Um Modelo de Dados Extensível para Configuração da Simulação de Redes de Sensores Sem Fio

Luciana Brasil Rebelo dos Santos[1], **Gian Ricardo Berkenbrock**[1], **Celso Massaki Hirata**[1]

[1]Divisão de Ciência da Computação – Instituto Tecnológio de Aeronáutica (ITA)
São José dos Campos – SP – Brazil

{lurebelo,gian,hirata}@ita.br

Abstract. *Simulation enables a detailed analysis of Wireless Sensor Networks (WSNs) behaviour in a cost effective way, and for this reason it is an important option for studying such systems. Although a vast number of WSN simulators is available, each of them was designed to satisfy specific needs. Because of the diverse aspects of WSNs, a unique simulation model may not be accurate and complete to address the analysis of all the WSN requirements. Additionally, the integration among the simulators is generally weak or non-existent. In this context, this work presents an abstract data model of WSNs for simulation modeling, allowing that different simulators share the same configuration data model, which enables simulation models interoperability.*

Resumo. *A simulação possibilita estudar Redes de Sensores Sem Fio (RSSFs) de forma rápida e econômica, tornando-se uma importante opção para estudo destas redes. Apesar do grande número de simuladores de RSSFs disponíveis, cada um deles foi projetado para atender necessidades específicas. Por causa dos aspectos singulares das RSSFs, os modelos existentes podem não ser precisos e completos para todas as áreas de simulação de RSSFs. Além disso, a integração entre esses simuladores é pequena ou inexistente. Nesse contexto, este trabalho apresenta uma forma de modelagem abstrata de RSSFs, permitindo que diferentes simuladores compartilhem um mesmo modelo de dados de configuração, promovendo interoperabilidade de modelos de simulação.*

1. Introdução

Uma Rede de Sensor Sem Fio (RSSF) é desenvolvida para detectar fenômenos, coletar dados, processá-los e transmitir a informação coletada para usuários [Ilyas and Mahgoub 2005]. As RSSFs são providas de nós que podem sensoriar fenômenos físicos, executar pequeno processamento e realizar comunicação sem fio. Estes nós são chamados de nós sensores. Nós sensores são pequenos dispositivos com reduzida capacidade de processamento, comunicação e energia e de baixo custo. Cada nó é equipado com uma variedade de sensores, tais como acústico, temperatura, sísmico e umidade. Os sensores são alimentados por baterias, geralmente não recarregáveis, o que torna a economia de energia essencial para o prolongamento do tempo de vida da rede.

Para realizar o monitoramento, os sensores são colocados diretamente no ambiente. Nem sempre esses locais são de fácil acesso, como por exemplo, no topo de um vulcão ativo [Werner-Allen et al. 2006] ou nas águas do mar [Dunkels 2007]. Nesses

casos, depois que os sensores são instalados, não há mais como resgatá-los. Como conseqüência, a implantação da rede torna-se uma tarefa difícil, consumidora de tempo, e por isso, de alto custo.

Devido a essas características particulares, a simulação é essencial para o desenvolvimento de aplicações e teste de novos protocolos de RSSF, conforme mencionado por [Egea-Lopez et al. 2005]. Muitos artigos publicados contêm resultados baseados apenas em simulação experimental [Curren 2005]. As vantagens são: menor custo, pois existem vários simuladores de RSSFs de código aberto disponíveis, praticidade em testar o comportamento da rede utilizando diferentes parâmetros, possibilidade de realizar validação de código, testar redes de larga escala e, dependendo das restrições do projeto, a simulação pode até indicar se a implantação da rede é ou não viável.

Em virtude da crescente popularidade das RSSFs, existe um grande número de simuladores disponíveis, como por exemplo NS-2 [Downard 2004], OMNeT++ [Varga et al. 2001], Castalia [Boulis 2009], COOJA [Osterlind et al. 2006], TOSSIM [Levis et al. 2003], AVrora [Titzer et al. 2005], ATEMU [Polley et al. 2004] e OPNET [Chang 1999]. Alguns grupos de simuladores possuem funções muito similares, enquanto outros grupos possuem funções complementares.

Contudo, por causa de seus aspectos singulares e limitações, os modelos existentes nos simuladores de RSSFs podem não levar a uma demonstração completa de tudo que precisa ser verificado [Park et al. 2001]. Vários problemas são encontrados nos simuladores, como modelos super-simplificados para alguns objetivos de simulação, dificuldade para fazer customizações e também em obter protocolos relevantes já existentes [Handziski et al. 2003]. Por causa da dificuldade em se obter um simulador que reúna todas as características pertinentes para um estudo completo de RSSFs, análises mais confiáveis e precisas podem ser obtidas combinando os modelos de diversos simuladores.

Ocorre que os simuladores de RSSFs, em geral, usam um conjunto de especificações de modelagem distintas. Além do mais, faltam formas de integração entre eles. Para cada simulador utilizado, uma nova modelagem é requerida, sendo necessário realizar o procedimento de verificação e validação do modelo que, segundo [Balci 1994], é uma tarefa que consome bastante tempo e custo.

Essa discussão mostra a necessidade de uma forma unificada de modelagem de dados para o desenvolvimento, que permita modelar, simular e implantar aplicações RSSFs. Com o intuito de dar o primeiro passo para alcançar essa situação, o presente trabalho propõe a definição de um modelo de dados extensível para configuração da simulação de RSSF. O objetivo é que o modelo de dados seja um modelo abstrato que possa ser reutilizado por diferentes simuladores. Além disso, é uma solução para evitar o retrabalho de verificação e validação de modelos, facilitando um estudo mais integrado de RSSFs.

As seções subseqüentes estão dispostas da seguinte maneira: a próxima seção mostra como ocorre a simulação nas RSSFs e explica, com exemplos, os tipos de simuladores existentes. Na Seção 3 é apresentado o modelo de dados proposto. A Seção 4 mostra um estudo de caso onde o modelo de dados é utilizado para realizar a simulação. A Seção 5 traz as conclusões.

2. Simulação de RSSFs

Limitações de recursos como memória, processador e bateria, a natureza dinâmica para realizar a implantação dos nós sensores e dificuldades para realizar depuração de código de RSSFs, tornam o projeto e implantação de aplicações de RSSFs uma tarefa que despende um grande esforço [Egea-Lopez et al. 2005]. A identificação de elementos básicos de RSSFs pode ajudar na definição de um campo comum para integração de simuladores. Nas próximas subseções, são descritos os modelos básicos de simulação de RSSFs que acredita-se serem comuns para a maioria dos simuladores.

2.1. Modelagem de Simulação para RSSF

Em [Park et al. 2001] é proposto um ambiente para a simulação de RSSFs, incluindo novos componentes, que não estão presentes em simuladores de redes clássicas, como por exemplo, modelo detalhado de bateria, consumo de energia e ambiente. A maioria dos simuladores de RSSF utiliza arquitetura semelhante a essa. Esse ambiente permite utilizar vários tipos de modelos para simular diferentes cenários.

A Figura 1, inspirada em [Park et al. 2001, Egea-Lopez et al. 2005], mostra uma abordagem de modelagem das RSSFs. São considerados os agentes, que são os geradores de eventos de interesse para os nós; o ambiente, que modela a propagação de eventos que são sensoriados pelos nós; os nós, que representam o equipamento físico de monitoramento; o canal sem fio que caracteriza a propagação de sinais de rádio entre os nós da rede; e o nó usuário ou nó *sink*, um nó especial que recebe dados da rede e os processa.

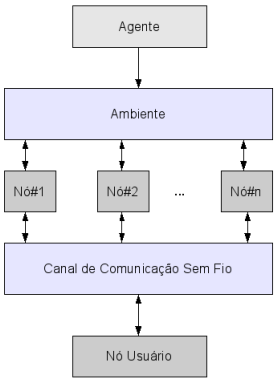

Figura 1. Arquitetura de uma RSSF

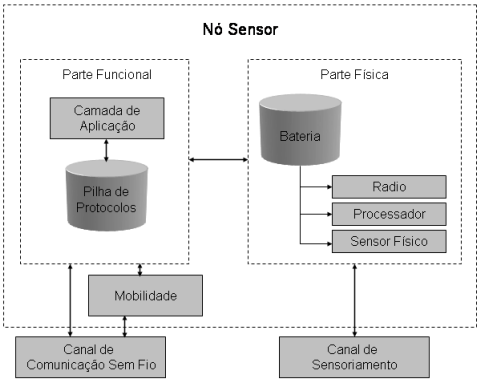

Figura 2. Arquitetura do nó sensor

O modelo de simulação do nó sensor é composto pela parte física do nó e pela parte funcional. A parte física normalmente é composta pelos sensores físicos, modelo de energia, equipamento de rádio e processador. Cada nó sensor, também, é equipado com a parte funcional, que compreende a pilha de protocolos de comunicação que interage com a camada de aplicação. A operação da pilha de protocolos depende da interação com o modelo de energia. Por exemplo, o protocolo de roteamento deve considerar as restricões de bateria para decidir a rota dos pacotes. Além desses componentes, o nó sensor interage com o modelo de mobilidade, que controla sua posição; com o canal de sensoriamento, que representa o ambiente onde se propagam os eventos de interesse; e com o canal de comunicação sem fio, por onde os nós sensores se comunicam. A arquitetura do nó sensor pode ser visualizada na Figura 2, adaptada de [Egea-Lopez et al. 2005].

Objetivando obter um modelo de dados de simulação comum, foram investigados os principais simuladores de RSSFs, cuja descrição é apresentada na próxima subseção.

2.2. Simuladores de RSSFs

Os simuladores de RSSFs podem ser categorizados de acordo com o nível de sua operação: simuladores de aplicação, simuladores de sistema operacional e simuladores de nível de instrução.

Simuladores de aplicação permitem verificar o desempenho de algoritmos e protocolos em uma primeira ordem de validação, antes de implementá-los em uma plataforma específica. Exemplos de simuladores deste tipo são:

- Castalia [Boulis 2009] é um simulador especializado em RSSF baseado no OMNet++[Varga et al. 2001]. Apresenta canal de comunicação sem fio realístico e modelo de rádio baseado em dados reais. Possui também modelos de mobilidade, processo físico, equipamento de sensoriamento e recursos de energia, como bateria, CPU e memória. Castalia é escrito em C++ e possui um arquivo de configuração de simulação, o omnetpp.ini.
- NS-2 [Downard 2004] é um simulador baseado em eventos discretos que oferece grande potencial para simular protocolos de roteamento e medições de consumo de energia com diferentes configurações e topologias. NS-2 foi escrito em C++ e provê uma interface de simulação através da OTcl. NS-2 não tem um modelo pronto para simular RSSFs, é preciso customizá-lo.

Simuladores de sistema operacional executam o código que será executado no nó sensor, ou seja, o código de um SO de redes de sensores é emulado em um PC. Exemplos de simuladores deste tipo são:

- TOSSIM [Levis et al. 2003] é o simulador do TinyOS [Hill et al. 2000]. Ao invés de compilar a aplicação do TinyOS para o nó, usuários podem compilá-la para o ambiente do TOSSIM, que é executado em um PC. TOSSIM permite que usuários depurem, testem e analisem algoritmos em ambientes controlados, podendo simular milhares de nós com diferentes configurações de rede. TOSSIM possui algumas limitações, por exemplo, não captura consumo de energia; outra restrição é que todos os nós devem rodar o mesmo código, então não é possível avaliar alguns tipos de aplicações heterogêneas. TOSSIM foi escrito em C++ e suporta duas linguagens de programação para configuração da simulação: Python e C++.
- COOJA [Osterlind et al. 2006] é o simulador do sistema operacional Contiki [Science 2005], que permite a simulação da aplicação e do SO. COOJA pode simular redes onde os nós sensores podem ser de diferentes plataformas ao mesmo tempo. A linguagem utilizada em seu arquivo de configuração é XML.

Simuladores de nível de instrução executam programas com o mesmo conjunto de instruções que é executado em uma plataforma de sensoriamento, simulando a execução do código detalhadamente, onde são observados os registros da CPU e memória. São exemplos desse tipo:

- ATEMU [Polley et al. 2004] é o emulador do processador AVR e da plataforma de sensoriamento MICA2. Nós sensores podem executar diferentes códigos ao

mesmo tempo. Possui restrições de escalabilidade executando com eficácia aproximadamente até cento e vinte nós. ATEMU é escrito em C e possui um arquivo de especificação de configuração escrito em XML.

- AVrora [Titzer et al. 2005] é um emulador mais eficiente do que o ATEMU e com corretude parecida com o TOSSIM, segundo [Curren 2005]. AVrora executa o código instrução-por-instrução, mas melhora a escalabilidade não realizando sincronização dos nós após cada instrução. É implementado em Java.

3. Modelo de Dados de Simulação para RSSFs

A necessidade de realizar simulação em diversas ferramentas para combinar resultados de simulação complementares, foi a motivação para a criação de um modelo de dados abstrato para configuração de simulação, que possa ser compartilhado por diversos simuladores. O Modelo de Dados de simulação a partir de agora é chamado de MD, e tem a habilidade de ser compartilhado por diversos simuladores.

Ferramentas de simulação usualmente consistem de uma biblioteca básica de simulação e alguma linguagem *script* para descrição da simulação [Egea-Lopez et al. 2005]. Esse fato pode ser observado nos exemplos de simuladores mostrados na seção anterior.

Por esta razão, este trabalho concentra-se na descrição da simulação, criando um arquivo comum de configuração da simulação, onde são colocados os parâmetros relevantes de simulação. MD é um arquivo extensível de configuração de simulação. Cenários de simulação descritos em um arquivo de configuração comum podem ser utilizados em diversos simuladores. Essa proposta é mostrada na Figura 3, onde o MD é convertido, através de regras de transformação, para o arquivo de configuração dos simuladores. A versão atual do MD inclui os simuladores Castalia e TOSSIM. Essa abordagem de integração permite utilizar simuladores que provêm algum tipo de arquivo de configuração de simulação. A linguagem utilizada para construir o MD é XML, pois é uma linguagem para troca de dados, que permite facilmente realizar extensões e alterações, possibilitando que novos simuladores possam ser adicionados futuramente.

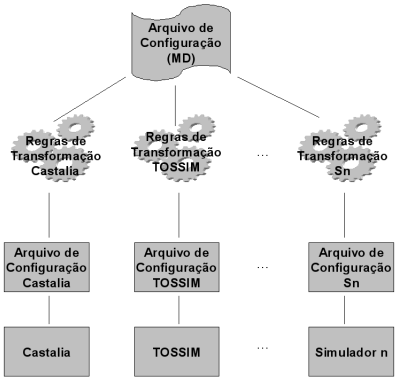

Figura 3. Abordagem utilizando o MD

```
<Topology
        Number_of_nodes="15"
        unit="metros"
        EnvironmentDimensionX="50"
        EnvironmentDimensionY="50"
        EnvironmentDimensionZ="0">
    <Type name="Uniform"/>
</Topology>
```

Figura 4. Descrição da topologia no MD

3.1. Descrição do MD

O MD é um arquivo de configuração da simulação. Isso significa que ele deve conter a descrição completa da simulação. A descrição da simulação é armazenada no MD na

forma de parâmetros e seus respectivos valores.

Os parâmetros de simulação incluídos no MD baseiam-se nos componentes de modelagem de RSSFs e nos simuladores estudados, ambos mostrados na seção anterior. Parâmetros podem ser facilmente incluídos no MD, caso seja necessário. Os simuladores que não suportam alguns parâmetros específicos, podem ignorá-los durante seu processo de modelagem de cenário

O MD inclui dois tipos de parâmetros: requisitos da aplicação, que permanecem fixos do ponto de vista da simulação; e dados específicos de entrada, que variam de acordo com o simulador que se está utilizando, visto que os simuladores têm diferentes dados de entrada. Para cada simulador, parâmetros que ainda não estão definidos no MD, devem ser incluídos. Atualmente, o MD inclui os principais requisitos e objetivos de uma aplicação RSSF e os dados de entrada necessários para os simuladores Castalia e TOSSIM, escolhidos por serem simuladores muito conhecidos e que atuam em diferentes níveis de operação (aplicação e SO), o que traz maior benefício de integração.

Um exemplo de como são inseridos os valores no MD é apresentado na Figura 4. É mostrada a descrição da topologia de uma RSSF com 15 nós sensores, distribuídos de maneira uniforme em um terreno de dimensões 50m por 50m.

Os passos para executar uma simulação utilizando o MD são mostrados a seguir. Esses passos são mostrados de forma ilustrativa na Figura 5.

1. O aquivo é construído de acordo com os requisitos, objetivos e restrições da aplicação. Os valores dos parâmetros que representam estes elementos são preenchidos.
2. Tendo como base o item anterior, o simulador é escolhido e os parâmetros que representam os dados de entrada relacionados a este simulador são preenchidos. O MD então é transformado no arquivo de configuração do simulador escolhido, através de regras de transformação.
3. A simulação é executada.
4. Se os resultados da simulação não forem satisfatórios, o processo retorna ao passo 1. Esse processo é repetido até que se chegue aos resultados esperados, quando diz-se que o MD está validado.
5. Uma vez que o MD está validado, pode-se utilizá-lo como entrada para outro simulador.

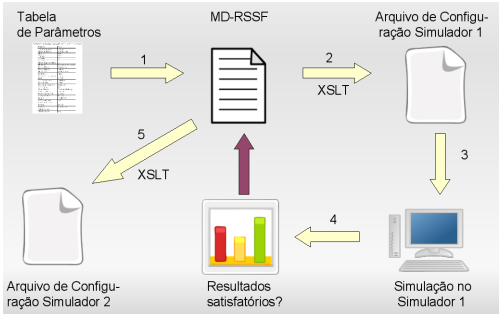

Figura 5. Passos para utilizar o MD

3.2. Transformando o MD em Modelos de Simulação

Quando o MD está completo para o estudo de simulação, ele deve ser transformado no arquivo de configuração do simulador escolhido para que a simulação possa ser realizada. Esse processo é efetuado através de regras de transformação, aplicando-se folhas de estilo da linguagem de transformação do XML, que é a XSLT. A transformação de modelos foi classificada em duas grandes categorias: transformação de modelo para código e transformação de modelo para modelo [Czarnecki and Helsen 2003]. XSLT é um exemplo de transformação de modelo para modelo.

Os arquivos de configuração de cada simulador são programas capazes de realizar simulação que, por sua vez, são arquivos texto, escritos em alguma linguagem *script* ou de alto nível. XSLT permite que arquivos XML sejam transformados em outros arquivos de diferentes formatos. A saída no formato texto é a utilizada no presente trabalho, para criar os arquivos de configuração. Deve ser criada uma folha de estilos para cada simulador incluído no MD.

4. Estudo de Caso

O estudo de caso mostra uma aplicação que utiliza o MD para realizar a simulação. A aplicação consiste em monitorar as condições ambientais durante a fase de envelhecimento da cachaça. Durante seu fluxo produtivo, a cachaça passa por várias etapas, dentre elas, tem-se o envelhecimento. No proceso de envelhecimento, as características sensoriais da cachaça se modificam, aprimorando suas qualidades, o que lhe agrega maior valor [SEBRAE-MG 2001].

4.1. Desenvolvimento da Simulação

O envelhecimento deve ser em lugar fresco, a uma temparatura na faixa de 15^0C a 20^0C, com umidade relativa na faixa de 70% a 90% [da Silva 2008], por um período mínimo de 12 meses. A seguir, são mostradas as restrições e os requisitos para montar o cenário de simulação.

Aplicação: coleta de amostras a cada 15 minutos para verificar as condições ambientais do galpão. Se algum valor coletado estiver fora das especificações, um sinal de alerta deverá ser enviado. Uma vez ao dia um relatório é emitido com a última leitura de temperatura e umidade coletadas.

Métricas de avaliação: *Cobertura*: 70% dos nós da rede ativos. *Tempo de vida*: quantidade maior ou igual a 70% dos nós funcionando.

Topologia: os sensores estão dispostos em forma de *grid*, espaçados um do outro a uma distância aproximada de 2,5 metros na mesma linha, conforme mostrado na Figura 6. De acordo com essa distribuição e dimensões do galpão, são necessários 90 sensores na aplicação. Existe 1 nó *sink*, localizado no centro do campo de sensoriamento.

Comunicação: No Castalia foi utilizado o modelo realístico por não haver muitas interferências externas. No TOSSIM, foi necessário configurar as qualidades das ligações individuais entre os nós, representadas pelos parâmetros *gain* e *noise*.

Com base nos requisitos apresentados, foi montado o MD para a simulação da fase de envelhecimento da cachaça. A Tabela 1 mostra os parâmetros e os valores constantes no MD para configuração da simulação da aplicação proposta.

Tabela 1. Parâmetros e seus valores para a definição do cenário de simulação

Parâmetro	Valor
Tempo de Simulação	12 meses
Número de Sensores	90 sendo 1 sink
Dimensões	8x80x0
Topologia	Grid
xGridSize	30
yGridSize	4
Número de processos físicos	2
Resultados	Cobertura e GastoEnergia
Processo Físico 1	Temperatura
Processo Físico 2	Umidade
Bateria Inicial	36288 (em Joule)
Equipamento	Micaz
Freqüência de Sensoriamento	15 minutos
Rádio	CC2420
Canal de Transmissão	Realístico
Protocolo MAC	TMAC
Protocolo de Roteamento	SimpleTree
MAC-dutyCycle	0.01
Nome Aplicação	WarehouseMonitoring
MaxAppPacketSize	40 (bytes)
PacketHeaderOverhead	8 (bytes)
ConstantDataPayload	8 (bytes)
Resource	2AAbatteries
MAC-listenInterval	1000
MaxNumberOfParents	1
FileGain	topo.txt
FileNoise	meyer-heavy.txt
Attenuation-exp-a	1
OnlyStaticNodes	true

4.2. Resultados Obtidos e Análise

Os parâmetros mostrados na Tabela 1 foram inseridos no MD, o qual foi sendo preenchido da forma como foi mostrado na Figura 4. Em seguida, foi aplicada a folha de estilos para transformação do MD no arquivo de configuração do Castalia e a simulação foi executada. Algumas alterações foram feitas no MD para se chegar aos resultados esperados, como por exemplo, quantidade de nós sensores para se alcançar a cobertura desejada. Uma vez que obteve-se os resultados esperados, o MD está validado. Em seguida, o MD foi utilizado para ser transformado no arquivo de configuração do TOSSIM, através da aplicação da folha de estilos do TOSSIM e a simulação foi realizada no TOSSIM.

Os resultados apresentando o tempo de vida da rede podem ser visualizados na Figura 7, que mostra as saídas do Castalia e TOSSIM. O tempo de vida da rede é maior no TOSSIM. Uma das causas pode ser o protocolo MAC utilizado no TOSSIM, que é diferente do protocolo TMAC utilizado no Castalia.

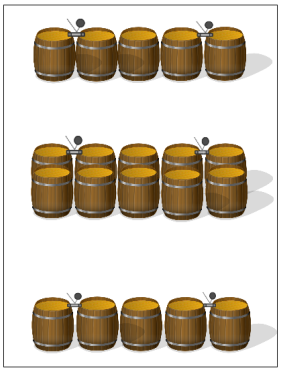

Figura 6. Distribuição dos nós

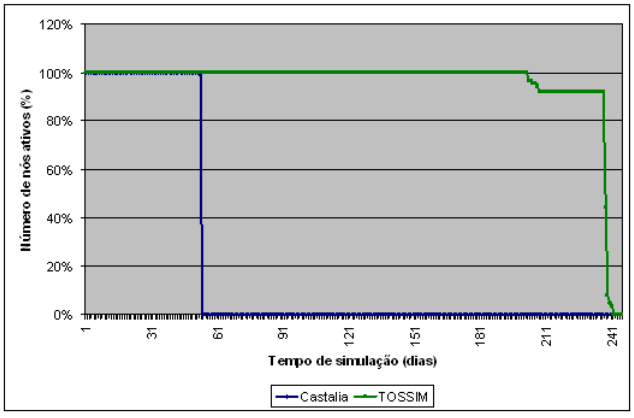

Figura 7. Tempo de vida da rede nas simulações

O objetivo do estudo de caso é analisar a utilização do MD na fase de configuração da simulação, como modelo extensível de dados de simulação. Verifica-se que foi possível utilizar de maneira satisfatória essa abordagem, pois os modelos gerados utilizaram-se de dados que são comuns aos dois simuladores ou próprios de cada simulador, e os resultados de simulação que foram coletados mostram-se coerentes. O modelo permitiu que os simuladores gerassem resultados consistentes de uma maneira prática. Com a configuração de um único arquivo, chegou-se a execução da simulação no Castalia e no TOSSIM.

5. Conclusões

Neste trabalho foi mostrada a necessidade de utilização de mais de um simulador para estudo e análise de aplicações de RSSFs. Verificou-se que a reutilização de modelos de simulação pelos simuladores evitaria o retrabalho de verificação e validação de modelos, que demanda bastante tempo e esforço.

Para alcançar reutilização de modelos, foi definido um modelo de dados para configuração de simulação, denominado MD (Modelo de Dados). O MD permite realizar a modelagem de cenários de simulação, proporcionando uma forma de modelagem abstrata, onde são colocados os parâmetros de configuração da simulação, podendo ser convertido para o arquivo de configuração específico de cada simulador, através de regras de transformação. O MD viabiliza o compartilhamento de modelos de simulação pelos diversos simuladores.

A avaliação do MD foi conduzida por meio de uma aplicação que utilizou o MD para construir seus arquivos de configuração de simulação. A aplicação mostra que essa forma de trabalho é consistente pois foi possível realizar a simulação em dois simuladores utilizando o mesmo modelo de dados, alcançando o objetivo que é a reutilização de modelo.

Referências

Balci, O. (1994). Validation, verification, and testing techniques throughout the life cycle of a simulation study. In *WSC '94: Proceedings of the 26th Conference on Winter Simulation*, pages 215–220, San Diego, CA, USA. Society for Computer Simulation International.

Boulis, A. (2009). Castalia version 2.1 user's manual.

Chang, X. (1999). Network simulations with opnet. In *WSC '99: Proceedings of the 31st Conference on Winter Simulation*, pages 307–314, New York, NY, USA. ACM.

Curren, D. (2005). A survey of simulation in sensor networks. *project report (CS580), University of Binghamton*.

Czarnecki, K. and Helsen, S. (2003). Classification of model transformation approaches. In *Proceedings of the 2nd OOPSLA Workshop on Generative Techniques in the Context of the Model Driven Architecture*. Citeseer.

da Silva, J. M. (2008). *Cachaça: O mais brasileiro dos prazeres*. Anhembi Morumbi, 2a edition.

Downard, I. (2004). Simulating sensor networks in ns-2.

Dunkels, A. (2007). Contiki in the baltic sea.

Egea-Lopez, E., Vales-Alonso, J., Martinez-Sala, A., Pavon-Marino, P., and García-Haro, J. (2005). Simulation tools for wireless sensor networks. In *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'05)*.

Handziski, V., Kopke, A., Karl, H., and Wolisz, A. (2003). A common wireless sensor network architecture. *Proc. 1. GI/ITG Fachgesprach Sensornetze(Technical Report TKN-03-012 of the Telecommunications Networks Group, Technische Universitat Berlin), Technische Universitat Berlin, Berlin, Germany*, pages 10–17.

Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D., and Pister, K. (2000). System architecture directions for networked sensors. In *ASPLOS-IX: Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 93–104, New York, NY, USA. ACM.

Ilyas, M. and Mahgoub, I. (2005). *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press.

Levis, P., Lee, N., Welsh, M., and Culler, D. (2003). TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, pages 126–137. ACM New York, NY, USA.

Osterlind, F., Dunkels, A., Eriksson, J., Finne, N., and Voigt, T. (2006). Cross-level sensor network simulation with cooja. In *Proceedings of the First IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp 2006)*, page 8.

Park, S., Savvides, A., and Srivastava, M. (2001). Simulating networks of wireless sensors. In *Proceedings of the 33nd Conference on Winter Simulation*, pages 1330–1338. IEEE Computer Society Washington, DC, USA.

Polley, J., Blazakis, D., McGee, J., Rusk, D., Baras, J., and Karir, M. (2004). Atemu: A fine-grained sensor network simulator. In *IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*. Citeseer.

Science, S. I. C. (2005). The Contiki Operating System.

SEBRAE-MG (2001). DIAGNÓSTICO DA CACHAÇA DE MINAS GERAIS.

Titzer, B. L., Lee, D. K., and Palsberg, J. (2005). Avrora: scalable sensor network simulation with precise timing. In *IPSN '05: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, page 67, Piscataway, NJ, USA. IEEE Press.

Varga, A. et al. (2001). The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM 2001)*, pages 319–324.

Werner-Allen, G., Lorincz, K., Welsh, M., Marcillo, O., Johnson, J., Ruiz, M., and Lees, J. (2006). Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2):18–25.