Usage Control for Data Handling in Smart Cities

Quyet H. Cao*†, Giyyarpuram Madhusudan*, Reza Farahbakhsh†, Noel Crespi†
*Orange Labs, France
Email:{quyet.caohuu, giyyarpuram.madhusudan}@orange.com
†Institut Mines-Telecom, Telecom SudParis, CNRS UMR 5157, France
Email: {reza.farahbakhsh, noel.crespi}@it-sudparis.eu

***Abstract*—Data in smart cities is commonly generated by a large variety of participants including institutional actors, equipment manufacturers, network operators, infrastructure providers, service providers, and end users. This data potentially undergoes several transformations such as aggregation and/or composition before finally being consumed. In this context of sharing data between diverse consumers, it is essential to provide the data producers the means by which they can exercise control over how and by whom the data is used. To date, usage control has received attention in the domains of the web and social networks, in terms of confidentiality, privacy and access control aspects. However, it has not yet been fully applied in a rigorous manner in the context of smart cites. In this paper we study usage control with the goal to address the problem of providing stakeholders more control over their data and enforcing accountable management of such data. We first propose a new data usage policy, called DUPO, which captures the diversity of obligations and constraints resulting from the usage control requirements for smart cities. Next, we apply a defeasible logic based approach on DUPO to formally define rule language, solve rule conflicts, and elaborate reasoning. We then introduce the data handling mechanism, which provides useful functionality to process consumer's request, ensuring the accountability of the policy enforcement, and traceability of the data usage. To this end we benefit from SPINdle reasoner to implement the proposed usage control module covered main functionalities of the mechanism.**

***Keywords*—Defeasible Logic, Usage Control, Data Handling, Smart Cities, and Information Accountability.**

I. Introduction

We are witnessing a new communication paradigm which goes beyond traditional people interactions, the one that is between devices under the umbrella of the Internet of Things (IoT) and the underlying technologies. In the very near future, most of the defined plans and ideas for smart cities will become a reality. In this era, deployment of the shared platforms for IoT enables the participation of citizens and groups of users in both the data collection and the emergence of new smart city services. Data can be collected from billions of interactions across a huge number of devices, forever altering the socioeconomic landscape [3]. In effect, it is the emergence of a marketplace for smart cities.

Currently, applications for smart cities are mostly developed in a vertical manner with no sharing of resources between different applications. Many of these vertical applications may benefit from using information sources from different origins to enhance their own services. These resources include the devices and the networks that are deployed but also the data generated by these devices. The landscape involves a diversity of actors, both public and private, who participate to provide a large variety of services. These applications include energy management for public buildings, waste management, public lighting, mobility management including intelligent parking solutions and a range of new services that are being conceived for smart cities [2]. The actors involved in these applications tend to vary with the specific domain as each comes with its ecosystem. However we can identify several broad categories of actors: institutional actors (such as districts, municipalities), equipment manufacturers, network operators, infrastructure providers and service providers. With the development of IoT, the range of actors involved will be enlarged to include micro companies, value added service providers (such as aggregations, compositions and mashups) and end users. The need for a horizontal platform to federate information from these disparate sources is particularly important [4]. The shared platform, which can be provided by the operator, for actors with differing and sometimes contradictory requirements brings its own set of challenges. For this horizontal approach to succeed, the platform needs to ensure that the business interests of the different participants are fully honored.

In the context of IoT current studies provide mechanisms such as access control or address privacy issues. However the issue of usage control has not been treated in a formal manner. In this paper we focus on the following challenges (i) A language for IoT data producers to express the constraints and obligations on the use of IoT data, (ii) The mechanisms to enforce the policies that have been put in place in the context of a shared data platform, and (iii) Accounting mechanisms to prove to the data producers that their policies have been respected. More specifically, we need a machine-interpretable model and a language for the constraints and conditions that can be used and understood by all actors who are involved in, covering (i) Spatio-temporal granularity, (ii) Abstraction/-masking of certain information, and (iii) Conditions by class of actor/purpose.

The main contributions of this paper are two-fold: Firstly, we propose a new data usage policy, called DUPO, and define its conceptual model, as well as a formal language based on defeasible logic, and a practical expression of DUPO. Secondly, a data handling mechanism is provided, in which we indicate an overview of the mechanism in perspectives of data owners, data consumers, and IoT shared platform. Next, we initial implement the Usage Control Module that covers the main functionalities of the mechanism.

The rest of this paper is organized as follows: Section II describes use case scenarios and section III discusses DUPO. Section IV presents the data handling mechanism, section V indicates related works and we conclude in Section VI.

II. Use case Scenarios

To illustrate our contributions, we introduce the architecture of a general scenario with a use case for fine-granular sensor data collection in the context of smart cities.

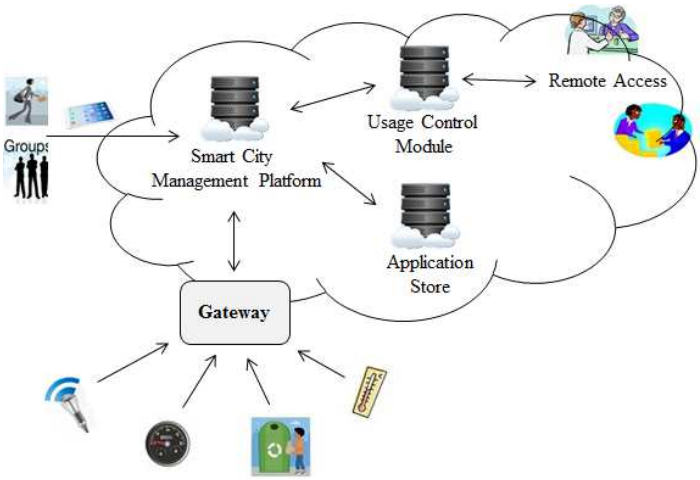

Fig. 1: Overall Architecture of the Use Case Scenario.

Fig. 1 shows the overall architecture, including the main components for our use case scenarios. A variety of smart city services communicate to a shared platform called *Smart City Management platform*. Examples of such services include intelligent parking solutions, waste management, public lighting, air quality monitoring, and participatory sensing applications, etc. The goal of the shared platform is to allow diverse applications to access data collected from a variety of sensors deployed over a large area. A critical element to enable such sharing is the possibility for data providers to exercise some control the usage of the data generated by their sensors. The *Usage Control Module* ensures that policies put in place by the data producers are respected by data consumers.

We could have different applications, $aParking$ for actors to be used for tracking the parking spaces in a city during a specific time in the intelligent parking solution, $aWaste$ for waste management, $aLighting$ for a public lighting system, $aAir$ for air quality monitoring, and aPS for participatory sensing, etc. sharing a common platform and some of their data between different them.

In the sections that follow we have chosen to illustrate *Usage Control* using an intelligent parking solution. Intelligent parking is a hot topic and several studies have proposed solutions in this domain [5]. This sort of solution is also available on the market and their integration is quite simple [1]. The data provider for this solution decides to share the data generated by parking sensors and apply a policy to control the usage of this parking data. Different participants such as municipal authorities, application developers and commercial operators could then request data from the parking solution and will be able to access data at the granularity and scope that the data producer has specified. In the following, we use examples to illustrate the usage control policy:

1) The data owner (the company that deploys and is owner of the parking sensors) will have full access to all the details generated by all the individual parking sensors.

2) For municipal authorities, the data owner is willing to make available average occupancy of parking places per street on an hourly basis.

3) However to commercial operators only statistical data will be made available over a zone and on a weekly basis.

We finally need the *Usage Control Module* in this scenario to deal with several questions as follows: (i) How do we define the required policies? (ii) What are the main criteria to define the policies? (iii) How do we ensure that these policies enforced correctly? (iv) How do we deal with potential incompatibilities between dependent policies? (v) How do we process the consumers' request and offer an explanation when the request is refused? and (vi) How do we trace data usage history?

III. DUPO: Data Usage Policy

This section introduces the new data usage policy, called DUPO, which includes the conceptual model, formal language, and practical expressions.

A. Conceptual Model

In Fig. 2, we propose a conceptual data usage model that allows a usage policy to be attached to a data set, a collection of data items. The policy is created by defining modal operators (Obligation, Forbidden, and Permission) on usage constraints and conditions: (i) class of actors, (ii) constraints (Spatiality, Temporality, and Abstraction), and (iii) class of purposes.

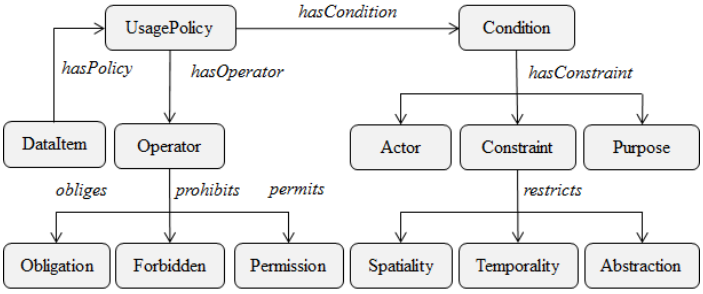

Fig. 2: Conceptual Data Usage Policy Model.

We aim to explain by examples how this model works. Let's consider that a commercial operator requests all the details of parking data over a street on an hourly basis. However, we have already the usage policy in the scenario that commercial operators are permitted only statistical data over a zone on a weekly basis. Thus, the consumer's request is refused. Otherwise, the related data items will be returned. This example basically covers the usage control requirements and related concepts in our model:

$$\begin{aligned} Actor &= (CommercialOperator), \\ Abstraction &= (Detail, StatisticalData), \\ Spatiality &= (StreetLevel, ZoneLevel), \\ Temporality &= (Hourly, Weekly), \\ Operator &= (Permission, Obligation). \end{aligned}$$

Next we present each component of the model in detail.

1) Data Items: A *Data Item* is an individual of the *Context Element* proposed in the NGSI 9/10 Information Model[1]. The *Context Element* is a container used to exchange information about an entity. It contains the following information: (i) an entity ID including the name and the type, (ii) a list of the context attributes, (iii) (optionally) the name of an attribute domain that logically groups together a set of context attributes, and (iv) (optionally) a list of metadata that apply to all the attribute values of the given domain. We formally define it by using XML DTD, as mentioned in Listing 1.

```
<!DOCTYPE DUPO[
<!ELEMENT DataItem(ContextElement)>
<!ELEMENT ContextElement(EntityID,
    AttributeDomainName?, ContextAttributeList,
    DomainMetadata?)>
<!ELEMENT EntityID(Id, Type)>
<!ELEMENT ContextAttributeList(ContextAttribute*)
    >
<!ELEMENT ContextAttribute(Name, Type,
    ContextValue, ContextMetadata+)>
<!ELEMENT DomainMetadata(ContextMetadata*)>
<!ELEMENT ContextMetadata(Name, Type, Value)>
...
]>
```

Listing 1: XML DTD Definition of Data Item.

2) Usage Constraints and Conditions: This is a collection of individual *Conditions*. The condition list optionally contains the following expressions:(i) Temporal Constraints for temporal granularity, (ii) Spatial Constraints for spatial granularity, (iii) Abstraction Constraints for masking of certain information, (iv) Conditions by Actors, and (v) Conditions by Purposes. We formally define it by using XML DTD, as shown in Listing 2.

```
<!DOCTYPE DUPO[
<!ELEMENT Condition(Temporality*, Spatiality*,
    Abstraction*, Actor*, Purpose*)>
<!ELEMENT Spatiality(SpatialScope*)>
<!ELEMENT Temporality(TemporalScope*)>
<!ELEMENT Abstraction(AbstractScope*)>
<!ELEMENT Actor(ActorScope*)>
<!ELEMENT Purpose(PurposeScope*)>
<!ELEMENT TemporalScope(Secondly?, Minutly?,
    Hourly?, Daily?, Weekly?, Monthly?, Yearly?,
    Any?)>
<!ELEMENT SpatialScope(Space?, Slot?, Street?,
    Zone?, Any?)>
<!ELEMENT ActorScope(DataOwner?,
    MulnicipalAuthority?, ComercicalOperator?)>
<!ELEMENT AbstractScope(Aggregation?, Detail?,
    Any?)>
<!ELEMENT PurposeScope(CommercialUse?, Any?)>
...
]>
```

Listing 2: XML DTD Definition of Condition.

3) Operators: This is a set of *model operators* (i) Obligation (ii) Forbidden, and (iii) Permission. The formal definition created using XML DTD is presented in Listing 3.

```
<!DOCTYPE DUPO[
<!ELEMENT Operator(Obligation?, Forbidden?,
    Permission?)>
...
]>
```

Listing 3: XML DTD Definition of Operator.

[1]NGSI 9/10 Information Model: https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/NGSI-9/NGSI-10_information_model

4) Usage Policies: A collection of rules which is created by defining *Operators* on the individual *Condition*. Listing 4 formally defines the definition of *Usage Policies* using XML DTD.

```
<!DOCTYPE DUPO[
<!ELEMENT UsagePolicy(Rule*)>
<!ELEMENT Rule(Operator?, Condition?)>
...
]>
```

Listing 4: XML DTD Definition of Usage Policy.

B. Formal Language

Defeasible Logic (DL) is a non-monotonic formalism that deals with incomplete and conflicting information, originally proposed by Nute [11]. The Data Usage Policy proposed in this paper (DUPO) is based on the general concept of DL. In particular, we build on earlier works extending DL with modal and deontic operators, as presented in Governatori [14] [15] and Antoniou [13] [16]. There are some proposed formalisms for dealing with reasoning, handling and solving the normative conflicts that arise between rules and exceptions. However, DL is one of the best solutions which can manage all aspects in an efficient and computationally tractable way [15]. Moreover, DL offers enhanced representational capabilities and low computational complexity [12]. According to [14], when DL is enriched with modal operators, the complexity also does not increase in most cases. In this section, we define the formal language as follows:

Let $PROP$ be a set of propositional atom. A set of literals $Lit = PROP \cup \{\neg p | p \in PROP\}$.

Let $MOD = \{O, P, F\}$ be the set of basic deontic modalities (Obligation, Permission, and Forbidden). A set of modal literals $ModLit = \{[X]l, \neg[X]l | l \in Lit, X \in MOD\}$.

Let Lbl be a set of arbitrary labels. R is a set of base and deontic rules. A base rule is expressed $r : A(r) \hookrightarrow C(r)$, while a deontic rule is $r : A(r) \hookrightarrow_X C(r)$, where (i) A unique label $r \in Lbl$, (ii) The antecedent (or body) $A(r) = a_1, ..., a_n$, $a_i \in Lit \cup ModLit, 1 \leq i \leq n$; (iii) An arrow $\hookrightarrow \in \{\rightarrow, \Rightarrow, \rightsquigarrow\}$, denotes the type of rules: strict rules, defeasible rules and defeaters, respectively, (iv) $X \in MOD$, and (v) The consequent (or head) $C(r) = b, b \in Lit$.

The different rules have the following meaning. The strict rules can never be defeated, while defeasible rules can be defeated by contrary evidence. Defeater rules are only used to prevent certain conclusions.

1) DUPO Theory: A DUPO theory is a defeasible theory $D = (F^{DUPO}, R^{DUPO}, >)$, where $i) F^{DUPO} \subseteq Lit \cup ModLit$ are facts, $ii) R^{DUPO} \subseteq R$ is a set of rules from usage policies, and $iii) >$ is a superiority relation for priorities among the non-strict rules in R^{DUPO}.

2) Theory Proof: A conclusion derived from D is a tagged literal and is classified as follows: $+\Delta q$ means that literal q is definitely provable in D; $-\Delta q$ means that literal q is definitely rejected in D; $+\partial q$ means that literal q is defeasibly provable in D; and $-\partial q$ means that literal q is defeasibly rejected in D.

A proof $P = (P(1), ..., P(n))$ in D is a finite sequence of tagged literals of type $+\Delta q$, $-\Delta q$, $+\partial q$ and $-\partial q$.

We denote the set of all strict rules in R by R_s, R_{sd} for the set of strict and defeasible rules, and $R[q]$ for the set of rules whose head is q. $P[1..i]$ denotes the initial part of the sequence of length i. The proof conditions for the conclusions are formally defined as follows [13]:

$+\Delta : If\ P(i+1) = +\Delta q\ then\ either$
$(1)\ q \in F\ or$
$(2)\ \exists r \in R_s[q] \forall a \in A(r) : +\Delta a \in P[1..i].$

$-\Delta : If\ P(i+1) = -\Delta q\ then$
$(1)\ q \notin F\ and$
$(2)\ \forall r \in R_s[q] \exists a \in A(r) : -\Delta a \in P[1..i].$

$+\partial :\ If\ P(i+1) = +\partial q\ then\ either$
$(1) +\Delta q \in P[1..i]\ or$
$(2)(2.1) \exists r \in R_{sd}[q] \forall a \in A(r) : +\partial a \in P[1..i]\ and$
$(2.2)\text{-}\Delta \neg q \in P[1..i]\ and$
$(2.3) \forall s \in R[\neg q]\ either$
$(2.3.1) \exists a \in A(s) : -\partial a \in P[1..i]\ or$
$(2.3.2) \exists t \in R_{sd}[q]\ such\ that$
$\forall a \in A(t) : +\partial a \in P[1..i]\ and\ t > s.$

$-\partial :\ If\ P(i+1) = -\partial q\ then$
$(1) -\Delta q \in P[1..i]\ and$
$(2)(2.1) \forall r \in R_{sd}[q] \exists a \in A(r) : -\partial a \in P[1..i]\ or$
$(2.2) +\Delta \neg q \in P[1..i]\ or$
$(2.3) \exists s \in R[\neg q]\ such\ that$
$(2.3.1) \forall a \in A(s) : +\partial a \in P[1..i]\ and$
$(2.3.2) \forall t \in R_{sd}[q]\ either$
$\exists a \in A(t) : -\partial a \in P[1..i]\ or\ t \not> s.$

C. Practical Expression

In this part, we use our formal language to express facts, rules for usage policies, and consumer's request related to the use case examples. We also present an example of data items.

1) Facts: We have a fact about a commercial operator (CO) that wish to request the data. It is presented as follows:

$$F^{DUPO} = \{CommercialOperator(CO)\}$$

2) Rules: We express all of usage policies related to the use case scenarios. For the data owners (DO), they have the permission to full access of all the details. This policy is represented with the use of defeasible rules, as follows:

$$\begin{aligned} R^{DUPO} = \{&r_{1,d} : DO \Rightarrow_P TemporalScope(any),\\ &r_{2,d} : DO \Rightarrow_P SpatialScope(any),\\ &r_{3,d} : DO \Rightarrow_P AbstractScope(any),\\ &r_{4,d} : DO \Rightarrow_P PurposeScope(any)\} \end{aligned}$$

For municipal authorities (MA), they have permission to access available average occupancy of parking places $(aggregation)$ per street on an hourly basis. This policy is represented with the use of defeasible rules, as follows:

$$\begin{aligned} R^{DUPO} = \{&r_{1,m} : MA \Rightarrow_P SpatialScope(street),\\ &r_{2,m} : MA \Rightarrow_F \neg SpatialScope(street),\\ &r_{3,m} : MA \Rightarrow_P TemporalScope(hourly),\\ &r_{4,m} : MA \Rightarrow_F \neg TemporalScope(hourly),\\ &r_{5,m} : MA \Rightarrow_P AbstractScope(aggregation),\\ &r_{6,m} : MA \Rightarrow_F \neg AbstractScope(aggregation)\} \end{aligned}$$

For commercial operators (CO), only statistical data will be made available over a zone and on a weekly basis. This policy is represented with the use of defeasible rules, as follows:

$$\begin{aligned} R^{DUPO} = \{&r_{1,c} : CO \Rightarrow_P SpatialScope(zone),\\ &r_{2,c} : CO \Rightarrow_F \neg SpatialScope(zone),\\ &r_{3,c} : CO \Rightarrow_P TemporalScope(weekly),\\ &r_{4,c} : CO \Rightarrow_F \neg TemporalScope(weekly),\\ &r_{5,c} : CO \Rightarrow_P AbstractScope(statistic),\\ &r_{6,c} : CO \Rightarrow_F \neg AbstractScope(statistic)\} \end{aligned}$$

3) Consumer's Request: We have a consumer's request that a commercial operator (CO) requests all the detail of the parking data over a street on a hourly basis. This request is represented with the use of defeasible rules, as follows:

$$\begin{aligned} r : CO, &[P]SpatialScope(street),\\ &[P]TemporalScope(hourly),\\ &[P]AbstractionScope(detail)\\ &\Rightarrow_O ConsumerRequest \end{aligned}$$

We have the conlusions, $-\Delta[O]ConsumerRequest$, $-\partial[O]ConsumerRequest$. It means that $ConsumerRequest$ is not defeasible provable in D, so the request is refused.

4) Data Items: In the Listing 5, we describe a data item example using Context Element XML format. It contains the current state $(line\ 9)$ of the parking sensor $(line\ 3)$ in location $(line\ 14)$ at timestamp $(line\ 21)$.

```
1  <contextElement>
2    <entityId type="ParkingSensor" >
3      <id>ps1</id>
4    </entityId>
5    <contextAttributeList>
6      <contextAttribute>
7        <name>currentState</name>
8        <type>integer</type>
9        <contextValue>1</contextValue>
10     </contextAttribute>
11     <contextAttribute>
12       <name>location</name>
13       <type>string</type>
14       <contextValue>parkingspace1</contextValue>
15     </contextAttribute>
16   </contextAttributeList>
17   <domainMetadata>
18     <contextMetadata>
19       <name>timestamp</name>
20       <type>dateTime</type>
21       <value>2015-03-16T15:23:17.234+0200</value>
22     </contextMetadata>
23   </domainMetadata>
24 </contextElement>
```

Listing 5: An example of Data Item.

In fact, we have a collection of data items in different data sets. Thus, when the consumer's request is defeasible provable. These data items will be filtered or aggregated following the request conditions before returning results to consumers. Every transaction of data usage will be stored as new data items and later reported to the data owners as possible.

IV. Data Handling Mechanism

This section provides an overview of the data handling mechanism and an initial implementation of the usage control module.

A. Overview

Fig. 3 shows the sequence of the proposed mechanism for data handling, which is in the following steps: (1) Create Usage Policy, (2) Send Data, (3) Request Data, (4) Process Data Usage Enforcement, (5) Load Usage Policy, (6) Process Data Usage Accountability, (7) Store Data Usage History, (8) Response Data and Justification, and (9) Visualize Data Usage History. Traditional flows are provided by steps (2), (3), and a part of step (8) proposed traditionally $(normal\ lines)$, but steps (1), (4), (5), (6), (7), (8), and (9) are added to implement usage control and accounting $(dotted\ lines)$.

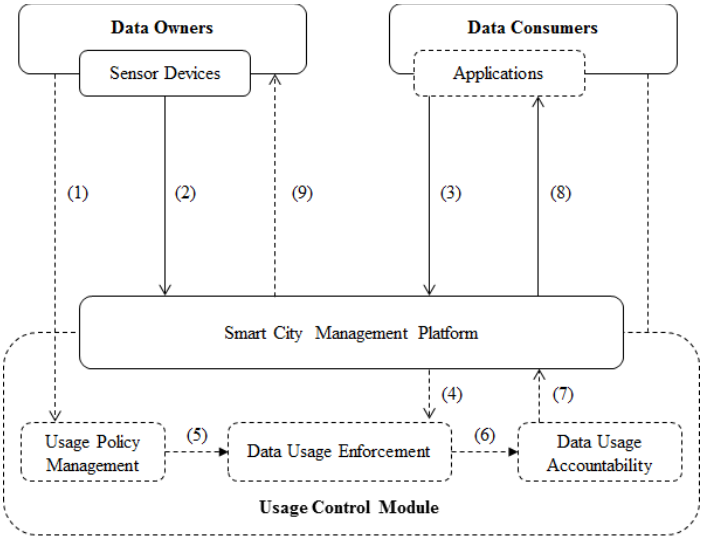

Fig. 3: Simplified Data Handling Mechanism.

Next, we present the main defined functionalities of the data owners, data consumers, and IoT trusted third party.

1) Data Owners: We first allow the data owners to create the usage policy for their data set, and then data from their sensor devices are sent to the shared platform. The data owners also can visualize the data usage history and adjust the usage policy as needed.

2) Data Consumers: The data consumers are able to request data using their applications. They can visualize not only the responsed data but also the justification for trusting the results.

3) IoT Shared Platform: The data collection and distribution will be managed in the *Smart City Management Platform*. The *Usage Control Module* is provided in order to incorporate transparency in how the data is used. It has three main functionalities as follows: *(i) Usage Policy Management* for managing data owner's policies, *(ii) Data Usage Enforcement* for reasoning consumer's requests, and *(iii) Data Usage Accountability* for tracing data usage history. We will explain more the module in the next section.

B. Usage Control Module

This part will provide more detail about the technical functionality of the usage control module which provided by the shared platform. We also aim at initial implementing it using SPINdle[2], a logic reasoner that can be used to compute the consequence of DUPO theories in an efficient manner [18].

1) Usage Policy Management: So far, we have explained how the usage policies are defined in the previous sections, from use case examples (section II) to defeasible logic rules (section III). However, we need a practical way to manage the policies. In this module, we suggest a rule based language which build on Defeasible SPINdle, and later extend to Defeasible RuleML[3] as possible. We will go further to provide a visual editor which supports data owners to define the policy for diversity of obligations and constraints mentioned above, and propose a DUPO service to load the related polices. For prototype, we use SPINdle systax to define facts and rules of usage policies in Listing 6.

```
1  # Facts
2  >> CO
3
4  # Usage policies
5  r1c: CO =>[P] SpatialScope(zone)
6  r2c: CO =>[F] -SpatialScope(zone)
7  r3c: CO =>[P] TemporalScope(weekly)
8  r4c: CO =>[F] -TemporalScope(weekly)
9  r5c: CO =>[P] AbstractScope(statistic)
10 r6c: CO =>[F] -AbstractScope(statistic)
11 ...
```

Listing 6: Facts and Usage Policies.

2) Data Usage Enforcement: This functionality aims to process the consumer's requests. Firstly, the request will be transformed into defeasible rules. We then use the DUPO service to load the related policies. Next, we deal with conflicts between the usage policies and consumer's request by setting superiority relation between the defeasible rules. Finally, we do reasoning to have the conclusions which are able to demonstrate that the related policies are correctly enforced. Listing 7 shows the consumer's request, the defined superiority relation, and reasoning conclusions from SPINdle engine.

```
1  # Consumer's request
2  r: CO,[P]SpatialScope(street),[P]TemporalScope(
       hourly),[P]AbstractionScope(detail) =>[O]
       ConsumerRequest
3
4  # Superiority relation
5  r1c > r
6  r2c > r
7  r3c > r
8  r4c > r
9  r5c > r
10 r6c > r
11 ...
12
13 # Conclusions
14 ==================
15 -D [O]ConsumerRequest(X)
16 -d [O]ConsumerRequest(X)
17 ...
```

Listing 7: Consumers' Request, Superiority Relation and Conclusions.

[2]SPINdle: http://spin.nicta.org.au/spindle/index.html
[3]Defeasible RuleML: http://ruleml.org/1.0/defeasible/defeasible.html

3) Data Usage Accountability: This functionality ensures the accountability of policy enforcement and traceability of the data usage history. We firstly define the justification [17] to the consumers based on extending of the SPINdle Inference Logger (as Listing 8). The data usage history is updated as every consumers' request is defeasible provable and responsed data is processed.

```
1 === Inference Logger ===
2 Rule_00000
3     +-- [DEFEASIBLE] Discarded :- [-d [O]
      ConsumerRequest(X)]
4 ...
```

Listing 8: Inference Logger.

V. Related Work

Usage control is concerned with how data is used after access to it has been granted. It has received attention in the domain of the social networks. In particular authors in [6] propose a simple accountability model and a platform that allows users to explore the consequences of different usage control choices. In addition, various efforts have been made in the web domain to address accountability and trust [10] [17]. Classic access control models have also been extended to implement obligations and conditions in the context of access to enterprise resources [8] and usage control mechanisms are well studied by authors cited in [9]. Recently, in the field of IoT, the work done by [7] they propose a privacy policy model that allow users give control over their data in context of smart grid application. However, our work is designed to bring a rigorous approach to usage control in the context of smart cities. We do not focus on the security aspects such as confidentiality, access control, and privacy. In fact, our paper deals with issues arising from applying usage control policies based on spatio-temporal granularity, abstraction/masking of certain information, and conditions by class of actors or purposes. In this work, policies has been built on deontic logic based on obligations, permissions and prohibitions, as in regular defeasible logic rules. Thus, we build on the work done in the area of DL by Governatori et al. [14] [15], and then focus on the accountability and traceability of data usage.

VI. Conclusion

In this paper we proposed a new policy model for usage control and its formalization using defeasible logic. We have also illustrated the enforcement of the above policies using SPINdle and the accounting mechanisms needed to ensure confidence to the data providers that the data is being used in a manner complaint to their policies. Two main contributions proposed in this study advance the state of the art in this domain. A new data usage policy, called DUPO, has been presented, including its conceptual model, formal language, and practical expression. We also introduce a novel data handling mechanism, including the definition and initial implementation of data owners', data consumers', and Usage Control Module' functionalities.

As future work, we aim to extend this study, with implementation and validating of the proposed models in the following directions: Firstly, experiment the proposed solution to check the performance related aspects to ensure that the system is efficient and scalable. Secondly, provide a visualization tool to help users customize their policies in a manner that allows them to explore the consequences of certain changes and finally enhancement to the language to allow monetization of data for example different levels of subscription or payment.

Acknowledgment

The research leading to these results was partially funded by the ITEA3 project CAP.

References

[1] A. Zanella, N. Bui, A. P. Castellani, L. Vangelista, M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, 2014.

[2] "Smart Cities - Le numrique au coeur de la ville intelligente, IDATE Study". [Online]. Available: http://www.idate.org/fr/Actualites/Smart-Cities_756.html

[3] "Global Agenda Council on Data-Driven Development, World Economic Forum". [Online]. Available: http://www.we forum.org/reports/global-agenda-council-data-driven-development-2012-2014

[4] FI-WARE Architecture. [Online]. Available: http://forge.fi-ware.org/plugins/mediawiki/wiki/fiware/index.php

[5] Y. Yin, D. Jiang, "Research and Application on Intelligent Parking Solution Based on Internet of Things," *Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, 2013, vol. 2, pp. 101-105.

[6] J. Pato, S. Paradesi, I. Jacobi, S. Fuming, S. Wang, "Aintno: Demonstration of Information Accountability on the Web," *IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT)*, 2011.

[7] S. Speiser, A. Wagner, O. Raabe, A. Harth, "Web Technologies and Privacy Policies for the Smart Grid," *Proceedings of the 39th Annual Conference of the IEEE Industrial Electronics Society (IECON), Springer*, 2013.

[8] J. Park, R. Sandhu, "The UCONABC usage control model," *ACM Trans. Inf. Syst. Secur. 7*, 2004.

[9] A. Lazouski, F. Martinelli, P. Mori, "Usage control in computer security: A survey," *Computer Science Review*, 2010, pp. 81-99.

[10] O. W. Seneviratne, "Augmenting the web with accountability," *In Proceedings of the 21st international conference companion on World Wide Web (WWW '12)*, 2012.

[11] D. Nute, "Defeasible logic," *Handbook of Logic in Artificial Intelligence and Logic Programming*, 1994, vol. 3, pp. 353-395.

[12] E. Kontopoulos, N. Bassiliades, G. Governatori, G. Antoniou, "A modal defeasible reasoner of deontic logic for the semantic web," *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2011, vol. 7, no. 1, pp. 18-43.

[13] G. Antoniou, D. Billington, G. Governatori, M. J. Maher, "Representation results for defeasible logic," *ACM Transactions on Computational Logic (TOCL)*, 2001, vol. 2, no. 2, pp. 255-287.

[14] G. Governatori, A. Rotolo, "BIO logical agents: Norms, beliefs, intentions in defeasible logic," *Autonomous Agents and Multi-Agent Systems*, 2008, 17, no. 1, pp. 36-69.

[15] G. Governatori, A. Rotolo, S. Villata, F. Gandon, "One License to Compose Them All:A Deontic Logic Approach to Data Licensing on the Web of Data," *In The Semantic WebISWC*, 2013, pp. 151-166.

[16] G. Antoniou, N. Dimaresis, G. Governatori, "A modal and deontic defeasible reasoning system for modelling policies and multi-agent systems," *Expert Systems with Applications*, 2009.

[17] E. Kontopoulos, N. Bassiliades, G. Antoniou, "Visualizing Semantic Web proofs of defeasible logic in the DR-DEVICE system," *Knowledge-Based Systems*, 2011.

[18] H. P. Lam, G. Governatori, "The making of SPINdle," *In : Rule Interchange and Applications. Springer Berlin Heidelberg*, 2009, pp. 315-322.