Modelos de Publicação/Subscrição na *Internet of Things*

Diogo Lima[1], Dulce Domingos[1], Hugo Miranda[1], Caio Fontana[2]

[1] Lasige, Universidade de Lisboa
dmlima@fc.ul.pt
{dulce, hmiranda}@di.fc.ul.pt

[2]Universidade Federal de São Paulo
Caio.fernando@unifesp.br

Resumo De modo a facilitar o acesso das aplicações aos dados fornecidas pelos sensores, estes são normalmente disponibilizados através de serviços web. Estes serviços web podem ser implementados diretamente nos sensores ou em intermediários. A utilização do paradigma publicador/subscritor na interacção entre as aplicações e os sensores (directamente ou através do intermediário) tem a vantagem de reduzir o número de mensagens recebidas pela aplicação. Ao nível das redes de sensores contribui ainda para o aumento do seu tempo de operação.
No entanto, quando estes serviços são implementados em intermediários, as subscrições são geridas pelos próprios intermediários, não usufruindo das vantagens do modelo publicação/subscrição ao nível da rede de sensores, mesmo que este esteja disponível.
Este artigo apresenta um mecanismo genérico que permite tirar partido das capacidades efetivas dos sensores que serve. O mecanismo beneficia, por um lado, da utilização da linguagem sensorML (para determinar as reais capacidades dos sensores que serve) e por outro de uma interpretação das expressões de subscrição. O artigo apresenta ainda uma prova de conceito por extensão ao middleware MuFFIN.

1 Introdução

O paradigma de *Internet of Things* (IoT) consiste em estabelecer uma ponte entre o mundo real e a sua representação nos sistemas de informação. Com a sua aplicação visa-se que qualquer sistema de informação, desde os painéis de trânsito presentes ao longo das autoestradas a sistemas que lidam com a otimização e reação a diferentes processos de negócio, disponham de informação atualizada sobre sensores espalhados pelo meio ambiente. A comunicação tem, no entanto, de ser devidamente planeada para não ser sobrecarregada por um volume excessivo de dados trocados e de modo a poder poupar as baterias dos sensores. Neste âmbito, é adotado um modelo de publicação/subscrição para concretizar a comunicação entre o sistema de informação e os sensores [1].

Porém, esta interação tem igualmente de ser o mais homogénea possível de modo a facilitar o acesso das aplicações aos dados fornecidos pelos sensores. Tal homogeneidade de interação é conseguida através da implementação de serviços web [2]. A norma *Web Services Notification* (WS-N) [3] estabelece um conjunto

de especificações que definem a forma como os serviços web interagem através de notificações ou eventos. No entanto, quando estes serviços são implementados em intermediários, as subscrições são geridas pelos próprios intermediários, não se usufruindo desta funcionalidade ao nível da rede de sensores, mesmo quando está disponível.

Neste âmbito, o presente artigo apresenta um mecanismo genérico que permite estender intermediários por forma a tirar partido das capacidades efetivas dos sensores que serve. Quando as redes de sensores são capazes de se coordenar e responder a subscrições, o intermediário passa-lhes a tarefa de publicação/subscrição. O mecanismo beneficia, por um lado, da utilização da linguagem sensorML [4] (para determinar as reais capacidades dos sensores que descreve) e, por outro lado, de uma decomposição das expressões de subscrição.

O mecanismo proposto foi aplicado no middleware MuFFIN e testado com uma rede de sensores constituída por cinco bases de sensores *CM4000* e placas de sensores *EM1000*, ambas fabricadas pela empresa *AdvanticSYS*.

Com vista a fazer face aos desafios propostos, o presente artigo encontra-se organizado da seguinte forma: a Secção 2 resume os aspetos previamente estudados e propostos pelo meio académico nas áreas da concretização de intermediários e de algoritmos de publicação/subscrição para redes de sensores. A Secção 3 descreve as nossas soluções para a comunicação entre o intermediário e a rede de sensores. A Secção 4 começa por apresentar o MuFFIN para depois descrever a implementação dos novos componentes introduzidos no sistema e a sua interação. Finalmente, as conclusões e trabalho futuro são elaboradas na Secção 5.

2 Trabalho Relacionado

A grande variedade de sensores disponíveis no mercado, cada um com as suas próprias especificidades, dificulta a sua integração nos sistemas de informação. As normas *Sensor Web Enablement* (SWE) [5] da *Open Geospatial Consortium* (OGC) visam fazer face a esta limitação escondendo os detalhes de comunicação e a heterogeneidade dos protocolos dos sensores das aplicações e permitindo que os sensores fiquem disponíveis via Web através de formatos bem definidos e interfaces de serviços Web. Seguir estas normas revela-se especialmente apropriado em sistemas de informação de grande escala, onde um vasto número de participantes tem de cooperar num conjunto de sensores partilhado, tendo sido já aplicado em situações reais como em sistemas de gestão de catástrofes [6] e de monitorização de tsunamis [7].

O desenho de uma framework SWE pode ainda ser conjugada com o paradigma de publicação/subscrição. A especificação *Sensor Event Service* (SES) [8] permite o acesso aos metadados e à informação recolhida pelos sensores seguindo um modelo *push-based*, onde os clientes definem os dados que querem receber através de filtros definidos nas suas subscrições. A Figura 1 apresenta um exemplo simples da implementação da especificação SES, onde o cliente subscreve dois serviços no intermediário, especificando os filtros dos mesmos. No primeiro caso,

o cliente está interessado em todos os eventos cuja temperatura seja superior a 10°C, enquanto o segundo filtro determina a notificação de eventos onde se regista uma temperatura superior a 20°C e uma humidade inferior a 80%.

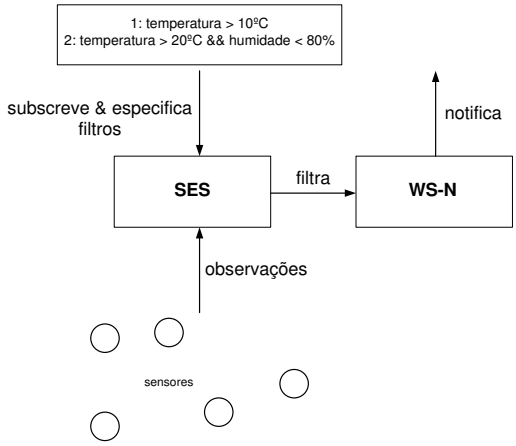

Figura 1. Exemplo da implementação da especificação SES.

Do ponto de vista dos sistemas baseados em conteúdos gerados pelos utilizadores a partir de redes de sensores já existem igualmente algumas concretizações. Por exemplo, o sistema apresentado em [9] pela fabricante IBM aborda um caso de vídeo vigilância num aeroporto, tendo por objetivo o desenvolvimento de outras arquiteturas *open source* para sistemas de vigilância. De forma mais abrangente, o sistema proposto em [10] pela Microsoft vai mais além com a implementação de um intermediário coordenador que permite recolher informação dos mais variados tipos de sensores e dispositivos móveis, disponibilizando-a posteriormente a aplicações clientes que os utilizadores queiram desenvolver. Mais recentemente, o projeto *Xively* [11] liderado pela companhia LogMeIn apresenta-se como o primeiro serviço em nuvem para a IoT, permitindo a gestão de dados proveniente de milhões de sensores e dispositivos móveis em todo o mundo e a simplicidade de desenvolvimento aplicacional para os utilizadores finais no uso desta plataforma. Estes sistemas seguem uma abordagem idêntica à norma SWE, onde as subscrições são geridas internamente pelo intermediário.

As arquiteturas apresentadas em [1, 12] gerem as subscrições ao nível dos *gateways*, que estão mais próximos das redes de sensores, permitindo reduzir o número de mensagens que são trocadas entre os *gateways* e os níveis acima.

No entanto, com a atual proliferação de redes de sensores, é igualmente possível aplicar um modelo de publicação/subscrição ao nível dos sensores. Os algoritmos de publicação/subscrição dependem em grande medida dos algoritmos de retransmissão de mensagens. Por exemplo, o algoritmo proposto em [13] permite suportar eficientemente um modelo de publicação/subscrição em redes não in-

fraestruturadas através, por um lado, da disseminação de subscrições para criar rotas de origem e, por outro, da análise da força de sinal das mensagens recebidas para otimizar essas mesmas rotas. Outros algoritmos de disseminação de mensagens podem ser igualmente usados para serviços de publicação/subscrição em redes de sensores, como por exemplo o algoritmo apresentado em [14], onde cada nó possui uma estimativa geográfica da sua vizinhança de forma a selecionar um conjunto de vizinhos que deverá retransmitir as mensagens, criando assim um mecanismo que permite assegurar a propagação das mensagens e a conectividade da rede. Outras abordagens para a seleção de nós retransmissores podem ser igualmente encontradas em [15, 16]. Neste artigo propomos a extensão dos intermediários de forma a que estes possam usufruir das protocolos de publicação/subscrição das redes de sensores, quando estão disponíveis.

3 Modelos de Publicação/Subscrição em redes de sensores

Tipicamente os intermediários seguem a abordagem da norma SWE tal como no exemplo apresentado na figura 1. O intermediário recebe as subscrições e, face às observações que recebe dos sensores, notifica os interessados (consumidores). De acordo com esta arquitetura, todas as observações são enviadas para o intermediário mesmo que não haja nenhum consumidor interessado.

O nosso objectivo passa assim por utilizar as capacidades das redes de sensores para a publicação/subscrição quando estas estão disponíveis, de modo a reduzir a troca de mensagens ao nível da rede de sensores, aumentando o tempo de vida dos sensores. Adicionalmente, estamos igualmente a aliviar a carga num potencial ponto de estrangulamento no intermediário. Nesta secção apresentamos três propostas de extensão dos intermediários de forma a que estes passem as subscrições para as redes de sensores, distribuindo o poder computacional e as mensagens entre intermediário e sensores. O modelo de publicação/subscrição seguido neste trabalho é baseado em conteúdos.

3.1 Subscrições para os sensores

O evoluir das tecnologias sem fios e do poder computacional dos sensores nos últimos anos permitiu o estabelecimento de redes mais complexas ao nível da gestão e da coordenação dos participantes, suportando modelos de comunicação como a publicação/subscrição. O objetivo desta primeira abordagem passa por tirar partido deste princípio na sua forma mais simples, difundindo pela rede de sensores filtros para os quais os dispositivos irão gerar notificações. Deste modo, a tarefa de subscrição e especificação de filtros abordada na secção anterior é pela primeira vez transferida do intermediário para a rede de sensores, como é apresentado na Fig. 2.

Porém, antes que o intermediário seja capaz de delegar as subscrições para uma rede de sensores, é necessário que este conheça as capacidades da mesma e

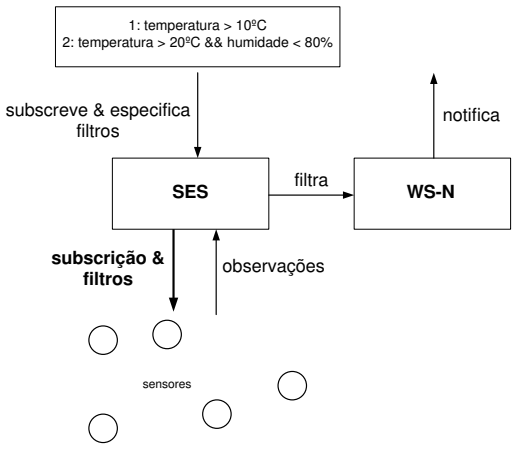

Figura 2. Delegação da responsabilidade de pub/sub para a rede de sensores

dos dispositivos que a integram. Tal é conseguido através da utilização da linguagem *Sensor Model Language* (SensorML) [4]. Esta linguagem especifica um modelo e uma codificação em XML que permite a descrição das características dos sensores, bem como de processos associados a esses sensores. Por ser independente de qualquer domínio, o SensorML é genérico fazendo com que a informação possa ser descrita de diversas formas, abstraindo das interfaces dos sensores.

Considerando o exemplo da Fig. 2 e considerando que a temperatura e a humidade são obtidas de duas redes de sensores independentes, para a rede de sensores de temperatura são enviadas duas subscrições ("temperatura $>$ 10ºC" e "temperatura $>$ 20ºC") e para a rede de sensores de humidade é enviada uma subscrição ("humidade $<$ 80%"). Com este exemplo mostramos que o intermediário terá de continuar a gerir as subscrições, já que é a este nível que o filtro ('temperatura $>$ 20ºC && humidade $<$ 80% ") tem de ser aplicado.

Seguindo esta abordagem, a rede de sensores tem de assumir a responsabilidade de guardar e gerir todas as subscrições ativas, sem que o intermediário possa ser libertado da filtragem dos dados devido ao caso onde tem de comparar filtros sobre observações de redes diferentes.

3.2 Composição de subscrições

De modo a fazer face à limitação da abordagem anterior, as funcionalidades do SES no intermediário têm de ser estendidas de modo a efetuarem a composição das subscrições antes de as passar para as redes de sensores, e assim reduzir a quantidade de subscrições que as redes de sensores têm de gerir. Para tal, é necessário criar uma lista de filtros ativos para cada uma das redes de sensores disponíveis. Posteriormente, ao receber uma nova subscrição, o intermediário percorre a lista e, fazendo a disjunção dos filtros, verifica se os novos filtros

não estão contidos já noutros existentes, ou vice-versa. Por exemplo, no caso das subscrições consideradas anteriormente, o tópico de "temperatura>20" está contido no tópico "temperatura>10", logo é apenas necessário enviar este último à rede de sensores para ambos serem cobertos.

Do ponto de vista do funcionamento do SES na fase de resposta às observações registadas, o comportamento é idêntico ao da abordagem anterior, aplicando um filtro para agregar a informação vinda de origens diferentes.

3.3 Decomposição de filtros

Finalmente, nesta última abordagem, consideramos o caso onde as redes de sensores têm a capacidade para gerir e responder a subscrições contendo um ou mais filtros, ou seja, no caso em que uma subscrição como "temperatura>20 && humidade<80%" pode ser passada à rede no todo.

Os filtros de temperatura e humidade têm sido até agora tratados num cenário onde estas observações são efetuadas em redes distintas, portanto os filtros davam origem a duas subscrições que eram posteriormente combinadas no intermediário. No entanto, pode dar-se o caso de todos os filtros poderem ser tratados numa mesma rede. Para tal, é necessário que o intermediário faça a decomposição da expressão de subscrição e identifique se os filtros podem ser passados no todo a uma rede de sensores ou se devem ser decompostos para serem enviados para várias redes.

4 Implementação

A implementação do mecanismo proposto na secção 3 foi realizada utilizando o middleware MuFFIN (de *Middleware Framework For the Internet of thiNgs*). O nosso protótipo inclui o MuFFIN em conjunto com cinco bases de sensores *CM4000* e placas de sensores *EM1000*, ambos fabricados pela empresa *AdvanticSYS*.

4.1 Middleware MuFFIN

O middleware MuFFIN [17] permite criar a ponte de comunicação entre as aplicações de alto nível e as redes de sensores e, para que esta seja possível, o MuFFIN disponibiliza uma interface de serviços web que torna todos os dispositivos acessíveis através de uma interface comum. O MuFFIN tem uma arquitetura idêntica à proposta pela norma SWE. O componente *DFN-Engine* gere as subscrições e os filtros. Os filtros são concretizados através de módulos, que podem ser compostos. A ligação entre o MuFFIN e as redes de sensores é feita através de *gateways*, os *ThingsGateway*(na concretização deste protótipo foi desenvolvido um *ThingsGateway* para os sensores *CM4000*). No caso do MuFFIN, antes de efetuarem as subscrições, os clientes têm de criar o módulo que concretiza o filtro da sua subscrição e associá-lo ao *ThingsGateway* respetivo. Por fim, o componente *Subscriptions* efetua as notificações para o clientes. A Figura 3 apresenta uma versão simplificada da estrutura interna do MuFFIN.

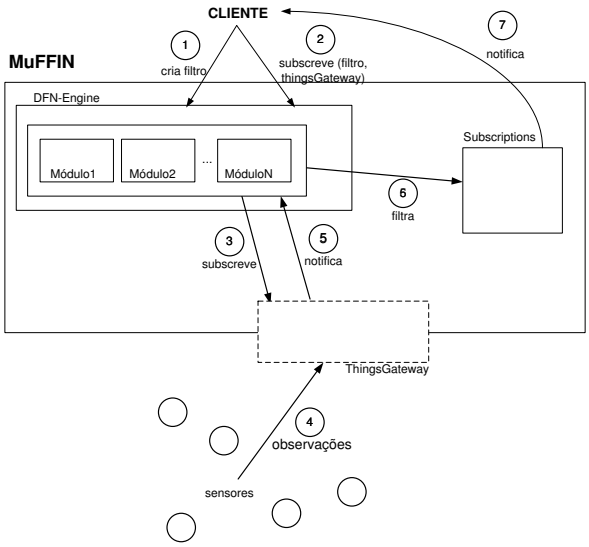

Figura 3. Estrutura do MuFFIN para o tratamento de subscrições e notificações

4.2 Extensão ao MuFFIN

Os clientes do MuFFIN têm de invocar dois serviços para efetuarem uma subscrição, já que previamente têm de instalar o módulo que concretiza o filtro. Para concretizar as subscrições de acordo com a norma WS-N, foi criado o serviço "subscribe" que efetua ambos os passos e adicionada uma nova camada no DFN-engine, designada por *Subscription Decider*.

Quando o *Subscription Decider* recebe um pedido de subscrição, se uma rede conseguir assumir a responsabilidade de uma subscrição, então este componente envia o pedido para o respetivo *ThingsGateway* da rede escolhida. Adicionalmente, este componente desenvolve um módulo de filtragem de dados a ser instalado no MuFFIN. Uma vez que os módulos são implementados na linguagem Java, e usando o *XML Schema Definition* (xsd) da norma WS-N, é feita uma tradução da subscrição para Java. Posteriormente, o módulo desenvolvido é instalado no componente *DFN-Engine* e associado ao(s) *ThingsGateway* que contenha(m) as observações visadas, tal como era efetuado na anterior versão do MuFFIN, presente em [17].

A Figura 4 apresenta a nova versão da estrutura interna do middleware MuFFIN, e como é que o novo componente *Subscription Decider* interage com os restantes.

O *Subscription Decider* determina as capacidades das redes disponíveis através da informação que está guardada na base de dados em SensorML. Esta informação é inserida quando os *ThingsGateway* são instalados no MuFFIN. Desta forma, o serviço *"installThingGateway"* foi alterado de forma a incluir mais um parâmetro, que corresponde ao ficheiro SensorML de descrição da rede que fica

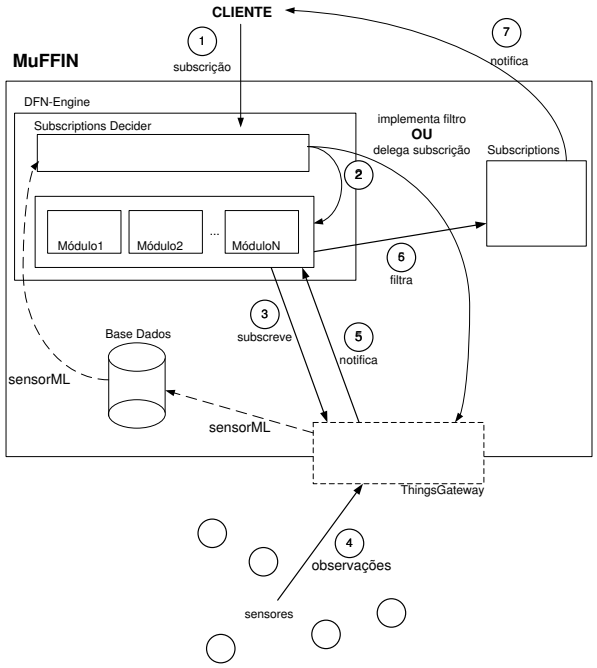

Figura 4. Nova estrutura do MuFFIN para o tratamento de subscrições e notificações.

guardado em base de dados. A Figura 5 apresenta um excerto de um ficheiro sensorML, onde estão definidas as principais características e capacidades da rede de sensores utilizada.

As subscrições são efetuadas ao *Subscription Decider* de acordo com a norma WS-N. A Figura 6 apresenta o exemplo do pedido de subscrição para o filtro de "temperatura $>$ 20ºC && humidade $<$ 80%". O *Subscription Decider* analisa o pedido decompondo-o numa expressão lógica, de modo a determinar se a subscrição pode passar no todo ou em parte para uma ou mais redes de sensores. Tal depende da capacidade da rede de sensores em assegurar a publicação/subscrição. No caso da Figura 5 este facto é especificado em primeiro lugar pelo valor booleano verdadeiro para o elemento "Pub/Sub", e posteriormente pela capacidade em respeitar os tópicos da subscrição, neste caso "Temperatura" e "Humidade".

4.3 Rede de sensores

A implementação da rede de sensores foi realizada através da utilização de cinco bases de sensores *CM4000*, com placas de sensores *EM1000* que permitem a captação de temperatura, humidade, luminosidade e aceleração. De forma a recolher os dados no computador foi usada a interface USB *UD1000*, sendo que todos estes dispositivos são fabricados pela empresa *AdvanticSYS*.

```
...
<sml:System>
     <gml:name>AdvanticSys Sensor Network</gml:name>
      <sml:characteristics>
               <swe:DataRecord>
                    <swe:field name="Temperatura">
                         <swe:Boolean>
                              <swe:value>true</swe:value>
                         </swe:Boolean>
                    </swe:field>
                    <swe:field name="Humidade">
                         <swe:Boolean>
                              <swe:value>true</swe:value>
                         </swe:Boolean>
                    </swe:field>
                    <swe:field name="Sensors">
                    <swe:Quantity>
                              <swe:value>5</swe:value>
                    </swe:Quantity>
                    </swe:field>
               </swe:DataRecord>
     </sml:characteristics>
     <sml:capabilities>
               <swe:DataRecord>
                    <swe:field name="Pub/Sub">
                         <swe:Boolean>
                              <swe:value>True</swe:value>
                    </swe:Boolean>
                    </swe:field>
               </swe:DataRecord>
     </sml:capabilities>
</sml:System>
...
```

Figura 5. Excerto do ficheiro SensorML descrevendo as principais características e capacidades da rede de sensores utilizada.

O componente *Subscription Decider* faz as subscrições para o *ThingsGateway* através do pedido de subscrição enviado pelos clientes. No caso do nosso protótipo, onde o tamanho máximo das mensagens que os sensores conseguem trocar é de 28 bytes, é necessário estabelecer um formato de mensagem que permita traduzir esse pedido de subscrição numa mensagem que o *gateway* e a sua rede de sensores consigam trocar e interpretar. Para tal, as subscrições são decompostas com recurso à linguagem *XPath* [18] para poder transformá-las em mensagens próprias entre *gateway* e sensores. Neste sentido, é apresentado na Fig. 7 o formato de mensagem genérico a ser difundido pela rede, sendo cada campo descrito de seguida:

- *Opcode*(1 byte): identificador do tipo de mensagem enviada. Assume o valor "01" se se tratar de uma subscrição, o valor "02" se for uma notificação
- *Número de tópicos*(1 byte): valor que permite contabilizar o número de ocorrências do triplo *tipo de dados, operador, valor* que constitui cada tópico na mensagem.
- *Tipo de dados*(1 byte): valor que permite descrever qual o conteúdo que está a ser enviado. Por exemplo, na representação utilizada, o valor "01" representa o tópico de "temperatura", "02" representa o tópico de "humidade", "03" representa a luminosidade, etc.
- *Operador*(1 byte): permite expressar o operador matemático que relaciona o tipo de dados com a seu valor pretendido. Por exemplo, o operador "00" representa a igualdade "=", o operador "01" significa a inferioridade "<", "02"

```
...
</s:Header>
<s:Body>
      <wsnt:Subscribe>
            <wsnt:ConsumerReference>
                  <wsa:Address>
                        http://www.examplo.pt/ReceberNotificacao
                  </wsa:Address>
            </wsnt:ConsumerReference>
            <wsnt:Filter>
                  <wsnt:TopicExpression Dialect="http://docs.oasis-open.org/wsn/t-
1/TopicExpression/Simple">
                        Temperatura
                  </wsnt:TopicExpression>
                  <wsnt:TopicExpression Dialect="http://docs.oasis-open.org/wsn/t-
1/TopicExpression/Simple">
                        Humidade
                  </wsnt:TopicExpression>
                  <wsnt:MessageContent
                        Dialect="http://www.w3.org/TR/1999/REC-xpath-19991116">
                        boolean(Temperatura>"15" && Humidade<"80")
                  </wsnt:MessageContent>
            </wsnt:Filter>
      </wsnt:Subscribe>
</s:Body>
```

Figura 6. Exemplo do conteúdo de uma mensagem de subscrição

significa a superioridade "$>$" enquanto os valores de "03" e "04" representam as relações de "$\leq$" e "$\geq$" respetivamente.

– *Valor*(2 bytes): valor real associado ao tipo de dados anterior que é desejado ou foi medido, e que constitui uma subscrição de um tópico ou uma notificação do mesmo.

A mensagem de subscrição entre os sensores é assim especificada através do conjunto {*tipo de dados, operador, valor*} que constitui um termo, podendo uma mensagem ter um ou mais termos, que são combinados entre si através da conjunção "$\wedge$". Desse modo, o pedido de subscrição apresentado na Fig. 6 iria assim resultar na seguinte mensagem de subscrição a ser difundida pela rede:
01 | 02 | 01 02 00 14 | 02 01 00 50.

Do ponto de vista da implementação dos protocolos de disseminação de mensagens utilizados, a fase difusão de tópicos ou subscrições do *gateway* para a rede foi estabelecida através de um simples *flooding*, visto termos poucos participantes na rede. Na fase inversa da propagação das mensagens de notificação até ao *gateway*, foi utilizado o módulo *CollectionC* disponibilizado no sistema operativo *TinyOS* [19] que já cria uma árvore de encaminhamento até a um nó raiz, neste caso definido para ser o *gateway* da rede. Para a implementação do *gateway* foi instalada a aplicação "BaseStation", igualmente disponibilizada pelo sistema operativo *TinyOS*, no interface USB *UD1000* anteriormente referido.

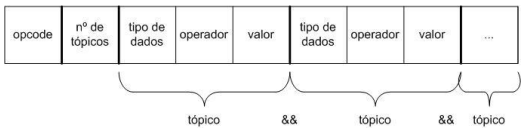

Figura 7. Formato de mensagem de subscrições e notificações usado pelos sensores

5 Conclusão

O paradigma de IoT tem conhecido uma importante evolução à medida que existe uma necessidade cada vez maior de aceder à informação disponibilizada pelas redes de sensores espalhadas pelo meio ambiente. De modo a facilitar o acesso das aplicações a essa informação, esta é normalmente disponibilizada através de serviços web. Estes serviços web podem ser implementados diretamente nos sensores ou em intermediários. No entanto, quando são implementados em intermediários, as subscrições são geridas pelos próprios intermediários, não usufruindo desta funcionalidade ao nível da rede de sensores, mesmo quando está disponível. Ao longo deste artigo apresentámos um mecanismo genérico que permite estender estas plataformas de intermediário por forma a tirar partido das capacidades efetivas dos sensores que serve. O mecanismo beneficia, por um lado, da utilização da linguagem sensorML (para determinar as reais capacidades dos sensores que serve) e por outro de uma decomposição das expressões de subscrição. De seguida, elaborámos uma prova de conceito estendendo o intermediário MuFFIN, onde foi adicionada a possibilidade das subscrições serem ou não passadas para a rede de sensores, podendo ainda ser delegadas no seu todo ou em parte.

Como trabalho futuro projetamos a integração dos componentes apresentados com o componente que gere as ontologias do MuFFIN para a definição dos filtros. Em relação à rede de sensores seria interessante possibilitar a junção de mensagens subscrição com mensagens cancelamento de subscrições. Nos casos onde surge uma nova subscrição que inclui outra anterior, a mensagem de subscição a ser difundida pela rede poderia igualmente levar o cancelamento da outra, de modo a poupar o número de mensagens.

Agradecimentos

Agradecemos ao Rui Pires e ao Professor Francisco Martins pelos esclarecimentos prestados sobre o MuFFIN. Este trabalho foi parcialmente suportado pela FCT através do projecto PATI (PTDC/EIAEIA/103751/2008) e do financiamento multi anual LaSIGE (U I 408 - 2011-2013, refª PEst-OE/EEI/UI0408/2011).

Referências

1. Liu, Y., Plale, B.: Survey of publish subscribe event systems. Technical report, Indiana University (2003)
2. Zeng, D., Guo, S., Cheng, Z.: The web of things: A survey (invited paper). Journal of Communications **6**(6) (2011)
3. OASIS: Web services notification (ws-notification) version 1.3. oasis. URL: https://www.oasis-open.org/committees/wsn (2006)
4. Botts, M.: Ogc implementation specification 07-000: Opengis sensor model language (sensorml). In: Open Geospatial Consortium: Wayland, MA, USA, 2007

5. Botts, M., Percivall, G., Reed, C., Davidson, J.: Ogc® sensor web enablement: Overview and high level architecture. In Nittel, S., Labrinidis, A., Stefanidis, A., eds.: GeoSensor Networks. Volume 4540 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2008) 175–190
6. Stasch, C., Walkowski, A.C., Jirka, S.: A Geosensor Network Architecture for Disaster Management based on Open Standards. In Ehlers, M., Behncke, K., Gerstengabe, F.W., Hillen, F., Koppers, L., Stroink, L., Wächter, J., eds.: Digital Earth Summit on Geoinformatics 2008: Tools for Climate Change Research. (2008) 54–59
7. Raape, U., Teßmann, S., Wytzisk, A., Steinmetz, T., Wnuk, M., Hunold, M., Strobl, C., Stasch, C., Walkowski, A., Meyer, O., Jirka, S.: Decision support for tsunami early warning in indonesia: The role of ogc standards. In Konecny, M., Zlatanova, S., Bandrova, T.L., eds.: Geographic Information and Cartography for Risk and Crisis Management. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg (2010) 233–247
8. Echterhoff, J., Everding, T.: OpenGIS Sensor Event Service Interface Specification. Technical Report OGC 08-133, http://portal.opengeospatial.org/files/?artifact_id=29576 (October 2008)
9. Shu, C.F., Hampapur, A., Lu, M., Brown, L., Connell, J., Senior, A., Tian, Y.: Ibm smart surveillance system (s3): a open and extensible framework for event based surveillance. In: Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference on. (2005) 318–323
10. Grosky, W., Kansal, A., Nath, S., Liu, J., Zhao, F.: Senseweb: An infrastructure for shared sensing. MultiMedia, IEEE **14**(4) (2007) 8–13
11. https://xively.com/
12. Leguay, J., Lopez-Ramos, M., Jean-Marie, K., Conan, V.: Service oriented architecture for heterogeneous and dynamic sensor networks. In: Proceedings of the second international conference on Distributed event-based systems. DEBS '08, New York, NY, USA, ACM (2008) 309–312
13. Schnitzer, S., Miranda, H., Koldehofe, B.: Content routing algorithms to support publish/subscribe in mobile ad hoc networks. In: Local Computer Networks Workshops (LCN Workshops), 2012 IEEE 37th Conference on. (2012) 1053–1060
14. Ludovico, J., Miranda, H.: Um algoritmo de difusão baseado na troca de resumos. In Atas do INFORUM 2011 - Terceiro Simpósio de Informática. Raul Barbosa and Luís Caires (eds.)Dep. de Engenharia Informática da Universidade de Coimbra. Coimbra, Portugal. (2011) 294–305
15. Miranda, H., Leggio, S., Rodrigues, L., Raatikainen, K.: A power-aware broadcasting algorithm. In: Personal, Indoor and Mobile Radio Communications, 2006 IEEE 17th International Symposium on. (2006) 1–5
16. Meisel, M., Pappas, V., Zhang, L.: Listen first, broadcast later: Topology-agnostic forwarding under high dynamics. Technical report, Technical Report 100021, UCLA Computer Science Department (2010)
17. Valente, B., Martins, F.: A middleware framework for the internet of things. In: Proceedings of AFIN 2011 - The 3rd International Conference on Advances in Future Internet. (2011) 139–144
18. http://www.w3schools.com/xpath/default.asp
19. http://www.tinyos.net/