Sistemas & Gestão 2 (3) 303-318
Programa de Pós-graduação em Sistemas de Gestão, TEP/TCE/CTC/PROPP/UFF

Um estudo de simulação para avaliar o tempo de comunicação e o consumo de energia em um ambiente de cooperação móvel

Carla Diacui Medeiros Berkenbrock[1], diacui@ita.br

Celso Massaki Hirata[1], hirata@ita.br

[1] Divisão de Ciência da Computação, Instituto Tecnológico de Aeronáutica - ITA

São José dos Campos, SP, Brasil

*Recebido: Outubro, 2007 / Aceito: Dezembro, 2007

RESUMO

A comunicação entre dispositivos móveis e servidores fixos não é simétrica. Dispositivos móveis gastam mais tempo e energia quando enviam mensagens do que quando as recebem. Aplicações cooperativas fazem uso intensivo de comunicação com o intuito de manter a consistência de dados. Uma maneira de reduzir não apenas o tempo de acesso como também o consumo de energia é armazenar os dados freqüentemente acessados na cache dos clientes móveis. Contudo, determinar se a cópia na cache do cliente está consistente com o servidor pode ser uma operação custosa. Neste artigo, analisa-se o tempo de acesso e o consumo energia através de um novo mecanismo para tratar a coerência de cache em um ambiente cooperativo móvel. Os estudos foram conduzidos com base em um modelo de simulação.

Palavras-Chave: Groupware móvel. Consumo de energia. Coerência de cache.

1. INTRODUÇÃO

Computer-Supported Cooperative Work (CSCW) é um campo de pesquisa multidisciplinar que estuda sistemas para integrar o processamento de informações e atividades de comunicação (ELLIS, 1991). O CSCW se preocupa em verificar como grupos trabalham e busca descobrir como a tecnologia, especialmente relacionada com computadores, pode ajudar no desenvolvimento do trabalho em grupo.

Muitos trabalhos utilizam o termo *groupware* como sinônimo de CSCW. Entretanto, a área de pesquisa CSCW estuda os conceitos relacionados aos trabalhos cooperativos de forma genérica e ampla. Um *groupware* é constituído da tecnologia empregada para possibilitar o trabalho em grupo, sendo assim, em se tratando de *groupware* o foco está mais restrito. De acordo com ELLIS e GIBBS (1989), sistemas de *groupware* são sistemas baseados em computador que suportam dois ou mais usuários engajados em uma tarefa comum e que fornecem uma interface para ambiente compartilhado. Com isso, para o estudo de *groupware* é necessária a existência de um sistema computacional. No entanto é

importante ressaltar que o objetivo do *groupware* não é o sistema computacional em si, mas sim a interação e colaboração entre os participantes do grupo cooperativo.

Aplicações de *groupware* permitem que equipes distribuídas geograficamente trabalhem de forma simultânea através de um ambiente computadorizado. Esses sistemas suportam a manipulação de documentos por um grupo através de um ambiente compartilhado.

A forma de interação entre os participantes de um *groupware* pode ser síncrona ou assíncrona. Um *groupware* síncrono apresenta a idéia de simultaneidade no tempo, isto é, as notificações de eventos são imediatamente propagadas para todos os participantes. Em um *groupware* assíncrono os participantes interagem em momentos distintos, sendo necessário à existência de registros de mudanças de estados.

Várias plataformas de *groupware* têm sido sugeridas para simplificar o desenvolvimento de aplicações de *groupware* síncronos e diminuir as preocupações do desenvolvedor com problemas como padrões de comunicação, sincronização, coordenação e controle de concorrência (KINDBERG, 1996; ROTH, 2002). Entretanto, um dos desafios relacionados às ferramentas existentes para trabalhos cooperativos é a falta de suporte a mobilidade (DIVITINI, 2004). Assim, existe uma necessidade de técnicas para o desenvolvimento de sistemas *groupware* adaptáveis para disponibilizar recursos e apoiar os usuários móveis.

A utilização de dispositivos móveis para apoiar a gestão de informações é recomendável em ambientes que exigem deslocamento físico dos usuários, por exemplo, nos corredores de uma universidade, de um hospital, ou mesmo nos transportes. Contudo, a computação móvel impõe mais dificuldades para a computação distribuída. A comunicação sem fio permite a mobilidade dos dispositivos, para tanto ela requer que usuário mantenha-se conectado sem a necessidade de uma infra-estrutura de comunicação cabeada fixa e, na maior parte dos casos, estática. O avanço da computação móvel em conjunto com as novas formas de conectividade permite a integração de dispositivos móveis em aplicações cooperativas, como por exemplo, sistemas de *groupware* (LUFF e HEATH, 1998). No entanto, as técnicas para realização de trabalhos cooperativos atuais não foram elaboradas para lidar com dispositivos de mão e nem com as restrições existentes nesses equipamentos. Entre essas restrições, podem-se citar o tamanho da tela dos dispositivos móveis, limitado poder de bateria e manutenção de coerência de cache.

A ocorrência de desconexões é comum em ambientes sem fio, o que torna importante a redução da comunicação durante o processo de cooperação. Nesse sentido, a replicação de dados é uma técnica chave para fornecer uma maior disponibilidade e desempenho em ambientes móveis (SAITO e SHAPIRO, 2005). As réplicas, armazenadas localmente nos dispositivos móveis, são conhecidas na literatura como cache (BARBARA e IMIELISKI, 1994; CHUNG e CHO, 1998; SAITO e SHAPIRO, 2005).

A replicação é uma forma de reduzir a quantidade de informações transferidas, a latência de acesso aos dados e o consumo de energia. No entanto, a replicação apenas reduz a latência das transações se o cliente encontrar as informações de seu interesse em sua cache local. Em ambientes sem fio com dispositivos móveis, a desconexão pode ser indesejada por ocasionar perda de sinal ou desejada quando o usuário necessita economizar bateria. Contudo, é importante ressaltar que, por causa das desconexões, torna-se difícil à verificação da validade da cache dos dispositivos móveis.

Métodos eficientes de coerência de cache são críticos para garantir que as aplicações em ambientes sem fio tenham um desempenho razoável (YUEN, 2000). Entretanto, a maioria dos métodos atuais para coerência de cache em ambientes móveis não foi projetada para trabalhar com sistemas de *groupware* (BARBARA e IMIELISKI, 1994; CHUNG e CHO, 1998; WANG, 2004). Adicionalmente, a maior parte dessas estratégias não contempla o conceito de *awareness*.

Awareness é um conceito que busca apresentar ao usuário um conhecimento e entendimento sobre o seu próprio grupo e sobre as atividades de seus colegas, fornecendo

um contexto compartilhado das atividades individuais. Para isso, os participantes de um *groupware* podem manter uma lista de informações enquanto estão compartilhando o espaço. Entre estas informações encontram-se elementos de identidade, localização, nível de atividade, ações, intenções, objetivos entre outros.

De acordo com FUKS (2007) *awareness* é diferente de percepção, pois a percepção em si está relacionada aos sentidos humanos, enquanto que *awareness* está relacionada à ciência de informações. Assim, informações de *awareness* precisam ser fornecidas pelo sistema, isso porque os sentidos humanos não conseguem captar eventos em locais remotos ou acompanhar os eventos ocorridos ao longo de uma sessão de trabalho cooperativo.

Informações de *awareness* limitadas podem resultar em perda do contexto e familiaridade necessária para interações e comunicações que são a chave da colaboração (BELLOTTI, 1996). Contudo, existe um custo relacionado ao fornecimento de informações de *awareness* em termos de sobrecarga de informações, estado real de tela, recurso de rede e privacidade (LITIU, 2000). As dificuldades na provisão da *awareness* aumentam quando trata de dispositivos móveis com interfaces limitadas, como *Personal Digital Assistants* (PDAs). Nesses dispositivos o sistema não pode entregar a mesma quantidade de informações que entregaria a um computador pessoal convencional.

Nesse contexto, o objetivo deste trabalho é apresentar a simulação para verificar a viabilidade de uma técnica de manutenção da coerência de cache e provisão de *awareness* em um ambiente de *groupware*, em termos de tempo de comunicação e duração de bateria. Considera-se que esse ambiente suporte a mobilidade dos usuários através de uma interface de comunicação sem fio. O artigo está organizado da seguinte forma. A seção 2 apresenta conceitos relacionados com a assimetria de comunicação e consumo de energia em ambientes móveis. A seção 3 contém a descrição da técnica proposta para prover a coerência de cache e *awareness* em aplicações de *groupware* móvel. A simulação da técnica, resultados obtidos e análise dos resultados estão descritos na seção 4. Finalmente, na seção 5, apresentam-se as conclusões e trabalhos futuros.

2. OBJETIVO DO ESTUDO DA SIMULAÇÃO

O consumo de energia em dispositivos equipados com bateria é uma questão crítica e requer atenção. De acordo com PERING (2006), a utilidade dos dispositivos móveis está diretamente relacionada com o tempo de vida de suas baterias antes que seja necessário trocá-las ou recarregá-las. Utilizando por exemplo à tecnologia WiFi, um PDA conectado a uma rede WLAN IEEE 802.11b tem a sua energia consumida em poucas horas (SHIH, 2002).

Outra questão importante é que a comunicação entre dispositivos móveis e os servidores fixos é de natureza assimétrica (MURTHY, 1998), ou seja, a largura de banda na direção *downlink* (servidor para cliente) é muito maior do que na direção *uplink* (cliente para servidor). Além disso, a transmissão de dados pelas unidades móveis (UMs) consome mais energia do que o recebimento de dados. A curta duração das baterias dos dispositivos móveis pode ocasionar interrupções inesperadas dos usuários, prejudicando o andamento do trabalho cooperativo. Com isso, para estender o tempo de vida das baterias, são desejáveis protocolos onde as UMs recebam mais mensagens do que as enviem.

Armazenar os dados freqüentemente acessados na cache das UMs é considerado um mecanismo efetivo para conservar largura de banda e energia (CAI e TAN, 1999). Quando os itens de dados estão disponíveis na cache, as UMs evitam utilizar o canal de *uplink*, reduzindo a transmissão de dados e isso implica em uma melhor utilização da largura de banda. Adicionalmente, nesse caso não é necessário consumir energia na transmissão de dados. A redução da transmissão de dados pode ainda representar economia de custos, dependendo do tipo de bilhetagem (tarifa) que esteja sendo aplicado.

Contudo, os efeitos das alterações realizadas por um usuário, devem se refletir na cache dos outros participantes do grupo cooperativo. As técnicas de coerência de cache

possibilitam a consistência entre as informações. Entretanto, as freqüentes desconexões e a mobilidade dos clientes dificultam a manutenção da coerência de cache com os dados do servidor.

Conforme SHEN (2005) existem três questões principais envolvidas no projeto da cache para dispositivos móveis: recolocação, busca prévia e invalidação. A recolocação (*replacement*) é o processo de selecionar um conjunto de itens de dados para serem descartados da cache de um dispositivo móvel, a fim acomodar novos dados. A busca prévia (*prefetching*) é o processo de selecionar previamente itens de dados para a cache dos dispositivos móveis, de modo a antecipar os acessos futuros. E por fim, a invalidação (*invalidation*) é o processo de invalidar itens de dados da cache para manter a coerência dos dados locais e o servidor original. O foco deste artigo está na invalidação. Assim, para manter a coerência de cache é necessário que as UMs sejam informadas quando os itens de dados de suas caches são alterados.

Uma maneira de informar as UMs sobre as alterações em sua cache é através da notificação de invalidação (NI) (BARBARA e IMIELISKI, 1994). A NI é uma abordagem utilizada para que a cache seja reutilizada após desconexões. Nessa abordagem, um servidor dissemina notificações contendo as alterações/invalidações a serem realizadas na cache das UMs. As abordagens baseadas em notificações fornecem uma coerência de cache fraca. Isto porque nessas abordagens as UMs devem confirmar com o servidor todas as alterações que realizaram em sua cache.

O servidor, responsável pelas notificações, pode ser do tipo *stateful* ou *stateless*. O servidor *stateful* tem conhecimento de quais UMs estão conectadas a ele e dos dados armazenados localmente nelas. Dessa forma, ao ocorrer uma alteração, ele envia notificações de invalidação (NIs) diretamente para as UMs afetadas. O servidor *stateless* não tem conhecimento sobre as UMs conectadas a ele, assim como, não possui conhecimento do conteúdo dos dados dessas UMs. Nesse caso, as mensagens de invalidação são enviadas para todas as UMs.

Diversos trabalhos mencionam a assimetria da comunicação e o consumo de energia em ambientes sem fio (SHIH, 2002; VIREDAZ, 2003; PERING, 2006). De acordo com PERING (2006) o consumo de energia e tempo de conclusão das operações são métricas efetivas para a avaliação de sistemas de computação móvel.

Neste artigo, a análise da assimetria na comunicação é realizada através do tempo de comunicação durante o processo de entrada em um grupo cooperativo. Nesta etapa, é verificado se o tempo de comunicação resulta em impactos ao desenvolvimento do trabalho em grupo. PERING (2006) estima que o consumo devido à comunicação sem fio seja aproximadamente 70% da energia total consumida por um dispositivo móvel. Os outros 30% correspondem ao consumo de processamento. Assim, para a análise da energia consumida pelas UMs em aplicação cooperativa, usa-se o consumo realizado pela técnica para manter a coerência de cache e *awareness* durante o processo de cooperação em relação ao consumo total e compara-se com a porcentagem estimada por Pering.

3. UMA TÉCNICA PARA COERÊNCIA DE CACHE E *AWARENESS* EM AMBIENTES DE *GROUPWARE* MÓVEL

Em dispositivos móveis maiores, como *notebooks*, a mobilidade é limitada por fatores como peso e tamanho do equipamento. Para que o *groupware* suporte uma maior mobilidade dos usuários, no ambiente proposto, são investigados problemas relacionados com pequenos dispositivos móveis, como PDAs. A utilização de PDAs é adequada considerando que ela fornece um elevado grau de acesso sem fio e mobilidade.

No ambiente, cada dispositivo móvel possui capacidade de armazenar dados localmente. Conforme apresentado na seção 2, os dados das UMs podem ser atualizados através de servidores *stateful* ou de servidores *stateless*. Neste trabalho, optou-se pela utilização de um ambiente híbrido, mesclando características das abordagens *stateless* e *stateful*. Da mesma forma que ocorre na abordagem *stateless*, a técnica utiliza envio

periódico de notificações de invalidação. Entretanto, as UMs podem consultar o servidor sempre que necessário e elas não precisam esperar pelo novo envio da notificação para receber as respostas das suas solicitações. Essa liberdade de responder aos questionamentos das UMs a qualquer momento é uma das características da abordagem *stateful*. Outra característica da abordagem *stateful* está relacionada ao conhecimento do servidor a respeito da UMs. No ambiente proposto, esse conhecimento é limitado, considerando que o servidor não terá informação completa a repeito das UMs (como por exemplo, ele não saberá o conteúdo e tempo do cache nas UMs). Entretanto, o servidor tem conhecimento a respeito da conexão das UMs.

Adicionalmente, o ambiente fornecerá suporte a informações de *awareness*. Para fornecer elementos de *awareness*, o ambiente permite que as UMs tenham conhecimento de informações relacionadas à conectividade e as atividades dos membros no grupo. As informações de *awareness* das atividades realizadas seguem o modelo de interface *What You See Is What I Did* (WYSIWID) (MARSHALL, 1991). Nesse modelo, as interações entre os usuários dão-se de forma assíncrona, pois a propagação das alterações para os demais usuários será adiada até que as condições presentes no servidor do *groupware* sejam estabelecidas.

Considerando um ambiente de *groupware* móvel, com possibilidade de desconexão dos participantes, manter a sincronização dos dados geralmente é dispendioso e pode ser inapropriado. A abordagem proposta neste artigo utiliza um ambiente flexível. Os participantes conectados ao sistema trabalham de forma síncrona, enquanto os participantes desconectados podem continuar a cooperação de forma assíncrona.

O ambiente representa as informações compartilhadas através de uma estrutura de dados do tipo grafo. Várias formas de representação que usam grafos podem ser adotadas, como por exemplo, redes de Petri e diagramas de classe da *Unified Modeling Language* (UML). Adicionalmente, vários trabalhos utilizam grafos para demonstrar métodos de coerência em ambientes compartilhados (CORREA e MARSIC, 2003; NEYEM, 2006).

O trabalho cooperativo é realizado a partir de um ambiente infra-estruturado, isto é, a rede sem fio envolve servidores com localização fixa (Estação de Suporte Móvel), que se comunicam com os dispositivos móveis (Unidades Móveis) via onda de radio. A Estação de Suporte Móvel (ESM) possui um ponto de acesso IEEE 802.11 (interface sem fio). Através desse ponto de acesso, a ESM fica responsável pela área geográfica chamada célula. Apenas as unidades móveis (UMs) que estiverem na área de cobertura dessa célula e que possuem energia suficiente para manter os serviços de rede poderão se comunicar com o servidor. Com isso, a ESM atua como um intermediário entre as UMs e a Estação Fixa (EF), sendo responsável pela comunicação. A EF é o servidor de *groupware*. A Figura 1 representa o ambiente de computação móvel considerado neste artigo.

3.1. ARQUITETURA DO AMBIENTE

A arquitetura é uma questão importante no projeto de aplicações cooperativas (DEWAN, 1999). Através dela é determinada à natureza dos componentes de um sistema e a forma como esses componentes interagem.

Em arquiteturas centralizadas, os clientes de um grupo utilizam um único programa de aplicação central, localizado no servidor. Esse programa processa as solicitações dos clientes e distribui as respostas a essas solicitações para que sejam exibidas nos clientes. Entre as vantagens de um modelo centralizado, pode-se destacar a facilidade de sincronização e consistência das informações. Contudo, um modelo centralizado implica no processamento seqüencial, onde deve-se assegurar o atendimento as solicitações na ordem em que elas ocorreram. Se a latência é baixa, não há problemas. No entanto, se ela for alta, o comportamento correto sistema pode ficar comprometido.

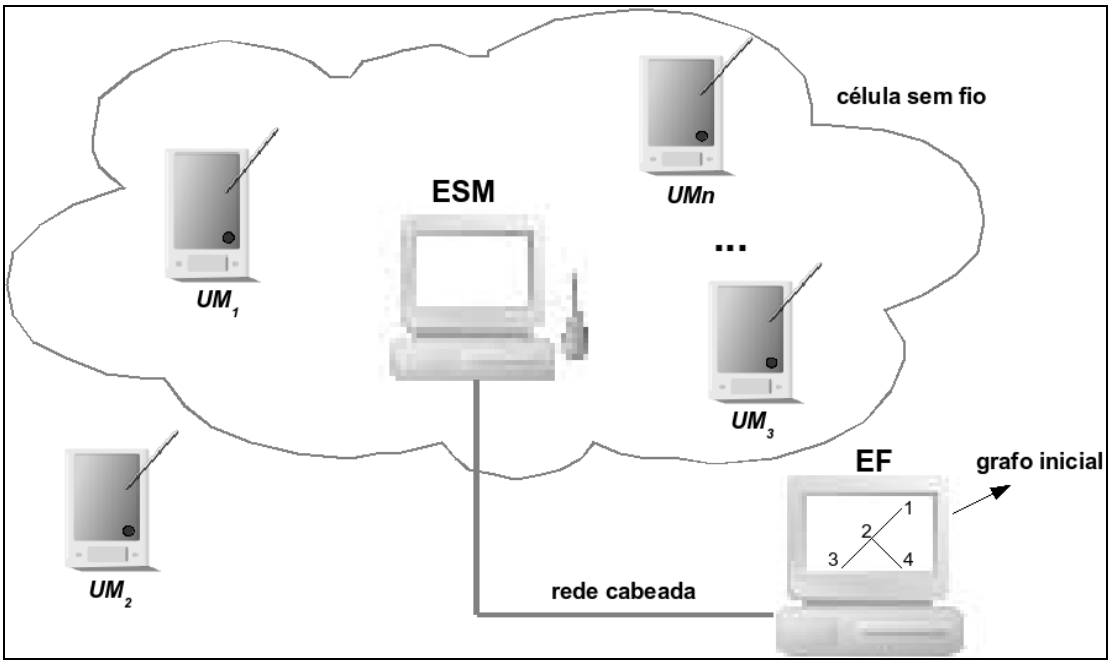

Figura 1. Ambiente experimental.

Por outro lado, nas arquiteturas replicadas cada cliente possui uma cópia do programa de aplicação. Os clientes junto com o servidor devem coordenar as ações realizadas localmente e as ações remotas e garantir sincronização entre as cópias. Um esquema replicado implica no processamento paralelo. As réplicas podem ser atualizadas por meio de notificações a respeito das mudanças ocorridas. Contudo, enquanto as mensagens relacionadas às atividades remotas podem ser recebidas com atraso, as atividades locais podem ser processadas imediatamente, possibilitando divergências entre as cópias.

Entre as duas arquiteturas mencionadas anteriormente estão às arquiteturas híbridas semi-replicadas (GREENBERG e ROSEMAN, 1996). Essa arquitetura contém tanto componentes centralizados quanto replicados.

O problema da coerência de cache é tratado neste trabalho através da arquitetura híbrida semi-replicada. Cada unidade móvel executa uma cópia do programa de aplicação e possui uma réplica das informações compartilhadas pelos participantes do *groupware*. As UMs podem alterar as suas réplicas locais (cache), entretanto, essas alterações apenas poderão ser efetivadas pelo servidor. O servidor recebe as requisições das UMs, e assim que elas são atendidas, as alterações são propagadas para os participantes conectados ao grupo cooperativo.

A arquitetura do ambiente proposto é apresentada na Figura 2. Os componentes principais dessa arquitetura são a EF e as UMs. Nesse ambiente, a EF se comunica com as UMs através do canal de *downlink*. Essa comunicação apenas será realizada se a UM possuir energia e estiver na área de cobertura da rede sem fio. As UMs podem obter réplicas dos grafos armazenados na EF. Adicionalmente, elas podem solicitar alteração dos grafos através do canal de *uplink*. No entanto, essa alteração apenas será realizada com a confirmação da EF.

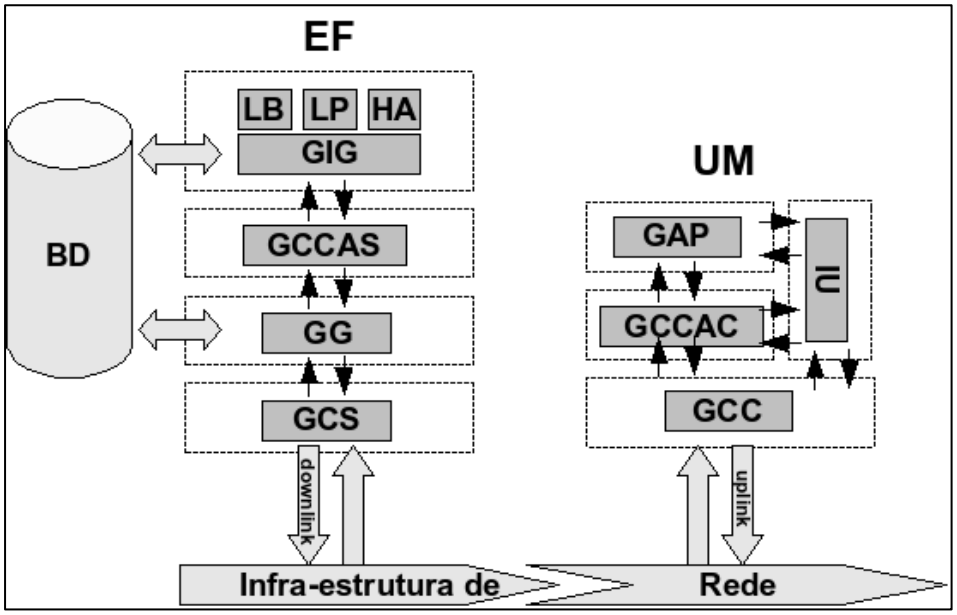

Figura 2. Arquitetura do ambiente.

A EF é composta pelos módulos descritos a seguir:

- Gerenciador de conexão no servidor (GCS) - Gerencia a conexão da EF com as UMs;
- Gerenciador de *groupware* (GG) - Responsável pelo gerenciamento das UMs no *groupware*. O GG recebe as solicitações das UMs. O GG também é responsável por retornar as respostas a essas solicitações e enviar as notificações dos eventos ocorridos;
- Gerenciador para coerência de cache e *awareness* no servidor (GCCAS) - Realiza o gerenciamento da coerência de cache e das informações de *awareness* na EF. É responsável pela atualização do rótulo de tempo da EF (RT_{EF}), rótulo de tempo das UMs (RT_{UM}), janela de desconexão (*w*) e contador de tempo de desconexão (*ct*). Essas variáveis possuem papel fundamental na realização da coerência de cache. Os valores RT_{UM} *e* RT_{EF} ajudam a verificar se as UMs deixaram de receber alguma notificação de alteração (*NA*). A *NA* contém as informações necessárias para que as UMs atualizem suas caches, sempre que ocorrer alguma alteração no grafo. Os rótulos de tempo indicam quando o grafo foi atualizado pela última vez. A atualização dos valores RT_{UM} *e* RT_{EF} ocorre a cada alteração do grafo na EF. O valor de *w* indica quanto tempo as UMs podem permanecer desconectadas, sem que seus dados sejam invalidados. Adicionalmente, o GCCAS é responsável por ativar o gerenciador de informações no *groupware* (GIG) quando for necessário realizar alguma alteração no banco de dados; e
- Gerenciador de informações no *groupware* (GIG) - Responsável pela alteração da lista de bloqueios (*LB*), lista de pendências (*LP*) e histórico de alterações (*HA*). As informações de bloqueios de nós do grafo, realizados pelo grupo, são armazenados na *LB*. A *LB* será utilizada para verificar se as caches das UMs que se desconectaram da rede precisarão ser invalidadas. A *LP* contém informações sobre as UMs que desejam trabalhar em um nó que já foi bloqueado. O *HA* contém as alterações realizadas nos últimos *w* segundos.

Cada UM é composta pelos seguintes módulos:

- Gerenciador de conexão no cliente (GCC) - Gerencia a conexão da UM com a EF;

- Gerenciador para coerência de cache e *awareness* no cliente (GCCAC) - Nele fica registrado o valor do rótulo de tempo da sua UM (RT_{UM}), janela de desconexão (*w*) e contador de tempo de desconexão (*ct*). Os valores de RT_{UM} e *w* são atualizados pela EF. O GCCAC também é responsável por ativar o gerenciador do armazenamento persistente (GAP) quando for necessário realizar alguma alteração na cache da UM;
- Gerenciador do armazenamento persistente (GAP) - Executa as alterações necessárias na cache do dispositivo móvel; e
- Interface com o usuário (IU) - Representa uma interface amigável para que as UMs possam acessar o ambiente. Através dessa interface a UM pode realizar consultas e alterações em suas réplicas locais, e confirmá-las na EF.

3.2. DESCRIÇÃO DA TÉCNICA

Os seguintes passos constituem uma descrição da técnica para coerência de cache e *awareness* em sistemas de *groupware* com suporte a mobilidade:

1. Após entrarem no *groupware*, as UMs solicitam o envio do grafo relacionado ao seu grupo cooperativo. Dessa forma, o grafo é armazenado na cache de cada UM. Adicionalmente, a EF envia um rótulo de tempo que será armazenado nas UMs para consistências futuras. Então, as UMs podem solicitar o bloqueio dos nós de interesse;
2. Ao receber uma solicitação de bloqueio da UM, a EF pode aceitá-la ou recusá-la. Em caso de aceite, a EF notifica os demais membros do grupo sobre o bloqueio. Os bloqueios realizados pelo grupo são armazenados na *LB*. Se a EF recusar o bloqueio, ela envia uma mensagem para a UM solicitante com o motivo da recusa. Se a recusa ocorre porque o nó solicitado já está bloqueado, a EF envia uma notificação para a UM que possui o bloqueio, avisando sobre o interesse do outro participante do grupo. A EF, então, adiciona a UM solicitante na *LP*. A UM apenas começa a trabalhar no grafo, após conseguir o bloqueio de algum nó. Utiliza-se a abordagem de prevenção de *deadlock* para evitar o *deadlock* de recurso. As UMs devem solicitar o bloqueio de todos os nós desejados de uma única vez. A EF tem o conhecimento do estado global da alocação dos nós através da *LB*;
3. Cada UM portadora de um grafo possui o RT_{UM}, enviado pela EF, indicando quando o grafo foi atualizado pela última vez. Após alterar o grafo, a UM pode confirmar as alterações com a EF. Antes de efetivar as alterações, a EF faz as seguintes verificações:
    - Se $RT_{UM} = RT_{EF}$, a EF registra as alterações da UM no *HA*. Em seguida, a EF atualiza RT_{EF} (o novo valor indicará o instante dessa última alteração) e envia uma *NA* para demais UMs, pertencentes ao grupo, que estão conectadas. A *NA* contém os novos dados do grafo e o novo rótulo de tempo (para que as UMs atualizem o valor do RT_{UM} com o novo valor do RT_{EF}). Nesse momento, o bloqueio do nó que foi atualizado é liberado. A EF, então, verifica na *LP* se existe alguma UM interessada em obter bloqueio do nó liberado. Caso positivo, a EF envia uma notificação para a UM interessada, questionando sobre o interesse em alterar o nó liberado;
    - Se $RT_{EF} - RT_{UM} <= w$, a EF envia para a UM uma mensagem contendo todas as alterações realizadas após o tempo RT_{UM}. Com isso, a UM pode atualizar os seus dados e reiniciar o processo de confirmação da alteração;
    - Se $RT_{EF} - RT_{UM} > w$, isso significa que ocorreram alterações no *groupware* que já não estão mais armazenadas no histórico. A EF, então verifica na *LB* se algum outro participante bloqueou o nó utilizado pela UM após o tempo RT_{UM}. Caso positivo, a EF não tem capacidade de verificar a coerência de cache. Sendo assim, será necessário invalidar as informações da UM. Caso contrário, a EF envia novamente o grafo para que a UM possa unir as suas alterações com o grafo atual;

4. Quando uma UM se desconecta da célula de comunicação, se ela possuir algum nó bloqueado, a EF avisa as demais UMs sobre a desconexão. Nesse caso, a partir do momento de desconexão, a EF e a UM desconectada iniciam o contador *ct*. É necessário que *ct* exista tanto na UM quanto na EF, pois é importante que a UM, ainda que desconectada, tenha consciência de quanto tempo ela poderá trabalhar. Adicionalmente, a EF precisa ter conhecimento do tempo de desconexão da UM. Enquanto *ct* for menor do que *w*, a EF não permite o bloqueio para nenhuma outra UM do nó que está sendo atualizado pela UM desconectada. Caso contrário, a EF liberará o nó bloqueado;
5. Quando a UM se reconecta a célula de comunicação, é verificado o valor de *ct*:
    - Se *ct* <= *w*, a UM desconectada pode solicitar confirmação de alteração com a EF. Porém, antes disso, a EF verificará o RT_{UM} e enviará uma mensagem contendo todas as alterações que ocorreram no grupo enquanto a UM permaneceu desconectada. Com isso, a UM atualiza os seus dados e reinicia o processo de confirmação da alteração;
    - Se *ct* > *w* será necessário verificar, na *LB*, se outro participante bloqueou o nó utilizado pela UM, durante o seu período de desconexão. Caso existam registros de bloqueio do nó na *LB*, será necessário invalidar as informações da UM. Caso contrário, a EF envia novamente o grafo para que a UM realize a união das suas alterações ao grafo atual.

4. MODELAGEM E EXPERIMENTAÇÃO

Durante o processo de cooperação, as UMs consomem energia para enviarem e receberem dados. Entretanto, conforme mencionado na seção 2, a largura de banda no canal de *downlink* é muito maior do que no canal de *uplink*. Com isso, as UMs demoram mais tempo e consomem mais energia para enviarem do que para receberem dados. Os experimentos conduzidos neste artigo consideram o tempo e a energia consumida no envio e recebimento de dados para verificar a eficiência da estratégia proposta.

Os experimentos foram conduzidos utilizando o simulador *Network Simulator* (NS-2) (BERKELEY, 2006). O NS-2 é um simulador para eventos discretos direcionado à pesquisa em redes de computadores. O modelo desenvolvido suporta tanto redes cabeadas (para comunicação da EF com a ESM) quanto redes sem fio (para comunicação das UMs com a ESM). Ele foi escrito na linguagem de programação *Tool Command Language* (Tcl).

Para simular a técnica proposta para coerência de cache e *awareness* foi desenvolvida uma aplicação chamada *Mobile Groupware Cache Coherence* (MGCC). Essa aplicação foi escrita utilizando a linguagem de programação C++. O modelo, implementado em Tcl, em conjunto com a aplicação MGCC foram definidos de acordo com a arquitetura apresentada na seção 3. A aplicação MGCC é inserida em cada UM.

Tabela 1. Parâmetros da simulação.

Notação	**Definição**	**Valor**
A	Área de simulação	500m X 500m
N	Número total de UMs	10
C_{UP}	Capacidade de transmissão no canal de *uplink*	19,2 Kbps
w	Janela de desconexão	65s
C_B	Capacidade da bateria	7600mWh
E_{UP}	Energia consumida por cada UM no envio de dados	81 mW
E_{DOWN}	Energia consumida por cada UM no recebimento de dados	30 mW

A Tabela 1 apresenta a configuração dos parâmetros utilizados nos experimentos. No ambiente é utilizado o protocolo de rede UDP e a camada MAC de acordo com o padrão 802.11. A capacidade de transmissão no canal de *uplink* é de 19,2 Kbps e no canal de *downlink* é de 2,0 Mbps. A energia consumida por cada UM no envio e recebimento de dados varia de dispositivo para dispositivo. Neste artigo, foi estimado o consumo de 81mW para envio e de 30mW para recebimento de dados. Os valores utilizados nas estimativas baseiam-se nos trabalhos de SHIH (2002), LIU (2006) e PAPADOPOULOS (2006). O modelo de simulação implementado não leva em consideração à distância em que as UMs estão da ESM no cálculo do consumo de energia.

O MGCC foi utilizado em um cenário com área de simulação de 500m X 500m, contendo 10 unidades móveis. Nesse cenário ocorre a média de uma alteração por minuto. A configuração da máquina utilizada para simular o ambiente de cooperação é: processador AMD Sempron 2300, 512 MB de memória e sistema operacional Open Suse Linux 10.1.

A Figura 3 apresenta o tempo médio para a troca de mensagens durante o processo de entrada em um grupo cooperativo. Através dessa Figura percebe-se a assimetria na comunicação. Em média, as mensagens enviadas pelas UMs levaram 190,06 ms para chegarem até a EF, ao ponto que as mensagens enviadas da EF para as UMs levaram em média 2,84 ms.

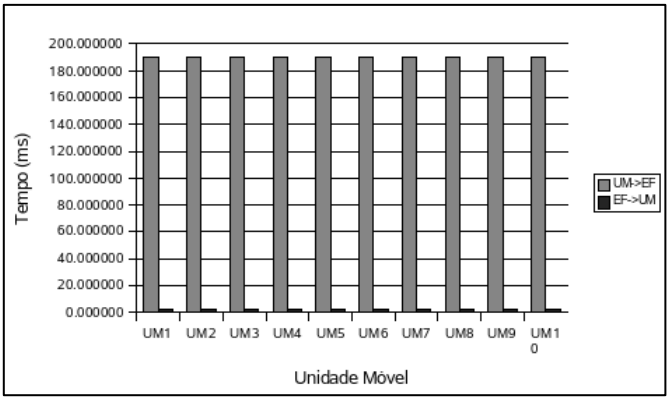

Figura 3. Tempo médio para envio de pacotes.

As Figuras 4, 5 e 6 apresentam a energia consumida na troca de mensagens entre as UMs e a EF, em um experimento contendo uma desconexão. Nas Figuras 7, 8 e 9 é mostrada a energia consumida para enviar e receber mensagens no sistema, em um experimento com 3 desconexões.

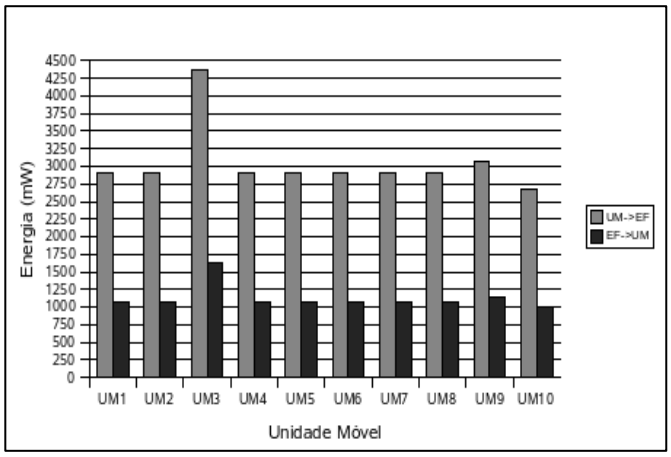

Figura 4. Desconexão da UM_3 com *ct* inferior a *w*.

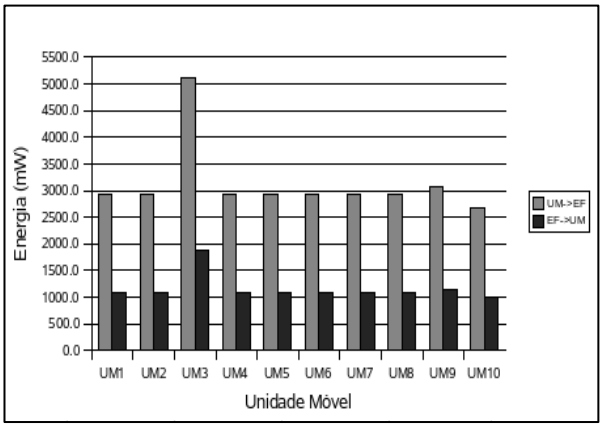

Figura 5. Desconexão da UM_3 com *ct* superior a *w*, sem novos bloqueios no nó em alteração.

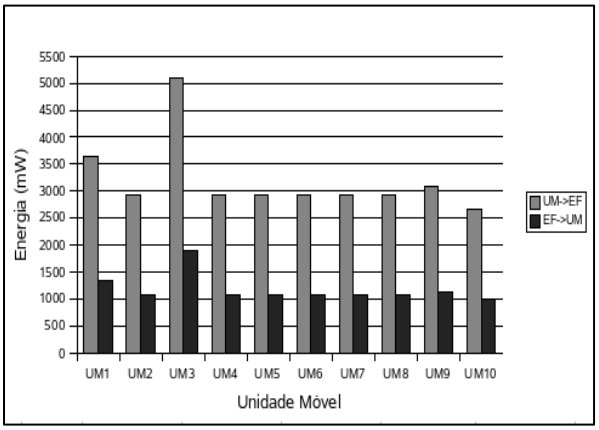

Figura 6. Desconexão da UM_3 com *ct* superior a *w*, com novo bloqueio no nó em alteração.

As desconexões apresentadas nas Figuras 4 e 7 possuem tempo inferior a *w*. Nesse caso, para garantir a coerência de cache, foi necessária a troca de um maior número de mensagens entre as UMs que ficaram desconectadas e a EF (média de 54 pacotes), se comparado com as UMs que permaneceram conectadas durante todo o experimento (média de 36 pacotes). O aumento no número de mensagens trocadas no sistema resulta em um maior consumo de energia. No caso de períodos de desconexão menores que *w*, a média de energia consumida para envio e recebimento de dados no sistema corresponde, respectivamente, 3.199,5 mW e 1.185 mW.

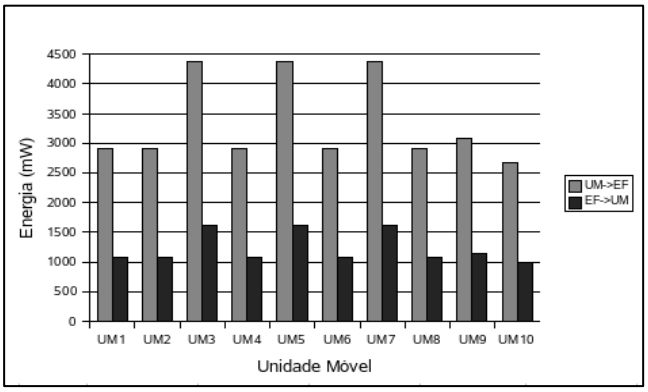

Figura 7. Desconexão da UM_3, UM_5 e UM_7 com *ct* inferior a *w*

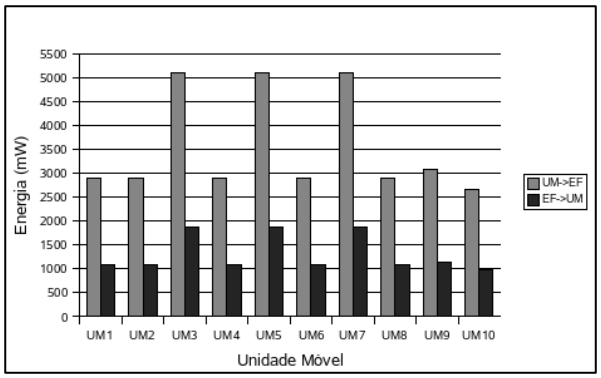

Figura 8. Desconexão da UM_3, UM_5 e UM_7 com *ct* superior a *w*, sem novos bloqueios nos nós em alteração.

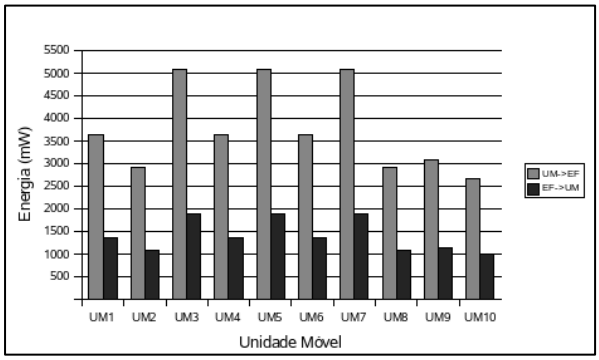

Figura 9. Desconexão da UM_3, UM_5 e UM_7 com *ct* superior a *w*, com novos bloqueios nos nós em alteração.

Quando o tempo de desconexão é maior do que *w* podem ocorrer duas situações. Na primeira situação, nenhum outro participante bloqueia o nó utilizado pela UM durante a sua desconexão. Nesse caso, a EF envia novamente o grafo para a UM incluir suas alterações ao grafo atual (Figuras 5 e 8). Na segunda situação, existe alguma outra UM na *LB*, bloqueando o nó que a UM desconectada estava atualizando. Com isso, os dados da UM que acabou de reconectar são invalidados (Figuras 6 e 9). Se a UM que teve os seus dados invalidados desejar continuar participando das atividades do grupo, ela necessitará reiniciar o processo de entrada no grupo cooperativo, o que resultará em um aumento no número de mensagens trocadas no sistema MGCC e, conseqüentemente, em um maior consumo de energia. Nessa segunda situação, a média de energia consumida no envio de dados pelas UMs é de 3.491,1mW, enquanto que a média de energia consumida no recebimento de dados é de 1.293mW. Nas duas situações apresentadas, a média de pacotes trocados entre as UMs desconectadas e a EF é 63. Contudo, na segunda situação ocorre um aumento no número de mensagens trocadas não apenas entre as UMs que ficaram desconectadas, como também nas UMs que adquirem os novos bloqueios.

4.1. ANÁLISE DOS RESULTADOS

Para uma aplicação de cooperação ser viável, a defasagem de tempo entre uma ação realizada pelo usuário e os resultados apresentados em tela deve ser aceitável. Ainda, a expectativa com o tempo de resposta aumenta de acordo com o tipo de mensagem a ser trocada. Os usuários ficam frustrados quando não conseguem ter resposta às suas solicitações em certo tempo. Nos experimentos realizados, foi analisado o tempo de comunicação para envio de mensagens durante o processo de entrada em um grupo cooperativo. Devido à assimetria na comunicação, as mensagens enviadas pelas UMs levaram mais tempo para chegarem a EF (média de 190,06ms) do que as mensagens enviadas da EF para as UMs (média de 2,84ms). Na comunicação no sentido UM para EF, o menor tempo de comunicação foi de 189,77ms, o maior tempo foi de 190,23ms, e o desvio padrão foi de 0,13ms. A comunicação no sentido EF para UM obteve um tempo de comunicação praticamente constante. Embora, através dos experimentos pode-se verificar a assimetria na comunicação, o tempo de comunicação para envio de mensagens durante o processo de entrada em um grupo não representou impacto negativo significante no trabalho cooperativo.

Conforme PERING (2006), a comunicação sem fio consome aproximadamente 70% da energia total de um dispositivo móvel. Nos experimentos realizados, o cálculo da energia consumida pelas UMs leva em consideração o consumo realizado para enviar e receber dados durante o processo de cooperação. Em situações onde o tempo de desconexão das UMs foi inferior a *w*, o melhor caso, o pior caso e o consumo médio de energia foram, respectivamente, de 3.663 mW, 5.994 mW e 4.386 mW, representando 48,19%, 78,86% e 57,71% da energia consumida por hora da bateria, com desvio padrão de 1.151,94mW. Em

situações onde as desconexões das UMs tiveram *ct* superior a *w* e sem novos bloqueios nos nós em alteração, o consumo médio de energia foi de 4.584,25mW, representando 60.34% da energia consumida por hora da bateria, com desvio padrão foi de 1.712,49mW. E finalmente, quando as desconexões das UMs possuíram *ct* superior a *w* e ocorreram novos bloqueios nos nós em alteração, o consumo médio de energia foi de 4.786,6mW, representando 62.98% da energia consumida por hora da bateria, com desvio padrão de 1.667,88mW. Quando o tempo de duração das desconexões foi superior a *w*, tanto com ou sem novos bloqueios dos nós em alteração, o consumo de energia no pior e melhor caso foram de 3.663mW e 6.993mW, representando, respectivamente 48,19% e 92,01% da energia consumida por hora da bateria. O consumo médio de energia no sistema apresentou uma pequena redução em relação ao valor estimado por Pering para um dispositivo móvel conectado a rede. Contudo, se analisarmos o pior caso para cada uma das situações consideradas na técnica para coerência de cache proposta, em todas as situações a porcentagem de consumo de energia foi maior do que o valor médio praticado pelas aplicações atualmente.

É importante ressaltar que a definição de um valor de *w* ideal, depende do tipo de aplicação. Ambientes com um valor de *w* pequeno podem resultar em elevado número de invalidações. No entanto, um valor alto para *w* pode resultar em retardos para obtenção de bloqueios, aumento da troca de mensagens no sistema e, conseqüentemente, um maior consumo de energia pelas UMs. Dessa maneira, acredita-se que seja essencial obter um conhecimento prévio sobre comportamento dos usuários móveis na operação da aplicação, para que se possa determinar um valor eficiente para a janela de desconexão.

5. CONCLUSÕES E TRABALHOS FUTUROS

A utilização de dispositivos móveis para a gestão de informações torna-se indicada em ambientes que exigem deslocamento físico dos usuários. Contudo, o armazenamento de informações em dispositivos móveis levanta questões associadas ao tempo de comunicação e poder de bateria desses equipamentos.

A comunicação entre dispositivos móveis e os servidores fixos é de natureza assimétrica. Com isso, armazenar os dados freqüentemente acessados na cache das UMs tem sido considerado uma forma muito útil e um mecanismo efetivo para conservar largura de banda e energia. Por outro lado, as freqüentes desconexões e a mobilidade dos clientes dificultam a manutenção da coerência de cache com os dados do servidor.

Nesse contexto, o objetivo deste trabalho foi apresentar a simulação para verificar a viabilidade de uma técnica de manutenção da coerência de cache e provisão de *awareness* em um ambiente de *groupware* móvel. Para simular a técnica proposta foi desenvolvida uma aplicação chamada *Mobile Groupware Cache Coherence* (MGCC).

Os experimentos foram conduzidos utilizando o simulador *Network Simulator*. Nele foi desenvolvido um modelo que representa a estrutura de um cenário de uso proposto onde se considera a assimetria na comunicação e consumo de energia para verificar a eficiência da técnica proposta. Através dos experimentos pôde-se verificar que o tempo de comunicação não representou um problema no tempo de resposta para envio de mensagens durante o processo de entrada em um grupo cooperativo. O consumo médio de energia no sistema apresentou valores que correspondem a porcentagens aceitáveis em relação à média de energia consumida em um dispositivo móvel conectado a rede. Contudo, se analisarmos o pior caso para cada uma das situações consideradas na técnica para coerência de cache proposta, o consumo de energia foi maior do que o valor médio praticado pelas aplicações atualmente.

Trabalhos futuros incluem o aperfeiçoamento do MGCC possibilitando a simulação de cenários de uso mais complexos. Pretende-se possibilitar que o sistema identifique o tamanho da janela de desconexão (*w*) de forma adaptativa. Assim, com base em informações relacionadas às desconexões e quantidade de bloqueios, o sistema poderá alterar o valor de *w*, reduzindo o tráfego na rede e a quantidade de invalidações.

6. REFERÊNCIAS

BARBARA, D.; IMIELISKI, T. Sleepers and workaholics: caching strategies in mobile environments. **In: SIGMOD '94: Proceedings of the 1994 ACM SIGMOD international conference on Management of data.** New York, NY, USA: ACM Press, 1994. p. 1–12. ISBN 0-89791-639-5.

BELLOTTI, V.; BLY, S. Walking away from the desktop computer: distributed collaboration and mobility in a product design team. **In: CSCW'96: Proceedings of the 1996 ACM conference on Computer supported cooperative work.** New York, NY, USA: ACM Press, 1996. p. 209–218. ISBN 0-89791-765-0.

BERKELEY, U.; LBL; USC/ISI; PARC., X. **The ns manual**. August 2006.

CAI, J.; TAN, K.-L. Energy-efficient selective cache invalidation.Wirel. Netw., **Kluwer Academic Publishers**, v. 5, n. 6, p. 489–502, 1999. ISSN 1022-0038.

CHUNG, H.; CHO, H. Data caching with incremental update propagation in mobile computing environments. **Australian Computer Journal**, v. 30, n. 2, p. 77–86, 1998. Disponível em:<citeseer.ist.psu.edu/chung98data.html>.

CORREA, C. D.; MARSIC, I. Software framework for managing heterogeneity in mobile collaborative systems. **ACM Press**, New York, NY, USA, p. 125–134, 2003.

DEWAN, P. Architectures for Collaborative Applications. **Computer Supported Co-operative Work**, v. 7, p. 169 - 193, 1999.

DIVITINI, M.; FARSHCHIAN, B. A.; SAMSET, H. Ubicollab: collaboration support for mobile users. **In: SAC '04: Proceedings of the 2004 ACM symposium on Applied computing**. New York, NY, USA: ACM Press, 2004. p. 1191–1195. ISBN 1-58113-812-1.

ELLIS, C. A.; GIBBS, S. J. Concurrency control in groupware systems. In: SIGMOD '89: **Proceedings of the 1989 ACM SIGMOD international conference on Management of data**. New York, NY, USA: ACM Press, 1989. p. 399–407. ISBN 0-89791-317-5.

ELLIS, C. A.; GIBBS, S. J.; REIN, G. Groupware: some issues and experiences. **Commun. ACM**, ACM Press, New York, NY, USA, v. 34, n. 1, p. 39–58, 1991. ISSN 0001-0782.

FUKS, H.; RAPOSO, A.; GEROSA, M.A.; PIMENTEL, M.; FILIPPO, D.; LUCENA, C.J.P. Inter- e Intra-relações entre Comunicação, Coordenação e Cooperação. **IV Simpósio Brasileiro de Sistemas Colaborativos, Anais do XXVII Congresso da Sociedade Brasileira de Computação**, 2007.

GREENBERG, S.; ROSEMAN, M. Groupware toolkits for synchronous work. **Computer-Supported Cooperative Work (Trends in Software 7)**, p. 135–168, 1996. Disponível em: <citeseer.ist.psu.edu/greenberg96groupware.html>.

KINDBERG, T. A framework for collaboration and interaction across the internet. **The International Workshop on CSCW and the Web**, Sankt Augustin, Germany,February, 1996.

LITIU, R.; PARAKASH, A. Developing adaptive groupware applications using a mobile component framework. **In: Computer Supported Cooperative Work**., 2000. p. 107–116.

LIU, W.; ZHANG, Y.; LOU, W.; FANG, Y. A robust and energy-efficient data dissemination framework for wireless sensor networks. Wirel. Netw., **Kluwer Academic Publishers**, Hingham, MA, USA, v. 12, n. 4, p. 465–479, 2006. ISSN 1022-0038.

LUFF, P.; HEATH, C. Mobility in collaboration. **In: CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work**. New York, NY, USA: ACM Press, 1998. p. 305–314. ISBN 1-58113-009-0.

MARSHALL, C. C.; HALASZ, F. G.; ROGERS, R. A.; JANSSEN, J. W. C. Aquanet: a hypertext tool to hold your knowledge in place. **ACM Press**, New York, NY, USA, p. 261–275, 1991.

MURTHY, V. K. Mobile computing deploying agents. **Twelfth International Conference on Information Networking**, IEEE, Koganei, Tokyo, Japan, p. 127–130, 1998.

NEYEM, A.; OCHOA, S. F.; PINO, J. A. A strategy to share documents in manets using mobile devices. **In: The 8th International Conference Advanced Communication Technology**, ICACT 2006. IEEE Computer Society, 2006. p. 1400–1404.

PAPADOPOULOS, M.-C. Improving awareness in mobile cscw. **IEEE Transactions on Mobile Computing**, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 5, n. 10, p. 1331–1346, 2006. ISSN 1536-1233. Member-Constantinos Papadopoulos.

PERING, T.; AGARWAL, Y.; GUPTA, R.; ROYWANT. Coolspots: reducing the power consumption of wireless mobile devices with multiple radio interfaces. **ACM Press**, New York, NY, USA, p. 220–232, 2006.

ROTH, J. Seven challenges for developers of mobile groupware. **Workshop Mobile Ad Hoc Collaboration**, CHI, 2002, Minneapolis, Minnesota, USA.

SAITO, Y.; SHAPIRO, M. Optimistic replication. **ACM Comput. Surv.**, ACM Press, New York, NY, USA, v. 37, n. 1, p. 42–81, 2005. ISSN 0360-0300.

SHEN, H.; KUMAR, M.; DAS, S. K.; WANG, Z. Energy-efficient data caching and prefetching for mobile devices based on utility. Mob. Netw. Appl., **Kluwer Academic Publishers**, Hingham, MA, USA, v. 10, n. 4, p. 475–486, 2005. ISSN 1383-469X.

SHIH, E.; BAHL, P.; SINCLAIR, M. J. Wake on wireless: an event driven energy saving strategy for battery operated devices. **ACM Press**, New York, NY, USA, p. 160–171, 2002.

VIREDAZ, M. A.; BRAKMO, L. S.; HAMBURGEN, W. R. Energy management on handheld devices. Queue, **ACM Press**, New York, NY, USA, v. 1, n. 7, p. 44–52, 2003. ISSN 1542-7730.

WANG, Z.; DAS, S. K.; CHE, H.; KUMAR, M. A scalable asynchronous cache consistency scheme (saccs) for mobile environments. **In: IEEE transactions on parallel and distributed systems.** IEEE Computer Society, 2004.

YUEN, J. C.-H.; CHAN, E.; LAM, K.-Y.; LEUNG, H. W. Cache invalidation scheme for mobile computing systems with real-time data. SIGMOD Rec., **ACM Press**, v. 29, n. 4, p. 34–39, 2000. ISSN 0163-5808.

A Simulation study for evaluating the communication delay and the energy consumption in a mobile cooperative environment

Carla Diacui Medeiros Berkenbrock[1], .diacui@ita.br

Celso Massaki Hirata[1], hirata@ita.br

[1] Divisão de Ciência da Computação, Instituto Tecnológico de Aeronáutica - ITA

São José dos Campos, SP, Brasil

*Received: October, 2007 / Accepted: December, 2007

ABSTRACT

The communication between mobile devices and fixed servers is not symmetrical. Mobile devices take longer and consume more energy when they send than when they receive messages. Collaborative applications make intensive use of communication in order to keep the consistency. A way to reduce not only the access time but also the energy consumption is caching the frequently accessed data in mobile clients. However, the task to determine whether the copy at the client's cache is consistent with the copy of the server can be an expensive. In this paper we analyze the access time and energy consumption through a new mechanism to deal with cache coherency in a mobile cooperative environment. We conduct our studies based on a simulation model.

Keywords: Mobile groupware. Power consumption. Cache coherence.