, Jörg. A Communication Middleware for Mobile and Ad-hoc Scenarios. In: International Conference on Internet Computing. 2002, Las Vegas (USA). **Proceedings…** 2002.,[S.I.]: CSREA, v.I, p.77-84.

[RYAN98] RYAN, R.; PASCOE, J.; MORSE, D. Enhanced Reality Fieldwork: the Context Aware Archaeological Assistant. **Computer Applications in Archaeology**. Oxford: Tempus Reparatum, 1998.

[RYAN99] RYAN, N.; PASCOE, J.; MORSE, D. R. FieldNote: a Handheld Information System for the Field. In: First International Workshop on TeloGeoProcessing, 1999, Lion (França). **Proceedings…** Lion: Claude Bernard University of Lyon, 1999, p.156-163.

[SATY01] SATYANARAYANAN, M. Pervasive Computing: Vision and Challenges, **IEEE Personal Communications**, vol. 8, n. 4, p10-17, ago. 2001.

[SMIT99] SMITH, J.R.; MOHAN, R.; LI, C.S. Scalable Multimedia Delivery for Pervasive Computing, In: ACM Multimedia, Orlando (EUA), 1999.**Proceedings**, 1999, p. 131-140.

[SRIV04] SRIVATSA, Mudhakar; GEDIK, Bugra; LIU, Ling. Scaling Unstructured Peer-to-Peer Networks With Multi-Tier Capacity-Aware Overlay Topologies. In: Tenth International Conference on Parallel and Distributed Systems. 2004, Newport Beach (EUA). **Proceedings.** Jul. 2004.

[STIL99] STILLER, B. et al. A Flexible Middleware for Multimedia Communication: Design, Implementation, and Experience. **IEEE Journal on Selected Areas in Communications**, v.17, n.9, p1580–1598, set. 1999.

[SUN99] SUN Microsystems. **Jini Technology Architectural Overview**, jan. 1999. Disponível em:
<http://www.sun.com/software/jini/whitepapers/architecture.html.> Acessado em: 01/01/04

[THAM03] THAM, Chen-Khong; JIANG, Yuming; GAN, Yung-Sze. Layered Coding for a Scalable Video Delivery System. In: IEEE/EURASIP Packet Video, 2003, Nantes, França. **Proceedings...** França, 2003.

[TRIP04] TRIPCEVICH, Nicholas. Flexibility by Design: How Mobile GIS Meets the Needs of Archaeological Survey. **Cartography and Geographic Information Science**, v. 31, n. 3, p. 137-151, 2004.

[UEVO03] UEVORA. Acesso à Informação a Qualquer Hora, em Qualquer Lugar - Estado Presente e Visão Futura do Wireless. **Relatório Técnico**. Disponível em: <http://www.alunos.uevora.pt/~l12059/es/t3/index.html>. Acesso em 01/05/2003.

[UPNP00] UPNP Forum. **UPnP Device Architecture**, jun. 2000. Disponível em: <http://www.upnp.org/download/UPnPDA10_20000613.htm>. Acessado em: 01/01/04.

[VEIZ97] VEIZADES, J. et al. Service Location Protocol. **IETF Request For Comments**, jun. 1997. Disponível em: <http://www.ietf.org/>. Acesso em 15/03/2002.

[WAN99] WAN, Guang; LIN, Eric. Cost reduction in location management using semi-realtime movement information. **Wireless Networks**, v.5, n.4, p.245–256, 1999.

[WEBE00] WEBER, Sam J.; JENNINGS, J. Architectural Issues for Pervasive Computing. In: Workshop on Software Engineering for Wearable and Pervasive Computing, 2000, Limerick (Irlanda). **Proceedings…** 2000.

[WILS00] WILSON, J. D. Reporting from the Field: Speech Transforms Data Collectors into Knowledge Workers. **Speech Technology Magazine**, fev. 2000.

[XU00] XU, D. et al. Resource-Aware Configuration of Ubiquitous Multimedia Service. In: IEEE International Conference on Multimedia and Expo, 2000. New York. **Proceedings…** 2000.

[YAU03] YAU, S.; KARIM, F. A Lightweight Middleware Protocol for Ad Hoc Distributed Object Computing in Ubiquitous Computing Environments. In: IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. Hakodate (Japão), 2003. **Proceedings...** 2003, p.172-179.

ANEXO A – Diagrama de Estados

O diagrama de estado é tipicamente um complemento para a descrição das classes. Este diagrama mostra todos os estados possíveis que objetos de uma certa classe podem se encontrar e mostra também quais são os eventos do sistemas que provocam tais mudanças. Os diagramas de estado não são escritos para todas as classes de um sistema, mas apenas para aquelas que possuem um número definido de estados conhecidos e onde o comportamento das classes é afetado e modificado pelos diferentes estados.

Diagramas de estado capturam o ciclo de vida dos objetos, subsistemas e sistemas. Eles mostram os estados que um objeto pode possuir e como os eventos (mensagens recebidas, timer, erros, e condições sendo satisfeitas) afetam estes estados ao passar do tempo.

Diagramas de estado possuem um ponto de início e vários pontos de finalização. Um ponto de início (estado inicial) é mostrado como um círculo todo preenchido, e um ponto de finalização (estado final) é mostrado como um círculo em volta de um outro círculo menor preenchido. Um estado é mostrado como um retângulo com cantos arredondados. Entre os estados estão as transições, mostrados como uma linha com uma seta no final de um dos estados. A transição pode ser nomeada com o seu evento causador. Quando o evento acontece, a transição de um estado para outro é executada ou disparada.

Uma transição de estado normalmente possui um evento ligado a ela. Se um evento é anexado a uma transição, esta será executada quando o evento ocorrer. Se uma transição não possuir um evento ligado a ela, a mesma ocorrerá quando a ação interna do código do estado for executada (se existir ações internas como entrar, sair, fazer ou outras ações definidas pelo desenvolvedor). Então quando todas as ações forem executadas pelo estado, a transição será disparada e serão iniciadas as atividades do próximo estado no diagrama de estados.

ANEXO B – Diagrama de Atividades

Diagramas de atividade capturam ações e seus resultados. Eles focam o trabalho executado na implementação de uma operação (método), e suas atividades numa instância de um objeto. O diagrama de atividade é uma variação do diagrama de estado e possui um propósito um pouco diferente do diagrama de estado, que é o de capturar ações (trabalho e atividades que serão executados) e seus resultados em termos das mudanças de estados dos objetos.

Os estados no diagrama de atividade mudam para um próximo estágio quando uma ação é executada (sem ser necessário especificar nenhum evento como no diagrama de estado). Outra diferença entre o diagrama de atividade e o de estado é que podem ser colocadas como "swimlanes". Uma swimlane agrupa atividades, com respeito a quem é responsável e onde estas atividades residem na organização, e é representada por retângulos que englobam todos os objetos que estão ligados a ela (swimlane).

Um diagrama de atividade é uma maneira alternativa de se mostrar interações, com a possibilidade de expressar como as ações são executadas, o que elas fazem (mudanças dos estados dos objetos), quando elas são executadas (sequência das ações), e onde elas acontecem (swimlanes).

Um diagrama de atividade pode ser usado com diferentes propósitos inclusive:

• Para capturar os trabalhos que serão executados quando uma operação é disparada (ações). Este é o uso mais comum para o diagrama de atividade.

• Para capturar o trabalho interno em um objeto.

• Para mostrar como um grupo de ações relacionadas podem ser executadas, e como elas vão afetar os objetos em torno delas.

• Para mostrar como uma instância pode ser executada em termos de ações e objetos.

• Para mostrar como um negócio funciona em termos de trabalhadores (atores), fluxos de trabalho, organização, e objetos (fatores físicos e intelectuais usados no negócio).

O diagrama de atividade mostra o fluxo sequencial das atividades, é normalmente utilizado para demonstrar as atividades executadas por uma operação específica do sistema. Consistem em estados de ação, que contém a especificação de uma atividade a ser

desempenhada por uma operação do sistema. Decisões e condições, como execução paralela, também podem ser mostrados na diagrama de atividade. O diagrama também pode conter especificações de mensagens enviadas e recebidas como partes de ações executadas.

Tese de Doutorado apresentada por **Thienne de Melo e Silva Mesquita Johnson** a Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **"Uma Arquitetura de Computação Previsiva para Trabalho de Campo",** elaborada sob a orientação do **Prof. Djamel Fawzi Hadj Sadok** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Paulo Romero Martins Maciel
Departamento de Sistemas de Computação - CIn / UFPE

Prof. Aluízio Fausto Ribeiro Araújo
Departamento de Sistemas de Computação - CIn / UFPE

Prof. Nelson Souto Rosa
Departamento de Sistemas de Computação – CIn / UFPE

Prof. Gledson Elias da Silveira
Departamento de Informática / UFPB

Prof. Carlos Alberto Kamienski
Centro Federal de Educação Tecnológica da Paraíba

Visto e permitida a impressão.
Recife, 31 de março de 2005.

Prof. JAELSON FREIRE BRELAZ DE CASTRO
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.