Noname manuscript No.
(will be inserted by the editor)

Enhancing Context Data Distribution for the Internet of Things using QoC-Awareness and Attribute-Based Access Control

Léon Lim(1), Pierrick Marie(2), Denis Conan(1), Sophie Chabridon(1), Thierry Desprats(2), Atif Manzoor(1)

the date of receipt and acceptance should be inserted later

Abstract The Internet of Things (IoT) enables producers of context data like sensors to interact with remote consumers of context data like smart pervasive applications in an entirely decoupled way. However, two important issues are faced by context data distribution, namely providing context information with a sufficient level of quality—i.e. quality of context, QoC—while preserving the privacy of context owners. This article presents the solutions provided by the INCOME middleware framework for addressing these two potentially contradictory issues while hiding the complexity of context data distribution in heterogeneous and large-scale environments. Context producers and consumers not only express their needs in context contracts but also the guarantees they are ready to fulfil. These contracts are then translated into advertisement and subscription filters to determine how to distribute context data. Our experiments on a first open source prototype show that QoC-based filtering and privacy protection using attributed-based access control can be performed at a reasonable cost.

Keywords IoT, Middleware, Distributed Event-Based Systems, Quality of Context, Privacy, Access Control Policy, Attribute-Based Access Control.

1 Introduction

The Internet of Things (IoT) paradigm allows the design of new applications for instance in the domains of smart cities, smart homes, or smart transportation. In addition to the communication standards proposed by the IETF (6LoWPAN, CoAP [25], etc.) or the IEEE (1888-2014), the development of these applications will benefit from new middleware solutions. Since many of the smart things interact by pushing events, the publish/subscribe communication model [11] that is at the root of distributed event-based systems (DEBS) [20] is an important

(1) Institut Mines-Télécom/Télécom SudParis, UMR CNRS 5157 SAMOVAR, 9 rue Charles Fourier, 91011 Évry, France, email: firstname.lastname@telecom-sudparis.eu
This work was performed while Atif Manzoor was on stay at Télécom SudParis
(2) Université de Toulouse, IRIT UMR 5505, France, email: firstname.lastname@irit.fr

enabler: Its interaction pattern decouples in space and time the things that produce events from applications that consume these events. In addition, semi-structured data models *à la* XML are preferred to structured data models that organise notifications as records of pairs (attribute name, value). This allows to stay as open as possible to inter-operate with approaches such as sensors as a service and with information processing approaches such as machine learning with ontologies. The first contribution that is presented is the DEBS infrastructure of the INCOME framework for the distribution of context data from the IoT. Existing P2P-based DEBS solutions assume either subject-based filtering or content-based filtering with structured data models [15]. They do not consider semi-structured data models.

The IoT enables the collection of a large variety of context data, coming from local ambient sensors and remote sources. In this work, "context is any information that can be used to characterize the situation of an entity" (a person, place, or object) [9]. Context data represent either useful raw data that have been directly acquired by a context manager through sensing, observing, or measuring some facts, or they represent data that have been processed, organized, structured or presented so as to make them meaningful and useful. These context data can then be exploited by pervasive applications to detect the current situation of the users and provide them with the relevant services corresponding to their precise needs. However, context data are known to be imperfect and uncertain by nature [14]. One way to limit the impact of uncertainty is to manipulate additional knowledge associated with context data in the form of meta-data that represent the Quality of Context (QoC) [6] through criteria such as freshness, precision or correctness.

Another aspect of the IoT requires particular attention and concerns the privacy of users. As the IoT is gaining momentum, the threats on users' privacy appear more clearly [13]. We have shown in [7] that QoC and privacy are closely related and must be addressed together in order to find a workable solution. This article extends our previous work dealing with QoC-aware context contracts [19] and proposes to define attribute-based access control policies for delivering context data to consumers depending on their intended use and their privacy guarantees.

To summarize, the contributions of this article concern two main directions. We first propose an open and flexible content-based publish/subscribe framework manipulating semi-structured data models. Secondly, the originality of our solution is to consider both the privacy expectations of context producers in order to not disclose their private information and the requirements of context consumers for receiving context data of a sufficient quality.

The structure of the article is as follows. Section 2 describes our INCOME framework. Section 3 describes the necessary elements for implementing a motivation scenario in a smart city with the INCOME framework. Section 4 presents an evaluation of the cost of our solution. Section 5 discusses some related works and Section 6 concludes the article and gives some perspectives.

2 The INCOME framework

The logical view of the INCOME framework in Figure 1 shows the different information flows (context data, privacy and QoC meta-data flows), and the functions involved in context management. Context producers—i.e. software

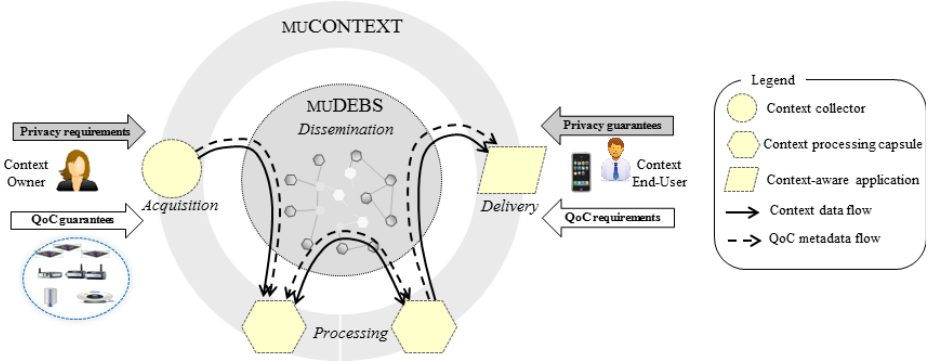

Fig. 1 INCOME Logical Architecture

entities or persons on which context data are collected—express in context contracts their privacy requirements but also the guarantees concerning the QoC they are ready to provide. Symmetrically, context consumers—i.e. applications or intermediate processing software entities—specify their QoC requirements and the guarantees they are ready to fulfil to respect the privacy of the context owners. These contracts are then translated into advertisement and subscription filters to determine how to distribute context data.

In the INCOME framework, context data distribution is the responsibility of muDEBS[1]. The other main framework of INCOME that is presented in this article is muContext[2] that manages context entities with their associated data models. muDEBS and muContext are developed as an open source software and are publicly available[3].

We present the main functionalities of muDEBS in Section 2.1 and the way it enforces privacy protection in Section 2.2. We then present QoC management that is part of muContext in Section 2.3.

2.1 Context data distribution

muDEBS is a framework offering content-based routing and is responsible for distributing context data. It is generic in the sense that it is data-model agnostic (data models are manipulated in muContext). The interface of muDEBS is the one of a distributed event-based system. Producers declare the kind of data they are willing to produce in "advertisements". Then, they publish these data in what are called "notifications" or "publications". Similarly, consumers declare the information they want to receive and react to notifications delivered to them through "subscriptions".

muDEBS organises an overlay network of brokers that connect producers and consumers. Producers and consumers are collectively called clients. A client is connected to only one broker at a time that is called the access broker. muDEBS

[1] muDEBS stands for MUltiscale Distributed Event-Based System.

[2] muContext stands for MUltiscale Context data manager.

[3] `https://fusionforge.int-evry.fr/www/mudebs/` and `https://fusionforge.int-evry.fr/www/mucontext/`

assumes that the data are semi-structured records serialised as XML documents. The rationale for the choice of XML is its openness to allow approaches such as sensors as a service or ontology-based inference engines that often bring to play XML languages. It follows that we use XPath to navigate through XML data as standardised by the W3C.

An advertisement expresses the set of publications that a producer is allowed to publish. A subscription expresses the set of publications that a consumer wants to consume. In practice, in our research prototype, a filter is a function, written in JavaScript, which evaluates XPath expressions and returns `false` when the notification does not match the filter, or returns `true` when it matches the filter. The rationale for the choice of JavaScript is its flexibility as a scripting language for research experimentations (at the expense of execution performance).

Operational modes. muDEBS provides two main operational modes for subscriptions and advertisements: (1) global subscription with local advertisement, and (2) local subscription with global advertisement. Of course, the other two modes are possible but are less used. In mode (1), advertisements are kept local to the access brokers of the producers, and brokers forward subscriptions to their neighbouring brokers according to a simple routing mechanism. The advantage of this mode is to minimise the notification traffic. In mode (2), subscriptions remain local to the access broker of the consumers, and advertisements are broadcast in the overlay network of brokers. This operational mode is suitable when consumers are very volatile and producers are more stable.

Figure 2 shows the operations in the first mode. A producer advertises a *local* filter F (depicted by a dotted segment), uniquely identified by id, by calling the `advertise` operation of its access broker B_1. B_1 then registers the advertisement filter. Later on, a consumer registers a *global* subscription filter F' (depicted by a solid segment) to its access broker B_4. The global subscription filter F' identified by id' is then installed on every broker building a spanning DAG (Directed Acyclic Graph) directed towards the consumer. When the producer publishes a notification n in the context of the filter F identified by id, the access broker filters out n if it does not match F. Otherwise the access broker of the producer evaluates all the subscription filters it is aware of. When n matches a subscription filter, let say F', the notification n is forwarded towards the subscriber of F' via the access broker of the subscriber, which is eventually notified of n.

The second operational mode is illustrated in Figure 3. A producer advertises a *global* filter F, identified by id. B_1 then registers the advertisement filter. This global advertisement filter is then propagated to all the brokers. Thereafter, a consumer registers a *local* subscription filter F', identified by id', to its access broker. When the producer publishes a notification n', it is forwarded to all the neighbouring brokers if it matches F. Each broker then evaluates subscription filters of local consumers and notifies them when there is a match.

2.2 Protecting privacy with attribute-based access control

Our approach focuses on authorisation mechanisms to check the counterparts that data consumers are ready to fulfil with respect to the privacy of the data owner. It considers the intended use that consumers will make of the data before

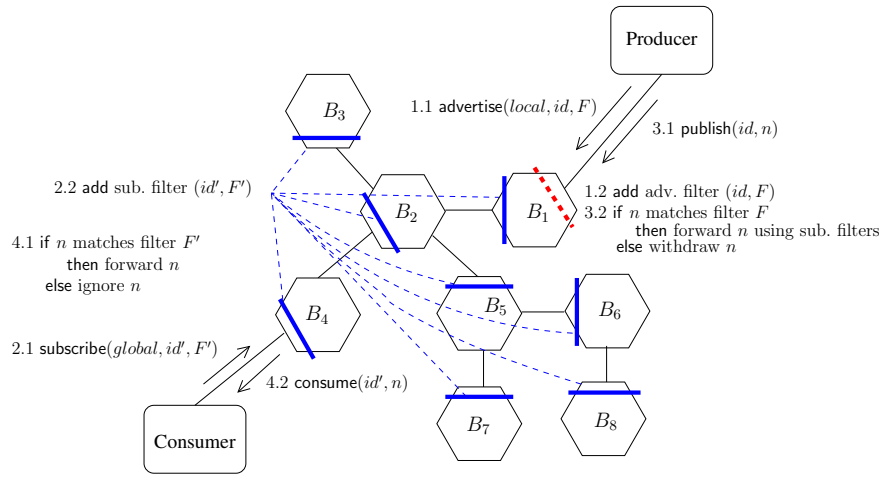

Fig. 2 Local advertisement and global subscription—Mode 1

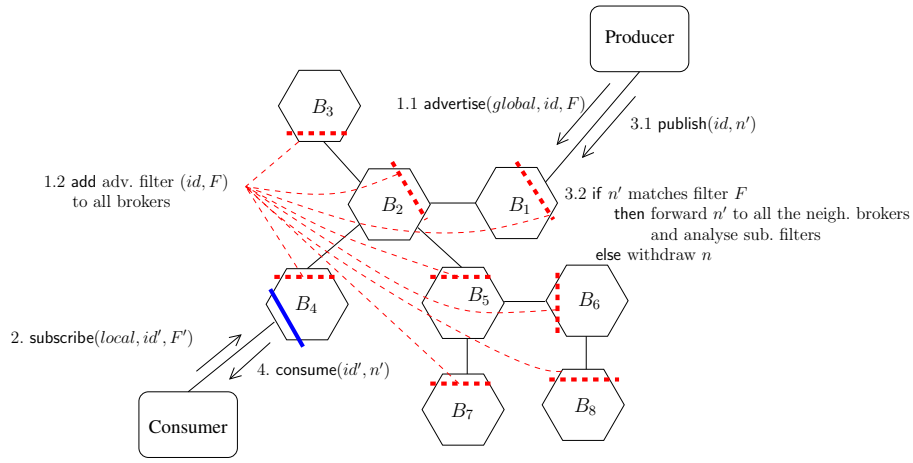

Fig. 3 Global advertisement and local subscription—Mode 2

granting them access. It is complementary to security solutions proposed for publish/subscribe systems [10]. In the case that the brokers are not trustworthy, security solutions such as [1] or [21] can be used to encrypt filters and also events inside the overlay of brokers in order to ensure confidentiality.

Attribute-Based Access Control (ABAC) has emerged as a flexible policy-based authorisation solution. A policy is attached to some object, or resource, and describes what operations may be performed upon this object. A subject, a person or a service, then requests an authorisation to perform some operations and provides attributes associated with the subject, object, requested operations, and possibly, conditions of the environment. If the attributes provided by the subject

satisfy the access control policy established by the object owner, then access is granted, otherwise it is denied. ABAC is implemented in the OASIS XACML (eXtensible Access Control Markup Language) standard [28]. The architecture for managing XACML policies relies on different components among which the Policy Enforcement Point (PEP) receives access requests from subjects and sends them back access decisions. In the current implementation of muDEBS, a PEP is included within each broker of the context distribution overlay.

In [17], we have identified models for a first set of attributes to be specified in privacy policies: purpose (intention of use), visibility (who has access), retention (for how long data may be retained) and QoC. Considering both the purpose and quality dimensions allows to condition the delivery of high quality data to a specific purpose. Following these models, context producers specify privacy requirements in context contracts, that are then registered in muDEBS as XACML policies. On their side, context consumers express in consumer contracts the privacy guarantees that they are committed to fulfilling, mentioning at least for what purpose they are requesting access to some specific context data. This means they indicate explicitly to the INCOME framework for what reasons they request data and what is their intended use. Privacy guarantees take the form of ABAC information that is registered by muDEBS with the subscription filters.

At publication time, muDEBS must first of all determine whether the ABAC information stored within a subscription filter does match the privacy policy associated with an advertisement in order to authorise or not access to the publication message. As muDEBS allows local and global advertisements and also local and global subscriptions, we must first determine where privacy filters must be evaluated.

We define the following abbreviations:

- la = local advertisement and $\neg la$ = global advertisement,
- ls = local subscription and $\neg ls$ = global subscription,
- p = advertisement with privacy requirement,
- $s1$ = publication path of size 1 (every publication is tagged with the path of brokers that have forwarded it).

It is sufficient to analyse the privacy requirements when the notification arrives at the first broker that knows both the advertisement and the subscription filters. We can then deduce the following predicate: $p \wedge [s1 \wedge (la \vee \neg ls) \vee (\neg la \wedge ls)]$. Next, it follows guidelines for the implementation of the matching of privacy requirements with privacy guarantees. When analysing a subscription filter, two cases are possible: (1) If the advertisement corresponding to the publication is present on the broker that performs the matching, then the ABAC information of the subscription is analysed with respect to the XACML policy associated with the advertisement; (2) If the advertisement corresponding to the publication is not present, then the publication is tagged with a boolean stating that there exists a privacy requirement. This allows to maintain the set of *ACE* (*A*ccess *C*ontrol already *E*nabled) subscriptions corresponding to subscriptions for which the ABAC information has matched the XACML policy. The *ACE* set is computed by the access broker of the producer and is used by the other brokers when the advertisement is not present.

We present in Algorithm 1 the pseudo-code of the `evaluate()` function of a subscription filter, that is a composite filter—i.e. a conjunction of simple filters

such as privacy filters, simple subscription filters... Lines 5 and 9–10 analyse the privacy filter but only when necessary. Lines 7–8 detect when access control is not enabled, and lines 2–3, 6, and 12 detect whether a privacy filter is missing.

Algorithm 1: Evaluate function of a subscription filter

```
(1)  function evaluate(Publication n)
(2)      if p then privacyFilterFound ← false          {a privacy filter is required}
(3)      else privacyFilterFound ← true                {publication with privacy requirements}
(4)      forall f ∈ filters do
(5)          if f is a PrivacyFilter then
(6)              privacyFilterFound ← true
(7)              if p ∧ ¬ls ∧ ¬s1 ∧ f ∉ ACE then
(8)                  return false          {not on producer's access broker and access control not
                     enabled}
(9)              if ¬[p ∧ [s1 ∧ (la ∨ ¬ls)] ∨ (¬la ∧ ls)]] then
(10)                 continue                {no necessity to analyse the privacy filter}
(11)     if ¬f(n) then return false                    {conjunction of filters}
(12) return privacyFilterFound
```

2.3 QoC-aware context contracts

muContext is a framework that offers distributed context management. It involves three categories of software entities (see Figure 1): context collectors, context processing capsules, and context-aware applications. Each of these categories of components implement a functional part of context management.

A context collector is a software entity dealing with the acquisition of raw context data—i.e. that have not yet been processed or transformed—and it associates QoC meta-data to raw context data. We have proposed in [18] the dedicated QoCIM (Quality of Context Information Model) meta-model, which offers a unified solution to model heterogeneous meta-data about QoC. QoCIM facilitates exploiting and manipulating criteria in an expressive, computable, and generic way.

A context capsule is a functional element that performs the processing of context information into information of a higher level of abstraction. It is a consuming and producing entity. Several categories of context data manipulation can be operated by a capsule: aggregation, filtering, fusion, inference... The context management operations do not only perform a transformation of the context data flow. They also analyse what the impacts are on the management of QoC meta-data during these manipulations that encompass more and less complex operations like add/retrieve QoC indicators, update the value of an indicator, filter on the presence of an indicator, or filter on the value of an indicator.

QoC-aware contracts facilitate the expression of QoC requirements and guarantees. Context producers express guarantees on the QoC of the data they provide. Conversely, consumers express QoC requirements. Decoupled contracting is based on advertisement and subscription filters. An advertisement filter allows to express guarantees related to one or more QoC indicators, while a subscription filter specifies a requirement concerning some QoC indicators. The specification of contracts and their translation into filters is of the responsibility of muContext while their implementation and evaluation are of the responsibility of muDEBS.

3 Illustrative smart city scenario

3.1 An asthma traveller in Paris

Bob arrives in Paris to visit the city. He likes to walk or ride a bike. However, he is an asthma patient very allergic to polluted air. Recently, he has been using a special wrist-mounted device equipped with temperature and ECG sensors, and connected via Bluetooth to his smart-phone. His host in Paris explains to him that thanks to the INCOME framework, a lot of context-aware applications are now available to inform and help the citizens in their daily life. The added-value of the INCOME framework is to evaluate the quality of the information it manipulates in order to serve adequately its users while guaranteeing their privacy. His host advises him the `FreeAir` application that will provide him with a city map showing the pollution level in the streets, alert him in case of high pollution and recommend him a less polluted itinerary. His host also recommends Bob to install locally on his phone a wrapper provided by INCOME to get his health situation so that the city medical emergency services could rescue him automatically in case of danger of death.

As Bob is particularly concerned with privacy, he agrees to provide his current location but only for health purposes. In the general case, he decides that an approximate location should be sufficient. But he also considers that being able to be rescued automatically by the nearest emergency services in case of life danger would be of valuable help. In such a break-the-glass situation, he agrees to disclose his most precise location and the vital sensors data collected by his wrist device.

3.2 Scenario implementation

Using the INCOME framework, Bob can deploy context collectors on his smart-phone to collect his location and vital sensors data. He can then configure easily his privacy requirements using the Kapuer tool [24][4], which helps to automate the elaboration of privacy policies through a learning process so that the users do not have to write them by themselves. Pollution collectors are deployed by the city management services to get measurements from sensors disseminated in the city. QoC meta-data are associated to these raw data (e.g. precision of the measurements) at collection time. INCOME context processing capsules then use raw context data to extract high level context information. Such a capsule can be installed on Bob's smart-phone to analyse the body temperature and ECG data collected from Bob's wrist device in order to determine his health situation. This capsule is both a consumer of vital sensors data and a producer of health situations.

Table 1 gives some elements of the context contracts defined for producer and consumer entities.

[4] `http://kapuer.org/en/index.html`

	Producers		
	Pollution level	Bob's GPS location	Bob's wrist device sensors
Privacy Requirements	N/A	Rule 1: Purpose = health and no hazard situation Rule 2: Purpose = health and life hazard situation	Purpose = health and life hazard situation
QoC Guarantees	QoC $\geq$ high	Rule 1: QoC $\geq$ medium Rule 2: QoC $\geq$ high	QoC $\geq$ high

	Consumers	
	FreeAir application	City Emergency services
QoC Requirements	Location with QoC $\geq$ medium Pollution level with QoC $\geq$ high	Location with QoC $\geq$ high
Privacy Guarantees	Purpose = health	Purpose = health and life hazard situation

Table 1 Elements of context contracts

4 Evaluation of the cost of privacy filtering and QoC-based filtering

The measurements are obtained on machines equipped with an Intel Core Duo P9400 processor clocked at 2.40 GHz and with 4 GiB of RAM. Without loss of generality, we present experimental results for configurations in which advertisements are local and subscriptions are global. Therefore, when measuring the execution time of the forwarding of a publication, we have to consider the cost of filtering on two categories of brokers: the access broker of the producer (named B_{prod} in the following) that applies advertisement and subscription filters, and the other broker towards the consumers (named B_{other} in the following) that applies only subscription filters. All the measurements correspond to the mean over 1000 publications with a warm up period of 500 publications.

We begin the evaluation with the expression of the execution time on brokers as a linear combination depending on the number of subscriptions. Afterwards, the cost of privacy filtering and QoC-based filtering are expressed as overheads.

Basic context-based filters. The decision of forwarding a notification depends on the evaluation of the different constraints composing a routing filter. Recall that notifications are represented as XML documents and that routing filters are built with JavaScript functions that contain XPath expressions. The first experiment consists in applying a routing filter to a single notification.

To obtain reference values of execution times, we measure the execution time of context-based filters with neither utilisation of privacy constraints nor analysis of QoC meta-data. The context data part of a notification message is structured with three main elements: (i) a *context observable* is an abstraction that defines something to watch over (observe); (ii) a *context entity* represents a physical or logical phenomenon (person, concept, etc.) to which context observables may be associated; (iii) a *context observation* is the state of an observable at a given time. An URI identifies the context entity and the context observable. We evaluate

context-based filters composed of only one constraint testing the URI. Listing 1 presents an example of such a constraint. It specifies the pollution observable measured by the sensor with $id = 45$ that is placed in the "Henri Martin" avenue in Paris. The results of this first experiment indicate that the execution time increases linearly with the number of subscriptions N_s: $T_{prod} \approx 3.39 \times N_s + 60$ ms and $T_{other} \approx 3.40 \times N_s + 21$ ms. The confidence intervals are 0.11 ms and 0.04 ms, respectively. The overhead in T_{prod} compared to T_{other} is due to the additional matching of the advertisement filter. Thereafter, the execution times 63 ms and 24 ms serve as references when analysing the cost of privacy filters and the cost of enriched context-based filters.

Listing 1 Example of context-based constraint

```
// Context−based constraint
if (xpath.evaluate("//observable[uri='#pollution' and
    entity[uri='paris://henri_martin_avenue./sensors/45/']]",
    doc, XPathConstants.NODESET).length == 0) {
    return false;}
```

4.1 Cost of privacy filters

We evaluate the cost of privacy filters—i.e. subscription filters that are associated with ABAC information—with the open source XACML 3.0 implementation Balana[5]. If not stated otherwise, a policy is defined by four attributes that are divided into the three following categories: "access-subject", "resource" and "action". An example of a policy used in the experiment is presented in Listing 2. The attributes INCOME and health belong to the category "access-subject". The attributes location and access belong to the categories "resource" and "action", respectively. A privacy filter may be used to match against one or several policies. In order to deal with a policy set, we use the XACML combining algorithm named "permit-overrides", which allows a single evaluation of "permit" to take precedence over any number of results of the categories "deny", "not applicable" or "indeterminate".

Listing 2 Example of XACML Policy

```
<?xml version="1.0" encoding="UTF−8"?>
<Policy ...
  RuleCombiningAlgId="urn:oasis:names:tc:xacml:3.0:rule−combining−algorithm:permit−overrides">
  <Description>Policy example used in performance evaluations</Description>
  <Target />
  <Rule RuleId="urn:oasis:names:tc:xacml:3.0:example:simple:ruleid:3" Effect="Permit">
    <Description>People from "INCOME" with purpose "health" can access data "location"</Description>
    <Target>
      <AnyOf>
        <AllOf>
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string−equal">
            <AttributeDesignator MustBePresent="true"
              AttributeId="urn:oasis:names:tc:xacml:3.0:example:attribute:group"
              DataType="http://www.w3.org/2001/XMLSchema#string"
              Category="urn:oasis:names:tc:xacml:1.0:subject−category:access−subject"/>
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">INCOME
              </AttributeValue>
          </Match>
        </AllOf>
      </AnyOf>
      <AnyOf>
        <AllOf>
          <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string−equal">
            <AttributeDesignator ...
```

[5] http://xacmlinfo.org/category/balana/

```
25                Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject" />
26              <AttributeValue ...>health</AttributeValue>
27            </Match>
28          </AllOf>
29        </AnyOf>
30        <AnyOf>
31          <AllOf>
32            <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
33              <AttributeDesignator ...
34                Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource" />
35              <AttributeValue ...>location</AttributeValue>
36            </Match>
37          </AllOf>
38        </AnyOf>
39        <AnyOf>
40          <AllOf>
41            <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
42              <AttributeDesignator ...
43                Category="urn:oasis:names:tc:xacml:3.0:attribute-category:action" />
44              <AttributeValue ...>access</AttributeValue>
45            </Match>
46          </AllOf>
47        </AnyOf>
48      </Target>
49    </Rule>
50    <Rule RuleId="urn:oasis:names:tc:xacml:3.0:example:simple:ruleid:4:default" Effect="Deny">
51      <Target />
52    </Rule>
53  </Policy>
```

As indicated in Section 2.2, the access broker of the producer B_{prod} computes the set of *ACE* subscriptions—i.e. subscriptions for which ABAC information has matched the XACML policy—that is used by the access broker of the consumer B_{other}. Therefore, the impact on the execution time at B_{other} is negligible. In the sequel, we describe only the results of B_{prod}.

Number of policies. We measure the execution time of a privacy filter when the number of policies maintained by the PEP at B_{prod} is increased. There is only one policy that matches the privacy filter, and the matching policy is the last one that has been added to the PEP.

Figure 4 shows that the impact of the number of non-matching policies on the execution time at B_{prod} is negligible when it is less than 500. When the number of policies is more than 500, the execution time increases linearly with the number of non-matching policies N_p : $T_{prod} \approx 0.09 \times N_p$ ms. The overhead is approximately 32% for 1000 non-matching policies.

Number of attributes. We now consider one (matching) policy and measure the execution time when the number of attributes used to define that policy is increased.

Figure 5 shows that the execution time increases moderately with the number of attributes. For a policy that is based on 32 attributes, the overhead is about 5.15%.

In conclusion, Section 2.2 analysed where privacy filters must be evaluated depending on the modes (local or global) of advertisements and subscriptions. Thanks to this analysis, privacy filters need not be evaluated by every broker, but only by the first broker in the path to the consumer that possesses the policy in its PEP. This contributes to limit the end-to-end overhead of access control.

4.2 Cost of QoC-based filters

We now evaluate the cost of QoC-based filters. We define two types of constraints relative to QoC meta-data. A QoC-criterion constraint controls if the meta-data

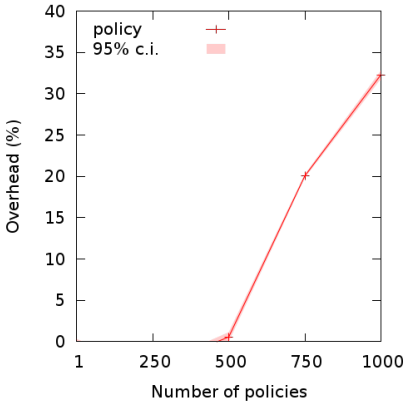

Fig. 4 Overhead of a privacy filter depending on the number of policies at B_{prod}

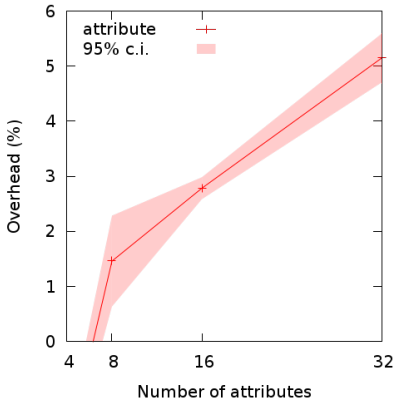

Fig. 5 Overhead of a privacy filters depending on the number of attributes at B_{prod}

of a notification include all the expected QoC criteria. It therefore evaluates the attribute *id* of the classes QoCIndicator, QoCCriterion and QoCMetricDefinition of the QoCIM meta-model [18]. In addition to testing the presence of QoC criteria, a QoC-value constraint also controls the value of each QoC indicator by evaluating the attribute *value* of the class QoCMetricValue. Listing 3 presents an example of each type of constraints. Both constraints specify the criterion with $id = 10.1$ and the QoC-value constraint specifies a QoC metric value larger than 40.

Figures 6 and 7 show that the overhead of adding QoC meta-data for B_{prod} and B_{other} is very low since it does not exceed 16% and 19% for 16 meta-data, respectively for B_{prod} and B_{other}. The overhead is more important when filtering on QoC values than when filtering on QoC criteria, but the highest gap is less than 5%. For example, in the case of B_{prod} with 8 QoC meta-data, the percentages are 7.2% and 10.5%, respectively, and the difference is approximately 3.3%.

Listing 3 Examples of QoC-criterion constraint and QoC-value constraint

```
// QoC−criterion constraint
if (xpath.evaluate("//qocindicator[@id='10'
    and qoccriterion[@id='[10.1]']/
    qocmetricdefinition[@id='10.1']]", doc,
    XPathConstants.NODESET).length == 0) {
    return false;}

// QoC−value constraint
if (xpath.evaluate("//qocindicator[@id='10' and
    qoccriterion[@id='[10.1]']/
    qocmetricdefinition[@id='10.1']
    and qocmetricvalue[@value>='40']]", doc,
    XPathConstants.NODESET).length == 0) {
    return false;}
```

Figures 8 and 9 show that the overhead is much more important when QoC constraints are added. For example, in the case of B_{other} with 16 constraints, the

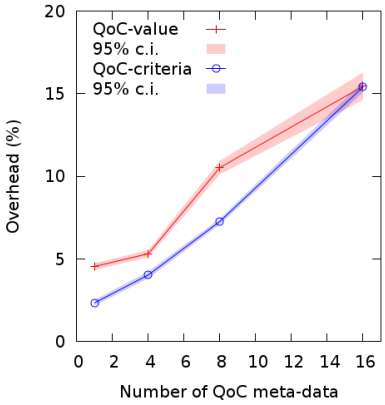

Fig. 6 Overhead of QoC-based filters depending on the number of meta-data at B_{prod}

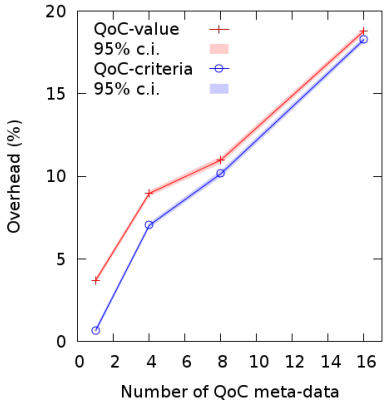

Fig. 7 Overhead of QoC-based filters depending on the number of meta-data at B_{other}

percentage is approximately 150% of 26 ms. This kind of filters with numerous QoC constraints are more than sufficient for implementing the `FreeAir` scenario.

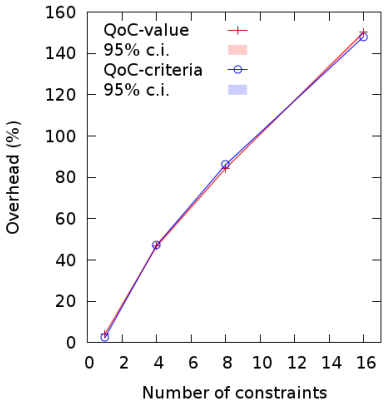

Fig. 8 Overhead of QoC-based filters depending on the number of constraints at B_{prod}

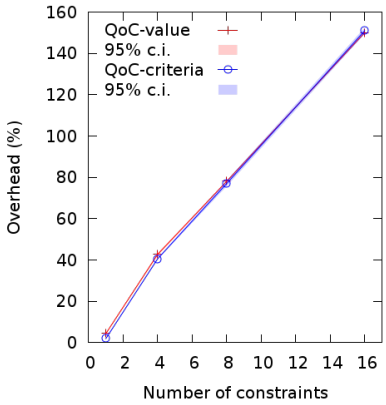

Fig. 9 Overhead of QoC-based filters depending on the number of constraints at B_{other}

5 Related works

[3] distinguishes "uninformed" and "informed" context data distribution, that is whether the routing according to context needs is performed blindly or not. The solution presented in this paper lies with informed context data distribution, which allows to consider rich filtering mechanisms exploiting the presence of QoC metadata and privacy constraints.

Regarding QoC, a few works have started to consider the uncertainty of context data during the dissemination phase. [12] proposes a context data distribution infrastructure for query-based applications. Cache management strategies then rely on QoC for keeping only fresh data in the cache. We instead consider that future pervasive applications will benefit from distributed solutions like the ones using the publish/subscribe paradigm more than from more centralised solutions that bring to play the traditional request/response model. [16] considers quality-aware data stream management systems based on a relational model and with a probabilistic processing for the evaluation of quality of context. Even though such an approach is promising, it remains very sensitive to the choice of the system parameters for the probabilistic processing. [22] describes a quality-aware publish/subscribe system for mobile sensor networks. It proposes to rely on location-based routing to deliver the subscriptions to the corresponding areas of interest. However, only consumers may express their QoC expectations and no advertisement is performed on the side of producers. We believe that a more powerful filtering can be obtained with content-based routing that benefits from both consumer requirements through subscriptions and producer guarantees through advertisements.

Concerning privacy protection in the IoT, most of the works offer confidentiality and anonymity using encryption mechanisms [1], k-anonymity or l-diversity models [27]. Under an open world assumption where data can be combined with external sources, recent works have shown that anonymization alone is not sufficient as re-identification becomes easy [26]. Our approach is complementary by granting access to consumers based on their intended use of context data. [4] presents a general architecture for integrating Role-Based Access Control (RBAC) into publish/subscribe systems. However, as discussed in [7], RBAC models were designed for stable computing environments involving limited mobility. As a consequence, they did not consider the notion of context. We rather consider Attribute-Based Access Control (ABAC) [8] that allows to manipulate any kind of attributes. In our approach, these attributes are expressed using semi-structured data enabling to specify flexible context-aware policies.

Another line of research is building a semantic web of things has emerged recently and proposes to exploit ontology-based solutions to build an application-oriented view of the IoT [2]. [5] presents a comparison of process-based context management, as proposed in this paper, and ontology-based context management and shows their complementarity. On the one hand, an ontology-based approach puts the stress on extracting knowledge from context data in conjunction with existing knowledge bases. On the other hand, a process-oriented approach focuses on handling dynamic context data in a scalable way.

6 Conclusion

This article proposes to add QoC-based filtering and attribute-based privacy policies in a DEBS infrastructure as a middleware solution for an efficient context data distribution in the IoT. We rely on a generic DEBS pattern and a generic QoC modelling approach that are both agnostic of the context model to address the heterogeneity of the IoT. Such heterogeneity also guided our choice of a

semi-structured data model to express rich and flexible filters able to manipulate any kind of attributes. The evaluation results on a first prototype implementation, available as open-source software, show that the cost of QoC-based and privacy filters is reasonable. We determine where along the dissemination path it is sufficient to evaluate privacy filters, contributing to limit the overhead of privacy protection. Our short-term work concerns the mobility of the elements of the DEBS infrastructure, first of all clients and then even brokers, which can be embedded in mobile vehicles such as buses. Our longer term perspectives target the enforcement of use control policies. As pointed out recently [23], data collection is unavoidable in an open world, and further work is needed on controlling the use of such data.

Acknowledgments This work is part of the French National Research Agency (ANR) project INCOME[6] (ANR-11-INFR-009, 2012-2015). The authors thank all the members of the project who contributed directly or indirectly to this article. We also want to thank the referees for their useful suggestions.

References

1. R. Barazzutti, P. Felber, H. Mercier, E. Onica, and E. Rivière. Thrifty Privacy: Efficient Support for Privacy-preserving Publish/Subscribe. In *6th ACM Int. Conf. on Distributed Event-Based Systems*, pages 225–236, New York, NY, USA, 2012. ACM.
2. Barnaghi, P. and Wei W. and Cory H. and Kerry Taylor. Semantics for the Internet of Things: Early Progress and back to the Future. *International Journal on Semantic Web and Information Systems*, 8(1):1–21, 2012.
3. P. Bellavista, A. Corradi, M. Fanelli, and L. Foschini. A Survey of Context Data Distribution for Mobile Ubiquitous Systems. *ACM Computing Surveys*, 44(4):24:1–24:45, August 2012.
4. A. Belokosztolszki, David M. Eyers, Peter R. Pietzuch, J. Bacon, and K. Moody. Role-based Access Control for Publish/Subscribe Middleware Architectures. In *2nd Int. Workshop on Distributed Event-based Systems*, pages 1–8, 2003.
5. A. Bouzeghoub, C. Taconet, A. Jarraya, N.K. Do, and D. Conan. Complementarity of Process-oriented and Ontology-based Context Managers to Identify Situations. In *Proc. 5th International Conference on Digital Information Management*, Thunder Bay, Canada, June 2010.
6. T. Buchholz, A. Kupper, and M. Schiffers. Quality of Context Information: What it is and why we Need it. In *10th Int. Workshop of HPOVUA*, Geneva, July 2003.
7. S. Chabridon, R. Laborde, T. Desprats, A. Oglaza, P. Marie, and S. Machara Marquez. A Survey on addressing privacy together with quality of context for context management in the internet of things. *Ann. Telecommun.*, 69, Issue 1:47–62, February 2014.
8. M. J. Covington and M. R. Sastry. A Contextual Attribute-based Access Control Model. In *Proceedings of the 2006 International Conference on On the Move to Meaningful Internet Systems*, pages 1996–2006, Berlin, Heidelberg, 2006. Springer-Verlag.
9. A.K. Dey. Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7.
10. C. Esposito and M. Ciampi. On Security in Publish/Subscribe Services: a Survey. *IEEE Communications Surveys & Tutorials*, (on-line), 2015.
11. P.T. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The Many Faces of Publish/Subscribe. *ACM Computing Surveys*, 35(2), June 2003.
12. M. Fanelli, L. Foschini, A. Corradi, and A. Boukerche. QoC-Based Context Data Caching for Disaster Area Scenarios. In *IEEE Int. Conf. on Communications, Kyoto, Japan, 5-9 June*, pages 1–5, 2011.
13. V. Garg, L. Camp, L. Lorenzen-Huber, K. Shankar, and K. Connelly. Privacy concerns in assisted living technologies. *Ann. Telecommun.*, 69(1-2):75–88, 2014.

[6] http://anr-income.fr

14. K. Henricksen and J. Indulska. Modelling and using Imperfect Context Information. In *1st IEEE PerCom Workshop CoMoRea*, pages 33–37, March 2004.
15. A.-M. Kermarrec and P. Triantafillou. XL Peer-to-Peer Pub/Sub Systems. *ACM Computing Surveys*, 46(2):16:1–16:45, November 2013.
16. C. Kuka and D. Nicklas. Quality Matters: supporting Quality-aware Pervasive Applications by Probabilistic Data Stream Management. In *The 8th ACM Int. Conf. on Distributed Event-Based Systems*, pages 1–12, May 2014.
17. S. Machara Marquez, S. Chabridon, and C. Taconet. Trust-based Context Contract Models for the Internet of Things. In *10th IEEE UIC/ATC Conference*, December 2013.
18. P. Marie, T. Desprats, S. Chabridon, and M. Sibilla. QoCIM: a Meta-model for Quality of Context. In *8th Int. Interdisciplinary Conf. on Modeling and Using Context*, volume 8175 of *LNCS*. Springer, October 2013.
19. P. Marie, L. Lim, A. Manzoor, S. Chabridon, D. Conan, and T. Desprats. QoC-Aware Context Data Distribution in the Internet of Things. In *1st Workshop on Middleware for Context-Aware Applications in the IoT, 15th Middleware Conf.*, pages 8–12, Bordeaux, France, December 2014. ACM.
20. G. Mühl, L. Fiege, and P.R. Pietzuch. *Distributed Event-Based Systems*. Springer, 2006.
21. M. Nabeel, S. Appel, E. Bertino, and A. P. Buchmann. Privacy Preserving Context Aware Publish Subscribe Systems. In Springer, editor, *7th Int. Conf. on Network and System Security*, June 2013.
22. E. Ngai and P. Gunningberg. Quality-of-information-aware Data Collection for Mobile Sensor Networks. *Pervasive and Mobile Computing*, 11:203–215, 2014.
23. PCAST (President's Council of Advisors on Science and Technology). Big Data and Privacy: A Technological Perspective. *www.whitehouse.gov/ostp/pcast*, May 2014.
24. A. Oglaza, R. Laborde, and P. Zaraté. Authorization Policies: Using Decision Support System for Context-Aware Protection of User's Private Data. In *5th IEEE UbiSafe Sympoisum, TrustCom*, pages 1639–1644, July 2013.
25. Z. Shelby, K. Hartke, and C. Bormann. Constrained application protocol (CoAP). *IETF, Request for Comments, https://tools.ietf.org/html/rfc7252*, June 2014.
26. L. Sweeney, A. Abu, and J. Winn. Identifying Participants in the Personal Genome Project by Name. *Social Science Research Network, http://dx.doi.org/10.2139/ssrn.2257732*, 2013.
27. R. Weixiong, C. Lei, and T. Sasu. Toward Efficient Filter Privacy-Aware Content-Based Pub/Sub Systems. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2644–2657, November 2013.
28. eXtensible Access Control Markup Language (XACML) Version 3.0. docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html, January 2013.