# Reduzindo a Implosão de Respostas em Protocolos de Descoberta de Serviços para Redes sem fio *Ad hoc* de Saltos Múltiplos

**Luciana dos S. Lima[1,2], Antônio Tadeu A. Gomes[2], Artur Ziviani[2], Pedro A. França[2], Bruno F. Bastos[2], Markus Endler[1]**

[1]Departamento de Informática (DI)
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Rua Marquês de São Vicente, 255 – 22.453-900 – Rio de Janeiro – RJ – Brasil
{lslima,endler}@inf.puc-rio.br

[2]Laboratório Nacional de Computação Científica (LNCC)
Av. Getúlio Vargas, 333 – 25.651-075 – Petrópolis – RJ – Brasil
{atagomes,ziviani,alcover,bfbastos}@lncc.br

***Abstract.*** *Providing service discovery in an efficient and scalable way in ad hoc networks is a challenging problem, in particular within multi-hop scenarios, due to the large number of potential participant devices and due to the typically limited resources of these networks. In this paper, we propose a strategy to reduce the reply implosion problem in service discovery protocols for multi-hop wireless ad hoc networks. The proposed scheme is based on suppression by vicinity (SbV) approach, thus considerably reducing the number of transmissions without compromising the efficiency of the service discovery. Our performance results show the scalability and the efficiency of the proposed solution.*

***Resumo***. *A descoberta de serviços de forma eficiente e escalável em redes ad hoc é um desafio, especialmente em cenários de saltos múltiplos, devido ao número potencialmente grande de dispositivos participantes e à escassez de recursos característica dessas redes. Neste artigo, é proposta uma estratégia para reduzir a implosão de respostas em protocolos de descoberta de serviços para redes sem fio ad hoc de saltos múltiplos. O mecanismo proposto se baseia na supressão de respostas por vizinhança (SbV), reduzindo de forma considerável o número de transmissões, sem comprometer a eficiência da descoberta de recursos. Nossos resultados de avaliação de desempenho atestam a escalabilidade e eficiência da solução proposta.*

## 1. Introdução

Protocolos de descoberta de serviços em redes em geral têm por objetivo reduzir a sobrecarga administrativa dos usuários e aumentar a usabilidade de recursos e serviços em ambientes distribuídos, possibilitando que dispositivos (e os serviços que eles oferecem) sejam descobertos, configurados e usados por outros dispositivos e serviços nesses ambientes, preferencialmente sem a necessidade de intervenção humana [Marin-Perianu et al. 2005]. A pesquisa na área de descoberta de serviços em redes sem

fio *ad hoc* é relativamente nova – em relação àquela para redes cabeadas e redes sem fio infra-estruturadas – e mais desafiadora, em particular, devido ao maior dinamismo determinado pela mobilidade de todos os dispositivos sem fio. Há na literatura diversas propostas de protocolos de descoberta de serviços em redes sem fio *ad hoc* [Mian et al. 2006]. O principal objetivo dessas propostas é fomentar a colaboração entre organizações virtuais espontâneas, formadas pela proximidade geográfica dos dispositivos em cenários mais próximos ao cotidiano de seus usuários, como em casa, no ambiente de trabalho e em momentos de lazer [Rittenbruch et al. 1998].

Os desafios relacionados à descoberta de serviços são ainda maiores quando se considera as redes sem fio *ad hoc* de saltos múltiplos (denominadas neste artigo por *MANETs*). Nas MANETs cada dispositivo atua como um roteador e, para tanto, precisa manter informações de roteamento localmente. Algumas propostas integram as funcionalidades de descoberta de serviços e de roteamento em MANETs em um único protocolo [Koodli and Perkins 2002, Harbird et al. 2005, Varshavsky et al. 2005]. Contudo, a instabilidade dessas redes, provocada pela mobilidade dos dispositivos e pela carência de recursos, como energia e memória, torna desafiador o armazenamento de informações de roteamento e, principalmente, a garantia de sua consistência. Por essa razão, existe um número considerável de propostas que oferece soluções para a descoberta de serviços no nível de aplicação, não utilizando qualquer informação de roteamento da MANET [Ratsimor et al. 2002, Helal et al. 2003, Chakraborty et al. 2006]. Nesses protocolos, a transmissão de mensagens de descoberta é geralmente realizada por difusão, utilizando *multicast* ou *broadcast*. Além da independência em relação às informações de roteamento da rede, a utilização de difusão permite também, em alguns casos, a descoberta simultânea de múltiplas instâncias de um mesmo serviço, possibilitando a manutenção de um grau médio de redundância do mesmo. Essa abordagem é indicada em casos como, por exemplo, de aplicações de computação distribuída em grades móveis sem fio [Kurkovsky et al. 2004], em que o serviço provido é o próprio poder de processamento dos dispositivos. Quando o protocolo de descoberta de serviço adota um mecanismo de difusão ele incorre, no entanto, no problema da implosão de mensagens de resposta [Tseng et al. 2003]. Esse problema decorre do volume potencialmente grande de respostas gerado pelos dispositivos provedores do serviço requisitado, em especial em redes de grande escala.

Neste artigo apresentamos um mecanismo de supressão de respostas por vizinhança (*Suppression by Vicinity* – SbV), que permite tratar o problema da implosão de respostas em protocolos de descoberta para MANETs baseados em difusão. O mecanismo SbV adota uma abordagem *peer-to-peer*, independente de protocolos de roteamento ou mesmo do endereçamento no nível de rede na MANET para garantir o seu funcionamento. Esse mecanismo pode ser empregado em conjunto com diferentes protocolos de descoberta, sejam eles reativos – em que serviços são descobertos sob demanda pelos dispositivos requisitantes – ou proativos – baseados no envio periódico de anúncios de serviços pelos dispositivos provedores –, embora seja realmente efetivo nas abordagens reativas, conforme atestam os nossos resultados experimentais.

Com o mecanismo SbV, cada dispositivo na MANET é capaz de suprimir o encaminhamento de respostas recebidas por ele, mas endereçadas a outros dispositivos. A supressão de uma resposta é feita com base no número de instâncias de serviço desejadas pelo dispositivo requisitante. Uma resposta é suprimida em um dispositivo se

a quantidade de respostas recebidas anteriormente, associadas à mesma requisição de serviço, for maior que o número de instâncias de serviço desejadas pelo requisitante. Nesse mecanismo, as respostas não são encaminhadas por *unicast* ao dispositivo requisitante, mas sim por *broadcast* salto-a-salto no nível de aplicação. Isso permite que um dispositivo leve em conta, para efeito de supressão de respostas, não somente as respostas anteriores para as quais o dispositivo fez parte do caminho até o destino (o dispositivo requisitante do serviço), como também outras respostas encaminhadas por dispositivos vizinhos, aumentando assim a eficácia da supressão.

O mecanismo SbV foi implementado e incorporado ao protocolo P2PDP (*Peer-to-Peer Discovery Protocol*), desenvolvido no contexto do *middleware* MoGrid [Lima et al. 2005]. Esse protocolo foi concebido para oferecer suporte à descoberta de serviços de computação distribuída em grades móveis baseadas em redes infra-estruturadas e redes *ad hoc* de salto único, permitindo a seleção dos dispositivos mais aptos a oferecer as instâncias do serviço desejado por um dispositivo requisitante. Neste artigo apresentamos as extensões ao protocolo P2PDP, decorrentes da utilização do mecanismo SbV, para redes *ad hoc* de saltos múltiplos. Em particular, mostramos por meio de simulações que a introdução do mecanismo SbV no P2PDP, bem como em outros protocolos de descoberta baseados em *broadcast*, possibilita uma redução significativa na sobrecarga de tráfego de comunicação associado ao processo de descoberta. Além disso, o mecanismo SbV torna a supressão de respostas maior à medida que as mesmas vão se aproximando do dispositivo requisitante. Desse modo, o algoritmo SbV aumenta a escalabilidade do protocolo de descoberta de serviços, no que diz respeito ao número de dispositivos participantes e à densidade da MANET.

O restante deste artigo encontra-se estruturado da seguinte forma. As principais propostas de protocolos de descoberta de serviços em MANETs são sumariadas na Seção 2. A Seção 3 descreve o algoritmo de supressão de mensagens de resposta por vizinhança (SbV). A Seção 4 introduz o protocolo de descoberta P2P (P2PDP), ao qual o algoritmo SbV foi incorporado, trazendo ainda alguns detalhes de implementação. Os resultados experimentais obtidos com a avaliação de desempenho do algoritmo SbV são explanados na Seção 5. A Seção 6 é reservada à apresentação das conclusões obtidas durante o desenvolvimento deste trabalho, apontando direções futuras.

## 2. Trabalhos Relacionados

Durante muito tempo, as pesquisas sobre descoberta de serviços seguiram o modelo clássico de aplicações cliente-servidor para computação distribuída. Nessa visão, servidores centralizados atuam como diretórios, armazenando um índice global dos serviços disponíveis na rede e gerenciando todo o processo de descoberta. Para ilustrar esse modelo, podemos mencionar os protocolos de descoberta de serviços remotos, hospedados em dispositivos conectados, através de uma rede cabeada, a um nó central confiável atuando como diretório. Dentre esse conjunto de protocolos, destacam-se Jini [Jini 1999] da Sun, Salutation [Salutation 1999] da IBM e SLP [Guttman 1999] do IETF. A maioria desses protocolos oferece extensões com suporte à descoberta de serviços sob demanda e adaptações para atender aos dispositivos móveis conectados através de redes sem fio infra-estruturadas. Entretanto, essas adaptações dependem de alguma extensão ao modelo tradicional, o que não se adequa às redes sem fio *ad hoc*,

sobretudo nas de saltos múltiplos, onde a topologia é volátil e o diretório pode não estar alcançável durante todo o tempo.

Há protocolos desenvolvidos especificamente para as redes sem fio *ad hoc* de salto único, como o DEAPspace [Nidd 2001] da IBM. Nesse protocolo, cada dispositivo anuncia periodicamente as informações sobre os serviços da rede, mantendo uma visão global do sistema em todos os dispositivos. O objetivo é garantir uma rápida convergência das informações sobre todos os serviços disponíveis na rede no menor tempo possível. Entretanto, a quantidade de mensagens de anúncio emitidas é muito elevada e o seu tamanho é considerável, o que inviabiliza a sua adoção nas MANETs, já que nessas redes a troca de mensagens de controle é feita por difusão.

Nos últimos anos, a pesquisa em protocolos de descoberta de serviços para redes sem fio *ad hoc* de saltos múltiplos tem se intensificado; a maioria das propostas oferece suporte aos mecanismos de descoberta de serviços sob demanda e à divulgação de anúncios de informações de serviços. O protocolo Konark [Helal et al. 2003], por exemplo, propõe uma melhoria ao mecanismo de anúncio do DEAPspace, utilizando o conceito de anúncio incremental, correspondente à diferença da visão do dispositivo em relação à visão global da rede, garantindo uma rápida convergência com a redução do número de mensagens de controle trocadas e do seu tamanho. ALLIA [Ratsimor et al. 2002] e DGSD [Chakraborty et al. 2006] se assemelham ao Konark; todas as três propostas permitem a sintonização de parâmetros como a periodicidade de envio de anúncios e a distância em número de saltos que uma requisição pode alcançar. Algumas propostas apresentam, ainda, mecanismos de encaminhamento seletivo de requisições de serviços baseados nos conceitos de políticas locais, como ALLIA, e de grupos de serviços, como DGSD. O *framework* RUBI [Harbird et al. 2005] propõe um mecanismo adaptativo de monitoramento da rede – com base nas visões locais que cada dispositivo tem da topologia – que permite controlar o modo como as informações sobre os serviços são disseminadas e recuperadas. Entretanto, nenhuma dessas abordagens explora a seleção das melhores respostas, provenientes dos dispositivos mais ricos em recursos, com a supressão daquelas menos adequadas, diminuindo, conseqüentemente, o tráfego de mensagens de controle na rede. Em [Varshavsky et al. 2005], as funções de descoberta e seleção de serviços são integradas aos protocolos de roteamento existentes, em uma abordagem *cross-layer*, mas essa proposta não oferece mecanismos para reduzir o problema da implosão das mensagens de resposta como o proposto neste trabalho.

## 3. Supressão de Respostas por Vizinhança (SbV)

O mecanismo SbV assume o uso de dois tipos de mensagens de descoberta: requisição e resposta. Além disso, esse mecanismo exige que cada dispositivo na MANET hospede uma estrutura de dados local (`pendingList`) usada no controle de supressões.

Para a operação do mecanismo SbV, cada mensagem de requisição deve carregar, além da descrição do serviço sendo requisitado, (i) uma identificação única de requisição (`reqID`), (ii) a identificação do último dispositivo transmissor da mensagem (`hopID`) e (iii) o número de instâncias de serviço desejadas (`numMaxReplies`).

Ao receber uma requisição, um dispositivo a registra como uma requisição pendente em `pendingList`. Além de armazenar os campos `reqID`, `numMaxReplies` e `hopID`, de uma requisição, cada entrada em uma `pendingList` possui um contador de

mensagens de resposta (numReplies) e um temporizador que determina o tempo de vida da entrada na estrutura (cleanUp). Após atualizar sua pendingList, o dispositivo pode responder a requisição, redifundí-la na MANET (por inundação), ou ambos. Antes que uma requisição seja redifundida ela deve ter seu campo hopID atualizado com a identificação do dispositivo transmissor. A atualização desse campo possibilita que um dispositivo na MANET mantenha em sua pendingList um caminho de retorno para as mensagens de resposta relativas a cada requisição repassada pelo dispositivo.

Mensagens de resposta são originadas por dispositivos provedores aos dispositivos requisitantes. O mecanismo SbV demanda que, além da identificação do dispositivo provedor e da descrição do serviço provido, a mensagem de resposta transporte também (i) a identificação da requisição correspondente (reqID) e (ii) a identificação do dispositivo do qual a requisição correspondente foi recebida (retPath) – o dispositivo obtém essa identificação a partir do valor do campo hopID na entrada correspondente à requisição em sua pendingList.

A Figura 1 traz o pseudocódigo do algoritmo de supressão, descrito a seguir.

```
//msg      corresponde à mensagem de resposta
//localID  corresponde ao identificador do dispositivo local
RecebeResp( msg, localID )
 1  SE primeiraCópia( msg ) ENTÃO //msg não é duplicada?
 2     SE MinhaResposta( msg ) ENTÃO //disp. local originou req. correspondente?
 3        ProcessaResposta( msg )
 4        Retorna
 5     entrada ← pendingList[msg.reqID] //entrada correspondente na lista
 6     SE entrada ≠ NULO ENTÃO //req. correspondente foi recebida?
 7        entrada.N_R ← entrada.N_R + 1
 8        SE entrada.N_R = entrada.N_M ENTÃO   //o nº de resps. é suficiente?
 9           SuprimeMinhaResposta( msg.reqID )  //suprime resp. do disp. local
10        SE msg.retPath = localID ENTÃO    //estou no caminho de retorno?
11           SE entrada.N_R ≤ entrada.N_M ENTÃO //é preciso encaminhar?
12              EncaminhaEmBroadcastDirecionado( entrada.hopID, msg )
13              Retorna
14           SENÃO
15              SuprimeRespostaDaRede( msg )
16     DescartaResposta( msg )
```

**Figura 1. O algoritmo de supressão por vizinhança SbV.**

No mecanismo SbV, as mensagens de resposta são transmitidas, por *broadcast* salto-a-salto, em direção ao dispositivo requisitante. Quando um dispositivo recebe uma resposta correspondendo a uma requisição dele originada, o dispositivo processa a mensagem e a retira da MANET. Se, ao invés, a resposta estiver endereçada a um outro dispositivo, o receptor verifica primeiramente se há uma entrada em sua pendingList relacionada à requisição correspondente. Em caso negativo, a resposta é descartada.[1]

[1] Em condições ideais, essa situação não poderia acontecer, devido ao mecanismo de inundação de requisições. Contudo, fatores como colisões ou o uso de esquemas inibidores de redundância em inundações [Tseng et al. 2003] podem ocasionar situações desse tipo.

Senão, o dispositivo incrementa o valor do campo `numReplies` ($N_R$) na entrada correspondente em sua `pendingList`. Se $N_R = N_M$, isso indica que já foram encaminhadas mensagens de resposta suficientes para atender à requisição do iniciador. Nesse caso, o dispositivo suprime uma eventual resposta sua que não tenha sido ainda transmitida. A seguir, o dispositivo compara sua própria identificação com o valor do campo `retPath` na resposta. Se os valores são iguais, o dispositivo se encontra no caminho de retorno da mesma. Nesse caso, se $N_R \leq N_M$, o dispositivo pode encaminhar a mensagem de resposta através do dispositivo que corresponde ao próximo salto no caminho de retorno da resposta. A identificação desse dispositivo é obtida pelo campo `hopID`, na entrada relacionada à requisição correspondente em sua `pendingList`. Se $N_R > N_M$, o dispositivo suprime a resposta recebida. Para permitir futuras supressões, a entrada correspondente na `pendingList` é mantida no dispositivo até que o temporizador `cleanUp` expire.

A Figura 2 ilustra o funcionamento do mecanismo SbV. Na figura, os dispositivos *w* e *z* estão no raio de transmissão do dispositivo *y* (área cinza). A Figura 2(a) traz a configuração inicial de `pendingList` em *z* e *y*. Na Figura 2(b) o dispositivo *y* recebe uma resposta para uma requisição com `reqID` = 1000 e incrementa o valor de $N_R$ do campo `numReplies` na entrada correspondente em sua `pendingList`. Na Figura 2(c), como *y* é o caminho de retorno da resposta, ele encaminha a resposta por *broadcast* local na direção de *w*, que corresponde ao próximo salto de *y* no caminho de retorno. O dispositivo *z* detecta a mensagem e, assim como *y*, incrementa o valor de $N_R$ do campo `numReplies` na entrada correspondente em sua `pendingList`, mas *z* não encaminha a mensagem, pois ele não está no caminho de retorno da mesma. Na Figura 2(d), *z* recebe uma nova resposta para a requisição com `reqID`=1000, mas, embora esteja em seu caminho de retorno, *z* não a encaminha, pois o campo `numMaxReplies` na entrada correspondente em sua `pendingList` indica que ele já encaminhou, ou detectou, $N_M$ mensagens.

Como resultado do funcionamento do mecanismo SbV, as respostas provenientes de alguns dispositivos suprimem respostas excedentes provenientes de outros dispositivos ao longo dos caminhos utilizados para encaminhar as respostas. Isso permite reduzir a implosão de respostas, que é intrínseca às abordagens de descoberta baseadas em *broadcast* [Tseng et al. 2003]. Além disso, conforme demonstram nossos resultados experimentais (vide Seção 5), esse mecanismo aumenta a supressão de respostas redundantes à medida que ela ocorre próxima do dispositivo requisitante, configurando, desse modo, uma solução escalável no que diz respeito ao número de dispositivos participantes e à densidade da MANET.

Vale destacar que em MANETs cujo controle de acesso ao meio físico é baseado no protocolo CSMA/CA (*Carrier Sense Multiple Access / Collision Avoidance*), o uso de um esquema de *broadcast* salto-a-salto, como proposto pelo mecanismo SbV, exclui a aplicação dos mecanismos de acesso baseados em confirmação ou no diálogo RTS/CTS (*Request-To-Send / Clear-To-Send*) oferecidos por esse protocolo, tornando as transmissões de mensagens menos confiáveis devido à maior probabilidade de perdas por colisão. A ausência de reconhecimento pode ser solucionada no próprio mecanismo SbV (vide discussão a respeito na Seção 6). Contudo, é interessante também que, em conjunto com o SbV, seja adotado algum mecanismo que garanta um certo assincronismo na transmissão das respostas, de modo a reduzir a probabilidade de

colisões. Isso pode ser conseguido, por exemplo, através de um retardo programado no envio das respostas [Duffield et al. 1999]. Na seção a seguir descrevemos a implementação de um mecanismo com essas características no protocolo P2PDP.

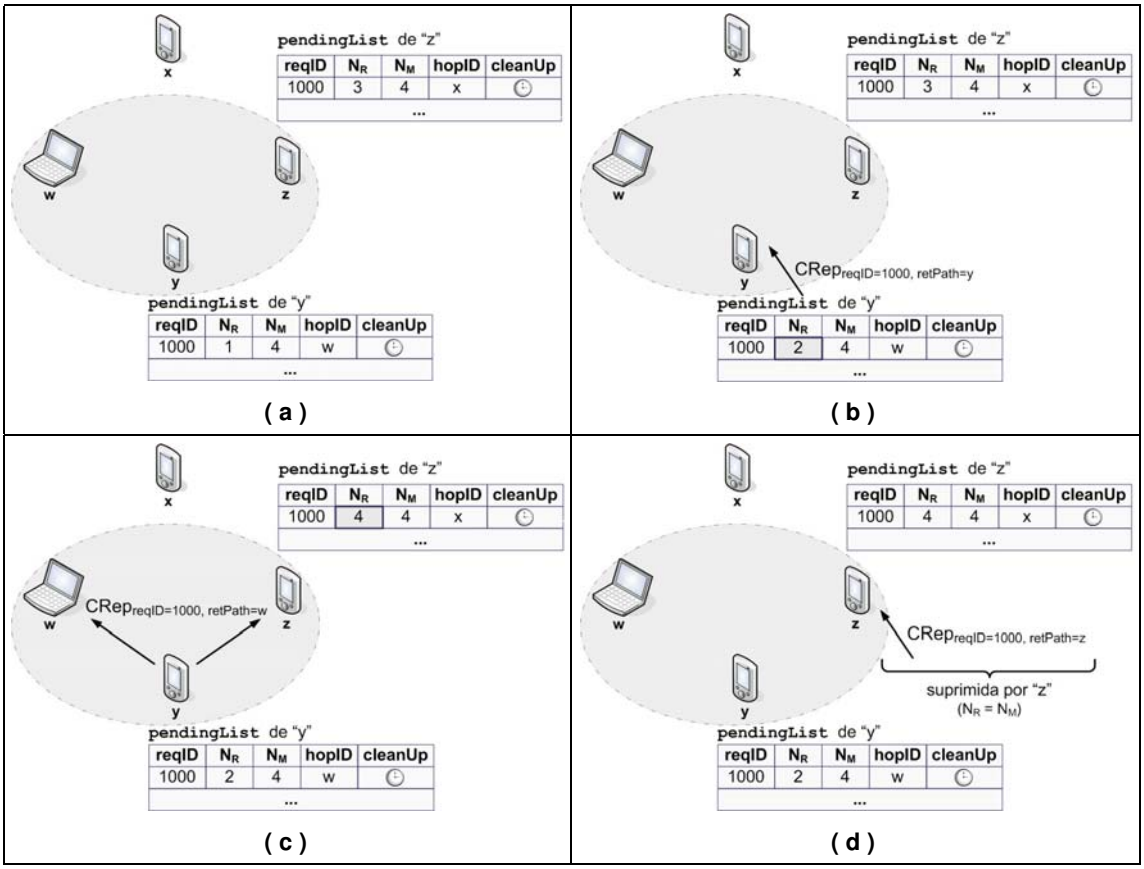


**Figura 2. Cenário ilustrando a supressão de mensagens de resposta com o SbV.**

## 4. O Protocolo de Descoberta *Peer-to-Peer*

No protocolo P2PDP (*Peer-to-Peer Discovery Protocol*) [Lima et al. 2005] são identificadas duas entidades principais – colaboradores e iniciadores – que correspondem aos diferentes papéis que um dispositivo pode desempenhar na rede sem fio. Colaboradores estão disponíveis para compartilhar seus recursos e serviços com os demais dispositivos da rede. Iniciadores submetem aos colaboradores requisições de descoberta sob demanda, em função das suas necessidades por recursos ou serviços. Qualquer dispositivo pode ser um iniciador na rede, a qualquer instante. Iniciadores enviam requisições para descoberta de serviços aos colaboradores e, com base nas respostas recebidas, montam uma lista contendo os colaboradores mais adequados.

Um colaborador adota dois critérios para decidir se está apto a prover o iniciador com o serviço solicitado. O primeiro critério atua como um controle de admissão, verificando se o colaborador de fato disponibiliza o serviço. O segundo critério define a adequação do colaborador em atender a solicitação. No P2PDP, o serviço solicitado é mapeado para uma requisição em termos da quantidade dos recursos necessários para a sua utilização, bem como da importância relativa entre os mesmos. A informação sobre

o estado desses recursos representa o *contexto de interesse* da aplicação que acionou o iniciador, ou seja, os recursos que são necessários para a utilização do serviço.

O protocolo P2PDP define duas mensagens de controle: `InitiatorRequest` (`IReq`) e `CollaboratorReply` (`CRep`). A Figura 3 ilustra o funcionamento básico do protocolo em MANETs. Na figura, assume-se que todos os dispositivos colaboradores na rede respondem ao dispositivo requisitante (*iniciador*). As mensagens e o funcionamento geral do protocolo são descritos ao longo desta seção, com enfoque nas extensões decorrentes da introdução do mecanismo SbV.

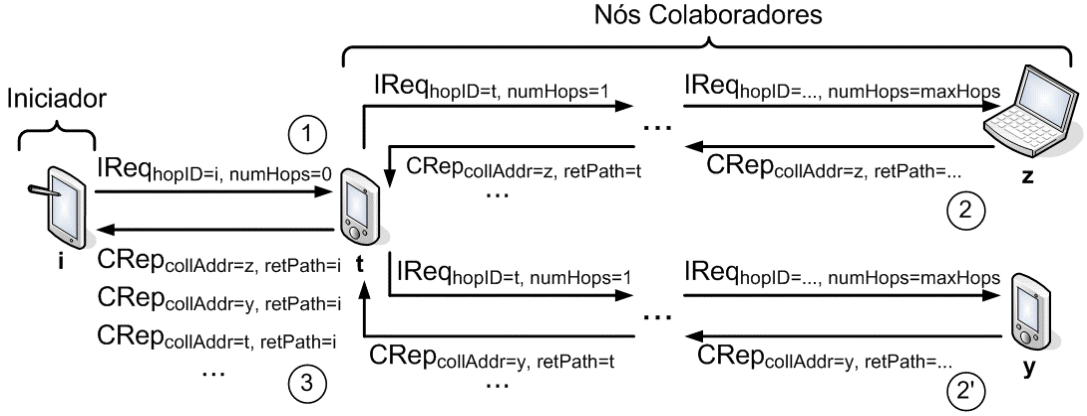


**Figura 3. Funcionamento básico do protocolo P2PDP.**

Mensagens `InitiatorRequest` (`IReq`) são enviadas dos iniciadores para os colaboradores por difusão (passo 1 da Figura 3). Uma mensagem `IReq` é composta pelos campos `reqID`, `hopID` e `numMaxReplies`, necessários ao funcionamento do mecanismo SbV (vide Seção 3), bem como (i) pelo retardo máximo de resposta que um iniciador admite (`maxReplyDelay`), (ii) pela informação de contexto de interesse (`ctxtInfo`) da aplicação solicitante e (iii) pelo diâmetro atual (`numHops`) e diâmetro máximo (`maxHops`) – em número de saltos – associados à propagação da requisição.

Quando um iniciador envia uma mensagem `IReq`, ele cria uma nova entrada representando a requisição em uma estrutura de dados local (`resumeList`). Cada registro em uma `resumeList` é constituído pelo ID da requisição (`reqID`) e pelo número de colaboradores (`numMaxReplies`) que ele deseja envolver na distribuição da tarefa. Tal entrada é associada a um temporizador (`resumeDelay`) configurado em função do retardo máximo de resposta que o iniciador admite. Quando esse temporizador expira, o iniciador sumariza todas as mensagens `CRep` que foram recebidas, associadas à requisição pendente, descartando, se preciso, as mensagens `CRep` sobressalentes.

Ao receber uma mensagem `IReq`, o colaborador registra a mensagem como uma requisição pendente em sua `pendingList`, necessária ao funcionamento do mecanismo SbV. O protocolo P2PDP acrescenta a essa estrutura de dados um segundo temporizador (`replyDelay`), cuja funcionalidade é descrita na subseção a seguir.

Mensagens `CollaboratorReply` (`CRep`) são encaminhadas pelos colaboradores em direção aos iniciadores em resposta às mensagens `IReq` (passos 2/2' e 3 da Figura 3). Além de transportar os campos `reqID` e `retPath`, necessários ao mecanismo SbV, uma mensagem `CRep` também informa ao iniciador sobre o endereço do dispositivo colaborador (campo `collabAddr`) – que pode ser usado posteriormente pelo dispositivo

iniciador para acionar efetivamente o serviço descoberto – bem como sobre a sua disponibilidade de recursos (campo `resInfo`), de acordo com a solicitação contida na requisição correspondente.

### 4.1. Retardo Programado de Mensagens `CRep`

No P2PDP, é possível estabelecer critérios flexíveis para categorizar as melhores respostas dependendo do perfil da aplicação em nome da qual o iniciador está atuando. Em [Lima et al. 2005], implementamos um critério de seleção de respostas para o P2PDP, no âmbito de grades móveis, baseado na informação de contexto de cada colaborador. Nesse trabalho, entende-se por *contexto* toda informação que represente o estado dos dispositivos móveis, incluindo o estado da conectividade, carga de CPU, energia disponível, memória disponível e espaço de armazenamento em disco.

Na implementação do protocolo P2PDP, cada dispositivo disposto a colaborar na provisão de um serviço específico retarda a transmissão de suas mensagens `CRep` de acordo com o temporizador `replyDelay`, associado à requisição em questão, na entrada correspondente em sua `pendingList`. Esse temporizador é configurado de modo que seu tempo de expiração seja inversamente proporcional à disponibilidade dos recursos no dispositivo que são necessários para prover o serviço requisitado. O objetivo é fazer com que os dispositivos com mais recursos sejam os primeiros a responder às requisições de descoberta de serviços. Se o número total de respostas produzido na MANET é maior do que o número máximo de respostas $N_M$ (= `numMaxReplies`) solicitado, o dispositivo que originou a requisição se encarrega de selecionar as mais adequadas dentre as $N_M$ primeiras respostas recebidas. Vale ressaltar que a determinação do retardo no envio da resposta é flexível no que tange aos recursos sendo considerados para a provisão do serviço e à importância relativa (peso) entre eles. A informação sobre quais recursos devem ser considerados no cálculo da adequação de um dispositivo é transportada pelas mensagens `IReq`, no campo `ctxInfo`.

Quando um dispositivo recebe uma requisição, ele consulta seu estado atual, em função dos recursos de interesse, para calcular o retardo de resposta. Para esse mecanismo funcionar a contento, todos os dispositivos na MANET devem empregar o mesmo critério no cálculo do retardo. Na implementação do P2PDP, um dispositivo colaborador configura o temporizador `replyDelay`, na entrada correspondente em sua `pendingList`, para $\tau$ unidades de tempo, conforme dado por

$$\tau = \left(1 - \omega \sum_{i=1}^{N} \left( \frac{\alpha_i P_i}{\sum_{j=1}^{N} P_j} \right) \right) \left(D_{\max} - 2HS\right), \quad \begin{matrix} 0 \le \alpha \le 1 \\ 0 < \omega \le 1 \end{matrix} \qquad (1)$$

Na Equação (1), $N$ representa o número dos diferentes tipos de recursos que o dispositivo colaborador deve considerar. $P_i$ corresponde ao peso que descreve a importância relativa de cada recurso do tipo $i$, $1 \le i \le N$. Tanto $N$ quanto $P_i$ são descritos como parte do campo `ctxInfo` transmitido na requisição. $\alpha_i$ corresponde ao nível normalizado de disponibilidade (no intervalo [0, 1]) do recurso do tipo $i$ no dispositivo colaborador. $D_{\max}$ corresponde ao retardo máximo de envio da resposta. $H$ e $S$ são parâmetros de sintonização usados para considerar os retardos de transferência que as

mensagens `IReq` e `CRep` podem experimentar. $H$ representa a distância em número de saltos entre o dispositivo colaborador e o que originou a requisição. $S$ permite a sintonização do valor de $\tau$ de acordo com o retardo de transferência experimentado em cada dispositivo. Os valores de $D_{max}$ e $H$ são obtidos, respectivamente, a partir dos campos `maxReplyDelay` e `hopCount` da mensagem `IReq`. Finalmente, $\omega$ indica a disposição ("boa vontade") do dispositivo colaborador em participar da provisão do serviço (no intervalo (0, 1]). $\omega$ é um parâmetro subjetivo e determinado pelo usuário que descreve o nível de interesse do usuário em permitir que o seu dispositivo colabore com os outros na MANET. $\tau$ não é definido para $\omega = 0$; tal valor significa que o usuário não deseja participar, desse modo, nenhuma resposta será gerada pelo dispositivo, que atuará apenas como um intermediário no encaminhamento de mensagens do protocolo.

É interessante notar que o mecanismo de retardo programado implementado pelo temporizador `maxReplyDelay` oferece implicitamente um assincronismo no envio das respostas, propiciando assim a possibilidade de redução no número de colisões dessas mensagens (vide discussão a respeito na Seção 6).

### 4.2. Detalhes de Implementação

O protocolo P2PDP foi desenvolvido no contexto do *middleware* MoGrid [Lima et al. 2005],[2] tendo sido implementado na linguagem Java, adotando o perfil CDC (*Connected Device Configuration*) do J2ME como plataforma de referência. O protocolo de descoberta foi testado com duas aplicações: uma aplicação P2P de compartilhamento de arquivos e uma aplicação de multiplicação distribuída de matrizes.

Na nossa implementação, dois serviços desempenham funções importantes para garantir o funcionamento do mecanismo de retardo programado: o *monitor* e o *gerente de contexto*. Cada dispositivo utilizando o P2PDP na MANET possui um monitor e um gerente de contexto residente. O monitor é responsável por coletar as informações de estado dos dispositivos móveis, incluindo o estado da conectividade, carga de CPU, carga da bateria, memória disponível e espaço de armazenamento. O monitor utilizado em nossa implementação corresponde à implementação disponível na arquitetura MoCA (*Mobile Collaboration Architecture*) [Sacramento et al. 2004]. O gerente de contexto recebe periodicamente do monitor as informações de estado coletadas e deduz, a partir de tais informações, a disponibilidade de recursos dos dispositivos. Em redes *ad hoc*, cada dispositivo tem seu gerente de contexto local.

## 5. Avaliação de Desempenho

Nós conduzimos uma série de experimentos com o mecanismo SbV utilizando o simulador *ns-2* [ISI 1995]. Todos os experimentos foram feitos considerando-se cenários com densidade fixa de dispositivos na MANET (usando topologias em grade, com número fixo de dispositivos em um mesmo raio de transmissão) de modo a avaliar apropriadamente o efeito do aumento no número de dispositivos na MANET sobre o mecanismo. A Figura 4 ilustra a execução de uma rodada de simulação em um cenário com 16 dispositivos.

---

[2] A implementação está disponível para *download*, sob licença acadêmica, no *site* http://martin.lncc.br.

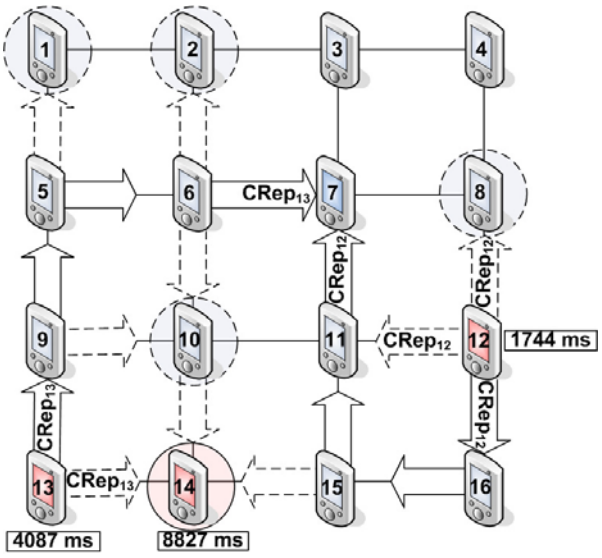


**Figura 4. Exemplo de cenário de simulação com 16 dispositivos.**

No cenário da Figura 4, o nó 7 atuou como iniciador, difundindo uma mensagem `IReq` com `numMaxReplies`=2 e `maxReplyDelay`=10s. Os colaboradores que oferecem o serviço requisitado pelo nó 7 são os nós 12, 13 e 14. Os valores calculados para o temporizador `replyDelay` pelos colaboradores 12, 13 e 14 foram, respectivamente, 1744 ms, 4087 ms e 8827 ms. Na Figura 4, as setas tracejadas representam as respostas encaminhadas aos nós que não se encontram no caminho de retorno dessas mensagens e os círculos tracejados representam o descarte das mensagens nesses nós. Ao escutar as mensagens de resposta dos nós 12 e 13, o nó 14 suprime a sua própria mensagem, pois a classifica como excedente, já que o valor do seu `replyDelay` é muito maior do que o dos demais; a supressão é representada na Figura 4 pelo círculo contornado pela linha sólida.

Os resultados apresentados nesta seção correspondem às médias dos resultados coletados em um total de cem rodadas por cenário de simulação com intervalo de confiança de 95%. Os experimentos visaram primordialmente a avaliação de duas métricas: a carga de mensagens de resposta na MANET e o diâmetro de supressão dessas mensagens. Os cenários de simulação consideraram dispositivos com baixa mobilidade. Nesses cenários, o tempo necessário para a descoberta equivale ao *round-trip time* (RTT) entre o envio da requisição e o recebimento das primeiras respostas – as de melhor qualidade – pelo nó requisitante. A movimentação dos nós, nesse caso, é insuficiente para mudar significativamente a topologia da rede a ponto de invalidar o caminho seguido por essas primeiras respostas, que corresponde ao caminho reverso da requisição na abordagem proposta neste artigo. A Tabela 1 apresenta os parâmetros usados na constituição dos cenários de simulação.

**Tabela 1. Parâmetros de simulação.**

| Parâmetro | Valores |
|---|---|
| Número de dispositivos (*N*) | 10 a 240 |
| Percentual de dispositivos colaboradores (*p*) | 20% a 80% |
| Número máximo de respostas (*R*) | 1 a 10 |
| Densidade de dispositivos | 5 |
| Distância entre dispositivos | 10m |

A carga média na MANET devido a mensagens de resposta foi medida calculando-se, para cada cenário, a média do número total de transmissões de pacotes envolvendo essas mensagens. Essa métrica permite-nos deduzir também se há uma redução global significativa no consumo de energia dos dispositivos na MANET devido à supressão de respostas, uma vez que transmissões são responsáveis por um alto consumo de energia. Para essa métrica, foi feita uma comparação entre dois protocolos de descoberta de serviços simples, um (chamado aqui de "BCast") baseado puramente em requisições por *broadcast* e respostas por *unicast* e outro baseado em requisições e respostas por *broadcast* incorporando o mecanismo SbV no encaminhamento de respostas. Em ambos os protocolos não há o emprego de anúncios de serviços. Os gráficos da Figura 5 ilustram os resultados das medições da carga de mensagens de resposta em função do número de dispositivos na MANET, para diferentes percentuais de dispositivos respondentes. As barras verticais nos gráficos indicam os intervalos de confiança. Podemos observar nesses gráficos que o uso do mecanismo SbV permite uma redução crescente, relativamente ao protocolo BCast, no número total de transmissões face a um número crescente de dispositivos na MANET. Podemos constatar resultados ainda mais acentuados de supressão quando se aumenta o percentual de dispositivos ($p$) na MANET com interesse em colaborar na provisão de serviços. Esses resultados indicam uma boa escalabilidade para protocolos que adotem o mecanismo SbV no tratamento de respostas, como o P2PDP.

O diâmetro de supressão das mensagens de resposta mede a distância – em número de saltos – dos dispositivos onde ocorrem as supressões até o dispositivo requisitante. Essa métrica permite-nos avaliar a distribuição da redução no consumo de bateria entre os dispositivos na MANET propiciado pelo mecanismo SbV. Os gráficos da Figura 6 ilustram os resultados das medições do diâmetro das supressões, para diferentes números de dispositivos e percentuais de dispositivos respondentes na MANET, sob a forma de uma distribuição cumulativa de probabilidades. Para ilustrar melhor a distribuição das supressões pela MANET, os resultados das medições são contrastados, nos gráficos da figura, com uma função distribuição cumulativa uniforme que é representada no gráfico pela linha contínua em diagonal. Podemos observar nesses gráficos uma melhor distribuição das supressões com o aumento no número de dispositivos e percentuais de dispositivos respondentes ($p$) na MANET, o que novamente atesta a escalabilidade da estratégia proposta.

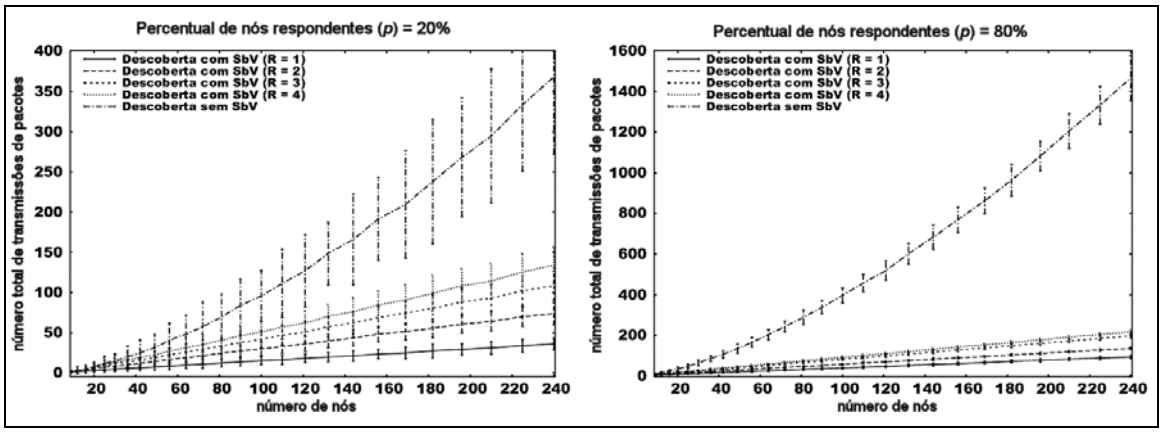


**Figura 5. Carga na MANET devido a mensagens de resposta.**

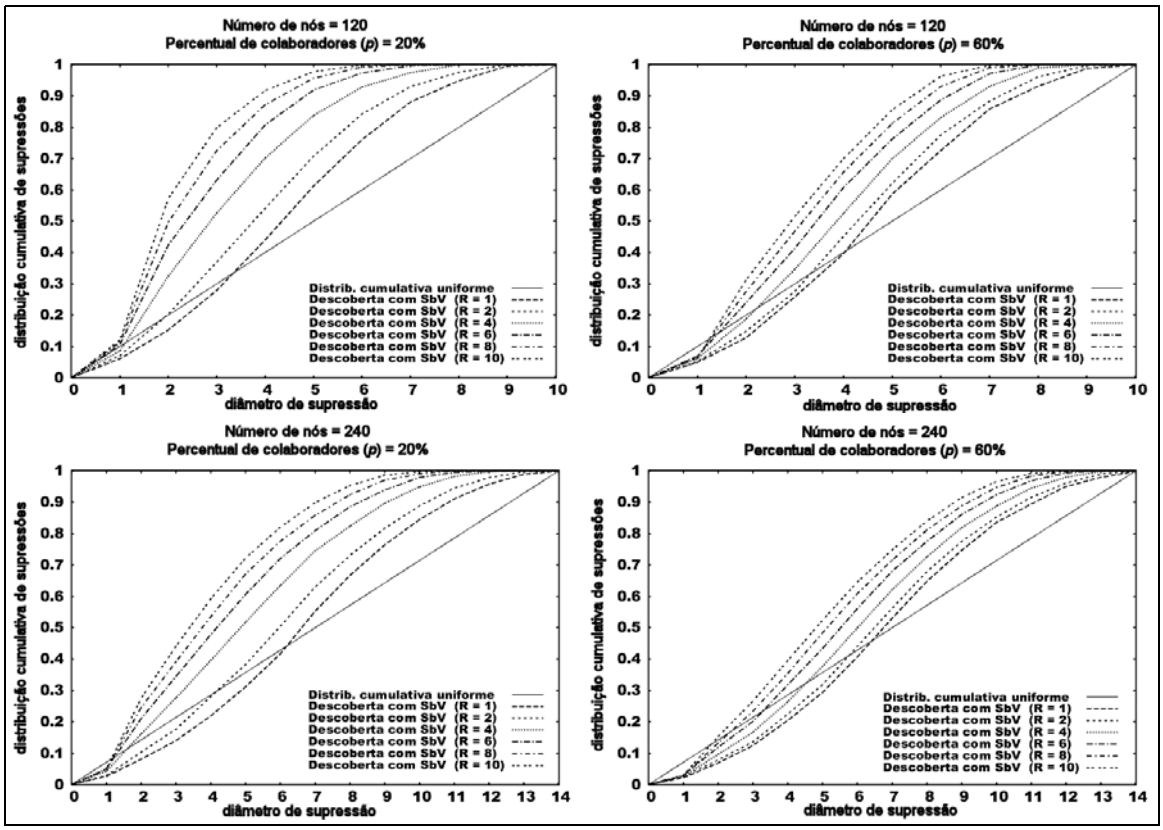


**Figura 6. Distribuição das supressões de respostas na MANET.**

## 6. Conclusões

Neste artigo apresentamos uma estratégia para reduzir a implosão de respostas em protocolos de descoberta de serviços para redes sem fio *ad hoc* de saltos múltiplos. O mecanismo proposto de supressão de respostas por vizinhança mostrou-se eficaz em controlar o volume de respostas recebido pelo requisitante do serviço, sem comprometer a qualidade das respostas recebidas no protocolo P2PDP – isto é, favorecendo as respostas dos colaboradores mais aptos. Pudemos observar ainda que o processamento adicional, gerado pelo mecanismo de supressão, é distribuído uniformemente entre todos os dispositivos, evitando que os vizinhos diretos do dispositivo requisitante sejam sobrecarregados, promovendo um balanceamento indireto no consumo de energia.

Durante o desenvolvimento deste trabalho foram identificados alguns aspectos que merecem uma investigação futura. O primeiro deles é a influência do parâmetro `maxReplyDelay` na eficiência do mecanismo de supressões. Ajustar esse parâmetro (por exemplo, em função do retardo de transferência das mensagens) é essencial para reduzir o tempo de descoberta sem aumentar o número de colisões de respostas, intento logrado com o assincronismo na transmissão dessas mensagens. Ainda nesse contexto, vale ressaltar a importância de se estudar o impacto da diferença de velocidade dos relógios dos dispositivos nos mecanismos de temporização implementados. Um segundo ponto importante é a definição de um mecanismo de reconhecimento salto-a-salto de respostas que contorne o problema da baixa confiabilidade das transmissões *broadcast* no protocolo CSMA/CA. Para isso, vislumbra-se o uso da própria transmissão em *broadcast* de uma resposta por um dispositivo como reconhecimento para a transmissão

dessa mesma resposta pelo dispositivo anterior, no caminho de retorno da mensagem. Vários aspectos do mecanismo SbV dependem da probabilidade de colisões; para avaliar a influência desse fator no funcionamento do mecanismo proposto é necessária a realização de testes em ambientes mais realísticos. Por fim, nossos cenários de simulação consideraram dispositivos com baixa mobilidade. Em cenários mais dinâmicos, o conceito de caminho de retorno usado pelas mensagens de resposta pode falhar, por exemplo, se algum dispositivo no caminho se afastar do mesmo, configurando rotas assímetricas para mensagens de requisição e resposta. Nesse sentido, estão sendo investigados mecanismos que permitem o uso alternativo de protocolos de roteamento *ad hoc* tradicionais quando é detectada uma falha no caminho de retorno.

## Referências


Chakraborty, D., Joshi, A., Yesha, Y., Finin, T. (2006) "Toward Distributed Service Discovery in Pervasive Computing Environments", In: *IEEE Transactions on Mobile Computing*, 5(2):97-112.

Duffield, N.G., Grossglauser, M., Ramakrishnan, K.K. (1999) "Distrust and Privacy: Axioms for Multicast Congestion Control", In: *Proc. 9th Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video*.

Guttman, E. (1999) "Service Location Protocol: Automatic Discovery of IP Network Services", In: *IEEE Internet Computing*, 3(4):71-80.

Harbird, R., Hailes, S., Mascolo, C. (2004). "Adaptive Resource Discovery for Ubiquitous Computing". In: *Proc. 2nd Workshop on Middleware for Pervasive and Ad-Hoc Computing*, 77:155-160.

Helal, S., Desai, N., Verma, V., Lee, C. (2003) "Konark – A Service Discovery and Delivery Protocol for Ad-hoc Networks", In: *Proc. 3rd IEEE Conference on Wireless Communication Networks*.

ISI – Information Sciences Institute. (1995–2007) "*The Network Simulator – ns-2*". [online]. http://www.isi.edu/nsnam/ns.

Koodli, R. and Perkins, C.E. (2002) "Service Discovery in on-demand Ad Hoc Networks", *IETF Internet Draft* (expired).

Kurkovsky, S., Bhagyavati, Ray, A. (2004) "Modeling a Grid-Based Problem-Solving Environment for Mobile Devices", In: *Proceedings of IEEE International Conference on Information Technology: Coding and Computing*.

Lima, L.S., Gomes, A.T.A., Ziviani, A., Endler, M., Schulze, B., Soares, Luiz F.G. (2005) "Peer-to-Peer Resource Discovery in Mobile Grids", In: *Proceedings of 3rd International Workshop on Middleware for Grid Computing*, ACM 1-59593-269-0/05/11.

Marin-Perianu, R.S., Hartel, P., Scholten, H. (2005) "*A Classification of Service Discovery Protocols"*, Technical Report TR-CTIT-05-25 Centre for Telematics and Information Technology, University of Twente, Enschede. ISSN 1381-3625.

Mian, A.N., Beraldi, R., Baldoni, R. (2006) "*Survey of Service Discovery Protocols in Mobile Ad Hoc Networks*", Technical Report 4/06, Dipartimento di Informatica e Sistemistica "Antonio Ruberti", Università degli Studi di Roma "La Sapienza", Italy.

Nidd, M. (2001) "Service Discovery in DEAPspace", In: *IEEE Personal Communications*, 8(4):39-45.

Ratsimor, O., Chakraborty, D., Joshi, A., Finin, T. (2002) "Allia: Alliance-based Service Discovery for Ad-Hoc Environments", In: *Proceedings of ACM Mobile Commerce Workshop*.

Rittenbruch, M., Kahler, H., Cremers, A.B. (1998) "Supporting Cooperation in a Virtual Organization", In: *Proc. 19th Int'l Conference on Information Systems*, pp. 30-38.

Sacramento, V., Endler, M., Rubinsztejn, H.K., Lima, L.S., Gonçalves, K., Nascimento, F.N., Bueno, G.A. (2004) "MoCA: a Middleware for Developing Collaborative Applications for Mobile Users", In: *IEEE Distributed Systems Online*, 5(10):2.

Sun Microsystems. (1999) "*Jini Technology*". http://www.sun.com/jini.

The Salutation Consortium Inc. (1999) "*Salutation Architecture Specification*". http://www.salutation.org.

Tseng, Y.-C., Ni, S.-Y., Shih, E.-Y. (2003) "Adaptive Approaches to Relieving Broadcast Storms in a Wireless Multihop Mobile Ad Hoc Network", In: *IEEE Transactions on Computers*, 52(5):545-557.

Varshavsky, A., Reid, B., de Lara, E. (2005) "A Cross-Layer Approach to Service Discovery and Selection in MANETs", In: *Proc. 2nd Int'l Conference on Mobile Ad-Hoc and Sensor Systems*.