Disciplinarum Scientia. Série: Naturais e Tecnológicas, Santa Maria, v. 17, n. 2, p. 205-216, 2016.
Recebido em: 30.05.2016. Aprovado em: 22.07.2016.

ISSN 2176-462X

DESCOBERTA DE SERVIÇOS PARA DISPOSITIVOS MÓVEIS EM AMBIENTES PERVASIVOS[1]

SERVICE DISCOVERY ON MOBILE DEVICES IN PERVASIVE ENVIRONMENTS

Thomás Fernandes Matos[2], Guilherme Chagas Kurtz[3] e Ricardo Giuliani Martini[4]

RESUMO

A Computação Pervasiva é um dos paradigmas da computação, que permite aos usuários o acesso não mais em apenas um dispositivo por vez, mas sim em diversos, que se fazem presentes aos mais variados lugares frequentados pelo usuário, tornando-se parte do seu cotidiano. Neste trabalho, é descrita uma aplicação para dispositivos móveis que visa facilitar a descoberta de serviços aos usuários. Por meio desta pesquisa, foi desenvolvida uma aplicação *Android*, utilizando a IDE (*Integrated Development Environment*) *Eclipse* com a configuração *Adt (Android Development Tools) Bundle*, a linguagem de programação Java e a biblioteca *clink* do *CyberGarage,* capaz de rastrear os dispositivos na rede e os serviços ofertados pelos mesmos, possibilitando aos usuários a utilização dos serviços implementados nesta aplicação. Nesse contexto, conclui-se que o desenvolvimento do presente trabalho beneficia a área da Computação Pervasiva, pois foi possível documentar e exemplificar o desenvolvimento de um *software* capaz de efetuar uma varredura na rede e interconectar os dispositivos existentes na mesma, e assim, possibilitar a utilização dos serviços ofertados, através do protocolo de comunicação UPnP (*Universal Plug And Play*), e disponibilizá-los aos usuários.

Palavras-chave: Android; aplicação; Computação Pervasiva; smartphone.

ABSTRACT

Pervasive computing is one of the paradigms of computing that allows users to connect not only one but several devices at the same time, since they are available in most of the places frequented by the users and so have become part of their daily lives. This paper describes an application for mobile devices that aims to facilitate the discovery of services to users. Through this research, an Android application was developed by using the Eclipse IDE (Integrated Development Environment), the ADT (Android Development Tools) Bundle, the Java programming language and the CyberGarage's clink library, which tracks the devices on the network as well as the services they provide, thus allowing users to use the services implemented in this application. Therefore, this study has contributed to Pervasive Computing since it was possible to document and illustrate the development of a piece of software that performs a full scan and interconnects existing devices on the network, thus enabling the use of the services provided through UPnP (Universal Plug and Play) communication protocol and making them available to all users.

Keywords: *Android application; Pervasive Computing; smartphone.*

[1] Trabalho de Iniciação Científica.

[2] Acadêmicos do Curso de Ciência da Computação - Centro Universitário Franciscano. E-mail: thomasmatos@gmail.com

[3] Docente do Curso de Ciência da Computação - Centro Universitário Franciscano. E-mail: guilhermechagaskurtz@gmail.com

[4] Orientador - Centro Algoritmi, Universidade do Minho, Braga, Portugal. E-mail: rgm@algoritmi.uminho.pt

INTRODUÇÃO

A Computação Pervasiva idealizada por Mark Weiser em 1991 e referida como o paradigma do século XXI, ao longo do tempo, ganhou espaço na vida das pessoas (WEISER, 1991). Essa é uma Computação que está em tudo o que se possa imaginar, está presente nos lares, nos automóveis, nos hospitais, etc.

Com o número de computadores no mundo sendo maior que o número de pessoas a ideia da Computação Pervasiva se fortifica ainda mais, juntamente com o avanço dos dispositivos móveis, pois ao invés de uma pessoa possuir somente um computador, ela possui diversos aparelhos computadorizados, móveis ou não, nos ambientes onde trabalha, mora e frequenta, utilizando-os para seu auxílio nas mais diversas atividades da sua rotina. Tudo isto, por meio dos dispositivos que geram um maior conforto e auxiliam no desempenho das atividades.

Neste âmbito da computação, em qualquer dispositivo, em todos os lugares e a todo o momento, torna-se mais frequente o desenvolvimento de sistemas para a integração entre diferentes tipos de dispositivos e aplicações, para acessar serviços disponíveis pelos mesmos. Os dispositivos móveis têm sido o grande alvo destes sistemas, pois a maioria das pessoas possuem *smartphones* e procuram cada vez mais funcionalidades para lhes auxiliarem no cotidiano.

A justificativa para o desenvolvimento deste trabalho é facilitar o uso de serviços entre usuários e dispositivos, para que as pessoas utilizem estes serviços oferecidos de forma cômoda, simplificando seu dia a dia nos ambientes onde costumam frequentar.

Neste trabalho, teve-se como proposta e objetivo geral, respectivamente, desenvolver uma aplicação para dispositivos móveis que permite aos usuários utilizarem serviços disponíveis na rede, por meio do protocolo de comunicação UPnP, podendo assim acessá-los e utilizá-los e efetuar um estudo sobre a Computação Pervasiva e suas características, bem como o processo de comunicação entre vários dispositivos distintos.

O restante deste artigo está estruturado da seguinte forma: na seção Computação Pervasiva, encontra-se a introdução, o referencial teórico, no qual é explanado sobre a Computação Pervasiva, abordando suas características e a sensibilidade de contexto, também é discernido sobre a descoberta de serviços em ambientes pervasivos, os protocolos de descoberta de serviços e os trabalhos correlatos. Já na Seção Material e Métodos, é explanado sobre a metodologia utilizada e é efetuado um estudo de caso: Controle de Mídia para validar o estudo realizado, de modo que se encontra todo o desenvolvimento dos processos e a implementação deste trabalho. Por último, a seção Conclusão, escrita a partir do estudo efetuado e do desenvolvimento deste artigo.

COMPUTAÇÃO PERVASIVA

O conceito de Computação Pervasiva surgiu com o cientista Mark Weiser em 1988, e que anos depois foi comentada em sua obra "O computador do século 21", em 1991. Para Weiser (1991), a Computação Pervasiva estará em tudo o que nos rodeia e, não somente nos computadores, ela estará tão integrada na nossa sociedade que se tornará mais transparente que a própria escrita. A Computação Pervasiva se define na terceira onda da computação, quando pessoas possuirão diversos computadores, visíveis ou não e, utilizáveis em diversos ambientes. As tecnologias mais profundas são aquelas que desaparecem, misturando-se com os objetos do cotidiano, de tal forma que se tornam praticamente invisíveis (WEISER, 1991).

Na visão de Saha e Mukherjee (2003), a Computação Pervasiva é um novo paradigma computacional, que provê a qualquer pessoa o acesso ao seu ambiente computacional, de qualquer lugar, a qualquer momento e de qualquer dispositivo. O ambiente pervasivo deve ser capaz de detectar todo e qualquer tipo de dispositivos que venham fazer parte dele, para isso, deve conter diversos tipos de sensores e oferecer uma gama de serviços computacionais (CHEN et al., 2003).

A partir do estudo efetuado por Saha e Mukherjee (2003), entende-se que as principais características da Computação Pervasiva são escalabilidade, heterogeneidade, integração, invisibilidade, mobilidade e a mais importante, sensibilidade de contexto, pois sem ela não seria possível concretizar a ideia de Computação Pervasiva.

SENSIBILIDADE DE CONTEXTO

O contexto é definido por Yamin et al. (2003) como toda a informação obtida a partir da infraestrutura computacional que deve ser relevante para a aplicação e, interessante ao usuário e, caso haja alguma alteração em seu estado, é disparado um alerta em forma de processo para que seja efetuada uma adaptação nesta aplicação.

Dey (2001) descreve contexto como toda e qualquer informação que seja relevante e possa ser usada para caracterizar a situação de uma entidade. Entende-se por entidade uma pessoa ou um lugar, sendo um objeto relevante para a interação entre aplicações, que devem ser sensíveis ao contexto e aos usuários.

Outro fator primordial, para a Computação Pervasiva alcançar suas propostas, é a descoberta de serviços, que por sua vez conta com os diversos protocolos de comunicação existentes que tratam da comunicação entre dispositivos.

DESCOBERTA DE SERVIÇOS EM AMBIENTES PERVASIVOS

Em ambientes inteligentes, os dispositivos, juntamente com os usuários, formam associações de cooperação dinâmica, conforme seu deslocamento pelo mesmo. Para que isto seja alcançado, cada dispositivo deve interagir com os integrantes deste ambiente por meio da oferta e procura de serviços a serem utilizados. É necessário que estes componentes possuam a capacidade de descobrir, utilizar e oferecer estes serviços aos demais dispositivos e aplicações que estão situados no mesmo ambiente pervasivo (CHEN, 2004).

Na visão de Murphy, Picco e Roman (2001), descoberta de serviços funciona como um serviço de diretórios, no qual são armazenadas as ofertas de serviços que poderão ser utilizados pelos usuários. Entre as principais funções estão a oferta, a descoberta, a escolha e a utilização de serviços que são disponibilizados através de diversos dispositivos. Para efetuar a comunicação entre aplicações e usuários, são utilizados protocolos de descoberta de serviços como, *Jini*, *Salutation* e o *Universal Plug and Play* (UPnP), entre outros.

PROTOCOLOS DE DESCOBERTA DE SERVIÇOS

Estes protocolos fornecem mecanismos necessários para descobrir, de forma dinâmica, os serviços que estão sendo disponibilizados em uma rede, fornecendo as devidas informações para efetuar a procura por serviços específicos, escolher o mais adequado em relação às características do usuário e utilizá-los. Normalmente esta descoberta envolve um provedor, um cliente e um servidor de diretórios ou busca. Os registros e a busca por serviços são componentes importantes de todos os protocolos (AVANCHA et al., 2002).

Baseado no estudo realizado, sobre a pesquisa de Lima (2007), o protocolo a ser utilizado neste trabalho é o UPnP, pois grande parte dos dispositivos existentes tem suporte para o mesmo, tornando assim mais fácil a comunicação com um maior número de dispositivos.

TRABALHOS CORRELACIONADOS

Os trabalhos de Leithardt e Passuelo (2009) e Cardoso et al. (2007) têm como objetivo o desenvolvimento de aplicações que sejam capazes de captar o contexto e armazená-lo para consultas posteriores. Os autores também também efetuam a descoberta e armazenamento de serviços e dispo-

sitivos na rede local, ofertando aos usuários e as aplicações, listas de serviços com base nos contextos de ambos que foram coletados.

Os respectivos trabalhos citados, em comparação ao estudo desenvolvido, têm em comum o fato de buscarem por serviços, dispositivos e aplicações, na rede, para assim ofertar estes serviços aos usuários.

A principal diferença entre a pesquisa e os trabalhos citados é que estes levam em consideração o contexto, para exibir os serviços encontrados aos dispositivos e aplicações que se conectam na rede local, sendo mais voltados a aplicações pervasivas do que a usuários. A importância do presente trabalho em relação aos outros é o fato de exibir de forma simples aos usuários os serviços ofertados na rede, porém não faz isso levando em consideração o contexto, como os trabalhos citados.

MATERIAL E MÉTODOS

Neste trabalho, foi utilizada a metodologia ágil FDD (*Feature Driven Development*) de desenvolvimento, pois é uma metodologia de modelo ágil, iterativo, incremental e que trabalha com funcionalidades, o que torna seu desenvolvimento mais compreensível e eficaz.

A aplicação proposta neste trabalho visou buscar dispositivos na rede, para que o usuário pudesse utilizar os serviços e as ações ofertados pelos mesmos.

As ferramentas utilizadas no desenvolvimento do presente aplicativo foram a IDE (*Integrated Development Environment*) *Eclipse* com a configuração *Adt (Android Development Tools) Bundle*, a linguagem de programação Java, o *SDK API 7: Android 2.1 (Eclair)* e a biblioteca *clink* do *CyberGarage*.

Primeiramente, entende-se que, na figura 1, a aplicação inicializa o Ponto de Controle, representado através de um objeto da classe *ControlPoint*, o qual cuida de toda a parte de descoberta de dispositivos e conexão. Além disso, o sistema controla também a entrada (método *deviceAdded()*) e a saída (método *deviceRemoved()*) dos dispositivos a partir destes métodos que escutam a rede, através de um "ouvinte". Além destes, o método *getFriendlyName(),* efetua a busca do nome do dispositivo selecionado pelo usuário através da listagem dos dispositivos descobertos encontrados na figura 2. No exemplo a seguir, o dispositivo selecionado foi o *G4m3R(G4m3R-PC:Windows Media Player)*, que é um computador com sistema operacional *Windows 7*.

Figura 1 - Código para controlar a entrada e saída de dispositivos da rede.

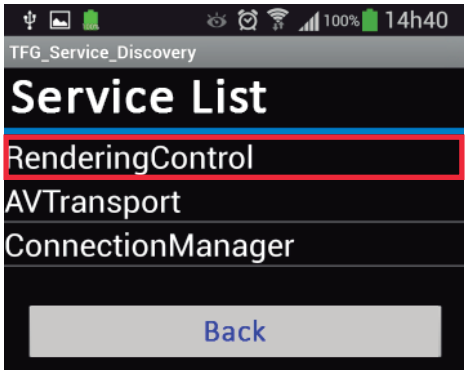
```
@Override
public void deviceAdded(Device dev) {
//Este método recebe por parâmetro o dispositivo que entrou na rede, coleta
  o nome do dispositivo e então o adiciona em uma lista de dispositivos e
  atualiza a tela Device List
    final String strDevice = dev.getFriendlyName();
    Handler refresh = new Handler(getMain
    refresh.post(new Runnable() {
        public void run()
        {
            if(adapter.getPosition(strDevice) == -1)
                adapter.add(strDevice);
        }
    });
}

@Override
public void deviceRemoved(Device dev) {
//Este método recebe por parâmetro o dispositivo que saiu na rede, coleta
  o nome do dispositivo e então o remove da lista de dispositivos e atualiza
  a tela Device List
    final String strDevice = dev.getFriendlyName();
    Handler refresh = new Handler(getMainLooper());
    refresh.post(new Runnable() {
        public void run()
        {
            adapter.remove(strDevice);
        }
    });
```

Figura 2 - Tela de listagem de dispositivos.

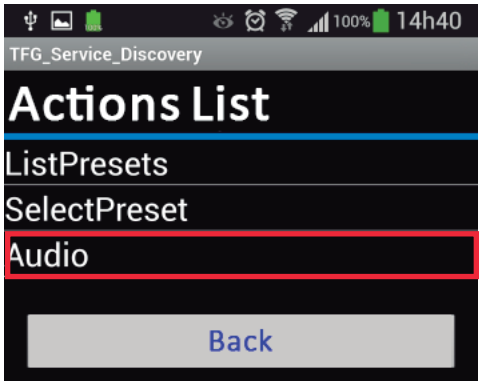
Na figura 2, encontra-se a lista de dispositivos (*Device List*), sendo selecionado o dispositivo desejado, a aplicação é direcionada para outra tela, quando é feita a listagem dos serviços (*Service List*), realizada pelo método *getServiceList()*. Tem-se na figura 3 a listagem geral dos serviços encontrados para o dispositivo selecionado. Nesta tela, o usuário escolhe o serviço que deseja utilizar. Neste caso, o serviço selecionado foi o *RenderingControl* que é referente a toda a parte de áudio.

Figura 3 - Tela de listagem de serviços.



Com o serviço selecionado, o usuário tem a opção de escolher a ação que deseja utilizar a partir da listagem de ações (*Actions List*) realizada pelo método *getActionList()*. Localiza-se a ação selecionada de *Audio* (Figura 4), na qual foram englobadas todas as ações referentes a áudio que foram implementadas.

Figura 4 - Tela de listagem de ações.



A seguir será descrito um estudo de caso no qual o usuário, a partir das telas detalhadas anteriormente, pôde utilizar os serviços de mídia de um dispositivo.

ESTUDO DE CASO: CONTROLE DE MÍDIA

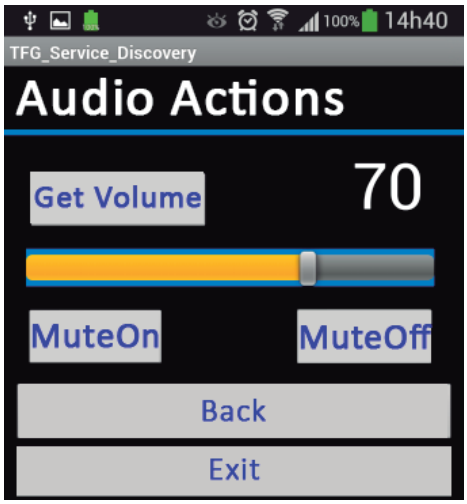
A fim de validar o estudo realizado, foi desenvolvido um aplicativo *Android* para controle de mídia como estudo de caso. Para elaborar os testes, o aplicativo proposto neste trabalho se conecta a um computador com o sistema operacional *Windows 7*, através do protocolo *UPnP*, o qual consegue captar os serviços oferecidos pelo *Windows Media Player* e utilizá-los, conseguindo aumentar e diminuir o volume, colocar no mudo, retirar do mudo e captar o volume. Além destes serviços, o aplicativo também consegue reproduzir, pausar, parar e ir para a última faixa.

O cenário a ser demonstrado, neste estudo de caso, visa utilizar dois serviços. O primeiro nomeado *RenderingControl*, o qual possui ações para manipulação de áudio. Para este estudo de

caso, as ações utilizadas para este primeiro serviço são captar o volume, colocar no volume desejado e colocar/retirar do mudo. Essas ações foram reunidas em uma única ação denominada *Audio*. Já o segundo serviço, nomeado *AVTransport*, possui ações de controle de mídia. As ações utilizadas para este segundo serviço são reproduzir, pausar, parar e ir para a última faixa. Todas estas ações foram reunidas em uma única ação nomeada *Player*.

Encontra-se, na figura 5, a tela de manipulação de áudio, na qual o usuário é direcionado para a tela do serviço *RenderingControl*. Nesta, o usuário pode aumentar e diminuir o volume, captar o volume, retirar do mudo e colocar no mudo.

Figura 5 - Tela de manipulação de áudio.



Para exemplificar, serão descritas duas das ações possíveis, captar volume (botão *Get volume*) e colocar no mudo (botão *MuteOn*) que podem ser vistos na figura 5. Tem-se, nas figuras 6 e 7, as implementações destas ações.

Figura 6 - Código que efetua a ação de pegar o volume.

```
public int getVolume() {
//Método que capta o volume atual do dispositivo e retorna este valor
  int vol = 0;
  try {
      Action action = service.getAction("GetVolume");
      action.setArgumentValue("InstanceID", "0");
      action.setArgumentValue("Channel", "Master");
      action.postControlAction();

      vol = (action.getArgument("CurrentVolume").getIntegerValue());
  } catch (Exception e) {
  }
  return vol;
}
```

No método *getVolume()* é acionada a ação *GetVolume*, que capta o volume e então, seus argumentos são atualizados para que seja possível captar o volume. O volume é captado pelo método *action.getArgument ("CurrentVolume").getIntegerValue()*.

Figura 7 - Código que efetua a ação de colocar no mudo.

```
public void muteOn( ) {
//Este método desabilita o volume do dispositivo através do boleano falso
    try {
        Action action = service.getAction("SetMute");
        action.setArgumentValue("InstanceID", "0");
        action.setArgumentValue("Channel", "Master");
        action.setArgumentValue("DesiredMute", false ? "0" : "1");
        action.postControlAction();
    } catch (Exception e) {
    }
}
```

Neste método *muteOn()*, é solicitada a ação de *SetMute*, que coloca o volume no mudo e então são definidos os seus argumentos para que isto seja possível, a partir do comando *action.setArgumentValue ("DesiredMute", false ? "0" : "1")*.

Já a partir da seleção da ação *Player*, o usuário é direcionado para outra tela, visível na figura 8, agora do serviço *AVTransport*. Nesta tela, o usuário pode executar as ações de reproduzir, pausar, parar e ir para a última faixa.

Figura 8 - Tela de controle de mídia.

Encontra-se, na figura 8, a ilustração a fim de exemplificar o uso do serviço *AVTransport*, e o detalhamento das ações de reproduzir (*play*) e pausar (*pause*). Nas figuras 9 e 10, tem-se as implementações destas ações.

Figura 9 - Código que efetua a ação de reproduzir (*play*).

```
public void play( ) {
 // Método que ativa a ação de  serviço de reprodução do dispositivo
    try {
        Action action = service.getAction("Play");
        action.setArgumentValue("InstanceID", "0");
        action.setArgumentValue("Speed", "1");

    } catch (Exception e) {
    }
}
```

O método *play()*, efetua a reprodução, a partir da ação de reproduzir que executa a reprodução conforme os seus argumentos, podendo definir a velocidade de reprodução em *actions.setArgument Value("Speed", "1")*.

Figura 10 - Código que efetua a ação de pausar ("*pause*").

```
public void pause( ){
 //Este método ativa a ação de pausar a reprodução do dispositivo
    try {
        Action action = service.getAction("Pause");
        action.setArgumentValue("InstanceID", "0");

    } catch (Exception e) {

    }
}
```

No método *pause()*, é simplesmente feita a chamada da ação pausa, e então é efetuada a ação de pausar.

Para todas as ações, é necessário capturá-las através do método *getAction("nome da ação")* e definir seus argumentos, os quais são específicos de cada ação.

CONCLUSÃO

Com base nos dados coletados e no estudo realizado na parte inicial deste trabalho, constata-se o grande avanço da Computação Pervasiva e o fundamental processo de descoberta de serviços. A utilização de protocolos de comunicação torna possível a comunicação entre diversos tipos de dispositivos, possibilitando a utilização deste paradigma em diversas áreas.

No desenvolvimento desta aplicação foram encontradas dificuldades em relação a documentação da biblioteca *clink*, por ser muito precária em relação a algumas especificações, o que tornou mais delicado o desenvolvimento do trabalho.

Esta pesquisa constitui-se numa contribuição importante para a Computação Pervasiva, pois é um estudo concreto e documentado realizado nesta área, com o desenvolvimento de um sistema que torna acessível ao usuário os serviços que estão sendo disponibilizados na rede e fornece apoio para que o mesmo possa utilizá-los de forma simples e eficaz. Além disso, beneficia futuros trabalhos na área da Computação Pervasiva, pois deixa uma base documentada sobre como efetuar a varredura e a intercomunicação entre dispositivos conectados a uma mesma rede e a utilização de alguns serviços ofertados pelos mesmos, isto tudo através da biblioteca *clink*.

REFERÊNCIAS

AVANCHA, S. E.; JOSHI, A.; FININ, T. Enhanced service discovery in Bluetooth. **Computer Communications**, University of Maryland Baltimore Country, Baltimore, MD, USA, v. 35, n. 6, p. 96-99, 2002.

CARDOSO, L.; SZTAJNBERG, A.; LOQUES, O. Proposta de uma Infra-estrutura de Suporte para Serviços de Contexto e Descoberta de Recursos. XXVII Congresso da SBC, Instituto de Computação, Universidade Federal Fluminense, Rio de Janeiro, RJ, 2007. **Anais...** Rio de Janeiro, 2007.

CHEN, H.; FININ, T.; JOSHI, A. An ontology for context-aware pervasive computing environments. **The Knowledge Engineering Review,** Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD, USA, v. 18, n. 3, p. 197-207, 2003.

CHEN, H. **An Intelligent Broker Architecture for Pervasive Context-Aware Systems**. 2004. 129f. Dissertation (PhD in Philosophy) - Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD, USA, 2004.

DEY, A. K. Understanding and Using Context. **Personal and Ubiquitous Computing,** College of Computing & GVU Center, Georgia Institute of Technology, Atlanta, GA, USA, v. 5, n. 1, p. 4-7, 2001.

LEITHARDT, V. R. Q.; PASSUELO, F. H. **Modelo Gerenciador de Serviços para Plataformas Pervasiva Sensíveis ao Contexto**. Fórum de Pós-graduação, Pontifícia Universidade Católica do Rio Grande do Sul, Caxias do Sul, RS, 2009. p. 123-124.

LIMA, S. L. **Um protocolo para descoberta e seleção de recursos grades móveis Ad Hoc**. 214f. Tese (Doutorado em Informática) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2007.

MURPHY, A. E.; PICCO, G.; ROMAN, G. C. Lime: a middleware for physical and logical mobility. **Anais**. 21st International Conference in Distributed Computing Systems. **Anais...** Proceedings, Windemere Hotel and Conference Center, Mesa, AZ, USA, 2001, p. 524-533.

SAHA, D.; MUKHERJEE, A. Pervasive Computing: A Paradigm for the 21st Century. **Computer,** Washington, DC, EUA, v. 36, n. 3, p. 25-31, 2003.

WEISER, M. The Computer for the 21st Century. **Scientific American,** New York, USA, p. 94-104, 1991.

YAMIN, A. et al. Towards Merging Context-Aware, Mobile and Grid Computing. **International Journal of High Performance Computing Applications,** Londres, v. 17, n. 2, p. 191-203, 2003.