Prototipagem rápida de ambientes ubíquos

José Luís Silva Óscar R. Ribeiro João M. Fernandes
José Creissac Campos
Dep. Informática / CCTC, Universidade do Minho, Braga, Portugal
{jlsilva, orribeiro, jmf, jose.campos}@di.uminho.pt

Michael D. Harrison
Newcastle University, United Kingdom
michael.harrison@ncl.ac.uk

Resumo

A experiência de utilização de ambiente ubíquos é um factor determinante no seu sucesso. As características de tais sistemas devem ser exploradas o mais cedo possível para antecipar potenciais problemas de utilização por parte do utilizador e para reduzir custos de re-design*. No entanto, o desenvolvimento antecipado de protótipos a serem avaliados no ambiente final pode ser disruptivo e tornar-se inaceitável. O desenvolvimento de protótipos de ambientes ubíquos pode ajudar, fornecendo indicações de como o utilizador irá reagir perante os ambientes. Este artigo descreve o APEX, uma plataforma de prototipagem rápida de ambientes ubíquos que junta a CPN Tools com um servidor de aplicações 3D existente. Os protótipos desenvolvidos com o APEX permitem que os utilizadores naveguem num mundo virtual, podendo experimentar muitas das características do* design *proposto. A arquitectura do APEX e a modelação baseada em CPN são descritas. Um exemplo ilustra a abordagem.*

Palavras-Chave

Computação ubíqua, modelação, ambientes virtuais

1. Introdução

A computação ubíqua impõe novos desafios de usabilidade para os *designers* e *developers* de sistemas interactivos. Dado que nestes sistemas os utilizadores estão imersos, o efeito que os sistemas tem sobre a experiência de utilização de um dado espaço torna-se um elemento importante que contribui para o sucesso do *design*. Um sistema bem sucedido deve, por exemplo, conseguir transformar um ambiente estéril num lugar no qual as pessoas estejam em harmonia com o ambiente e seus propósitos. Um exemplo de criação de tal experiência num ambiente ubíquo pode ser o de efectuar um *check-in* num aeroporto e mover-se com o mínimo de atraso através das várias etapas da partida. Outro exemplo pode ser um ambiente que fornece informação adaptada aos utilizadores de uma biblioteca, facilitando-lhes localizar a prateleira onde o livro desejado se encontra. A noção de *experiência* é difícil de capturar como um requisito que pode ser demonstrado num sistema. Actualmente, não existem técnicas que possam ser utilizadas permitindo analisar especificações contra diferentes noções de experiência (ver [Harrison 08] para uma discussão deste tema).

Este artigo aborda a prototipagem de ambientes ubíquos projectados para melhorar ambientes físicos reais através da utilização de sensores, ecrãs públicos e dispositivos pessoais. Um aspecto particularmente relevante destes sistemas é a forma como os utilizadores interagem com o ambiente, quer através de interacções explícitas com o sistema, quer de interacções implícitas que surgem através de mudanças do contexto. Aqui o contexto pode incluir a posição ou as diferentes etapas que tem de ser cumpridas pelo utilizador por forma a atingir um determinado objectivo (por exemplo: *check-in*, controlo do passaporte, digitalização do bilhete de embarque).

O objectivo do trabalho é possibilitar a avaliação de ambientes ubíquos precocemente, antes mesmo da sua efectiva instalação *in situ*. O artigo é baseado no pressuposto que protótipos serão utilizados para explorar o impacto que o *design* de um dado ambiente ubíquo terá nos utilizadores consoante eles se mexem, interagem e acedem a serviços dentro desse ambiente. Para evitar custos de desenvolvimento desnecessários, o *design* é a explorado desde cedo através de protótipos baseados em modelos, recorrendo a ambientes virtuais. O artigo descreve um plataforma de prototipagem (APEX) que utiliza modelos de redes de Petri coloridas [Jensen 07] (CPN - Coloured Petri Nets). O APEX interliga um modelo CPN com um servidor de aplicações 3D (OpenSimulator[1]). A plataforma permite a prototipagem de ambientes ubíquos, possibilitando que os

[1] http://opensimulator.org (acedido em: 14 de Junho de 2010)

utilizadores naveguem na simulação do ambiente virtual através do controlo (actualmente através do rato e do teclado) de um avatar que o representa, tornando possível avaliar problemas de usabilidade, incluindo noções de experiência de utilização.

A estrutura do artigo é a seguinte. A secção 2 apresenta trabalhos relacionados presentes na literatura. A secção 3 descreve a arquitectura do APEX. A plataforma é ilustrada através de uma biblioteca inteligente que detecta a presença dos seus utilizadores e os guia até as prateleiras onde se encontram os livros que estes pretendem requisitar. A secção 4 descreve como o exemplo é modelado. A secção 5 descreve a utilização da plataforma e a secção 6 apresenta conclusões e trabalho futuro.

2. Literatura relacionada

Apesar de avanços consideráveis no desenvolvimento de sistemas ubíquos, continua a existir uma tendência para que o seu desenvolvimento e avaliação sejam feitos principalmente através de experimentação, desenvolvendo protótipos de dispositivos a avaliar em sistemas parcialmente desenvolvidos (ver [Davies 05] para uma discussão deste tema). A questão de como avaliar os sistemas num contexto real permanece um tópico em aberto [Abowd 05]. Outro importante aspecto de avaliação é saber como explorar a experiência criada por um dado sistema. Nesse campo, existe uma literatura substancial proveniente do campo do *design* (ver, por exemplo, [Buchenau 00]). Um exemplo de uma abordagem típica no campo do *design* é desenvolver protótipos não-funcionais que potenciais utilizadores podem transportar no contexto onde o sistema real irá ser utilizado. O objectivo é obter informações acerca do modo como o *design* proposto irá ser experimentado.

O APEX é projectado para satisfazer 3 requisitos. O primeiro é permitir o desenvolvimento rápido de protótipos. Embora existam várias plataformas de computação ubíqua (e.g., [Braubach 02, Garlan 02, Harter 01]), o desenvolvimento de protótipos que permitam avaliar a reacção dos utilizadores a ambientes ubíquos é uma área ainda relativamente pouco explorada.

O segundo requisito é a utilização de ambientes 3D para construir simulações que podem ser exploradas de forma realística por utilizadores. Servidores de aplicações 3D, tais como o SecondLife™[2] ou o OpenSimulator fornecem uma via de desenvolvimento rápido de mundos virtuais. O OpenSimulator em particular tem a vantagem de disponibilizar o código fonte, o que significa que pode ser programado permitindo uma maior configurabilidade e extensibilidade.

O terceiro requisito é possuir uma abordagem para a modelação de sistemas ubíquos. As redes de Petri constituem uma linguagem de modelação gráfica expressiva e tem sido utilizada para descrever ambientes virtuais. Abordagens anteriores baseadas em redes de Petri incluem: Hybrid high level Nets (HyNets) [Massink 99], Flownets [Smith 99], Interactive Cooperative Objects (ICO) [Navarre 05] e CPNs [Jensen 07].

Um objectivo desta plataforma é integrar a abordagem de modelação com abordagens analíticas que permitam validar propriedades nos ambientes ubíquos que sejam relevantes aquando da sua utilização. Neste contexto, a utilização de CPNs surgiu como escolha lógica dado o bom suporte ao nível de ferramentas de animação e análise de modelos.

Embora possam ser encontradas na literatura várias abordagens à prototipagem de computação ubíqua, somente um número limitado refere aspectos de interacção do utilizador com o ambiente e da experiência obtida pelo utilizador. Alguns sistemas (e.g. [Li 04]) recorrem as técnicas de *Wizard of Oz* para evitar a instalação de sensores no ambiente. Outros, tais como 3DSim [Shirehjini 05], UbiWorld [Disz 97] ou o trabalho de O'Neill et al. [O'Neill 09] têm visões similares à nossa.

A abordagem de O'Neill et al. é a que mais se aproxima, utilizando modelos e um motor de jogos 3D para a prototipagem de ambientes ubíquos (3DSim e UbiWorld assemelham-se mais a *frameworks* de programação). Acreditamos que o uso de servidores de aplicações 3D (OpenSimulator) tem algumas vantagens comparativamente com a utilização de um motor de jogos. Os servidores de aplicações 3D suportam a criação de ambientes virtuais em tempo real utilizando ferramentas de construção de objectos e são facilmente extensíveis através do carregamento de módulos. No caso dos motores de jogos, o ambiente deve ser previamente criado utilizando um editor de mapas. Utilizar um servidor de aplicações 3D significa que a abordagem é flexível. Uma variedade de clientes, personalizáveis em aparência, podem aceder o mundo virtual em simultâneo utilizando diferentes protocolos e o desenvolvimento dos ambientes pode ser efectuado através de linguagens de programação variadas.

3. Arquitectura

O APEX utiliza a CPN Tools[3] para modelar o comportamento de ambientes virtuais. Esse comportamento é então ligado ao servidor de aplicações 3D (OpenSimulator). Modelos CPN dos diferentes tipos de dispositivos presentes no ambiente (e.g. sensores, ecrãs, dispositivos pessoais) devem ser criados. O objectivo é desenvolver um estilo genérico de CPN relevante para a modelação de ambiente virtuais, incluindo modelos que podem ser instanciados no espaço real no qual o sistema foi definido para operar.

O OpenSimulator permite a criação interactiva de ambientes virtuais suficientemente ricos permitindo aos utilizadores visualizar as características dos sistemas reais que estes representam. Ambientes e dispositivos pré-definidos podem ser utilizados nesse processo de criação.

Uma vez o modelo CPN e o ambiente criados, um componente do APEX é responsável por ligá-los. Vários utilizadores podem estabelecer ligações à simulação com diferentes pontos de vista no servidor OpenSimulator. Os

[2] http://secondlife.com (acedido em: 14 de Junho de 2010)

[3] http://wiki.daimi.au.dk/cpntools/ (acedido em: 14 de Junho de 2010)

utilizadores podem navegar e interagir com a simulação do ambiente ubíquo, permitindo avaliar questões relacionadas com o design proposto.

A visão global da arquitectura da plataforma é apresentada na figura 1. Esta é composta por 3 partes:

1. um *componente comportamental* responsável por gerir o comportamento do protótipo, incluindo a descrição, análise e validação do comportamento do ambiente virtual;
2. um *componente arquitectural* responsável por gerir a aparência física do protótipo, incluindo a gestão da simulação 3D e da construção do ambiente virtual;
3. um *componente de comunicação/execução* responsável pela troca de dados entre todos os componentes da plataforma.

3.1. Componente comportamental

Este componente utiliza a linguagem de modelação CPN para descrever o comportamento do ambiente virtual em resposta a acções do utilizador e a mudanças do contexto. Um modelo CPN genérico base foi desenvolvido contendo:

1. um módulo para inicializar a simulação e para estabelecer a conexão entre o modelo CPN representado na CPN Tools e o OpenSimulator;
2. um módulo que recebe dados do utilizador (por exemplo, identidade e posição) a partir do OpenSimulator quando este se move, utilizando-os para actualizar *tokens* apropriados do modelo CPN;
3. um módulo descrevendo o comportamento de cada um dos tipos de dispositivos do sistema.

Um exemplo de um modelo CPN para o *componente comportamental* é apresentado na secção 4.

3.2. Componente arquitectural

Este componente utiliza o OpenSimulator para definir características 3D da simulação apresentada aos utilizadores e permite a navegação destes na simulação. As características 3D da simulação incluem a localização, o aspecto visual e a física de cada objecto no ambiente. Estas definições são efectuadas utilizando uma ferramenta de visualização, e.g. o Hippo Viewer[4] ou o Linden Lab's Second Life viewer[5]. Existem outros visualizadores compatíveis disponíveis (ver http://opensimulator.org/wiki/Connecting). Actualmente alguns desses visualizadores permitem somente a exploração do ambiente sem fornecer qualquer ferramenta de edição.

Este componente é também responsável por permitir a navegação livre e a interacção no ambiente. A interacção pode ser efectuada explicitamente pelos utilizadores utilizando dispositivos (virtuais) e implicitamente através de mudanças do contexto. A plataforma permite a conexão de vários utilizadores de, possivelmente, diferentes localizações ao mesmo ambiente virtual através da Web. Ambientes e objectos pré-definidos podem ser guardados/carregados utilizando ficheiros OAR (Opensim ARchive files). Todas as entidades (objectos, terrenos, texturas, etc.) são codificadas nesses ficheiros no formato utilizado pelo OpenSimulator para guardar dados em ficheiros.

3.3. Componente de comunicação/execução

Este componente é uma DLL (*dynamic-link library*) responsável pelo carregamento do ambiente ubíquo e por utilizar os modelos CPN para o dirigir. A comunicação é baseada no Comms/CPN [Gallasch 01], uma biblioteca desenvolvida para conectar a CPN Tools com processos Java e C. Como os módulos para o OpenSimulator são desenvolvidos em C#, um novo pacote de comunicação C#/CPN teve de ser desenvolvido. Este pacote envia informações para a CPN Tools quando mudanças no ambiente ocorrem e é também responsável por modificar o ambiente em resposta a dados enviados pela CPN Tools. Adicionalmente, este componente controla o carregamento/gravação de objectos e ambientes no OpenSimulator.

4. Exemplo: modelação com CPNs

O exemplo utilizado para ilustrar o sistema é uma biblioteca inteligente. Todos os livros são identificados com etiquetas RFID e são dispostos em prateleiras que tem LEDs associadas. São utilizados ecrãs para fornecer informação aos utilizadores da biblioteca. Um utilizador registado pode entrar/sair da biblioteca através das portas.

Quando um utilizador registado chega à porta de entrada, esta abre e o ecrã associado apresenta os livros que foram previamente requisitados por ele (e.g. através da página Web da biblioteca). De seguida o sistema guia, com informação em ecrãs presentes na biblioteca, o utilizador até aos livros requisitados através do uso de sensores que reconhecem a posição do utilizador em tempo real. Assim que os utilizadores se aproximam da localização dos referidos livros (distância configurável no modelo CPN), uma das luzes com uma cor específica liga-se. Uma vez que vários utilizadores podem estar à procura de livros em localizações próximas, este método facilita a distinção dos diferentes pedidos, dado que cada utilizador utilizará uma cor diferente, previamente indicada quando este chega à biblioteca. Quando o utilizador se dirigir para a porta de saída, uma lista personalizada dos livros requisitados será apresentada no ecrã perto da porta e esta será aberta caso o utilizador possa requisitar os livros. À saída do utilizador, os RFIDs dos livros são detectados e consequentemente requisitados automaticamente.

4.1. Inicializando a simulação

As condições iniciais da simulação estão definidas no módulo CPN apresentado na figura 2. A execução da transição "`initialise simulation`" inicializa

[4] http://mjm-labs.com/viewer/ (acedido em: 14 Junho de 2010)

[5] http://secondlife.com/support/downloads (acedido em: 14 Junho de 2010)

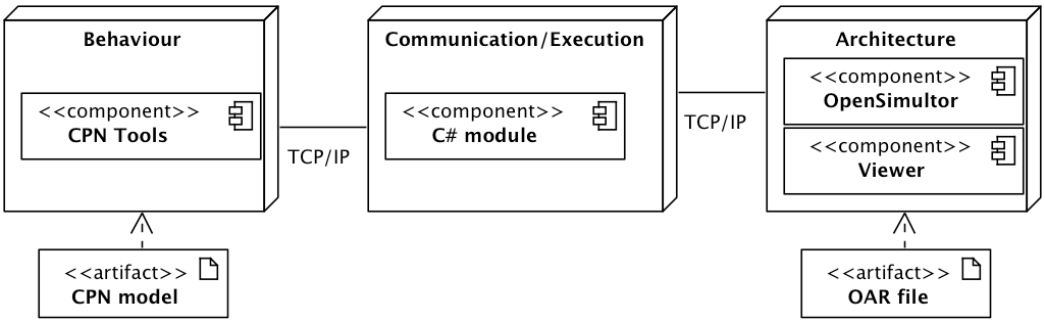

Figura 1. Arquitectura da plataforma APEX

a configuração inicial da simulação. Neste caso a configuração incluí 3 lugares: "`users`", "`gates`" e "`bookshelves`". As *Etiquetas de fusão* (e.g., etiqueta `books` no lugar `bookshelves`) permitem que instancias desses lugares apareçam em outras partes do modelo CPN. Esses lugares são chamados *fusion places*. A transição "`initialise simulation`" cria uma nova conexão e utiliza dois lugares para controlar a execução do modelo CPN: "`init`" para limitar a execução da transição a uma ocorrência e "`run`" para informar os outros módulos CPN que a simulação está em execução.

4.2. Leitura das posições do utilizador

A figura 3 apresenta o módulo CPN responsável por colectar os dados dos utilizadores. A transição "`read user id`" lê o identificador de um utilizador. Um *token* com o identificador do utilizador associado é introduzido no lugar "`read user ids`". Este é utilizado para ler a nova posição do utilizador através da transição "`read and update user position`" que também actualiza o *token* do respectivo utilizador. A expressão "`isThisUser(u,uId)`" na guarda desta transição garante que o *token* do utilizador que é actualizado corresponde ao identificador que foi previamente lido. Neste modelo, o número de utilizadores permanece constante durante cada sessão de uma simulação, i.e. os utilizadores que se podem conectar à simulação em tempo de execução devem estar previamente inseridos no modelo (variável "`initial_USERS`").

Os módulos CPN leitores de posições dos utilizadores provenientes da simulação e os que descrevem o comportamento dos dispositivos executam concorrentemente. A precedência das transições dos dispositivos sobre a aquisição de dados é garantida através da guarda "`not (hadASignificantMovement(u))`" sobre a transição "`read and update user position`". Um movimento de um utilizador é considerado significativo (para um dispositivo) quando a nova posição está "próxima" do dispositivo. Fica desta forma garantido que não ocorrem comportamentos indevidos devido a leituras antecipadas.

4.3. Modelação dos dispositivos do sistema

Cada tipo de objecto dinâmico (dispositivo) presente no ambiente ubíquo simulado precisa de um correspondente módulo CPN que descreva o seu comportamento. Os *fusion places* são a base do processo de criação desses módulos. Para criar os modelos desses módulos os *fusion places* necessários provenientes do modelo da figura 2 devem ser clonados nesses modelos. Neste exemplo os lugares "`users`", "`gates`" e "`bookshelves`" contêm os *tokens* relativos aos objectos dinâmicos presentes. Estes *tokens* são utilizados para modelar o comportamento do sistema. Isto é efectuado através da combinação de *fusion places*, lugares normais, transições, funções (descritas na linguagem CPN ML associada à CPN Tools) e condições. As transições tem um papel importante nesse processo já que são elas que estabelecem a conexão entre os modelos CPN e o simulador, através da utilização de funções CPN ML. As funções também são utilizadas para descrever algum comportamento que não seja expressado estruturalmente pela rede de Petri. Um exemplo do modelo de um desses dispositivos é o modelo da porta de entrada apresentado na figura 4.

A porta de entrada é equipada com um sensor que reconhece quando um utilizador se aproxima para entrar na biblioteca. A transição "`show info and open gate`" representa a acção da porta de entrada. Esta mostra os livros requisitados no ecrã e abre a porta de entrada. Estas acções ocorrem quando o sensor da porta detecta a chegada de um utilizador (modelado com a condição "`isArrivingToGateArea(u,g)`"). Quando a porta está aberta e outro utilizador registado entra na zona da porta, a transição "`add a user`" é executada incluído o utilizador no conjunto de utilizadores perto da referida porta.

Quando a transição "`show default and close gate`" é executada, informação por omissão é mostrada no ecrã e a porta é fechada. Para que isto aconteça um utilizador tem de se afastar da porta e mais nenhum utilizador pode estar perto dela. No caso de outros utilizadores estarem perto da porta, a transição "`remove a user and update info`" remove o referido utilizador do conjunto de utilizadores que estão perto da porta. Caso existisse informação para esse utilizador no ecrã, essa é trocada por informação relevante para outro utilizador que ainda esteja perto da porta.

Utilizando a ferramenta *State Space tool* fornecida com a CPN Tools podem validar-se propriedades dos modelos.

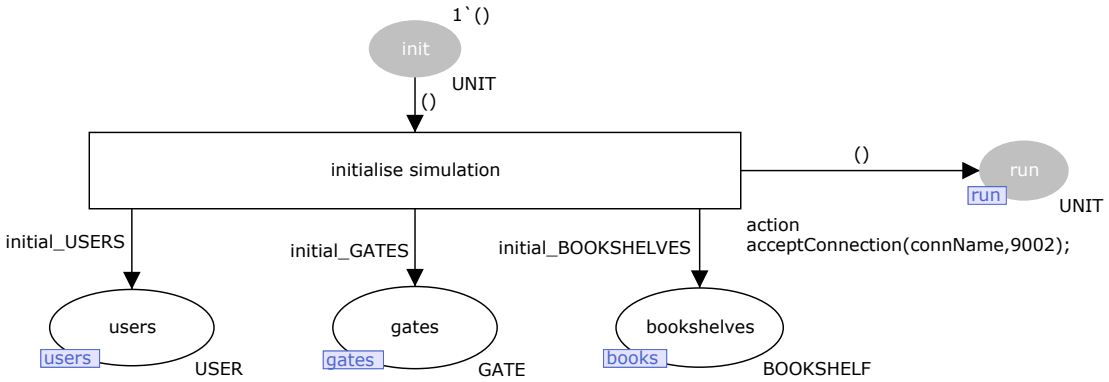

Figura 2. Módulo CPN para inicializar a simulação da biblioteca.

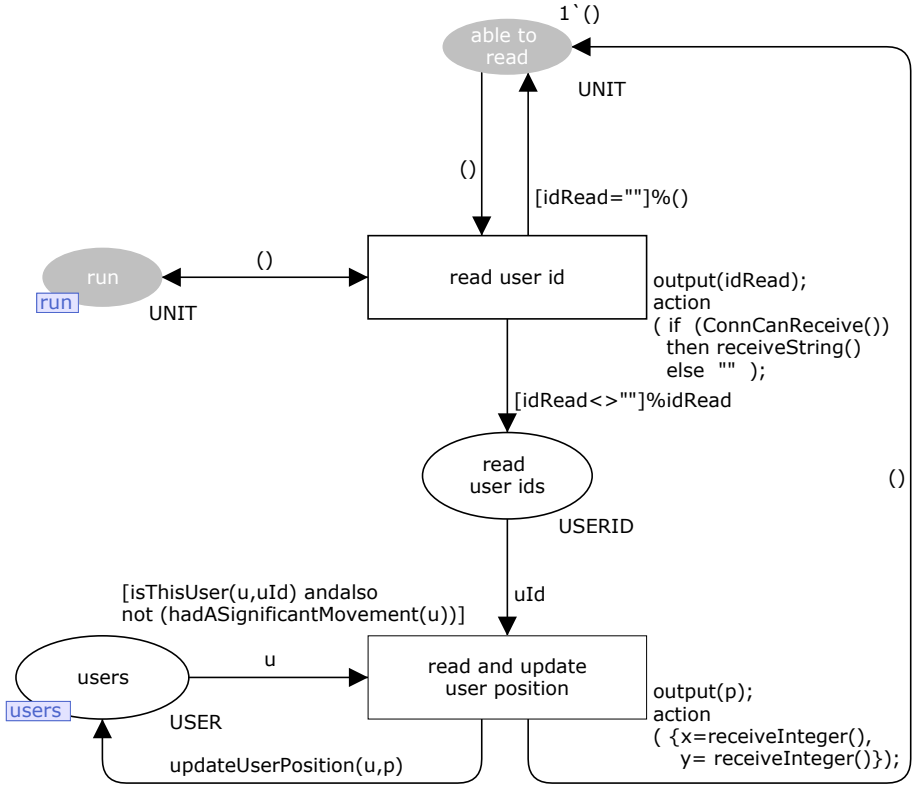

Figura 3. Módulo CPN para aquisição de dados dos utilizadores.

Por exemplo, a propriedade de acessibilidade, i.e. se todos os lugares são acessíveis ou se um dado lugar é acessível a partir de outro lugar numa dada situação, são exemplos de propriedades que podem ser facilmente validadas através da utilização de funções fornecidas pela *State Space tool* (c.f. *AllReachable()*, *Reachable(node,node)*).

5. Utilização

Como foi referido na secção 1, o objectivo do APEX é suportar o *design* e análise de sistemas ubíquos. A pessoa responsável pelo desenvolvimento do protótipo cria modelos CPN, tal como ilustrado na secção 4. Dependendo da necessidade de novos tipos de dispositivos, poderá ter que desenvolver novos modelos ou apenas reutilizar modelos existentes.

Cada dispositivo e cada utilizador é representado no modelo CPN por um *token* inserido no respectivo lugar etiquetado como *lugar de fusão* (e.g. veja o lugar *users* na figura 4). Cada um desses *tokens* tem um identificador que é comum aos identificadores dos objectos presentes na simulação. Estes dados são usados pelas funções CPN ML conjuntamente com instruções (e.g. *open*, *close*) para indicar mudanças que devem ser reflectidas no OpenSimulator. Estas indicações são responsáveis por reflectir mudanças nos objectos do ambiente de acordo com o estado do modelo CPN. Como exemplo, a figura 5 mostra um pedaço de código que procura objectos que devem ser modificados, de acordo com instruções recebidas do modelo CPN, e efectua as respectivas alterações. Nesse pedaço de código, a posição da porta é alterada de acordo com as acções *open*

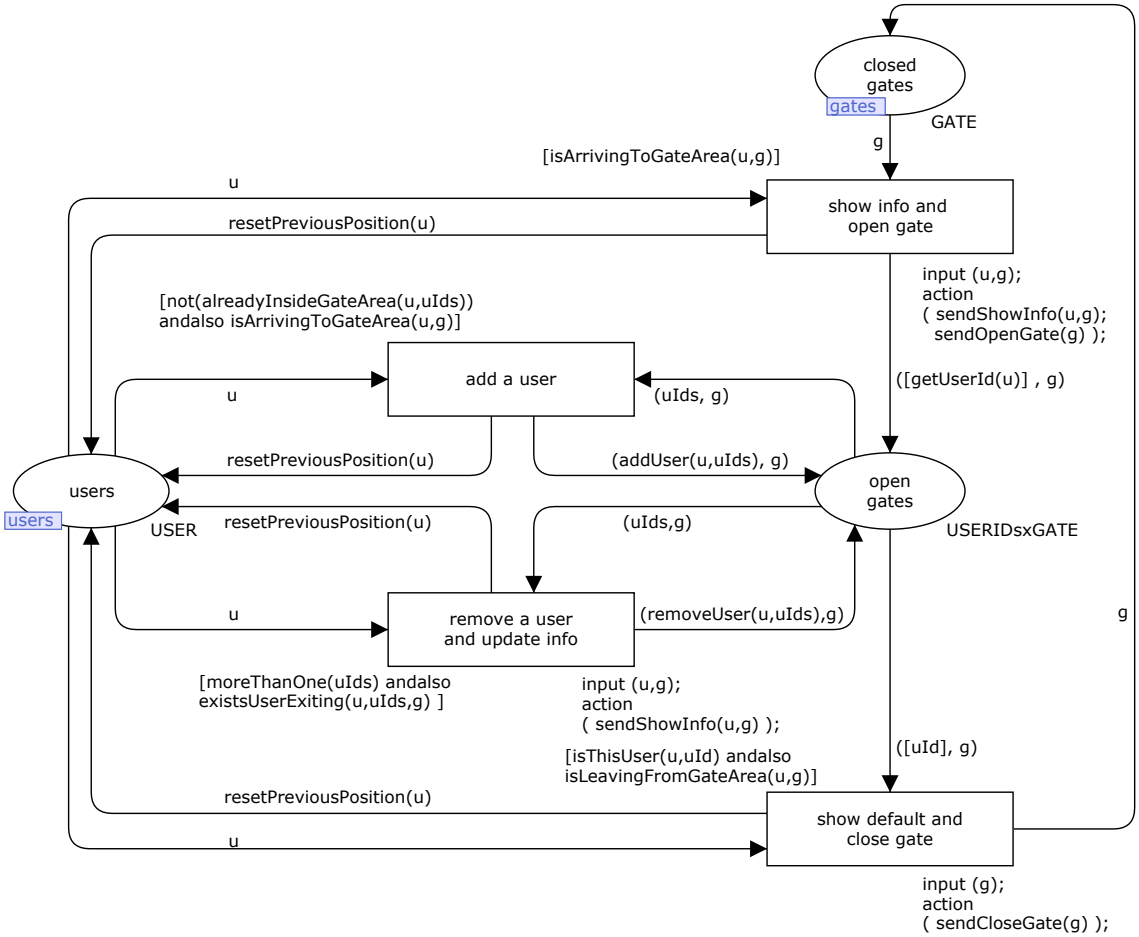

Figura 4. Módulo CPN para o dispositivo *porta de entrada***.**

e *close* que são recebidas.

Uma vez o servidor OpenSimulator e a CPN Tools instalados e configurados, o *componente de comunicação/execução* e o modelo CPN carregados, passa a ser possível a exploração do mundo virtual.

Existem dois modos possíveis de utilização do APEX. O modo de desenvolvimento e o modo de utilizador. No primeiro modo, o *developer* tem de alterar/estender o modelo CPN base de modo a reflectir o comportamento de novos dispositivos presentes no ambiente. No caso da definição de novos dispositivos, pequenas modificações tem também de ser efectuadas no módulo de comunicação/execução (módulo C#) de modo a que este saiba como reflectir no novo objecto do ambiente as instruções do modelo CPN. No modo de utilizador, este somente tem de descarregar um visualizador e configurá-lo de modo a poder aceder através da Web ao ambiente simulado. O acesso ao servidor é efectuado através da utilização de uma conta de utilizador criada para o efeito. Nesse modo são utilizados modelos CPN e ambientes virtuais já desenvolvidos sendo que o utilizador experimenta o sistema através de uma simples conexão ao servidor.

Para além disso, a criação, remoção e modificação de objectos pode ser efectuada através da ferramenta de visualização (ver figura 6) *gritando* (forma de interacção do avatar com o mundo disponibilizada pelo visualizador) os seguintes comandos:

- load-oar ficheiro - responsável por carregar um ambiente/objecto a partir de um ficheiro;
- save-oar ficheiro - responsável por guardar um ambiente/objecto num ficheiro;
- clear - remove todos os objectos presentes no ambiente, ficando somente os utilizadores e o terreno.

6. Conclusões e trabalho futuro

A experiência dos utilizadores em ambientes ubíquos é um factor determinante no seu sucesso. Possibilitar a exploração precoce das características de tais sistemas irá ajudar a antecipar potenciais problemas de utilização e reduzir o custo de *re-design*. No entanto, o desenvolvimento de protótipos no ambiente físico de destino é na maior parte dos casos inviável. Isto acontece devido aos custos de desenvolvimento de tais protótipos e por poder levar a uma ruptura do ambiente a que o sistema em desenvolvimento se destina. Alternativas devem ser exploradas, fornecendo um adequado grau de experiência no uso de sistemas ubíquos, mas evitando os custos da implementação real.

```
foreach (KeyValuePair<Scene, List<SceneObjectGroup>> kvp in HelloWorldModule.scene_prims)
 {
          foreach (SceneObjectGroup sog in kvp.Value)
          {

                  //Object: gate

                  if (sog.Name.Equals("gate"))
                  {
                      if (action.Equals("open"))        //Open the gate
                          sog.AbsolutePosition = new Vector3(130.7f,130.9f,25.9f)
                      else                              //Close the gate
                          if (action.Equals("close"))
                             sog.AbsolutePosition = new Vector3(128.6f,130.9f,25.9f)

                  }

                  //Object: screen

                  …
          }
 }
```

Figura 5. Código relativo ao comportamento dos objectos do OpenSimulator

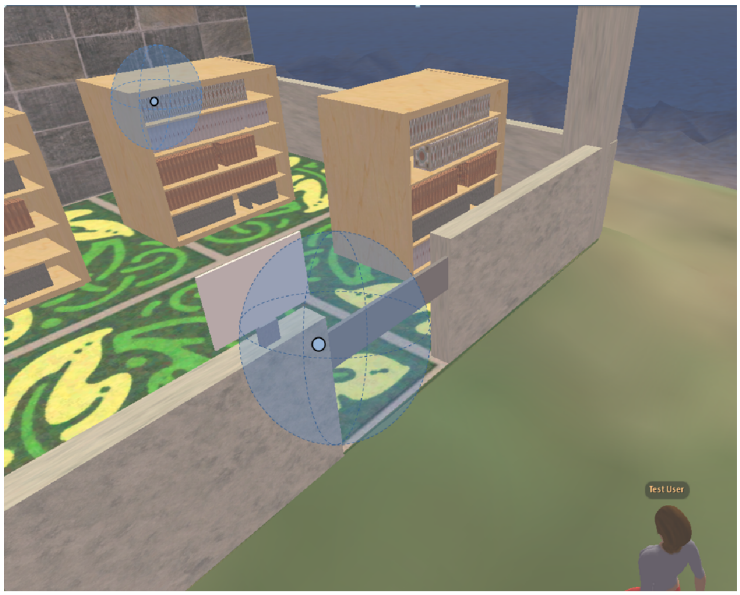

Figura 6. Interface do visualisador

Este artigo apresenta uma proposta para tais alternativas. Uma plataforma de prototipagem baseada em simulação para sistemas de computação ubíquos. Esta plataforma fornece, juntamente, a expressividade das redes de Petri, com a possibilidade de exploração de uma simulação virtual 3D do sistema modelado. O desenvolvimento de modelos e ambientes 3D é acelerado através da utilização do modelo CPN base e de ambientes pré-definidos. Isto permite que potenciais utilizadores possam explorar a simulação do sistema antes da sua implementação, sendo possível ter uma abordagem de prototipagem com um baixo custo.

Desenvolvimentos futuros da plataforma envolvem a sua avaliação com os utilizadores e *developers*. A avaliação dos utilizadores refere-se à fidelidade dos resultados, i.e. se os ambientes prototipados podem ser usados eficazmente permitindo aos utilizadores uma experiência suficientemente rica do *design*. A avaliação dos *developers* foca-se na agilidade da abordagem, na facilidade com que os protótipos podem ser desenvolvidos de uma forma precisa para ambientes ubíquos.

Trabalhos futuros no desenvolvimento da plataforma referem-se a questões técnicas que visam melhorar o suporte a utilizadores e *developers*. Estes incluem:

- A possibilidade de adicionar novos utilizadores à simulação em tempo de execução, sem que estes tenham que estar previamente inseridos no modelo;
- Reduzir a quantidade de informação trocada entre a CPN Tools e APEX para a mínima possível – isto é relevante para prever que a CPN Tools execute além dos recursos;
- Conectar a simulação a dispositivos móveis, tais como PDAs, via bluetooth – isto irá permitir uma ex-

periência de utilização mais realista e imersiva.

Agradecimentos

Este trabalho é suportado pela Fundação para a Ciência e Tecnologia (FCT, Portugal) através da bolsa de doutoramento SFRH/BD/41179/2007.

Referências

[Abowd 05] G.D. Abowd, G.R. Hayes, G. Iachello, J.A. Kientz, S.N. Patel, M.M. Stevens, e K.N. Truong. Prototypes and paratypes: designing mobile and ubiquitous computing applications. *IEEE Pervasive Computing*, 4(4):67–73, 2005.

[Braubach 02] L. Braubach, A. Pokahr, D. Moldt, A. Bartelt, e W. Lamersdorf. Tool-supported interpreter-based user interface architecture for ubiquitous computing. Em *Interactive Systems*, volume 2545 de *Lecture Notes in Computer Science*, páginas 89–103. Springer-Verlag, 2002.

[Buchenau 00] M. Buchenau e J.F. Suri. Experience prototyping. Em *Proceedings Designing Interactive Systems (DIS'00)*, páginas 424–433. ACM Press, 2000.

[Davies 05] N. Davies, J. Landay, S. Hudson, e A. Schmidt. Rapid prototyping for ubiquitous computing — guest editors' introduction. *IEEE Pervasive Computing*, 4(4):15–17, 2005.

[Disz 97] T.E. Disz, M.E. Papka, e R. Stevens. UbiWorld: an environment integrating virtual reality, supercomputing, and design. Em *Proceedings of the Heterogeneous Computing Workshop*, páginas 46–59, April 1997.

[Gallasch 01] G. Gallasch e L.M. Kristensen. Comms/CPN: A communication infrastructure for external communication with design/CPN. Em K. Jensen, editor, *3rd Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools (CPN'01)*, páginas 75–90. DAIMI PB-554, Aarhus University, 2001.

[Garlan 02] D. Garlan, D.P. Siewiorek, A Smailagic, e P Steenkiste. Project Aura: toward distraction-free pervasive computing. *IEEE Pervasive Computing*, páginas 22–31, April-June 2002.

[Harrison 08] M.D. Harrison, J.C. Campos, G. Doherty, e K. Loer. Connecting rigorous system analysis to experience centred design. Em E. Law, E. Hvannberg, e G. Cockton, editores, *Maturing Usability: Quality in Software, Interaction and Value*, Human Computer Interaction Series, páginas 56–74. Springer-Verlag, 2008.

[Harter 01] A. Harter, A. Hopper, P. Steggles, A. Ward, e P. Webster. The anatomy of a context-aware application. *Wireless Networks*, 1:1–16, 2001.

[Jensen 07] K. Jensen, L.M. Kristensen, e L. Wells. Coloured petri nets and CPN tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer (STTT)*, 9(3-4):213–254, 2007.

[Li 04] Y. Li, J.I. Hong, e J.A. Landay. Topiary: a tool for prototyping location-enhanced applications. Em *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, páginas 217–226. ACM, 2004.

[Massink 99] M. Massink, D.J. Duke, e S.P. Smith. Towards hybrid interface specification for virtual environments. Em D.J. Duke e A. Puerta, editores, *Design, Specification and Verification of Interactive Systems '99*, páginas 30–51. Springer-Verlag, 1999.

[Navarre 05] D. Navarre, P. Palanque, R. Bastide, A. Schyn, M. Winckler, L. Nedel, e C. Freitas. A formal description of multimodal interaction techniques for immersive virtual reality applications. Em *INTERACT 2005*, volume 3585 de *Lecture Notes in Computer Science*, páginas 170–183. Springer-Verlag, 2005.

[O'Neill 09] E. O'Neill, D. Lewis, e O. Conlan. A simulation-based approach to highly iterative prototyping of ubiquitous computing systems. Em *2nd International Conference on Simulation Tools and Techniques*, páginas 1–10. ICST, 2009.

[Shirehjini 05] Ali A. Nazari Shirehjini e Felix Klar. 3DSim: rapid prototyping ambient intelligence. Em *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, 2005.

[Smith 99] S. Smith, D. Duke, e M. Massink. The hybrid world of virtual environments. *Computer Graphics Forum*, 18(3):C287–C307, 1999.