HLS: Modelo para Desenvolvimento de Aplicações Sensíveis à Localização*

Cícero Raupp Rolim, Nélson Buis Sonntag, Jorge Luis Victoria Barbosa
Universidade do Vale do Rio dos Sinos (UNISINOS),
São Leopoldo – RS, Brasil, 93022-000
cicero.rolim@gmail.com, {nelsons, jbarbosa}@unisinos.br

Resumo

O crescimento do poder computacional dos dispositivos portáteis como PDAs, handhelds *e* notebooks *é uma realidade na última década. Paralelamente, as redes sem fio (por exemplo, Wi-Fi e bluetooth), tiveram um crescimento vinculado a estes equipamentos, facilitando a comunicação e troca de informações entre os mesmos. Nesse escopo surgiu a computação ubíqua. No escopo da computação ubíqua, as aplicações devem ser sensíveis à rede, recursos, localização física e contexto, ou seja, podem ter seu comportamento alterado durante sua execução, devido à mobilidade constante dos dispositivos móveis. Este trabalho apresenta o HLS, um modelo para desenvolvimento de aplicações sensíveis à localização utilizando o ambiente do Holoparadigma. O HLS é um modelo que inclui um servidor de localização de dispositivos móveis, um módulo que é integrado ao ambiente de execução do Holoparadigma para alteração do estado de execução das aplicações, e uma ferramenta para administração de ambientes com servidor de localização baseado em árvores de contexto.*

1. Introducão

Atualmente, os estudos sobre mobilidade em sistemas distribuídos vêm sendo impulsionados pela proliferação de dispositivos eletrônicos portáteis (por exemplo, telefones celulares, *handhelds*, *tablet PCs* e *notebooks*) e pela exploração de novas tecnologias de interconexão baseadas em comunicação sem fio (tais como, *WiFi*, *Bluetooth*, *WiMAX* e GSM). Este novo paradigma distribuído e móvel é denominado Computação Móvel [17]. Além disso, a mobilidade aliada à difusão da comunicação sem fio (*wireless*) permitiu aos serviços computacionais serem conscientes do contexto [2]. O acréscimo de pesquisas relacionadas com a Adaptação [1] trouxe a possibilidade de suporte computacional contínuo, a qualquer momento e em qualquer lugar (Computação Ubíqua [18]). Por sua vez, os sistemas de localização [9,10] estão viabilizando o uso preciso desse tipo de computação de acordo com a posição física do usuário, criando assim, serviços baseados em localização [12]. A busca pelo suporte à localização física em aplicações ubíquas é um desafio que motiva diversas plataformas tais como o Aura [6], o Gaia [16], o *one.world* [8] e o PLACE [13].

Neste contexto foi proposto o Holoparadigma (de forma abreviada, Holo [4,5]), um novo modelo para o desenvolvimento de sistemas móveis e ubíquos. O Holo e seu ambiente de execução estão sendo desenvolvidos no MobiLab/Unisinos [11]. Este artigo apresenta o HLS (*Holo Location System*), um modelo que suporta o gerenciamento de informações de localização em ambientes ubíquos desenvolvidos com o Holoparadigma.

Este trabalho está organizado da seguinte forma. A seção 2 aborda de forma resumida o Holoparadigma. A terceira seção apresenta o modelo proposto. Por sua vez, a seção quatro descreve um protótipo e discute os resultados obtidos em testes de precisão de informações de localização e durabilidade da carga de baterias em dispositivos móveis. Essa seção discute ainda uma aplicação do sistema de localização do HLS. A aplicação consiste em um sistema de educação ubíqua (LOCAL [3]) criado no MobiLab. As duas últimas seções apresentam os trabalhos relacionados e as considerações finais.

2. Holoparadigma

O Holoparadigma [4,5] (abreviado como Holo) é um modelo orientado ao desenvolvimento de sistemas móveis e ubíquos. Holo é baseado em uma abstração de contexto chamada ente, usada para suportar a mobilidade. Um ente elementar é organizado em três partes: comportamento, interface e história. O

* Este trabalho recebeu apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq - Brasil (Edital 15/2007 - Universal)

comportamento define o conjunto de ações que o ente é capaz de executar. Dentre essas, as que podem ser acessadas externamente são descritas na interface. A história consiste em um espaço de tuplas que fica encapsulado no ente. Um ente composto possui a mesma organização, suportando ainda a existência de outros entes na sua composição (entes componentes). Nesse caso, a história é compartilhada pelos entes componentes.

No Holoparadigma existem dois tipos da mobilidade: (i) mobilidade lógica e (ii) mobilidade de código. A mobilidade lógica relaciona-se com o deslocamento em nível de modelagem, sem considerações sobre a plataforma de execução. A mobilidade de código relaciona-se com o deslocamento entre nodos de uma arquitetura distribuída. Mobilidade lógica e mobilidade de código são independentes.

A execução de um programa Holo cria uma estrutura hierárquica de entes, denominada: Árvore de Entes (*HoloTree*). A árvore suporta o encapsulamento de contextos em níveis de composição, conforme proposto pelo Holoparadigma. A HoloTree suporta ainda o aspecto dinâmico da política de grupos, mudando continuamente durante a execução de um programa. A Figura 1a mostra um exemplo da organização hierárquica da *HoloTree* e a Figura 1b uma possível mobilidade lógica de um ente elementar.

O ambiente de execução do Holo é baseado em uma máquina virtual chamada HoloVM [7] e no HNS (*Holo Naming System*), um sistema que integra todas as máquinas virtuais que estão executando código Holo. Antes da proposta apresentada nesse artigo, o HNS não suportava informações de localização de equipamentos. Ao integrar um servidor de localização com o HNS foi possível disponibilizar informações de posicionamento de equipamentos para o ambiente distribuído Holo, e assim utilizar este recurso como forma de alterar o estado da aplicação Holo em tempo de execução, ou seja, fornecer o suporte a aplicações sensíveis à localização no Holoparadigma.

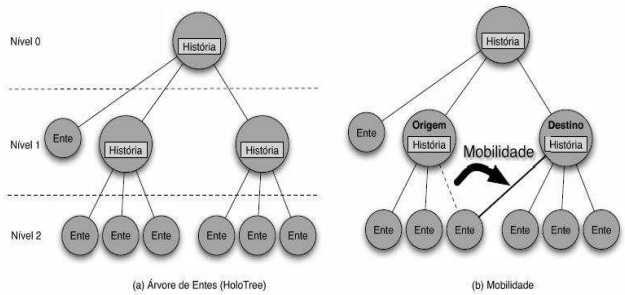


Figura 1. Árvore de Contextos (HoloTree).

3. Modelo HLS

Essa seção apresenta o modelo *Holo Locating System*, de forma abreviada HLS. Este modelo integra o suporte a aplicações sensíveis à localização no ambiente Holo.

3.1. Arquitetura

O HLS é um modelo que permite a criação de aplicações sensíveis à localização. O modelo possui três componentes: (1) suporte à localização de dispositivos (Servidor de Localização); (2) suporte automático à atualização da aplicação no Holo (*Context Changer)* e (3) administração de localização de equipamentos usando uma árvore de contextos (HTV - *Holo Tree View)*.

Na Figura 2 é apresentada a estrutura de requisições do modelo, onde é possível observar a comunicação dos dispositivos móveis com o Servidor de Localização. O servidor recebe dos equipamentos móveis a informação de potência de sinal das antenas de pontos de acessos conhecidos. O servidor também recebe as informações dos computadores *desktop*, através do protocolo SNMP. Usando essa informação, o HLS calcula a localização de um dispositivo em um determinado momento. O *Context Changer* é integrado ao HNS (suporte à HoloTree). Quando ocorrem mudanças no posicionamento dos dispositivos, o HNS recebe uma mensagem do Servidor de Localização, informando qual o novo contexto do dispositivo. O *Context Changer* também pode realizar consultas ao Servidor de Localização, para economizar recursos de rede e processador. Finalmente a *Holo Tree View* (HTV) recebe informações no formato de árvores de contextos, as quais podem ser oriundas do Servidor de Localização e do HNS.

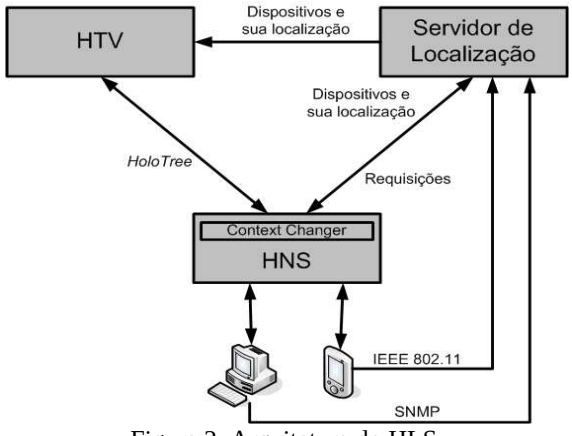


Figura 2. Arquitetura do HLS.

3.2. Servidor de Localização

O Servidor de Localização, denominado SELIC, permite que aplicações acessem informações sobre a localização de dispositivos. A sua tarefa é integrar diversos sistemas de localização (GPS, RFID, IEEE 802.11, etc) e disponibilizar as informações através de um protocolo padrão. Cada sistema de localização é um módulo no SELIC. As informações de localização dos dispositivos são disponibilizadas através de uma interface de alto nível.

A API do SELIC foi concretizada através de uma interface de *Web Services*. Esta escolha foi feita pela facilidade de integração com outras aplicações. *Web services* são uma solução utilizada na integração de sistemas e na comunicação entre diferentes aplicações. As bases para a construção de um *Web Service* são os padrões XML e SOAP. Os dados são transferidos no formato XML, encapsulados pelo protocolo SOAP.

A precisão das informações de posicionamento é uma questão importante, e para isso, o SELIC tem suporte a múltiplos sistemas de localização, o que torna possível comparar as informações e escolher a de maior coerência. A Figura 3 apresenta o SELIC com suporte aos sistemas de localização GPS, IEEE 802.11 e RFID.

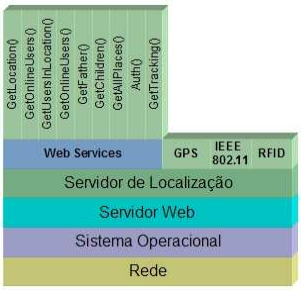


Figura 3. Arquitetura do SELIC.

O SELIC tem suporte a *tracking* de dispositivos. Quando o *tracking* estiver habilitado para um equipamento, as informações de posicionamento são registradas em uma base de dados para futuras consultas de aplicações. Um exemplo de utilização de *tracking* é a auditoria para descobrir o perfil de deslocamento de um equipamento. O tempo de *tracking* de um dispositivo é configurada em informações armazenadas em um arquivo XML. Por exemplo, se o *tracking* do dispositivo "iPaq 3" é quatro meses, as informações acima deste tempo serão descartadas. Com isso é possível manter a base de dados com informações relevantes, melhorando o desempenho em consultas e economizando recursos de armazenamento e memória.

A interface do SELIC possui os seguintes métodos:

- *Auth*: Método para autenticação no sistema;
- *GetAllPlaces*: Método para retornar a lista de locais que podem usufruir do sistema de localização;
- *GetChildren*: Método para retornar os contextos que são filhos do local passado como parâmetro;
- *GetFather* : Método para receber o local que está no topo da árvore de localização;
- *GetLocationByUserId* : Retorna a localização atual pelo *User Id* do usuário;
- *GetLocationByUserName*: Retorna a localização atual pelo nome do usuário;
- *GetOnlineUsers*: Informa os usuários que estão *online* no sistema;
- *GetUsersInLocation* : Retorna uma lista com todos os usuários que estão no local passado como parâmetro;
- *GetTracking*: Método para receber o histórico de deslocamento de um dispositivo (*tracking*).

3.3. Context Changer

A primeira medida para construir uma aplicação sensível a localização no Holoparadigma é a construção da *HoloTree*. A *HoloTree* deve ser baseada na árvore de contexto dos ambientes do SELIC. Isso é necessário para que o programa Holo seja um mapeamento exato da situação do SELIC, mantendo a integridade referencial da aplicação com os locais do mundo físico. Os serviços suportados em cada ambiente devem ser mapeados como ações para o seu representante na estrutura, permitindo que sejam agrupados e organizados no mesmo contexto.

O *Context Changer*, de forma abreviada CC, é um módulo integrado ao HNS com a função de manter os entes da *HoloTree* que representem dispositivos móveis, sincronizados com a sua localização física. Os entes que representam equipamentos (*laptops, handhelds, PDAs ou tablet pcs)* alteram sua posição na árvore de acordo com as mudanças de contextos no mundo físico. Quando um dispositivo troca o seu local no ambiente físico, esse ente é realocado na *HoloTree*, permitindo que o mesmo acesse o seu pai para uso dos recursos disponíveis no contexto atual. Na Figura 4 é apresentado o funcionamento do *Context Changer*. Nela o *parser* processa as requisições que chegam ao HNS. Quando o *parser* identifica que uma mensagem é enviada pelo SELIC, ele realiza uma chamada para o módulo *Context Changer* passando como parâmetro a mensagem. A mensagem então é processada pelo CC,

que altera somente os entes que representam equipamentos no SELIC.

O CC é quem mantém sincronizado os entes que estão nas HoloVMs e que representem dispositivos físicos no mundo real. A base do mapeamento é um arquivo em XML que descreve quais são os nomes dos entes que representam equipamentos. A utilização do arquivo de mapeamento torna fácil a integração de novos dispositivos ao sistema, permitindo adicionar ou remover equipamentos conforme a necessidade da aplicação.

Quando o SELIC detecta que um dispositivo teve seu contexto alterado, ele repassa esta informação ao HNS. Esta informação é processada pelo *Context Changer* que é encarregado de pesquisar o ente e alterar o seu pai na *HoloTree*, gerando uma mobilidade na árvore. Depois de efetuar esta operação, se o novo pai possuir ações relevantes ao contexto, elas já poderão ser acessadas pelo ente filho que foi movido.

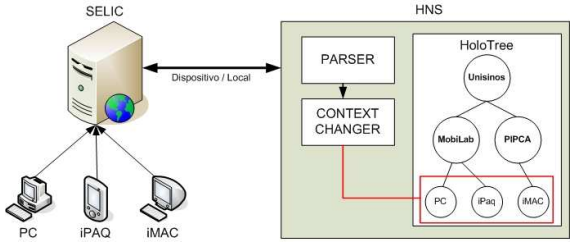


Figura 4. Funcionamento do *Context Changer*.

3.4. Holo Tree View

A existência de um servidor de localização traz novas funcionalidades que podem ser exploradas para a administração de usuários e dispositivos. Um fator que necessita especial atenção e pode ser explorado em ambientes ubíquos é a administração de recursos e equipamentos. Após verificar a necessidade de uma ferramenta para visualização da *HoloTree*, que está presente no HNS, e da árvore de ambientes, disponível no SELIC, foi desenvolvida a *Holo Tree View* (HTV). A HTV apresenta as informações baseadas em árvores de contextos de forma visual.

Com a HTV é possível acompanhar o deslocamento dos equipamentos ou entes. Com essa informação o administrador pode verificar equipamentos que estão em locais não apropriados, identificar locais que estão superlotados ou acompanhar o caminho que um ente dinâmico está percorrendo na HoloTree.

Outra funcionalidade que pode ser utilizada é o detalhamento do *tracking* de um dispositivo. Quando a HTV mostra o *tracking*, o percurso é apresentado com uma seqüência numérica. Essa seqüência também trás o tempo que o dispositivo permaneceu em cada local.

4. Protótipo e Aplicação

Um protótipo do servidor de localização (seção 3.2) encontra-se em funcionamento no MobiLab/Unisinos [11]. Os equipamentos de computação móvel usados na criação e avaliação do protótipo foram doados pela empresa HP Computadores. A seção 4.1 descreve o protótipo e discute resultados de testes relacionados com a precisão de informações de localização e com a durabilidade da carga de baterias de dispositivos móveis. Por sua vez, a seção 4.2 apresenta a aplicação do HLS na criação de um sistema de educação ubíqua, chamado LOCAL [3].

4.1. SELIC

O servidor de localização foi implementado com o ambiente de desenvolvimento *Microsoft Visual Studio 2005*, utilizando a linguagem de programação C#. Essa escolha foi determinada pela utilização do *Microsoft .NET Framework* e a possibilidade de portabilidade para outras plataformas, como por exemplo Linux utilizando o compilador e *framework Mono*. Outro fator importante foi à integração do *Microsoft Visual Studio 2005* com *Web Services*, possibilitando que o SELIC pudesse ser utilizado em diferentes aplicações.

4.1.1. Arquitetura do Protótipo. O protótipo está organizado em três partes:

- *Web Services*: Cada equipamento, sensor ou aplicação utiliza a *interface* de serviços baseada em *Web Services* para comunicação com o SELIC. Nos *Web Services* estão os métodos para autenticação no sistema, consulta de posicionamento de dispositivos e consulta na árvore de ambientes;
- Cliente: Módulo responsável pela autenticação no sistema e integração com os *Web Services*. Após o usuário instalar o cliente no seu dispositivo móvel, torna-se possível determinar sua localização física. Por exemplo, se o sistema de localização for o IEEE 802.11, é informada ao *Web Services* a potência de sinal das antenas de pontos de acesso conhecidos. Atualmente, o cliente pode ser executado nos ambientes *Windows* (*desktops* e Tablet PCs tc 1100) e *Windows Mobile* (iPaqs hx 4770). A Figura 5 mostra a tela de autenticação do sistema;
- *Engine*: Consiste na principal parte do sistema. O *Engine* é responsável pelos cálculos de posicionamento dos equipamentos. Cada equipamento reporta

informações dos sistemas de localização ao *Web Service* do SELIC. Essa requisição é colocada numa fila, que então é processada pelo *Engine*. Os sistemas são acoplados ao *Engine* através de uma classe abstrata.

Atualmente, o *Engine* suporta os seguintes sistemas de localização:

- IEEE 802.11: Cada ambiente possui um mapeamento com a potência de sinal de antenas de pontos de acessos conhecidos. Para cada metro quadrado do ambiente são colhidos *n* pontos. Quanto maior o número de pontos, maior a precisão do sistema. O mapeamento é armazenado em um banco de dados. Quando o *Engine* processa uma requisição IEEE 802.11, ele busca no banco de dados do mapeamento, procurando por pontos que se enquadram na potência de sinal. O ambiente que estiver com mais pontos consiste na atual posição do dispositivo. No *Engine* também foi implementado uma análise estatística, para contornar o problema da constante variação de potência de sinal. Essa análise considera os últimos dez ambientes onde o dispositivo foi encontrado e também os últimos cinco minutos. O tempo pode ser configurado pelo usuário.
- SNMP: O SNMP é utilizado no SELIC em equipamentos com pouca mobilidade (por exemplo, *desktops*). Cada equipamento é incluído numa tabela com o seu local atual, seu identificador de dispositivo, IP e senha. O *Engine* busca por todos dispositivos nesta tabela e realiza uma requisição SNMP para verificar se o dispositivo está *online*. Caso haja resposta, o dispositivo é registrado no *tracking* do sistema.

Figura 5. Tela de autenticação no SELIC (Cliente).

4.1.2. Avaliação da Precisão da Localização. Um dos fatores mais importantes de um servidor de localização é a sua precisão. A precisão determina a granulosidade das informações de localização suportadas nas aplicações (por exemplo, *room level*, *floor level*, *build level* ou *area level*). O protótipo foi criado para ter precisão *room level* utilizando como principal sistema de localização o IEEE 802.11. O sistema abrange nove salas no segundo andar do prédio onde está localizado o MobiLab [11]. A infra-estrutura de rede sem fio é composta de quatro antenas *wireless* Cisco Aironet distribuídas no andar. A Figura 6 mostra o cenário onde foram realizados os testes.

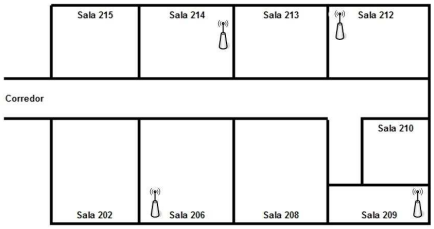


Figura 6. Cenário do teste do SELIC.

A Figura 7 apresenta o gráfico obtido no teste de precisão do SELIC. O equipamento utilizado foi um HP Compaq tc1100 Tablet PC, com processador Centrino de 1.8 MHz, 512 MB de memória RAM e placa *wireless* ORiNOCO Silver Pcmcia. Além disso, o teste considera quatro densidades de pontos para o mapeamento dos contextos. Um mapeamento consiste no número de pontos coletados por metro quadrado (veja o eixo X no gráfico) para cada antena do sistema. Essa coleta é armazenada no banco de dados do *Engine* (veja seção 4.1.1). Foram escolhidos três locais para amostra na sala 6B208. Cada local teve sua precisão avaliada cem vezes para cada mapeamento.

A Figura 7 mostra que o SELIC possui uma precisão que torna viável as aplicações *room level*. Uma densidade maior do que 2 pm^2 resulta em uma precisão acima de 91%. Além disso, conforme esperado, observa-se no gráfico um aumento na precisão das informações de localização, na medida em que aumenta a densidade de pontos coletadas no mapeamento.

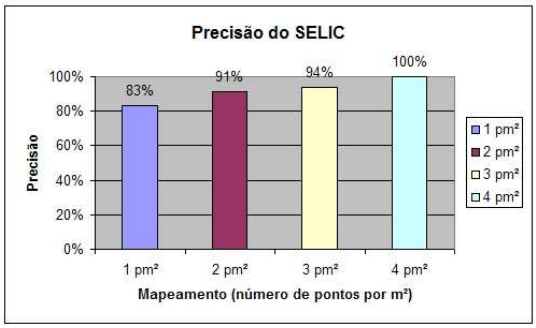


Figura 7. Gráfico de Precisão do SELIC.

4.1.3. Consumo de Energia. A utilidade de um dispositivo móvel está atribuída à capacidade de sua bateria, que por sua vez depende de sua durabilidade, onde está vinculada à utilização do acesso wireless e ao uso de determinados aplicativos.

O cliente do SELIC usa o sistema *WiFi* como base para a localização. A consulta de potências de pontos de acessos conhecidos e o envio dessa informação para o *Web Service*, ocasiona um consumo de bateria. Este processo é realizado em um intervalo de tempo pré-determinado, que pode ser ajustado pelo usuário.

O teste apresentado nessa seção avalia o consumo de bateria de um *PocketPC* quando está em execução o cliente do SELIC. Os equipamentos utilizados foram dez iPAQs HP hx4700. O teste foi realizado da seguinte forma para cada um dos equipamentos: No primeiro momento as baterias foram carregadas até atingirem 100% de carga; O cliente do SELIC teve seu arquivo de configuração alterado para que a comunicação com o *Web Service* fosse de *n* segundos. O teste considerou os valores 15, 30, 45 e 60; O sistema foi inicializado registrando o horário da primeira requisição; Quando o dispositivo teve sua bateria completamente descarregada, foi registrado o horário de desligamento; Foi calculada a média de durabilidade para a série de dispositivos, usando o número de minutos entre o seu desligamento e o horário de inicialização da aplicação.

A Figura 8 mostra a variação da durabilidade da carga da bateria em minutos (eixo Y), em função do tempo de envio de informações de localização (eixo X). A comunicação com o *Web Service* utiliza XML e um socket. Cada vez que uma requisição é realizada ao *Web Service* há um consumo de processador, memória e acesso à rede. Quanto menor esse tempo, maior a quantidade de recursos necessários para o processamento. No sistema quanto menor o tempo de coleta de informações de antenas e comunicação com o *Web Service* do SELIC, maior é a precisão de localização. Por conseqüência quanto maior a precisão, maior o consumo de energia.

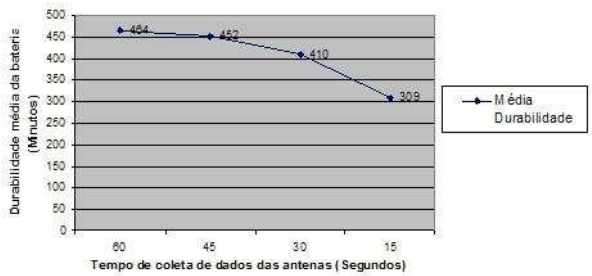


Figura 8. Durabilidade de bateria.

4.2. LOCAL: Um Sistema de Educação Ubíqua baseado em Localização

Recentemente, a aplicação da computação ubíqua na educação ocasionou o surgimento de uma nova frente de pesquisa denominada Educação Ubíqua [15]. Neste cenário, o MobiLab criou o modelo LOCAL (*LOcation and Context Aware Learning*) [3]. LOCAL usa informações de localização e de contexto como auxílio ao processo de ensino e de aprendizagem. Um sistema de localização acompanha a mobilidade dos aprendizes e, baseado nas suas posições físicas, explora oportunidades educacionais. A criação do LOCAL é baseada no HLS.

A arquitetura do LOCAL é formada por seis componentes (Figura 9). O primeiro é um sistema de perfis de usuário, que armazena dados relevantes ao processo de ensino e de aprendizagem. O segundo é o SELIC que atua como sistema de localização. O terceiro é um Assistente Pessoal (AP) que acompanha o usuário, executando em seu dispositivo móvel. O quarto é um repositório de objetos de aprendizagem, que armazena e indexa o conteúdo relevante ao processo pedagógico. O quinto componente é um sistema de envio de mensagens contextuais e o sexto é um motor de análise (Tutor) que realiza inferências usando dados fornecidos pelos sistemas de perfis e de localização.

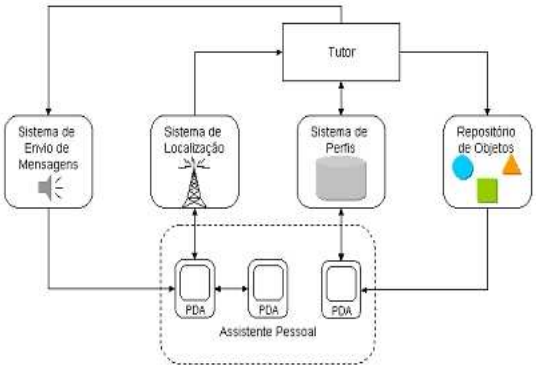


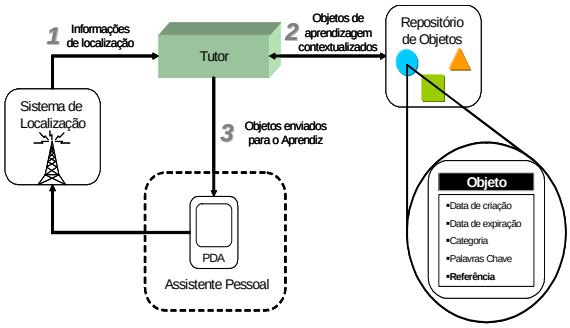
Figura 9. Arquitetura do LOCAL.

O AP possui as seguintes funcionalidades: (1) suporte a autenticação do aprendiz, ou seja, seu ingresso no sistema LOCAL; (2) armazenamento das informações persistentes do perfil; (3) suporte ao sistema de localização, permitindo o desligamento do mesmo se for de interesse do aprendiz; (4) suporte ao recebimento de avisos oriundos do sistema de mensagens. Atualmente o cliente do SELIC é integrado ao AP, ou seja, as funcionalidades do AP e do SELIC estão disponíveis no mesmo aplicativo.

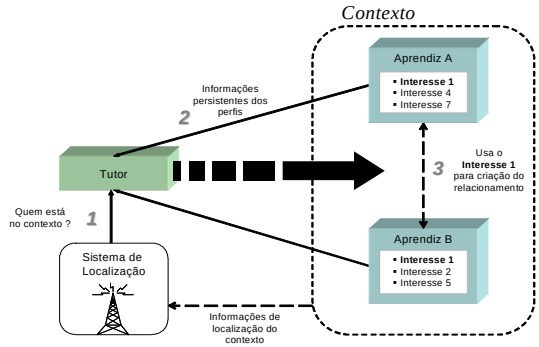
O Tutor usa os perfis e as informações de localização para inferência de oportunidades de ensino e de aprendizagem. Existem dois tipos de atuação: (1) envio de objetos de aprendizagem para os aprendizes de acordo com as oportunidades pedagógicas que surgem durante o seu deslocamento pelos contextos e (2) estímulo à interação entre aprendizes.

No primeiro, o SELIC informa para o Tutor, o contexto onde está o aprendiz (Figura 10a, passo 1). O Tutor usa essa informação, aliada ao perfil do aprendiz, para determinação dos objetos relevantes no contexto (passo 2). Os objetos são encaminhados para o aprendiz (passo 3). Este processo pode ser induzido por dois eventos: (1) a mudança de contexto do aprendiz ou (2) a inserção de novo material no repositório de objetos (neste caso, apenas os últimos dois passos são executados).

No segundo tipo de atuação do Tutor, as informações disponíveis nos perfis podem ser usadas para criação de vínculos entre os aprendizes. A Figura 10b exemplifica a atuação do Tutor estimulando a interação entre aprendizes através de similaridade. O Tutor descobre quem está em um contexto (passo 1), detecta um interesse em comum entre os aprendizes (passo 2) e envia mensagens para ambos estimulando a interação (passo 3).



(a) Envio contextualizado de objetos



(b) Estímulo à interação

Figura 10. Atuação do LOCAL.

5. Trabalhos Relacionados

No Gaia [16] o suporte à localização é realizado através do serviço *MiddleWhere* [14]. O *MiddleWhere* integra tecnologias de múltiplos sistemas de localização de objetos móveis (pessoas, equipamentos ou objetos físicos). Existe um mapeamento do mundo físico para *o MiddleWhere*, permitindo que exista relacionamentos entre ambientes e objetos móveis, ou seja, é possível relacionar todos os dispositivos que estão numa sala em um determinado momento.

O one.world [8] oferece um modelo de programação para a construção de aplicações ubíquas. No one.world o serviço de localização é denominado *Device Access Service* (DAS). O DAS coleta informações dos sensores (RFID) e converte para dados e eventos de aplicações. Os eventos são repassados a um módulo chamado *Proximity Service*, que realiza o *tracking* dos dispositivos.

O PLACE (*Pervasive Location Aware Computing Environments*) [13] é um projeto desenvolvido pela Universidade Purdue. O foco está em prover um suporte a aplicações sensíveis à localização. Seu objetivo é combinar funcionalidades de múltiplos sistemas de localização, como IEEE 802.11 e GPS. A principal funcionalidade da arquitetura PLACE é a habilidade de usuários móveis realizarem consultas a servidores de localização.

Uma comparação do HLS com os trabalhos relacionados permite as seguintes constatações:

1 - O SELIC possui a vantagem de sua interface ser baseada em Web Services, possibilitando a integração com qualquer aplicação, e não o restringindo apenas ao Holoparadigma. Essa vantagem foi demonstrada na aplicação LOCAL (seção 4.2);

2 - O gerenciamento das informações de localização nas aplicações Holo é realizado automaticamente pelo Context Changer. Nos demais trabalhos, o gerenciamento das informações é responsabilidade do programador e precisa ser implementada junto com a aplicação.

6. Conclusões

Este trabalho apresentou uma proposta para suporte a aplicações sensíveis à localização no Holoparadigma. Esta solução engloba um servidor de localização (SELIC), um módulo para gerenciamento automático de localização nas aplicações Holo (*Context Changer*) e uma ferramenta para visualização de ambientes baseados em árvores de contextos.

No SELIC destaca-se a interface de comunicação baseada em *Web Services*, o que permite seu uso de

forma simplificada. A modularização dos sistemas de localização usados no SELIC permite que sejam integrados novos sistemas sem a necessidade de alterar o protocolo com as aplicações (*Web Services*). Esta modularização também permite ao SELIC escolher o sistema de localização que tenha o maior grau de confiança na sua precisão, perante uma situação onde um equipamento é localizado por dois. Os testes do protótipo comprovaram a viabilidade do SELIC para aplicações de precisão *room level*. Além disso, o consumo de bateria dos dispositivos foi testado, permitindo uma avaliação do impacto do SELIC nessa característica dos equipamentos móveis. Entre os possíveis trabalhos futuros podem ser citados: (1) portar o cliente do SELIC para telefones celulares, permitindo o uso de informações de localização nesse tipo de dispositivo; (2) integrar o suporte GPS no servidor de localização, aprimorando assim, a precisão do SELIC em ambientes abertos (*outdoor*); (3) desenvolver novas aplicações que utilizem diretamente o SELIC e, em especial, aplicações em Holo que explorem as informações de localização gerenciadas automaticamente pelo *Context Changer*.

7. Referências

[1] Augustin, I. et al. *ISAM: a Software Architecture for Adaptative and Distributed Mobile Applications*. IEEE Symposium on Computers and Communications, Messina, Vol. 7, 2002, pp. 333-339.

[2] Augustin, I. et al. *ISAM, Joing Context-awareness and Mobility to Building Pervasive Applications*. Imad Mahgoub; Mohammad Ylias (Org.). Mobile Computing Handbook. New York, CRC Press, 2004, pp. 73-94.

[3] Barbosa, J. L.V.; Hahn, Rodrigo; Rabello, Solon A; Barbosa, Débora N. F. . LOCAL: *a Model Geared Towards Ubiquitous Learning*. In: 39th ACM Technical Symposium on Computer Science Education (SIGCSE), 2008, Portland. Proceedings of the ACM SIGCSE 2008. New York: ACM Press, 2008. p. 432-436.

[4] Barbosa, J. L.V. et al. Gholo: *a multiparadigm model oriented to development of grid systems*. Future Geneation Computing Systems. Vol. 21, No.1, (2005). pp. 227-237.

[5] Barbosa, J. L.V. et al. Holoparadigm: *a Multiparadigm Model Oriented to Development of Distributed Systems*. International Conference on Parallel and Distributed Systems,Vol. 8, pp. 165-170.

[6] Garlan, D. et al. *Project* Aura: *Toward Distraction-Free Pervasive Computing*. IEEE Pervasive Computing. Vol.4, (April-June, 2002). pp.22-31.

[7] Garzão, A.S.; Barbosa, J. L.V. *A virtual machine with concurrency, mobility and blackboards support*. Conferência Latinoamericana de Informática (CLEI), Vol. 24, 2003.

[8] Grimm, R. *System support for pervasive applications*. Phd Thesis, University of Washington, 2002.

[9] Hightower, J.; Gaetano, B. *Location Systems for Ubiquitous Computing*. IEEE Computer. Vol.34, No.8, (August, 2001). pp. 57-66.

[10] Hightower, J.; LaMarca, A. and Smith,I. *Practical Lessons from Place Lab. IEEE Pervasive Computing*. Vol. 5, No. 3, (July, 2006). pp. 32-39.

[11] MobiLab. Laboratório de Pesquisa e Desenvolvimento em Computação Móvel. Last Accessed Ago 2008 [Available at http://www.inf.unisinos.br/~mobilab].

[12] MobileIN Technologies. *Location Based Services (LBS)*. Last Accessed May 2007 [Available at http://www.mobilein.com/location_based_services.htm].

[13] Mokbel, M. F. et al. Place: *A query processor for handling real-time spatiotemporal data streams*. VLDB: International Conference on Very Large Data Bases, 2004, pp. 1377-1380.

[14] Ranganathan, A. et al. Middlewhere: *a middleware for location awareness in ubiquitous computing applications*. ACM/IFIP/USENIX international conference on Middleware. 2004, pp. 397-416.

[15] Rogers, Y.; et al. *Ubi-learning Integrates Indoor and Outdoor Experiences*. ACM Communications, v. 48, n. 1, Jan. 2005. p.55-59.

[16] Román, M. et al. Gaia: *A Middleware Infrastructure to Enable Active Spaces*. IEEE Pervasive Computing. Vol.1, (Oct-Dec, 2002). pp. 74-82.

[17] Satyanarayanan, M. *Fundamental Challenges in Mobile Computing*. ACM Symposium on Principles of Distributed Computing. New York: ACM Press, 1996.

[18] Satyanarayanan, M. Pervasive computing: vision and challenges. IEEE Journal. Vol. 8, No. 4, (August 2001), pp. 10-17.